БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ

Единен център за наука и подготовка на кадри по математика и механика

Реализация на функционален процесор за обработка на символна списъчна информация

ДИСЕРТАЦИЯ

на инж. Георги Иванов Попов

за присъждане на научна степен
"Кандидат на математическите науки"

Научен ръководител: ст.н.с. к.ф.м.н. П. Бърнев

София, 1975г.

Увод

Представената дисертация е описание на извършена реализация на процесор за обработка на символна списъчна информация. Този процесор, наречен HELP, е от компилиращо-интерпретиращ тип и служи за въвеждане, транслиране и управление на изпълнението на потребителски програми, написани на езика за програмиране HELP.

В първа глава са дадени някои основни дефиниции за списъчни структури. По-голямо внимание е отделено на представянето и функционалния подход при обработката на еднопосочните символни списъци.

Методите за реализацията на процесора HELP са представени във втора глава. Обяснени са вграждането и моделирането като средства за намаляване на времето и усилията за реализиране на сложни програми-процесори.

Трета глава представлява описание на абстрактна машина за обработка на еднопосочни символни списъци. Процесорът HELP е кодиран на езика на тази абстрактна машина.

В четвърта глава е направено описание на системата за програмиране HELP. Най-напред са описани синтаксиса и семантиката на езика HELP, а след това и самият процесор HELP. Дадени са примери за програмиране на HELP. В края на главата е направено сравнение между системите HELP и LISP.

Пета глава представлява подробно описание на етапите на реализацията на процесора HELP за електронната изчислителна машина FACOM 230-45S. Приложени са блок-схеми на етапите на реализацията и листинги на използваните изходни масиви.

Реализацията е извършена в Центъра по кибернетика на Министерството на снабдяването и държавните резерви.

Авторът на дисертацията дължи да изкаже своята благодарност на научния си ръководител ст.н.с. к.ф.м.н. П. Върнев от Единния център за наука и подготовка на кадри по математика и механика.

1. Въведение в обработката на списъчна информация

1.1. СПИСЪЧНИ СТРУКТУРИ

Електронните изчислителни машини намират все по-широко приложение при решаването на задачи от най-различно естество. При голяма част от задачите се налага обработване на нечислова информация, което изисква структура на данните и операции, различни от тези за обработка на числова информация.

Основата за построяване на структурата на данните е концепцията за явно или неявно указване на връзките между елементите на структурата. Неявното указване на връзките се използува широко в много конвенционални езици за програмиране. Съществуват, обаче, приложения на изчислителните машини, при които програмата оперира главно със структурата на данните, а не със стойностите на елементите. Неявното указване на връзките в тези случаи дава незадоволителни резултати поради следните причини:

- не е възможно лесно и ясно неявно указване на връзките между елементите на сложни динамични структури,
- промените в структурите водят до интензивно физическо преместване на елементите им в оперативната памет на изчислителната машина, което значително намалява ефективността на обработката.

Обобщено казано, при програми, които не променят структурата на данните, а боравят със стойностите на елементите, е желателно неявно указване на

връзките в структурите /обикновено при обработка на числова информация/, докато при програми, които променят структурата на данните, е желателно явно указване на връзките в структурите /обикновено при обработка на нечислова информация/.

Векторът е пример за неявно указана структура. Подразбира се, че елементът с индекс i е разположен в оперативната памет на изчислителната машина непосредствено след елемента с индекс $i-1$ и непосредствено преди елемента с индекс $i+1$.

Най-простата явно свързана структура е еднопосочният списък. Всеки елемент на този списък освен информацията съдържа и указател /връзка/ към последователя си. Това означава, че е възможно само еднопосочно сканиране на списъка.

Двупосочният списък е явно свързана структура, в която всеки елемент съдържа освен информацията два указателя. Единият указател сочи последователя, а другият - предшественика на елемента. Следователно, възможно е двупосочно сканиране на списъка.

Плекс се нарича списък, всеки елемент на който освен информацията съдържа произволен брой указатели към други елементи.

Явно указаните структури често са структури, съставени от структури, а понякога съдържат самите себе си като елементи.

1.2. ЕДНОПОСОЧНИ СПИСЪЦИ

Всеки елемент на еднопосочния списък може да се представи като кутийка с две части /фиг.1.1.а/. Едната част съдържа информацията, а другата част - указателя /връзката/. Твърде удобно е, когато двете части на елемента са напълно симетрични, а размерите на информационния атом и на указателя съвпа-

дат. Имената CAR и CDR нямат мнемонично значение, а просто указват двете половини на елемента.

Елементът на еднопосочния списък може да бъде атом или структура. Списък, който съдържа само атоми, се нарича линеен /фиг.1.1.б/. Списъкът се нарича разклонен, ако съдържа поне един елемент, който е структура /фиг.1.1.в/. Тези елементи на главния списък, които представляват структури, се наричат подсписъци.

Фиг.1.1.б и фиг.1.1.в представляват графично представяне на еднопосочни списъци и дават "физическа" интерпретация на проблема.

Символичното представяне на еднопосочните списъци /S-изразите/ е "математическа" интерпретация и се дефинира по следния начин:

- Атомът е S-израз.
- Ако e_1 и e_2 са S-изрази, тогава $(e_1.e_2)$ е S-израз.

Двойката $(e_1.e_2)$ съответствува на списъчен елемент, съдържащ e_1 в CAR полето и e_2 в CDR полето.

При графичното представяне е лесно да се покаже края на списъка - CDR полето на последния елемент се оставя празно. Но това не може да се приложи към S-изразите, защото $(e_1.\ )$ не представлява правилен S-израз, поради което се въвежда специален атом NIL. Формули /1.1/ и /1.2/ представляват S-изрази на списъците от фиг.1.1.б и фиг.1.1.в.

(С.(П.(И.(С.(Ъ.(К.NIL)))))) /1.1/

(+.((↑.((+.(X.(Y.NIL))).(2.NIL))).
((*.(B.((+.(X.(Y.NIL))).NIL))).NIL))) /1.2/

Формули /1.1/ и /1.2/ ясно показват бързото нарастване на броя на скобите в S-изразите и съответното намаляване на четливостта. За да се избегне този недостатък на S-изразите, преминава се към така наречената запетайкова форма на представяне на еднопосочните списъци. Даден списък се записва в за-

петайкова форма чрез записване на съдържанията на CAR полетата на елементите на списъка, разделени със запетайки и заградени в скоби. Формули /1.3/ и /1.4/ представляват запетайкова форма на списъците от фиг.1.1.б и фиг.1.1.в.

(С,П,И,С,Ъ,К) /1.3/

(+,(↑,(+,X,Y),2),(*,B,(+,X,Y))) /1.4/

Броят на двойките скоби в запетайковата форма е равен на броя на атомите NIL в S-изразите.

На фиг.1.2 са показани трите начина на представяне на прости списъци.

При запетайковата форма на представяне особено внимание трябва да се обърне на понятието празен списък. Празният списък е списък, който няма елементи. Той се получава чрез прилагане на селектора CDR /раздел 1.3/ към списък, който има само един елемент. В запетайкова форма празният списък се представя чрез ( ).

Подсписъкът $L_1 = (+,X,Y)$ от фиг.1.1.в представлява общ подсписък на подсписъците $L_2 = (↑,(+,X,Y),2)$ и $L_3 = (*,B,(+,X,Y))$. Трябва да се отбележи, че при S-изразите и запетайковата форма общите подсписъци се записват многократно, всеки път когато се срещнат. Това е сериозна слабост на тези начини на представяне, която изключва полезната конструкция на еднопосочен списък, позната като цикличен списък. Цикличният списък е списък, който съдържа себе си като подсписък.

CAR и CDR полетата на елементите са напълно симетрични и само по предварителна условеност се приема, че CDR полето се използува за указател. S-изразите запазват тази симетричност. Списъчен елемент, който съдържа атом X в CAR полето и атом Y в CDR полето, се представя чрез (X.Y). Запетайковата форма на представяне е несиметрична, защото се базира на условеността, че CDR полето на елемента може да съдържа само указател, но не и атом. Поради това (X.Y) не може да се представи в запетайкова форма.

Следователно, най-мощно е графичното представяне на списъците, следват S-изразите и най-слаба е запетайковата форма. Понякога се използува смесено представяне от S-изрази и запетайкова форма.

1.3. ОПЕРАЦИИ ВЪРХУ ЕДНОПОСОЧНИ СПИСЪЦИ

Адресирането на списъците се извършва чрез базови регистри. Базовият регистър е нищо по-вече от указател към структурата и позволява на програмата да адресира необходимите елементи. При графичното представяне на списъците базовият регистър се представя чрез стрелка. На фиг.1.4 са показани четири базови регистри, които адресират първите елементи на списъка и съответните подсписъци:

b_1: (+,(↑,(+,X,Y),2),(*,B,(+,X,Y)))

b_2: ((↑,(+,X,Y),2),(*,B,(+,X,Y)))

b_3: ((*,B,(+,X,Y)))

b_4: (+,X,Y)

Най-основната операция върху еднопосочните списъци е придвижването на указателя от базовия регистър, което позволява на програмата да адресира списъчните елементи. Това може да се направи чрез занасяне на съдържанието на CAR или CDR полетата на текущия адресиран елемент в базовия регистър и се изразява чрез:

b := CDR b или

b := CAR b .

Трябва да съществува и механизъм за създаване на нови елементи или промяна на старите. На лист хартия е възможно създаване на нови елементи, но в паметта на изчислителната машина това не е възможно. Създаването на нови елементи се трансформира в прехвърляне на елементи от един списък в друг,

поради което е необходимо да съществува един специален списък на свободните елементи. Елементите на този списък ще бъдат вземани или връщани в процеса на изпълнение на програмата. Фиг.1.4 показва действията за преместване на един елемент от списъка на свободните елементи b_c в списък b_1. Това се изразява чрез следната последователност от присвоявания:

$b_2 := CDR\ b_c$

$CDR\ b_c := b_1$

$b_1 := b_c$

$b_c := b_2$

Спомагателният регистър b_2 се използува за да се предотврати загубването на връзката към останалите елементи на списъка на свободните елементи след вземането на първия елемент.

Математическият подход към обработката на списъци игнорира съществуването на списъчни елементи. Списъците се представят чрез S-изрази или в запетайкова форма, а операциите върху списъците се дефинират като функции на изразите и тези функции създават нови изрази без да е необходимо да се променят съществуващите. Малък брой функции се разглеждат като първични, а всички други функции се изграждат чрез използуване на първичните или преди дефинирани непървични функции. Съществуват три типа първични функции:

- Селектори - Селекторите са функции, чиито стойности са данните, извлечени от аргументите.
- Конструктори - Конструкторите са функции, чиито стойности са изразите, конструирани от аргументите.
- Твърдения - Твърденията са функции, чиито стойности са true или false в зависимост от свойствата или връзките между тяхните аргументи.

При еднопосочните списъци единствената информация, която може да се

извлече от елементите им, е съдържанието на CAR или CDR полетата. Следователно, само две първични селекторни функции са необходими. Те са:

CAR(e)

CDR(e) .

Един елемент може да бъде изграден от два израза, единият от които става съдържание на CAR полето, а другият - на CDR полето. Следователно, само един първичен конструктор е необходим. Той е:

CONS(e_1,e_2) .

Наборът от твърдения не е така очевиден. Установено е, че само две твърдения са необходими като първични функции. Те са:

ATOM(e)

EQ(e_1,e_2) .

ATOM(e) има стойност true - когато аргументът е атом и
стойност false - при друг случай.

EQ(e_1,e_2) има стойност true - когато двата аргумента са идентични атоми,
стойност false - когато двата аргумента не са идентични атоми и
неопределена стойност - при друг случай.

За да се разбере действието на функционалния подход при обработването на списъци, нека бъде разгледан списъка (↑,(+,X,Y),2). За избягване на многократното преписване на израза, нека той се представя с "e". Това се прави само за удобство при обясненията. Стойността на функцията CAR(e) е атомът ↑, а стойността на функцията CDR(e) е списъкът ((+,X,Y),2). Стойността на израза CDR(CAR(e)), обаче, е недефинирана. Това се дължи на ограничението, че аргументите на селекторните функции трябва да бъдат списъци.

Прилагането на селекторна функция към израз, който представлява списък, е точен еквивалент на присвояване на съдържанието на едно от полетата, принадлежащи на някой от елементите на списъка, на базовия регистър на резултантната структура. Да предположим, че b_1 е базов регистър на списъка, пред-

ставляващ израза e и да предположим, че b_2 ще бъде базов регистър на списъка, представен чрез функцията CAR(CDR(e)). Желаният резултат ще се получи от изпълнението на следните оператори за присвояване:

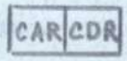
$$b_2 := \text{CDR } b_1$$
$$b_2 := \text{CAR } b_2 .$$

Конструкторът CONS(e_1,e_2) конструира израз с CAR поле представляващо e_1 и CDR поле представляващо e_2:

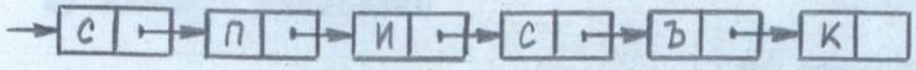
$$\text{CONS}(e_1,e_2) \rightarrow (e_1.e_2)$$

Функцията CONS(+,CONS(X,(Y))) конструира списъка (+,X,Y) показан на фиг.1.5. При всяко прилагане на функцията CONS се създава нов елемент. Стойността на CONS е указател към създадения елемент, който има стойност първия аргумент на CONS в CAR полето си и стойност втория аргумент на CONS в CDR полето си. На пръв поглед изглежда, че няма ограничения за вида на аргументите на конструктора CONS, т.е. да бъдат атоми или списъци. Но това важи когато се използува символичното представяне - S-изразите. Ако е желателно да се използуват предимствата на запетайковата форма, налага се въвеждане на ограничението, че e_2 трябва да бъде списък, т.е. CONS(e_1,e_2) е недефинирана, ако e_2 е атом. Като пример може да се посочи функцията CONS(X,Y), където X и Y са атоми. В този случай се получава списъчната структура, показана на фиг.1.6.а. Този списък не може да бъде изразен в запетайкова форма, което е недостатък на тази форма на записване на списъчни структури.

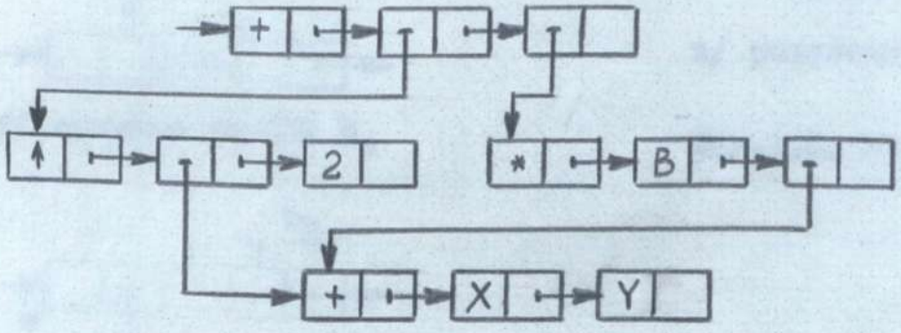
LISP позволява смесено записване, което съчетава удобствата на запетайковата форма без налагане на ограничения върху аргументите на функцията CONS. При този начин на записване последният елемент на всеки списък и подсписък може да се запише като точкувана двойка, но ако CDR полето на този последен елемент е NIL, записването може да бъде както при запетайковата форма. На фиг.1.6 са показани примери на смесено записване.



а/ елемент на еднопосочен списък

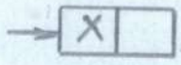


б/ еднопосочен линеен списък



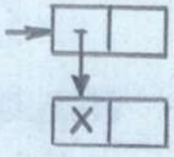
в/ еднопосочен разклонен списък

Фиг.1.1. Еднопосочни списъци



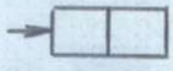
(X.NIL)

(X)



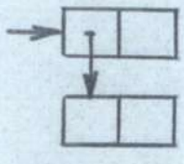
((X.NIL).NIL)

((X))



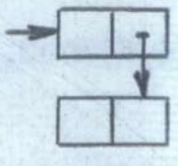
(NIL.NIL)

(( ))



((NIL.NIL).NIL)

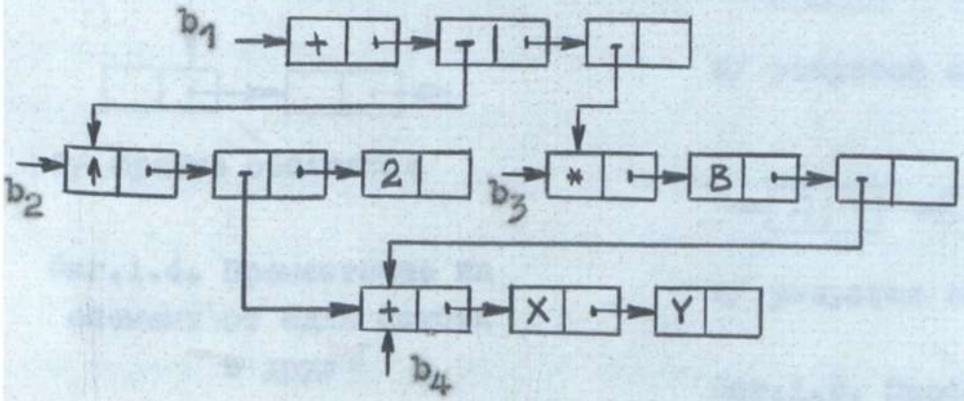
((( )))



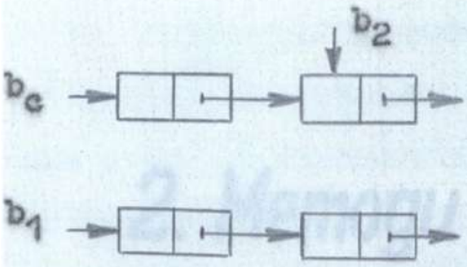
(NIL.(NIL.NIL))

(( ).( ))

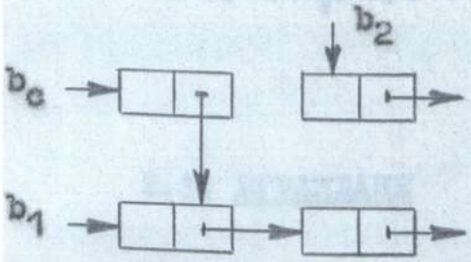
Фиг.1.2. Трите вида представяния на някои прости списъци



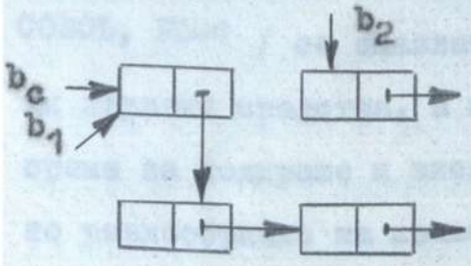
Фиг.1.3. Базови регистри на еднопосочни списъци



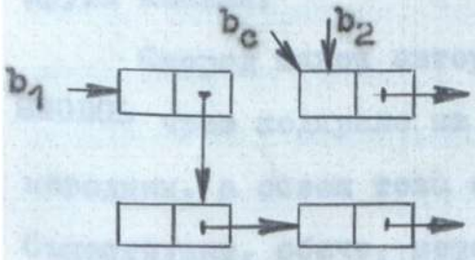
а/ начално състояние



б/ промяна на CDR b_c

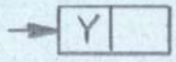


в/ промяна на b_1

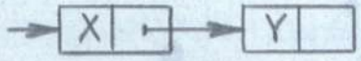


г/ крайно състояние

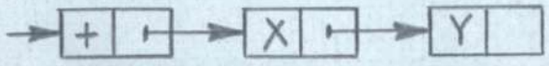
Фиг.1.4. Преместване на елемент от един списък в друг



а/ списъкът (Y)

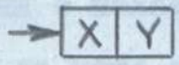


б/ резултат от CONS(X,(Y))



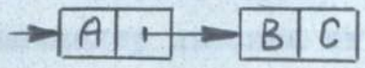
в/ резултат от CONS(+,CONS(X,(Y)))

Фиг.1.5. Изпълнение на CONS



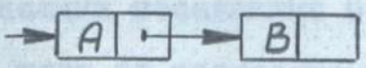
(X.Y)

а/ CONS(X,Y)



(A,B.C)

б/ резултат от CONS(A,CONS(B,C))



(A,B)
(A,B.NIL)

в/ резултат от CONS(A,CONS(B,NIL))

Фиг.1.6. Смесено представяне на списъци

2. Методи за реализация на процесори за обработка на нечислова информация

2.1. ВГРАЖДАНЕ

За кодирането на транслатори и други процесори за обработка на нечислова информация най-разпространените езици за програмиране / FORTRAN, ALGOL-60, COBOL, PL-1 / се оказват твърде неудобни, тъй като те не притежават подходящи изразни средства, а използуването на език от тип асамблер изисква много време за кодиране и високо квалифицирани програмисти. Освен това при голямото разнообразие на изчислителни машини е желателно да съществува възможност реализиран за дадена машина процесор да може лесно да бъде реализиран и за други машини.

Според някои автори реализацията на процесори от типа на LISP или SNOBOL чрез кодиране на език от тип асамблер изисква от четири до шест човекогодини, а освен това една такава реализация е напълно машинно-зависима. Съществуват, обаче, методи за намаляване на времето и усилията за реализация на подобни процесори и обмен на реализации независимо от оборудването. Тези методи се базират на фундаменталната идея за разделяне на процесорите на две съставни части:

- основни операции и типове данни, характерни за съответния процесор
- алгоритъм, който използува основните операции и типове данни за да обезпечи желаното действие на процесора.

Например, основен тип данни в SNOBOL са символните низове, а основни операции - съчленяване, подбиране по образец и заместване. В LISP основен тип данни са еднопосочните символни списъци, а обработката на тези списъци се осъществява от петте основни функции CAR, CDR, CONS, ATOM и EQ. Примери за операции за реализиране на алгоритма са преходите, условията и някои други конвенционални процедурни операции, които в голяма степен са независими от структурата на обработваните данни и следователно инвариантни в различните процесори.

Горната идея води непосредствено до концепцията за вграждане, т.е. използуване на съществуващ процедурно-ориентиран език /например FORTRAN или PL-1/ за описание на процедурните операции на алгоритма и реализация на основните операции за обработка на данните като функции и подпрограми, вградени в алгоритма. Обикновено тези функции и подпрограми се кодират на асамблер, защото езикът на вграждане не може да обезпечи ефективност при изпълнението на специфичните основни операции. Всяка нова реализация на процесора за друга изчислителна машина, притежаваща транслатор от езика на вграждане, се свежда единствено до ново кодиране само на тези функции и подпрограми. При метода на вграждане, чрез използуването на език от високо ниво като език на вграждане, се намалява работата по кодирането и следователно се намаляват усилията за реализирането. Като недостатък на метода на вграждане може да се посочи по-малката ефективност на създадените процесори по отношение на използувана оперативна памет и време за изпълнение.

Ако функциите и подпрограмите за реализация на основните операции се кодират на езика на вграждане, вграждането е пълно. В този случай то е еквивалентно на обикновено програмиране на език от високо ниво. Пълно вграждане се използува при реализиране на процесори за спомагателни цели и малка употреба, при което не е необходимо те да бъдат оптимизирани по отношение на време за изпълнение и използувана оперативна памет и освен това е желателно

лесното им реализиране на различни изчислителни машини.

Процесът на вграждане може да бъде и многостепенен. Многостепенното вграждане може да се сравни с въвеждането на подцели, достигането на които води до достигане на главната цел и опростява и улеснява извършването на реализацията. Този метод на въвеждане на подцели е широко разпространен при решаването на сложни проблеми по евристичен път.

Макар вграждането да изглежда прост и праволинеен процес, съществуват подробности, които трябва да бъдат взети под внимание за качественото му прилагане.

2.2. МОДЕЛИРАНЕ

Обикновено за език на вграждане се избира удобен за конкретния случай, широко разпространен и реализиран на много машини език за програмиране. В много случаи, обаче, такъв подходящ реализиран език не съществува или поне този език не е твърде разпространен. Тогава най-добре е за език на вграждане да се избере най-подходящия позволяващ пълно вграждане специализиран език за програмиране или дори да се създаде такъв език. Множеството операции на този специализиран език представляват от себе си абстрактен модел на изчислителна машина. Необходимо е операциите на абстрактната машина да бъдат преобразувани в операции на съответната реална изчислителна машина, за която е предназначена реализацията. Това означава всеки оператор от езика на абстрактната машина да бъде заменен с оператор или група оператори от езика на реалната машина, като под език на реалната машина се разбира не само асамблер, но и всеки реализиран на тази машина език от по-високо ниво. Преобразуването от език на абстрактна машина в език на реална машина може да бъде извършено от процесор от компилиращ тип за обработка на символни низове чрез подбиране по образец и заместване, какъвто процесор представлява например реализацията

на SNOBOL. Ако такъв процесор не съществува, той трябва предварително да бъде създаден.

В какво се състои предимството на моделирането, ако за съответна реализация трябва да бъде реализиран допълнително още един, при това специален процесор, който в случая може да се нарече метапроцесор. За изясняване на този въпрос трябва да бъде разгледана същността на процесорите за обработка на символни низове чрез подбиране по образец и заместване и привързването на дадена реализация към изчислителната машина за която е предназначена.

Споменатият тип метапроцесори възприемат като входни данни подлежащата на преобразуване изходна програма заедно с набор от дефиниции, определящи заместването на всеки оператор от изходния език с един или група оператори от обектния език, а като резултат от работата на метапроцесора се извежда преобразуваната на обектен език програма. Даден метапроцесор се използува за преобразуване не само на един конкретен език в друг също конкретен език, а за преобразуване на всяка двойка от едно множество изходни и обектни езици. Коя ще бъде двойката изходен и обектен езици за дадено изпълнение се определя от дефинициите. Сложността на изходните и обектните езици, както и сложността на дефинирането, зависят от възможностите на метапроцесора, но трябва да се отбележи, че един сравнително несложен метапроцесор може да послужи за преобразуване на значително развити езици за програмиране. Веднаж реализиран, метапроцесорът има самостоятелно значение и област на приложение.

Моделирането при реализацията на процесор е форма на пълно вграждане, като основните операции представляват операции на абстрактната машина, а език на вграждане е езикът на абстрактната машина. Една и съща програма за абстрактната машина може да бъде транслирана в програми за различни реални изчислителни машини просто чрез подходяща промяна в дефинирането на операциите на абстрактната машина. Част от дефинирането може да бъде идентично за всички или някои от реалните изчислителни машини.

За успешно приложение на моделирането е необходимо да бъде изяснен и подходът към операциите за вход-изход. Обикновено тези операции са твърде сложни и зависят от конфигурацията на изчислителната машина. Опитът на редица реализатори показва, че най-добре е абстрактната машина да бъде изолирана от конкретните входно-изходни устройства чрез един модул, който да обработва входно-изходните операции. Целта на този модул е да обезпечи канали за връзка между програмата и изчислителната среда. Всеки канал представлява съвокупност от логическо устройство на разположение на програмата и конкретно физическо устройство от конфигурацията на машината. Когато програмата иска изпълнение на операция за дадено логическо устройство, модулът изпълнява тази операция на съответното физическо устройство.

Поради голямото разнообразие на входно-изходни устройства съществува голямо разнообразие на входно-изходни операции, но всяка една от тях принадлежи към една от следните категории:

- четене - прехвърляне на данни от устройството към програмата,
- запис - прехвърляне на данни от програмата към устройството,
- управление - операции за управление на работата на устройството без да се извършва прехвърляне на данни.

Горното подразделяне е твърде удобно, защото отделните категории са независими една от друга и обезпечават основа за дефиниране на логически устройства, независими от конфигурацията на изчислителната машина. При това положение входно-изходният модул се състои от подпрограми за споменатите три категории операции. Чрез тези подпрограми се отчитат особеностите на входно-изходните устройства и се добавя необходимата управляваща информация за фактическото изпълнение на логическите операции от съответните физически устройства. По този начин става възможна промяна на използуваните входно-изходни устройства без никакви промени в реализирания процесор. В случая се правят съответни промени само в подпрограмите на входно-изходния модул. Възмож-

но е също включване на нови устройства или изключване на други такива, но най-важното предимство на използуването на входно-изходен модул е улесняването на обмена на процесори между различни изчислителни машини и организации.

2.3. ПРИВЪРЗВАНЕ

Беше изяснено, че за прилагане на моделирането е необходимо наличието на подходящ метапроцесор за обработка на символни низове чрез подбиране по образец и заместване. Трябва ли този метапроцесор да бъде реализиран на машината, за която се извършва реализацията? Това не е необходимо. Ако реализацията на съответен процесор е предназначена за машината М, а подходящ метапроцесор е реализиран вече на машината Н, необходимо е текстът на процесора на езика на абстрактната машина А, заедно с наборът от дефиниции, определящи заместването на операторите от езика на абстрактната машина А с оператори от езика на реалната машина М, да бъдат подадени като входни данни на метапроцесора на машината Н. В резултат от работата на изчислителната машина Н ще се получи програма за изчислителната машина М. Този подход към реализацията се нарича полупривързване към машината за която е предназначена реализацията, защото се използува и друга изчислителна машина, притежаваща подходящ метапроцесор.

При полупривързването съществуват и някои трудности. Освен необходимостта от изчислителна машина с реализиран и в експлоатация метапроцесор, малко вероятно е безпогрешното кодиране на дефинициите от първи път, което ще наложи няколко пъти да се използува изчислителната машина Н с реализирания метапроцесор. Освен трудният достъп до машината Н, допълнителни затруднения може да причини и недостатъчното познаване на машината Н, операционната й система и метапроцесора. По-важна, обаче, е кодовата съвместимост на входа и изхода на машините М и Н. Различието във външното кодово представяне

на символите трябва да се отстрани чрез съответно прекодиране, което се явява като допълнително бреме. Поради наличието на такива трудности може да се премине към пълно привързване.

Пълното привързване предполага използуване на само една изчислителна машина - тази, за която е предназначена реализацията. Ако тя не притежава подходящ метапроцесор, такъв трябва да бъде създаден, а ползата от него за програмното осигуряване на машината ще бъде голяма и разностранна. Освен това, един такъв по-сложен метапроцесор може да бъде реализиран като се използува отново вграждане, моделиране и съответно привързване.

Трябва да се подчертае, че няма пречки за преминаване от пълно привързване към полупривързване, ако обстоятелствата позволяват това. Възможно е двата метода да се използуват съвместно при една и съща разработка, включваща няколко етапа на вграждане и моделиране.

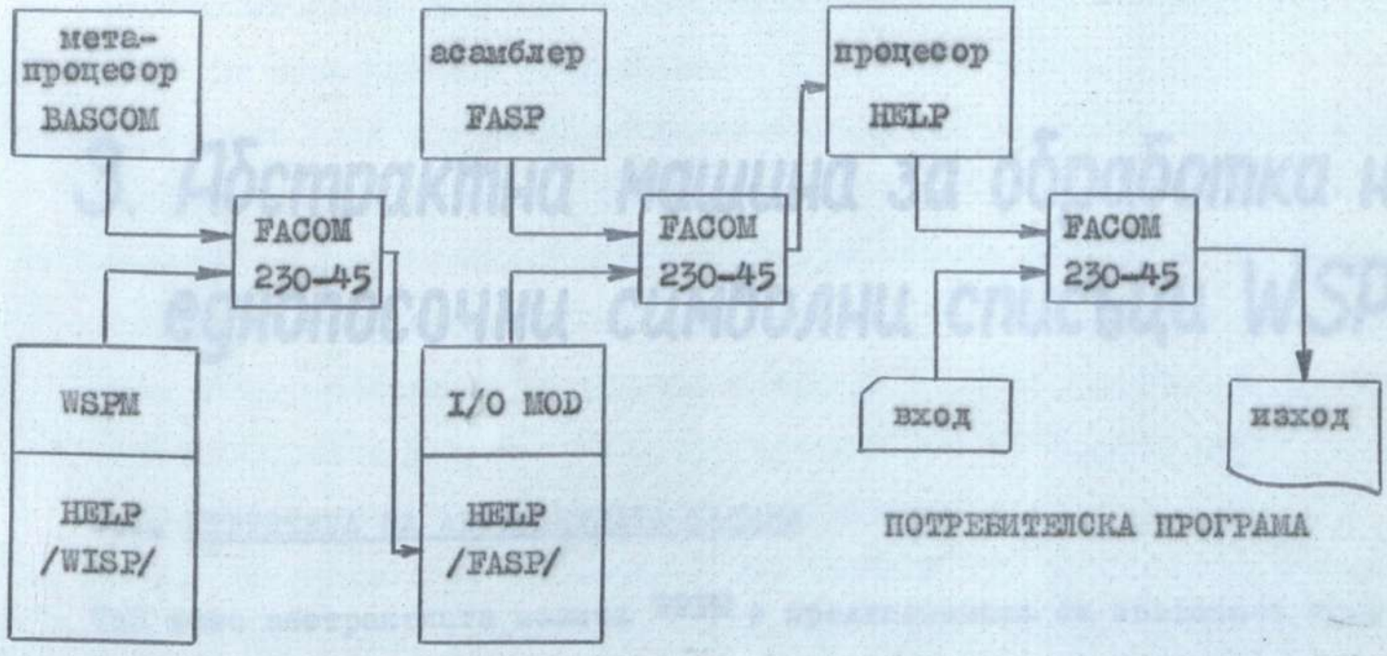
2.4. БЛОКОВА СХЕМА НА РЕАЛИЗАЦИЯТА

Функционалният процесор за обработка на символна списъчна информация HELP е реализиран по метода на вграждане и моделиране и при пълно привързване към изчислителната машина FACOM 230-45S, с която е оборудван Центъра по кибернетика на Министерството на снабдяването и държавните резерви. Този процесор от компилиращо-интерпретиращ тип служи за въвеждане, транслиране и управление на изпълнението на потребителска програма, написана на езика HELP. При реализирането на процесора HELP е използувано пълно вграждане в езика WISP, създаден специално за удобно описване на процесори за функционална обработка на еднопосочни символни списъци. Процесорът HELP е кодиран на езика WISP, а за да се получи като асамблерова програма, всеки оператор на езика WISP е дефиниран чрез команди на асамблера на машината FACOM 230-45S, т.е. абстрактната машина WSPM е моделирана на асамблера на реалната изчислителна

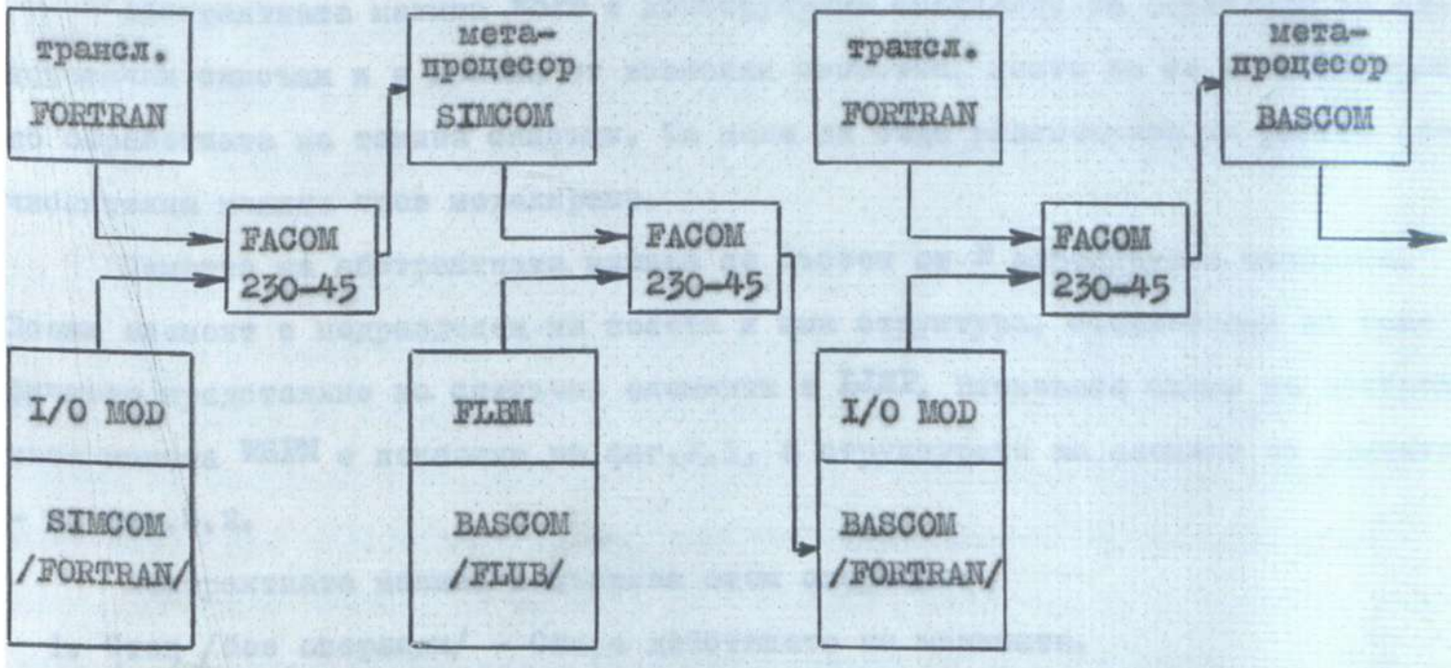
машина. Асамблеровият текст на процесора HELP се асамблира, за да се получи машинна програма. Блок-схемата на реализацията на HELP е показана на фиг.2.1. От тази блок-схема се вижда, че е необходим метапроцесорът BASCOM, който трябва да бъде реализиран предварително.

Блок-схемата на реализацията на метапроцесора BASCOM е показана на фиг.2.2. Използувано е пълно вграждане на BASCOM в езика FLUB, удобен за описване на ограничени по възможности процесори за обработка на символни низове. Преобразуването на програма на езика FLUB в програма на FORTRAN се извършва от метапроцесора SIMCOM, който е аналогичен на BASCOM, но значително по-прост. Метапроцесорът BASCOM е кодиран на езика FLUB, а за да се получи като фортранова програма, всеки оператор на езика FLUB е дефиниран чрез фортранови оператори, което означава, че абстрактната машина FLBM е моделирана на FORTRAN. Фортрановият текст на метапроцесора BASCOM се транслира, за да се получи машинна програма. Метапроцесорът SIMCOM е кодиран на FORTRAN. При реализацията на BASCOM като език на вграждане се използува също FORTRAN. Това позволява метапроцесорът BASCOM да бъде лесно реализиран на всяка изчислителна машина, притежаваща транслатор от FORTRAN.

В блоковите схеми от фиг.2.1 и фиг.2.2 активните програми са показани в горната част, а данните - в долната част на чертежите. Модулите, обработващи входно-изходните операции, са показани на съответните им места в блоковите схеми. Те се прибавят в последния етап, когато съответният процесор се получава като машинна програма.



Фиг.2.1. Блок-схема на реализацията на функционалния процесор за обработка на символна списъчна информация HELP



Фиг.2.2. Блок-схема на реализацията на метапроцесора за обработка на символни низове BASCOM

3. Абстрактна машина за обработка на еднопосочни символни списъци WSPM

3.1. СТРУКТУРА НА АБСТРАКТНАТА МАШИНА

Тъй като абстрактната машина WSPM е предназначена за създаване чрез вграждане на процесор за обработка на символна списъчна информация, тя трябва да притежава типове данни и основни операции за обработка на символни списъци. В такъв случай вграждането на процесора в езика на тази абстрактна машина ще бъде удобно и ефективно.

Абстрактната машина WSPM е конструирана специално за обработка на еднопосочни списъци и е лишена от всякакви свойства, които не се отнасят пряко до обработката на такива списъци. Тя може да бъде реализирана на реална изчислителна машина чрез моделиране.

Паметта на абстрактната машина се състои от N адресируеми елемента. Всеки елемент е подразделен на полета и има структура, съответваща на графичното представяне на списъчни елементи в LISP. Блоковата схема на абстрактната машина WSPM е показана на фиг.3.1, а структурата на елемент от паметта - на фиг.3.2.

Абстрактната машина изпълнява осем операции:

1. Стоп /без операнди/ - Спира действието на машината.
2. Безусловен преход /един операнд/ - Извършва безусловен преход на адрес, указан от операнда.

3. Увеличаване /един операнд/ - Увеличава стойността на операнда с размерността на един елемент от паметта.
4. Присвояване /два операнда/ - Занася стойността на втория операнд в полето, адресирано от първия операнд.
5. Вход-изход /два операнда/ - Стойността на първия операнд е входно-изходна команда. Стойността на втория операнд при прехвърляне на редове между входно-изходните устройства и буфера е адрес на елемент, указващ входно-изходното устройство, а при прехвърляне на символи между буфера и паметта - адрес на информацията в паметта.
6. Преход при равно /три операнда/ - Ако стойностите на втория и третия операнди са равни, извършва се преход на адрес, указан от първия операнд. В противен случай се изпълнява следващата операция.
7. Преход при неравно /три операнда/ - Ако стойностите на втория и третия операнди не са равни, извършва се преход на адрес, указан от първия операнд. В противен случай се изпълнява следващата операция.
8. Преход при по-малко /три операнда/ - Ако стойността на втория операнд е по-малка от стойността на третия операнд, извършва се преход на адрес, указан от първия операнд. В противен случай се изпълнява следващата операция.

Всяка операция на абстрактната машина може да има до три операнда. Операндите са винаги цели числа, но могат да бъдат модифицирани по три начина. Те могат да се използуват или като стойности или като адреси. Стойността на операнда е съдържанието на полето, обозначено чрез модифицираното цяло число или самото цяло число, ако модифицирането е непосредствено. Непосредственият операнд не може да се използува като адрес. Директните и индиректните операнди адресират съответни полета от елементите на паметта.

За адресиране на списъците се използуват указатели, които се записват

в CDR полетата на базови регистри за адресация. За реализацията на всеки базов регистър се използува по един елемент от паметта на абстрактната машина, а в AF полетата на тези елементи се записва единица. За обозначение на базовите регистри се използуват единични символи, като на всеки символ съответствува един базов регистър. Възможно е използуване на допълнителни базови регистри, които се именуват чрез символни низове.

3.2. ЕЗИК НА АБСТРАКТНАТА МАШИНА

Програмата за абстрактната машина се състои от последователност от редове. Всеки ред съдържа един или по-вече оператора. Всеки оператор завършва с точка или със запетая. Запетаята само завършва оператора, а точката освен това предизвиква игнориране на останалата част от реда, която може да бъде използувана за коментар и/или номериране на редовете. Всеки ред може да има етикет, който започва от първа позиция на реда. Ако първата позиция е шпация, редът е без етикет. Етикетът е произволен низ от символи, започващ с буква. Точка или запетая трябва да следват етикета и по този начин той се разглежда като оператор.

В табл.3.1 са представени операторите на WISP за осемте операции на абстрактната машина WSPM.

Операторът може да бъде разчленен на части: TO, INCR, op1, op2, =, NE и т.н. Тези части трябва да бъдат отделени една от друга с поне една шпация. Шпациите, които предшествуват първата част, както и шпациите, които предшествуват точка или запетая, завършващи оператора, се пренебрегват.

Операндите могат да бъдат както единични символи, така и символни низове. Всеки операнд може да бъде модефициран по един от посочените в табл. 3.2 начини.

ОПЕРАЦИЯ	ОПЕРАТОР
Стоп	STOP.
Безусловен преход	TO op1.
Увеличаване	INCR op1.
Присвояване	op1 = op2.
Преход при равно	TO op1 IF op2 = op3.
Преход при неравно	TO op1 IF op2 NE op3.
Преход при по-малко	TO op1 IF op2 LT op3.
Вход-изход	IO op1 ON op2.

Табл.3.1. Операции и оператори на абстрактната машина за обработка на еднопосочни символни списъци

НАЧИН НА МОДИФИЦИРАНЕ	ВИД НА ОПЕРАНДА		
	ЕДЕН. СИМВОЛ	СИМВОЛЕН НИЗ	
Непосредствено	'A	SAM	100
Директно адресиране	A	(SAM)	(100)
Индиректно адресиране на AF	AF A	AF SAM	AF 100
Индиректно адресиране на BPF	BPF A	BPF SAM	BPF 100
Индиректно адресиране на CAR	CAR A	CAR SAM	CAR 100
Индиректно адресиране на CDR	CDR A	CDR SAM	CDR 100

Табл.3.2. Модифициране на операндите

Апострофът се използува за обозначаване на непосредствен операнд, представен от еденичен символ. По този начин точката и запетаята могат също да бъдат използувани като непосредствени операнди без да бъдат възприемани за край на операторите. Това става чрез комбинациите "'." или "',". Горното

се отнася и за шпацията и за самия апостроф. Тези четири символа могат да се използуват като операнди само с непосредствено модифициране.

При непосредствено модифициране операндите, представени с еденични символи, се дефинират като адреси от паметта, равни на вътрешното представяне на съответните символи, т.е. тези операнди адресират базовите регистри на абстрактната машина.

При операнди, които са символни низове, първият символ в низа определя дали операндът е абсолютен или символичен. Ако първият символ е цифра, операндът е абсолютен, а низът трябва да бъде цяло число. Ако първият символ не е цифра, операндът е символичен и трябва да бъде дефиниран някъде в програмата. Символичният операнд е дефиниран, ако е използуван като етикет на някой ред от програмата, а съответна стойност ще му бъде присвоена от транслатора. Има едно изключение, което се отнася за низа NIL, стойността на който е вътрешното представяне на символа CR /физически край на ред от програмата или връщане на каретката на пишуща машина като вход/.

Освен осемте основни оператора, за улесняване на програмирането могат да бъдат дефинирани псевдо оператори, допълнителни оператори и разширени оператори. Допълнителните оператори представляват удобна форма на някои основни често използувани оператори с фиксирани операнди. Разширените оператори служат за изразяване на често използувани комбинации от операции, които могат и е удобно да се представят чрез един оператор на WISP. Псевдо операторите се използуват за резервиране на памет, завършване на програмата и други подобни действия.

Ще бъдат представени някои прости примери за програмиране на WISP, които включват всички възможни операции и операнди, без да бъдат обхванати всички възможни комбинации поради големия им брой.

Пример: Да се трансформира списъка B = (A,C,D) от фиг.3.3.а в списък B = (A,C,D,X), представен на фиг.3.3.в.

Първи вариант:

```
    E = CDR B.
    E = CDR E, E = CDR E.
    CAR E = 'X.
```

Втори вариант:

```
    E = B.
    LOOP, TO END IF CAR E = NIL.
    E = CDR E, TO LOOP.
    END, CAR E = 'X.
```

Трети вариант:

```
    E = B, TO END IF CAR E = NIL.
    LOOP, E = CDR E, TO LOOP IF CAR E NE NIL.
    END, CAR E = 'X.
```

Четвърти вариант:

```
    E = 'B.
    LOOP, E = CDR E, TO LOOP IF CAR E NE NIL.
    CAR E = 'X.
```

На фиг.3.4 е представена графически разликата между присвояванията E = B и E = 'B.

3.3. ВХОДНО-ИЗХОДНИ ОПЕРАЦИИ

При входно-изходните операции се използува буферен регистър, чрез който се организира предаването на информация по два начина:

1/ Предаване на еденичен символ между буфера и паметта на абстрактната машина.

2/ Предаване на един ред от символи между буфера и съответно входно-изходно устройство.

Операционните кодове на входно-изходните команди са представени в табл.3.3. Тези кодове се явяват като първи операнди в операторите за вход-изход. За предаване на еденични символи вторият операнд указва полето, вземащо участие в предаването на данни. За предаване на ред вторият операнд указва входно-изходното устройство, което взема участие в предаването на данни.

КОД	ОПЕРАЦИЯ
0	Няма операция
1	Извличане на текущия символ от буфера, превръщане във вътрешен код и запомняне в полето, указано от втория операнд.
2	Ако стойността на втория операнд е вътрешен код на символ, тя се превръща във външен код и се занася в текущата позиция на буфера. В противен случай операцията се пренебрегва.
3	Четене на следващия ред от указаното входно устройство в буфера. Вторият операнд адресира елемент, чийто CAR поле указва логическия номер на входното устройство. При EOF /край на масива/ в CDR полето на този елемент ще се занесе 1 и предаване на данни няма да има. Ако е указана неправилна за устройството операция, в CDR полето ще се занесе 2.
4	Извеждане на съдържанието на буфера към изходното устройство, указано от втория операнд. При EOM /край на носителя/ в CDR полето на адресиращия елемент се занася 1. Ако е указана неправилна за устройството операция, в CDR полето на елемента ще се занесе 2.
5	Запис на EOF /край на масив/.
6	Пренавиване.

Табл.3.3. Входно-изходни команди

При входно-изходните операции за предаване на еденични символи буферът се индексира чрез съдържанието на CAR полето на базовия регистър NIL. MAX е максималният възможен индекс за буфера и трябва да бъде равен на максималната възможна дължина на редовете в зависимост от периферните устройства, поради което MAX е зависим от конкретната реализация. Когато се изпълни входно-изходна команда за предаване на еденичен символ, индексът, записан в NIL елемента, се увеличава с еденица. При регистриране на CR в буфера, индексът се рестева на еденица. След като максималният брой символи бъдат записани в буфера, базовият регистър NIL ще съдържа нула, защото съдържанието му се увеличава по модул (MAX+1). Входно-изходните команди за предаване на еденични символи се изпълняват нормално при условие $1 \leq (NIL) \leq MAX$. Ако това условие не е изпълнено, извеждането на символ води до занасянето на стойността на втория операнд в NIL, а въвеждането на символ се игнорира. Индекс нула е необходим при редове с фиксирана дължина за обработка на неявно указания символ за край на реда CR.

Пример: Да се прехвърлят символите от 10 до 12 позиция на буфера в списък B.

```
(NIL) = 10.                    Зареждане на индекса за четене от буфера
X = B.                         X е указател към елементите на списъка
IO 01 ON CAR X, X = CDR X.
IO 01 ON CAR X, X = CDR X.
IO 01 ON CAR X.
```

Да се изведат същите символи в позиции от 1 до 3 на буфера.

```
(NIL) = 01.                    Зареждане на индекса за запис в буфера
X = B.                         X е указател към елементите на списъка
IO 02 ON CAR X, X = CDR X.
IO 02 ON CAR X, X = CDR X.
IO 02 ON CAR X.
```

Трябва да се отбележи, че за записване на цяло число, състоящо се от един символ, се прибавя една нула пред съответната цифра. Това се прави за да се различават целите числа от нула до девет от представянето на символите цифри.

При входно-изходните операции за предаване на редове вторият операнд адресира елемент, който се нарича спецификатор на операцията и чието CAR поле съдържа логически номер на използуваното входно-изходно устройство, а в CDR полето се регистрира завършека на операцията. Ако CDR полето съдържа:

0 - Успешно завършване на операцията

1 - EOF или EOM

2 - Неправилна операция за съответното устройство

2 - Специални, зависими от устройствата кодове

Обикновено се приемат едно основно входно устройство INPUT и две основни изходни устройства PRINT и PUNCH, но е възможно да бъдат включени и други устройства.

За обезпечаване на удобство при програмирането, предвидени са няколко допълнителни форми на оператори за вход-изход, представени в табл.3.4.

ОСНОВНИ ОПЕРАТОРНИ ФОРМИ	ДОПЪЛНИТЕЛНИ ОПЕРАТОРНИ ФОРМИ
IO 01 ON op2.	op2 = LINE BUFFER.
IO 02 ON op2.	LINE BUFFER = op2.
IO 03 ON op2.	READ op2.
IO 04 ON op2.	WRITE op2.
IO 05 ON op2.	ENDFILE op2.
IO 06 ON op2.	REWIND op2.

Табл.3.4. Допълнителни операторни форми

Пример: На фиг.3.5 е показана малка програма за четене от входното

устройство със спецификатор INPUT, проверка на прочетените символи и замяна на всички звездички с шпации и извеждане на изходното устройство със спецификатор PUNCH. Прието е, че входните и изходните редове са с една и съща дължина, а помощният списък I има достатъчно елементи за да побере информацията от един ред. Първоначалното зареждане на спецификаторите се извършва чрез псевдо-операцията ELEMENT af bpf car cdr.

3.4. ПОДПРОГРАМИ

Използуването на подпрограми в WISP е много опростено. Преди извършване на преход към подпрограма адресът за връщане трябва да бъде запомнен и след завършване на изпълнението на подпрограмата да бъде възстановен. За тази цел трябва да бъде резервиран един елемент от паметта на абстрактната машина.

Използуването на подпрограми е така широко застъпено, че е необходимо абстрактната машина да притежава следните допълнителни оператори:

ENTRY етикет - за вход в подпрограма
EXIT етикет - за изход от подпрограма
USE етикет - за преход към подпрограма

На фиг.3.6 е показано използуване на подпрограма само с основни оператори, а на фиг.3.7 - използуването на същата подпрограма чрез допълнителни оператори.

3.5. СПИСЪК НА СВОБОДНАТА ПАМЕТ

В началото на работата на абстрактната машина трябва да бъде въведена програмата и освен това да бъде създаден списък на свободните елементи от

паметта. Това се извършва от процедурата START, която може да се разглежда като хардуерна процедура при натискане на бутон "НАЧАЛНО ВЪВЕЖДАНЕ" на реална изчислителна машина. Тази част от паметта, която остава след въвеждане на програмата служи за разполагане на списъчните структури, подлежащи на обработка. Първоначално всички елементи на тази част от паметта трябва да бъдат организирани като списък на свободната памет. При създаването на списъка на свободната памет, който е явно свързана структура, се използува фактът, че елементите на паметта на абстрактната машина образуват вектор и чрез операцията INCR могат да бъдат адресирани последователно всички елементи на паметта между адреси BOT и TOP /фиг.3.8/.

За създаване на списъка на свободната памет се използуват два базови регистра, в единия от които /например базов регистър 8/ се занася адреса BOT, а в другия /например базов регистър 9/ - адреса TOP. На фиг.3.9 е показана процедурата, която създава явно свързване чрез зареждане на CDR полето на всеки елемент с указател към следващия елемент от вектора на паметта. Освен това се нулират флаговете на елементите и така създаденият списък на свободната памет се адресира чрез базовия регистър FREE.

3.6. ОСНОВИ НА ПРОГРАМИРАНЕТО НА WISP

3.6.1. Организация и използуване на речник

Речникът представлява набор от символни низове, наричани символи, всеки от които е свързан със стойност, която може да бъде също символен низ. При използуването на речника се задава символ и в резултат от изпълнението на програмата за преглед на речника се получава стойността на този символ.

Различават се два режима на програмата за преглед в зависимост от

действията, когато зададеният символ не е намерен в речника. В този случай единият режим връща нулева стойност, а другият режим обезпечава включване на символа в речника, присвоявайки му нулева стойност.

На програмата за преглед трябва да бъде зададен указател към речника и указател към представянето на зададения символ, а също така и режима на прегледа. Това е отразено в следните условия:

1/ Вход в програмата за преглед се извършва чрез оператора "USE LOOKUP."

2/ При входа в програмата е необходимо базовият регистър D да указва базовия регистър на речника. D ще бъде променян от програмата.

3/ При входа в програмата базовият регистър S ще указва линеен списък, представляващ зададения символ. CAR и CDR полетата на последния елемент на списъка ще съдържат NIL. S не се променя от програмата, но символът може да бъде включен в речника.

4/ При входа в програмата базовият регистър M трябва да определя режима на преглед. При M = NIL зададеният символ няма да бъде включен, а при всяка друга стойност на M този символ ще бъде включен в речника, ако не е бил намерен. M не се променя от програмата.

5/ Всеки символ от речника завършва с елемент, чийто CAR поле съдържа NIL, а CDR поле - указател към стойността на символа. При изхода от програмата за преглед базовият регистър Y ще указва завършващия елемент на намерения символ. Ако M = NIL и символът не е намерен, при изхода Y ще има стойност NIL.

Горните условия са независими от структурата на речника. Разположението на стойностите на символите от речника е също независимо, тъй като в резултат на прегледа се получава само указател към определена стойност. Макар работата на програмата LOOKUP да зависи от структурата на речника, изброените условия позволяват стандартна връзка на главната програма с програ-

мата за преглед на речник, така че структурата на речника може да се промени и програмата за преглед LOOKUP да се замени с друга, обезпечаваща същата стандартна връзка, без да бъдат правени никакви изменения в главната програма. Ще бъдат представени два вида организация на речници и съответни програми за преглед.

Един много прост начин за организиране на речник е представен на фиг.3.10. Речникът представлява списък, в който всеки символ е представен чрез линеен подсписък. Програмата за преглед сканира този списък, сравнявайки всеки подсписък с линейния списък, представляващ зададения символ. На фиг.3.11 е представена съответна програма за преглед LOOKUP. При реализирането на програмата трябва да се имат предвид приетите условия за стандартна връзка. Базовият регистър Z обезпечава спомагателен указател за сканиране на списъка S, представляващ зададения символ, в процеса на сравняване на този символ със символите от речника. Базовият регистър на речника се разглежда като фиктивен символ в речника, предотвратяващ специалното третиране на празен речник. В този случай базовият регистър на речника трябва да съдържа NIL в CDR полето си.

Речникът от фиг.3.10 има проста организация и преглед, но е неефективен по отношение на използувана памет и особено по време за намиране на търсения символ. Един начин за избягване на тези недостатъци е йерархически организираният речник от фиг.3.12, на който съответствува програмата за преглед от фиг.3.13. Структурата на този речник представлява дърво, а програмата за преглед осигурява движение по дървото. На фиг.3.14 е показано съдържанието на четирите полета на елементите на речника от фиг.3.12.

Изтриването на символи от речника може да стане по два начина - стойността на изтрития символ да се направи нулева или символът да се изключи напълно от речника. Първият начин е неефективен когато в процеса на работа

голям брой символи трябва да се изтриват от речника и да се записват нови. Вторият начин е прост за реализация при линейно организирания речник от фиг.3.10. Единственият недостатък на дървовидно организирания речник в сравнение с линейно организирания е по-трудното изтриване на символи от речника по втория начин.

3.6.2. Атоми

AF полетата на елементите определят дали тези елементи са атоми или елементи на списъци. Базовите регистри на абстрактната машина за обработка на еднопосочни символни списъци представляват атоми със специално предназначение, но съществува възможност за създаване на други атоми, които се различават от базовите регистри по това, че не служат за адресиране на списъчни структури и могат да се състоят от по-вече от един знак. Тези атоми не могат да се въвеждат или извеждат директно от входно-изходните оператори на WISP, а са необходими специални програми. Необходимо е също така да бъде организиран речник на тези атоми, всеки от които има като стойност базов регистър, указващ атома. На фиг.3.15 е показана структурата на речника на атомите.

Нека програмите за въвеждане и извеждане на атоми да отговарят на следните условия:

1/ Вход в програмата за въвеждане се извършва чрез "USE INATOM", а при изход от тази програма базовият регистър X указва базовия регистър на въведения атом.

2/ Вход в програмата за извеждане се извършва чрез "USE PRATOM", като при входа в програмата базовият регистър X трябва да указва базовия регистър на атома за извеждане.

3/ Речника на атомите се адресира от базовия регистър A.

Извличането на атомите от текста зависи от правилата за тяхното външно представяне. Поради това, в програмата за въвеждане на атоми, представена на фиг.3.16, този процес е заменен с коментар. Базовият регистър D трябва да указва базовия регистър на речника, т.е. базовия регистър A. Базовият регистър M не трябва да съдържа NIL за да може всеки нов атом да се включи в речника. Използуването на подпрограмата LOOKUP е известно от параг.3.6.1. Ако атомът не е открит в речника, след изхода от LOOKUP, CDR Y ще съдържа NIL, след което този атом ще бъде включен в речника.

За извеждане на атоми може да се използува програмата от фиг.3.17. Освен изброените по-горе условия, за отбелязване е, че когато се открие разклонение в речника, Y трябва да се придвижи по първия клон на дървото.

3.6.3. Стекове

Списъчните структури са много удобни за реализиране на стекове. Запаметяване на информация в стека може да се извърши чрез прибавяне на нов елемент в началото на списъка, представляващ стека. При извличане на информация от стека, от началото на списъка се отнема един елемент. Използуването на стекове в процесорите за обработка на списъчна информация е така широко застъпено, че в WISP са предвидени два разширени оператора:

PUSH DOWN op.

POP UP op.

Операндът и в двата оператора трябва да бъде базов регистър, представляващ указател към стека.

Съществуват два начина за осъществяване на операцията PUSH DOWN. Единият беше описан по-горе и се състои във включване на нов елемент в началото на съществуващия списък-стек. При този метод съществува една особеност, която трябва да бъде взета под внимание. Да предположим, че два базови ре-

гистра A и B указват началото на стека. Ако бъде изпълнена операцията PUSH DOWN A , новият елемент ще бъде свързан в началото на списък A, а базовият регистър B ще указва втория текущ елемент на стека. Може би това е желателно за някои случаи, но обикновено интерес представлява само най-горния елемент на стека. За да се запазят всички указатели към върха на стека, при операцията PUSH DOWN новият елемент от списъка на свободната памет се включва като втори елемент на списъка-стек. Съдържанието на първия елемент се копира в този нов елемент, а в CDR полето на първия елемент се занася адреса на новия елемент. Новата информация се занася в CAR полето на първия елемент. Това представлява втория подход, който ще бъде използуван в представените примери. Разбира се, изборът на съответен подход зависи от приложението на стека и предпочитанията на реализатора.

На фиг.3.18 е показана програма за отпечатване на S-израз в скобъчна форма, представен като списъчна структура в паметта на абстрактната машина. При входа в програмата Y трябва да указва структурата за извеждане. Списъкът S се използува като стек, но неговата предишна стойност се възобновява след завършване на програмата. X е указател, сканиращ списъка за отпечатване. Проверява се CAR полето на всеки списъчен елемент и ако то указва базов регистър, отпечатва се съответен атом, като се използува познатата подпрограма PRATOM. Ако CAR X указва подсписък, текущата стойност на X се занася в стека S , а в X се занася указател към този подсписък и програмата продължава работата си както преди. Когато се открие край на подсписък, в X се занася стойността на CAR полето на най-горния елемент на стека, след което се изпълнява операцията POP UP S и сканирането се възобновява от съответна точка на разклонение. Ако няма по-вече елементи за сканиране програмата завършва работата си.

На фиг.3.19 е показана програма, аналогична по предназначение на тази

от фиг.3.18, но използуваща стек за запомняне и на адреси от програмата, към които трябва да бъде върнато управлението в съответни моменти.

Фиг.3.19 илюстрира също различен ред на операциите по занасяне на информация в и извличане на информация от стека. При стека от фиг.3.18 операцията PUSH DOWN се извършва преди занасянето, следователно POP UP трябва да се извърши след извличането на информацията от стека, докато при стека от фиг.3.19 действията се извършват в обратен ред. Кой начин ще бъде избран зависи от приложението на стека и удобството при работа. Ако за стек се използува списък, стойността на който трябва да се възобнови след завършване на програмата която използува стека, то трябва да се извърши една операция PUSH DOWN при започване на програмата и една операция POP UP при завършване на програмата, ако се използува занасяне на информация преди операцията PUSH DOWN.

3.6.4. Проследяване на списъци

Примерите от параграф 3.6.3 за отпечатване на списък използуват процес на сканиране на всеки елемент от списъка, който процес се нарича проследяване на списък.

Съществуват два основни метода за проследяване на списъци, познати под имената "движение в дясно" и "движение в ляво". Те се различават по действието си при откриване на разклонение на списъка. При движение в дясно първо се проследява списъка, адресиран от CDR полето на елемента на разклонение, а след това се проследяват подсписъците, докато при движение в ляво първо се проследява подсписъка, адресиран от CAR полето, след което се продължава проследяването на списъка.

Примерите от параграф 3.6.3 използуват движение в ляво и стек за запомняне на точките на разклонение. В някои случаи, обаче, е невъзможно из-

ползуването на стек поради липса на свободна памет, както например в процедурата за събиране на неизползуваните елементи от паметта, представена в параграф 3.6.5. Тази процедура служи за създаване на нов списък на свободната памет, когато старият списък на свободната памет се изразходва, а това изисква проследяване на всички списъци в паметта.

За да се разбере начина на проследяване на списъци без използуване на стек, необходимо е да се разгледа същността на еднопосочните списъци, адресирани от базов регистър. Елементът, указан от базовия регистър, от своя страна указва други два елемента. Ако указаният елемент е базов регистър, той се разглежда като атом и не се проследяват указателите му. Този базов регистър може да бъде базов регистър на абстрактната машина или програмен базов регистър за адресиране на атом от речника на атомите. Връзките в еднопосочния списък позволяват сканиране на списъка от базовия му регистър до съответен атом, но не и обратно. При известни условия, обаче, сканирането в двете посоки става възможно.

Начинът на реализиране на движение в двете посоки при линеен списък е показан на фиг.3.20. Чрез използуване на три спомагателни указателя A, B и C и прилагане три пъти на операциите MOVRGT, списъкът от фиг.3.20.а се трансформира в двойката списъци от фиг.3.20.б.

```
MOVRGT, C = CDR B.
        CDR B = A.
        A = B, B = C.
```

Условие за завършване на движението в права посока е AF B = 1. Фиг.3.20.в показва резултата от движение в обратна посока чрез еднократно прилагане на MOVLFT върху списъците от фиг.3.20.б.

```
MOVLFT, C = CDR A.
        CDR A = B.
        B = A, A = C.
```

Обратното движение може да продължи до достигане на базовия регистър на списъка, т.е. възобновяване на изходното състояние преди започване на проследяването. Условията индициращи достигане на изходното състояние са A = B и AF B = 1.

Проследяването на разклонен списък, показано на фиг.3.21, е малко по-трудно от описаното по-горе проследяване на линеен списък. Това се дължи на обстоятелството, че след срещане на разклонение ще трябва да се проследи и съответния подсписък. Освен това, при проследяване на подсписъците е възможно както движение в дясно, така и движение в ляво. Ако се използува движение в ляво, подсписъците ще се сканират през време на правото движение за проследяване на списъка, докато при движение в дясно те ще се сканират през време на обратното движение. В примерите за проследяване на списъци ще се използува движение в ляво.

В случай на разклонен списък се използува и BPF полето на елементите на разклонение. Когато програмата за проследяване открие такъв елемент, тя зарежда BPF полето му с 1, което позволява при обратното движение да бъде направено решение за посоката на продължаване в зависимост от това дали се използува дясно или ляво движение, т.е. дали указателят за обратното движение е в CAR полето или в CDR полето на елемента на разклонение. Разклонението се открива от операторите

```
C = CAR B.
TO MOVDWN IF AF C = 00.
```

Операциите MOVDWN се използуват за зареждане на BPF полето на елемента на разклонение и започване на движение по разклонението:

```
MOVDWN, CAR B = A, BPF B = 01.
        A = B, B = C.
```

Операциите MOVLFT трябва да бъдат променени за извършване на проверка на

индикатора BPF.

```
MOVLFT, TO MOVUP IF BPF A = 01.
        C = CDR A.
        CDR A = B.
        TO CONTIN.
MOVUP,  C = CAR A.
        CAR A = B.
CONTIN, B = A, A = C.
```

На фиг.3.22 е показана програма за отпечатване на списък, аналогична на програмите от фиг.3.18 и фиг.3.19, която обаче не използува стек, а описания в този параграф метод. В допълнение тя възобновява стойността на базовия регистър Y и освен това използува по-малко памет, защото вместо стек се използува само един специален указател W. Но тази програма изисква повече време, което я прави за предпочитане само тогава, когато няма свободна памет за реализиране на стек.

3.6.5. Автоматично поддържане на паметта

В примерите от предните параграфи поддържането на списъка на свободната памет се извършва от потребителя-програмист. Ако е необходимо да се вземат или да се върнат елементи в списъка на свободната памет, програмистът трябва да напише съответни оператори. Поддържането на списъка на свободната памет по този начин е твърде обременителна работа. Освен това с увеличаване на сложността на програмата става все по-трудно да се реши дали изследваният подсписък е част от друг списък или вече не е необходим. Ако тези елементи се върнат в списъка на свободната памет, но в друг списък се запазят връзки към тях, получават се грешки в работата на програмата, действието на които не може да се предвиди.

Поддържането на паметта включва два процеса - вземане на елементи от списъка на свободната памет и връщане на елементи в него. В предните примери се срещаха операторите:

```
TO FAIL IF (FREE) = NIL.          Проверка за изчерпване
Z = (FREE), (FREE) = CDR FREE.    Вземане на елемент
```

Действието на тези оператори може да се замени с разширената операция:

```
op = NEW ELEMENT.
```

Например, нов елемент може да се вземе от списъка на свободната памет и да се свърже към базовия регистър Z посредством оператора:

```
Z = NEW ELEMENT.
```

Вземането на елемент от списъка на свободната памет е по-сложно от връщането поради това, че при вземане трябва да се проверява дали списъка на свободната памет не е изчерпан. Фактически, връщането може да стане чрез разширената операция POP UP, представена в параграф 3.6.3. Един удобен подход за решаване на проблема за връщане на елементи, обаче, е просто тези елементи да се игнорират до момента на изчерпване на списъка на свободната памет. В този момент обикновено ще има много елементи, които са изключени от съществуващите списъци и не се използват по-вече. Предназначение на програмата GETNEW е да събере тези неизползувани елементи и да ги организира в нов списък на свободната памет. По такъв начин потребителят-програмист е освободен от програмирането на тези процеси за поддържане на паметта. В този случай е необходимо когато се иска нов елемент от списъка на свободната памет да се използува програмата GETNEW чрез операторите:

```
USE GETNEW.
Z = (RESULT).
```

Тези операции предполагат, че GETNEW поставя адреса на новия елемент в CDR полето на елемента RESULT.

Според дефиницията на абстрактната машина, освободените неизползвани елементи не могат да бъдат достигнати от никой базов регистър. Програмата за събирането на тези елементи трябва да сканира всички списъци и да отбележи по някакъв начин всеки техен елемент. Това сканиране се извършва без използуване на стек поради изчерпване на списъка на свободната памет в момента на започването на този процес. След отбелязването на използуваните елементи, всички останали елементи са неизползувани и може да бъдат организирани в нов списък на свободната памет, след което отметките на използуваните елементи се изтриват.

Съществуват три проблема при събирането на неизползуваните елементи:

1/ Намиране на всички базови регистри, служещи за адресиране на списъци.

2/ Отбелязване на използуваните елементи без разрушаване на информацията в тях.

3/ Решаване кога да се преустанови сканирането на дадено разклонение на списък.

За да се направи възможно решението на тези проблеми, необходимо е да се установят някои правила-ограничения при програмирането:

1/ Всички базови регистри, включително тези, създадени по време на изпълнение на програмата, трябва да бъдат свързани във верига чрез CAR полетата си. По такъв начин всеки базов регистър ще може да бъде обходен при сканирането, независимо от кой от тях се започне.

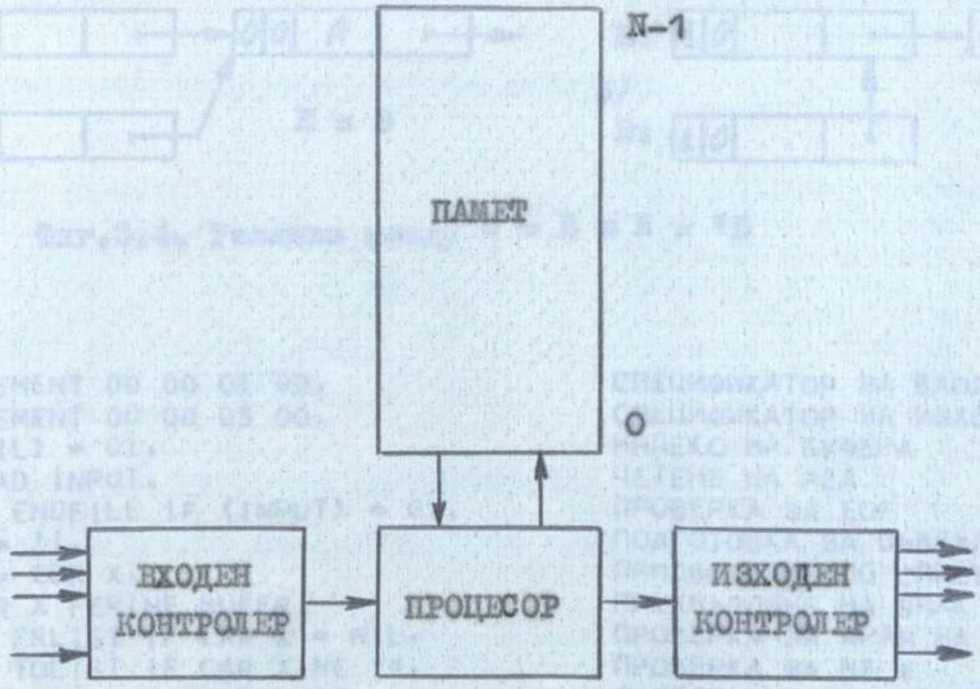
2/ BPF полетата на всички базови регистри и елеминти трябва да бъдат нулирани.

3/ AF полетата на базовите регистри трябва да съдържат единици, докато AF полетата на другите елементи на паметта са нулирани.

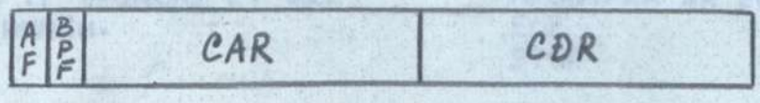
Първото правило осигурява сканиране на всички списъци. Второто правило обезпечава BPF полето на елементите за целите на програмата за сканира-

не /параграф 3.6.4/. Ако се вземат предвид първото и второто правило, третото правило не е необходимо, но осигурява удобства при работа. Понеже AF полето носи в този случай спомагателна информация, то се използува за отбелязване на използуваните елементи. След събирането на неизползуваните елементи, AF полетата на всички елементи освен базовите регистри трябва да бъдат нулирани. Сканирането на дадено разклонение се преустановява при AF=1. Този елемент може да е базов регистър или вече сканиран елемент. Ако е елемент, то всичките му връзки вече са изследвани и сканирането на този клон трябва да завърши, а ако е базов регистър, сканирането на базовите регистри се извършва чрез CAR свързването им. Следователно откриването на AF=1 предизвиква прекратяване на сканирането на дадено разклонение.

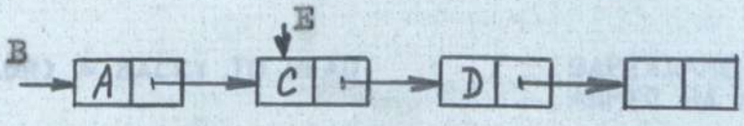
На фиг.3.23 е показана програмата за сканиране TRACER, а на фиг.3.24 - програмата GETNEW, осигуряваща автоматичното поддържане на списъка на свободната памет и използуваща TRACER като подпрограма.



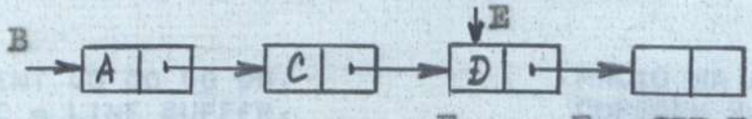
Фиг.3.1. Блокова схема на абстрактната машина за обработка на еднопосочни списъци WSPM



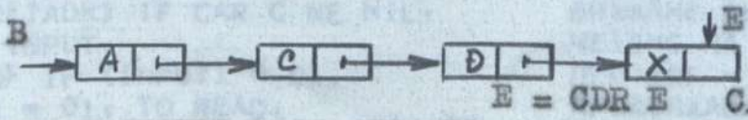
Фиг.3.2. Елемент от паметта на WSPM



а/ списъкът B след присвояването E = CDR B.



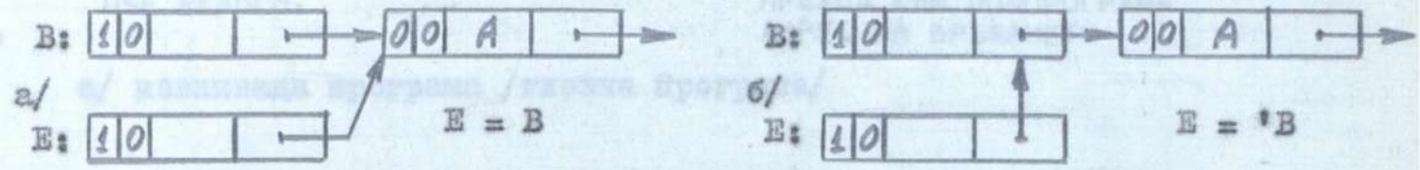
б/ придвижване на указателя E чрез E = CDR E.



E = CDR E и CAR E = 'X.

в/ краен резултат след второто

Фиг.3.3. Сканиране и трансформиране на списък



Фиг.3.4. Разлика между E = B и E = 'B

```
INPUT,    ELEMENT 00 00 01 00.          СПЕЦИФИКАТОР ЗА ВХОДНО У-ВО
PUNCH,    ELEMENT 00 00 03 00.          СПЕЦИФИКАТОР ЗА ИЗХОДНО У-ВО
          (NIL) = 01.                   ИНДЕКС НА БУФЕРА
GETIN,    READ INPUT.                   ЧЕТЕНЕ НА РЕД
          TO ENDFILE IF (INPUT) = 01.   ПРОВЕРКА ЗА EOF
          X = 'I.                       ПОДГОТОВКА ЗА ВЪВЕЖДАНЕ В СПИСЪК I
TOLIST,   X = CDR X.                    ПРИДВИЖВАНЕ ПО СПИСЪКА
          CAR X FERINE BUFER.           ПРЕХВЪРЛЯНЕ НА ЗНАК ОТ БУФЕРА
          TO FRLIST IF CAR X = NIL.     ПРОВЕРКА ЗА КРАЙ НА ИНФОРМАЦИЯТА
          TO TOLIST IF CAR X NE '*.     ПРОВЕРКА ЗА НЕ *
          CAR X = ' , TO TOLIST.        ЗАМЕСТВАНЕ НА * С ШПАЦИЯ
FRLIST,   X = 'I.                       ПОДГОТОВКА ЗА ИЗВЕЖДАНЕ ОТ СПИСЪК I
LOOP,     X = CDR X.                    ПРИДВИЖВАНЕ ПО СПИСЪКА
          LINE BUFFER = CAR X.          ПРЕХВЪРЛЯНЕ НА ЗНАК В БУФЕРА
          TO LOOP IF CAR X NE NIL.      ПРОВЕРКА ЗА КРАЙ НА ИНФОРМАЦИЯТА
          WRITE PUNCH.                  ПЕРФОРИРАНЕ НА РЕД
          TO GETIN IF (PUNCH) LT 01.    ПРОВЕРКА ЗА EOM
ENDFILE,  ENDFILE PUNCH.                EOF
          STOP.                         STOP
```

Фиг.3.5. Програма с използуване на входно-изходни операции

```
          (RETADR) = BACK, TO READ.     ЗАРЕЖДАНЕ НА АДРЕСА ЗА ВРЪЩАНЕ
BACK,                                   АДРЕС НА ВРЪЩАНЕТО
```

а/ извикваща програма /главна програма/

```
RETADR,   ELEMENT 00 00 00 00.          МЯСТО ЗА АДРЕСА ЗА ВРЪЩАНЕ
READ,     CAR C = LINE BUFFER.          ПОРЕДЕН ЗНАК ОТ БУФЕРА
          TO READ IF CAR C = ' .        ПРЕНЕБРЕГВАНЕ НА ШПАЦИИТЕ
          TO (RETADR) IF CAR C NE NIL.  ВРЪЩАНЕ АКО НЕ CRLF
          READ INPUT.                   ЧЕТЕНЕ НА СЛЕДВАЩ РЕД
          TO EOF IF (INPUT) = 01.       ПРИ EOF - ЗАВЪРШВАНЕ
          (NIL) = 01, TO READ.          ПРОДЪЛЖАВАНЕ НА ЧЕТЕНЕТО
```

б/ извиквана програма /подпрограма/

Фиг.3.6. Използуване на подпрограма чрез основни оператори

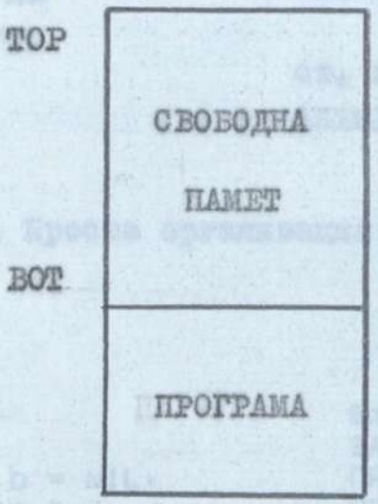
```
        USE READCH.                     ПРЕХОД КЪМ ПОДПРОГРАМА
BACK,                                   АДРЕС НА ВРЪЩАНЕТО
```

а/ извикваща програма /главна програма/

```
        ENTRY READCH.                   ВХОД В ПОДПРОГРАМАТА
READ,   CAR C = LINE BUFFER.            ВЗЕМАНЕ НА ПОРЕДНИЯ ЗНАК ОТ БУФЕРА
        TO READ IF CAR C = ' .          ПРОВЕРКА ЗА ШПАЦИЯ
        TO GETOUT IF CAR C NE NIL.      ПРОВЕРКА ЗА CRLF
        READ INPUT.                     ЧЕТЕНЕ НА РЕД
        TO EOF IF (INPUT) = 01.         ПРОВЕРКА ЗА EOF
        (NIL) = 01, TO READ.            ЗАРЕЖДАНЕ НА ИНДЕКС НА БУФЕРА
GETOUT, EXIT READCH.                    ИЗХОД ОТ ПОДПРОГРАМАТА
```

б/ извиквана програма /подпрограма/

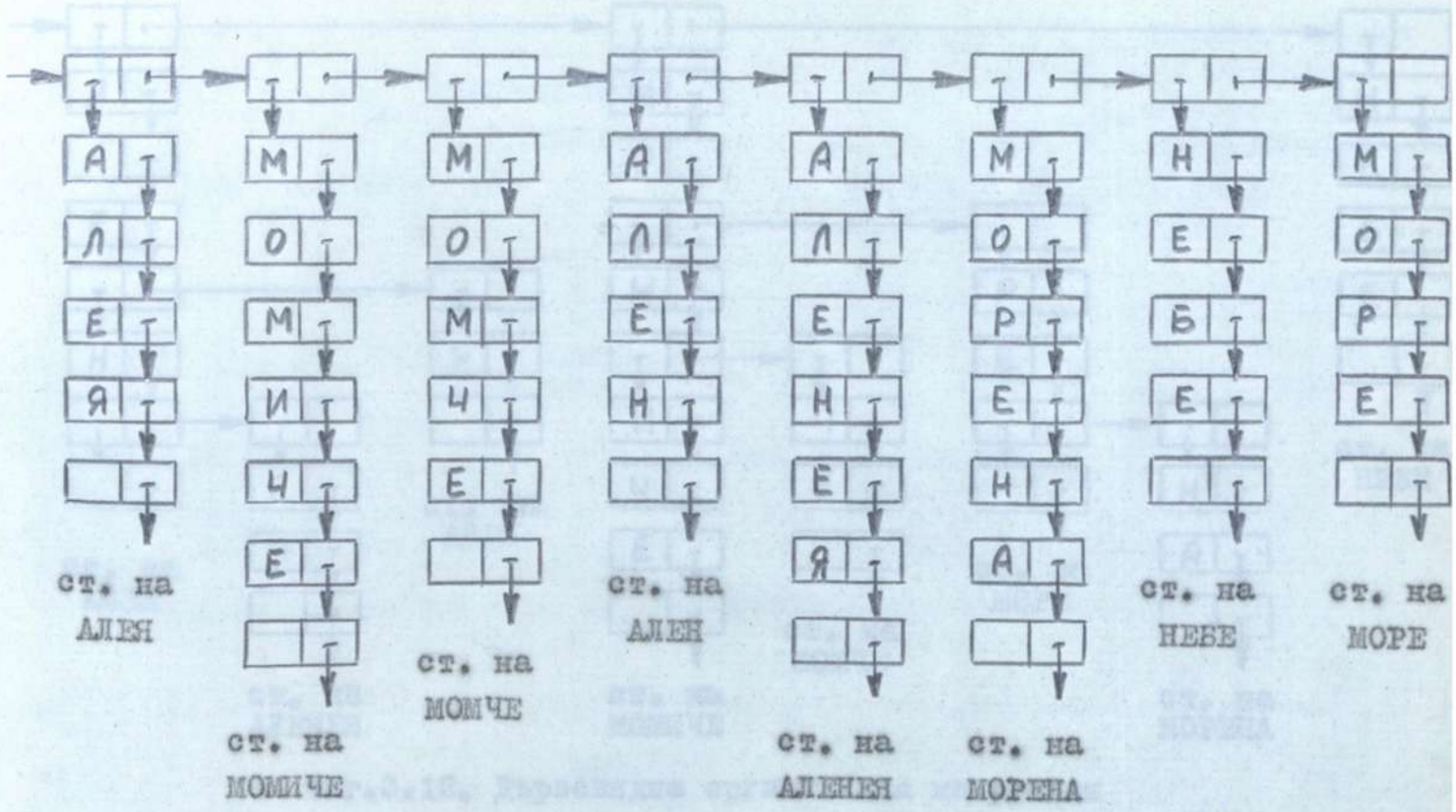
Фиг.3.7. Използуване на подпрограма чрез допълнителни оператори



Фиг.3.8. Разпределение на паметта

```
        A = 8.                          ЗАРЕЖДАНЕ НА УКАЗАТЕЛ А
LOOP,   B = A, INCR A.                  ПРИДВИЖВАНЕ КЪМ СЛЕДВАЩИЯ ЕЛЕМЕНТ
        CDR B = A.                      СВЪРЗВАНЕ НА ЕЛЕМЕНТА
        AF B = 00, BPF B = 00.          ИЗЧИСТВАНЕ НА ФЛАГОВЕТЕ
        TO LOOP IF A NE 9.              КРАЙ НА СВОБ. ПАМЕТ, НЕ
        AF A = 00, BPF A = 00.          ИЗЧИСТВАНЕ НА ФЛАГОВЕТЕ НА ПОСЛ.ЕЛ.
        CDR A = NIL.                    КРАЙ НА СП. НА СВОБ. ПАМЕТ
        (FREE) = 8.                     ПРИКАЧВАНЕ КЪМ FREE.
```

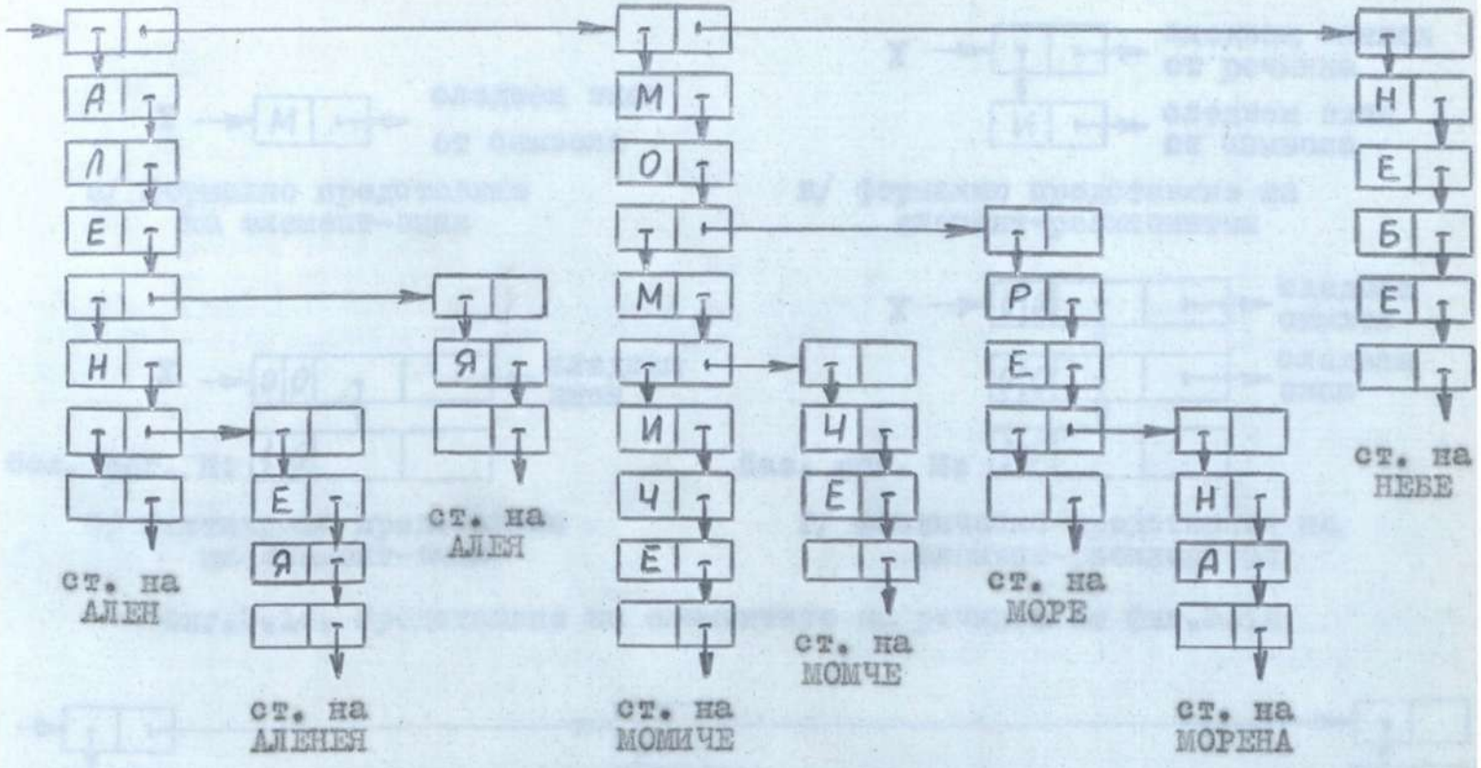
Фиг.3.9. Създаване на списък на свободната памет



Фиг.3.10. Проста организация на речник

```
        ENTRY LOOKUP.                    ВХОД В ПРОГРАМАТА
NEXT,   Z = S.                           ЗАР. НА СП. УКАЗАТЕЛ Z
        TO ABSENT IF CDR D = NIL.        ПРОВ. ЗА КРАЙ НА РЕЧНИКА
        D = CDR D, Y = CAR D.            НЕ, СЛЕДВАЩ СИМВОЛ ОТ РЕЧНИКА
CHECK,  TO NEXT IF CAR Y NE CAR Z.       ПРОВ. ЗА НЕСЪВПАДЕНИЕ
        TO BACK IF CAR Y = NIL.          ПРОВ. ЗА КРАЙ НА СИМВОЛ
        Y = CDR Y, Z = CDR Z.            НЕ, СЛЕДВАЩА СТЪПКА
        TO CHECK.                        ПРОДЪЛЖАВАНЕ НА СРАВНЯВАНЕТО
INSERT, TO FAIL IF (FREE) = NIL.         ПРОВ. НА СП. СВОБ. ПАМЕТ
        CDR D = (FREE), D = (FREE).      ВЗЕМАНЕ НА ЕДИН ЕЛЕМЕНТ
        (FREE) = CDR FREE, CDR D = NIL.  ОБН. НА УКАЗ. КЪМ СП. СВОБ. ПАМЕТ
        CAR D = Z, Y = 'Z.               ВКЛ. НА ЗАД. СИМВОЛ В РЕЧНИКА
TOEND,  Y = CDR Y.                       СКАНИРАНЕ НА СИМВОЛА
        TO TOEND IF CAR Y NE NIL.        ПРОВЕРКА ЗА КРАЙ
BACK,   EXIT LOOKUP.                     ИЗХОД СЪС СЪОТВЕТНА ИНДИКАЦИЯ
ABSENT, TO INSERT IF M NE NIL.           ПРОВЕРКА ЗА ВКЛЮЧВАНЕ
        Y = NIL, EXIT LOOKUP.            ИЗХОД БЕЗ ВКЛЮЧВАНЕ
```

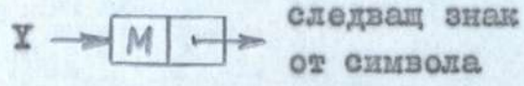
Фиг.3.11. Програма за преглед на речник с организация, показана на фиг.3.10



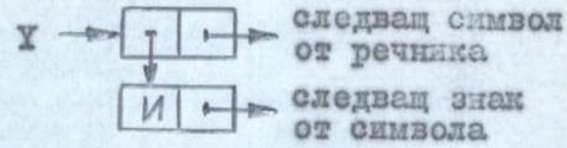
Фиг.3.12. Дървовидна организация на речник

```
          ENTRY LOOKUP.                    ВХОД В ПРОГРАМАТА
          Z = S.                           ЗАРЕЖДАНЕ НА СПОМ. УКАЗАТЕЛ Z
NEXT,     TO ABSENT IF CDR D = NIL.        ПРОВЕРКА ЗА КРАЙ НА РЕЧНИКА
          D = CDR D, Y = CAR D.            НЕ, СЛЕДВАЩ СИМВОЛ ОТ РЕЧНИКА
ZHECK,    TO NEXT IF CAR Y NE CAR Z.       ПРОВЕРКА ЗА НЕСЪВПАДЕНИЕ
          TO BACK IF CAR Y = NIL.          ПРОВЕРКА ЗА КРАЙ НА СИМВОЛА
STEP,     Y = CDR Y, Z = CDR Z.            НЕ, СЛЕДВАЩА СТЪПКА
          TO CHECKSUB IF CAR Y NE CAR Z.   ПРОВЕРКА ЗА НЕСЪВПАДЕНИЕ
          TO STEP IF CAR Y NE NIL.         ПРОВЕРКА ЗА КРАЙ НА СИМВОЛА
          EXIT LOOKUP.                     ДА, ИЗХОД НАМЕРЕН СИМВОЛ
CHRCKSUB, D = Y, Y = CARD.                 ПРОВЕРКА ЗА ПОДРЕЧНИК
          TO CHECK IF AF Y = 00.           ДА, ПРЕТЪРСВАНЕ
          TO NOINSERT IF M = NIL.          ПРОВЕРКА ЗА ВКЛЮЧВАНЕ
          TO FAIL IF (FREE) = NIL.         ПРОВЕРКА НА СП. СВОВ. ПАМЕТ
          Y = (FREE), (FREE) = CDR FREE.   ВЗЕМАНЕ НА ЕДИН ЕЛЕМЕНТ
          CAR Y = CAR D, CDR Y = CDR D.    ЗАПОМНЯНЕ НА ТЕК. ЗНАК И УКАЗАТЕЛ
          CAR D = Y.                       ВКЛЮЧВАНЕ В ПОДРЕЧНИКА
INSERT,   TO FAIL IF (FREE) = NIL.         ПРОВЕРКА НА СП. СВОБ. ПАМЕТ
          CDR D = (FREE), D = (FREE).      ВЗЕМАНЕ НА ЕДИН ЕЛЕМЕНТ
          (FREE) = CDR FREE, CDR D = NIL.  ОБН. НА УКАЗ. КЪМ СП. СВОБ. ПАМЕТ
          CAR D = Z, Y = 'Z.               ВКЛ. НА ЗАД. СИМВОЛ В РЕЧНИКА
TOEND,    Y = CDR Y.                       СКАНИРАНЕ НА СИМВОЛА
          TO TOEND IF CAR Y NE NIL.        ПРОВЕРКА ЗА КРАЙ
BACK,     EXIT LOOKUP.                     ИЗХОД СЪС СЪОТВЕТНА ИНДИКАЦИЯ
ABSENT,   TO INSERT IF M NE NIL.           ПРОВЕРКА ЗА ВКЛЮЧВАНЕ
NOINSERT, Y = NIL, EXIT LOOKUP.            ИЗХОД БЕЗ ВКЛЮЧВАНЕ
```

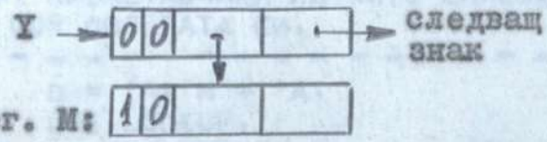
Фиг.3.13. Програма за преглед на речник с организация, показана на фиг.3.12



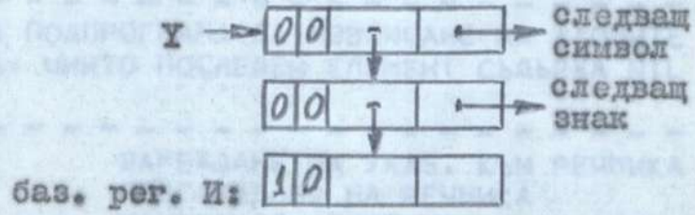
а/ формално представяне на елемент-знак



в/ формално представяне на елемент-разклонител

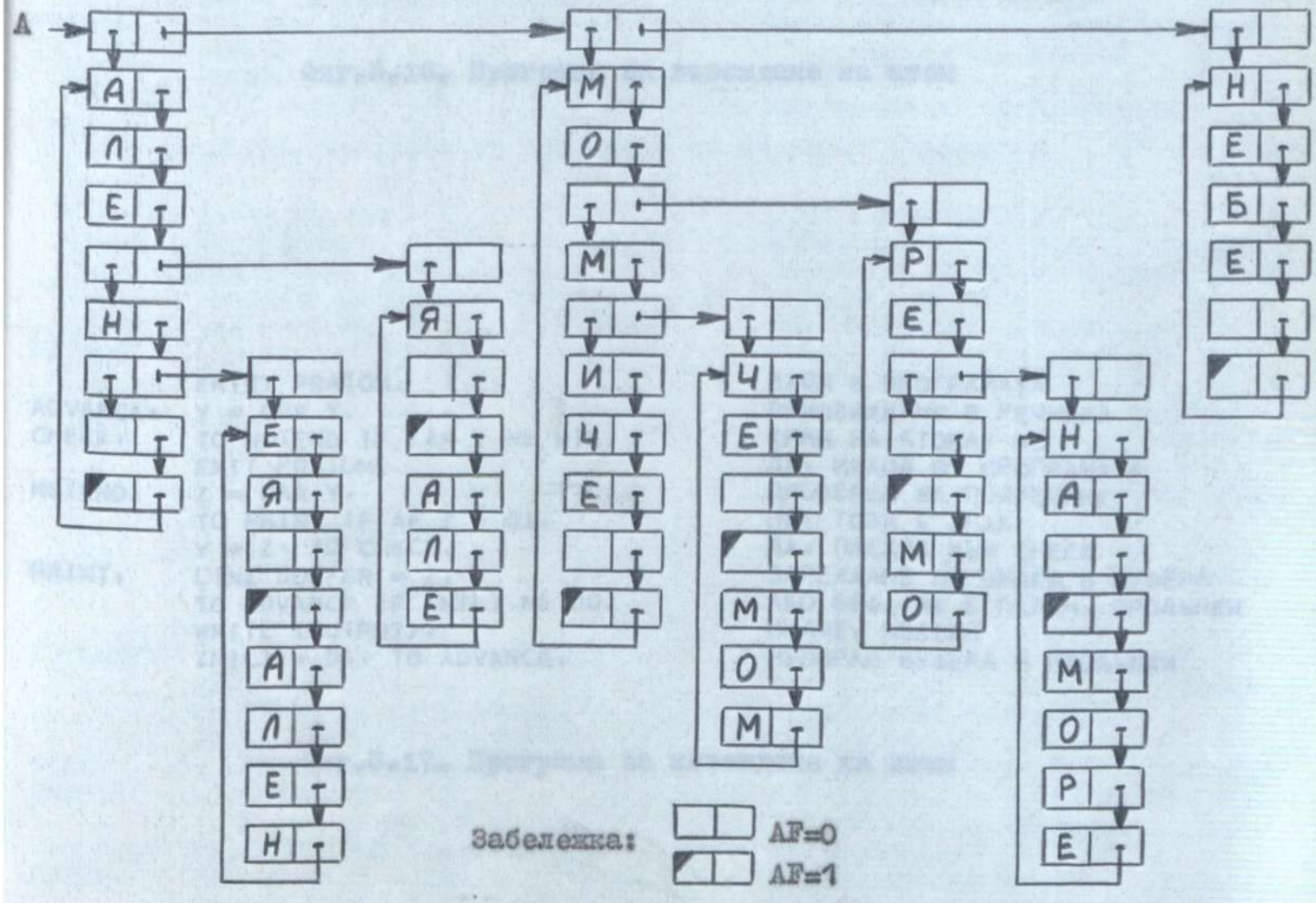


б/ фактическо представяне на елемент-знак



г/ фактическо представяне на елемент-разклонител

Фиг.3.14. Представяне на елементите на речника от фиг.3.12



Фиг.3.15. Речник на атомите

```
          ENTRY INATOM.                          ВХОД В ПРОГРАМАТА
. - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
. НА ТОВА МЯСТО ТРЯБВА ДА БЪДЕ ВКЛЮЧЕНА ПОДПРОГРАМА ЗА ИЗВЛИЧАНЕ НА АТОМИТЕ ОТ
. ТЕКСТА, ПРЕДСТАВЯЙКИ ГИ КАТО СПИСЪК S, ЧИЙТО ПОСЛЕДЕН ЕЛЕМЕНТ СЪДЪРЖА NIL В
. CAR И CDR ПОЛЕТАТА СИ.
. - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          D = 'A, M = 'A.                        ЗАРЕЖДАНЕ НА УКАЗ. КЪМ РЕЧНИКА
          USE LOOKUP.                            ПРЕГЛЕЖДАНЕ НА РЕЧНИКА
          TO NEWATOM IF CDR Y = NIL.             НАМЕРЕН ЛИ Е АТОМА, НЕ
          Y = CDR Y, EXIT INATOM.                ДА, ИЗХОД С УКАЗ. КЪМ БАЗОВ РЕГ.
NEWATOM,  TO FAIL IF (FREE) = NIL.               ПРОВЕРКА НА СП. СВОБ. ПАМЕТ
          CDR Y = (FREE), Y = (FREE).            ВЗЕМАНЕ НА НОВ ЕЛЕМЕНТ
          (FREE) = CDR FREE, CDR Y = S.          ОБН. НА УКАЗ. КЪМ СП. СВОБ. ПАМЕТ
          AF Y = 01, EXIT INATOM.                ПОСТАВЯНЕ НА AF В 1 И ИЗХОД ОТ ПР.
```

Фиг.3.16. Програма за въвеждане на атом

```
          ENTRY PRATOM.                          ВХОД В ПРОГРАМАТА
ADVANCE,  Y = CDR Y.                             ПРИДВИЖВАНЕ В РЕЧНИКА
CHECK,    TO NOTEND IF CAR Y NE NIL.             КРАЙ НА АТОМА, НЕ
          EXIT PRATOM.                           ДА, ИЗХОД ОТ ПРОГРАМАТА
NOTEND,   Z = CAR Y.                             ПРОВЕРКА ЗА ПОДРЕЧНИК
          TO PRINT IF AF Z = 01.                 НЕ, ТОВА Е ЗНАК
          Y = Z, TO CHECK.                       ДА, ПРЕХОД КЪМ CHECK
PRINT,    LINE BUFFER = Z.                       ЗАРЕЖДАНЕ НА ЗНАКА В БУФЕРА
          TO ADVANCE IF (NIL) NE 00.             АКО БУФ. НЕ Е ПЪЛЕН, ПРОДЪЛЖИ
          WRITE (OUTPUT).                        ИНАЧЕ, ИЗВЕДИ
          (NIL) = 01, TO ADVANCE.                НУЛИРАЙ БУФЕРА И ПРОДЪЛЖИ
```

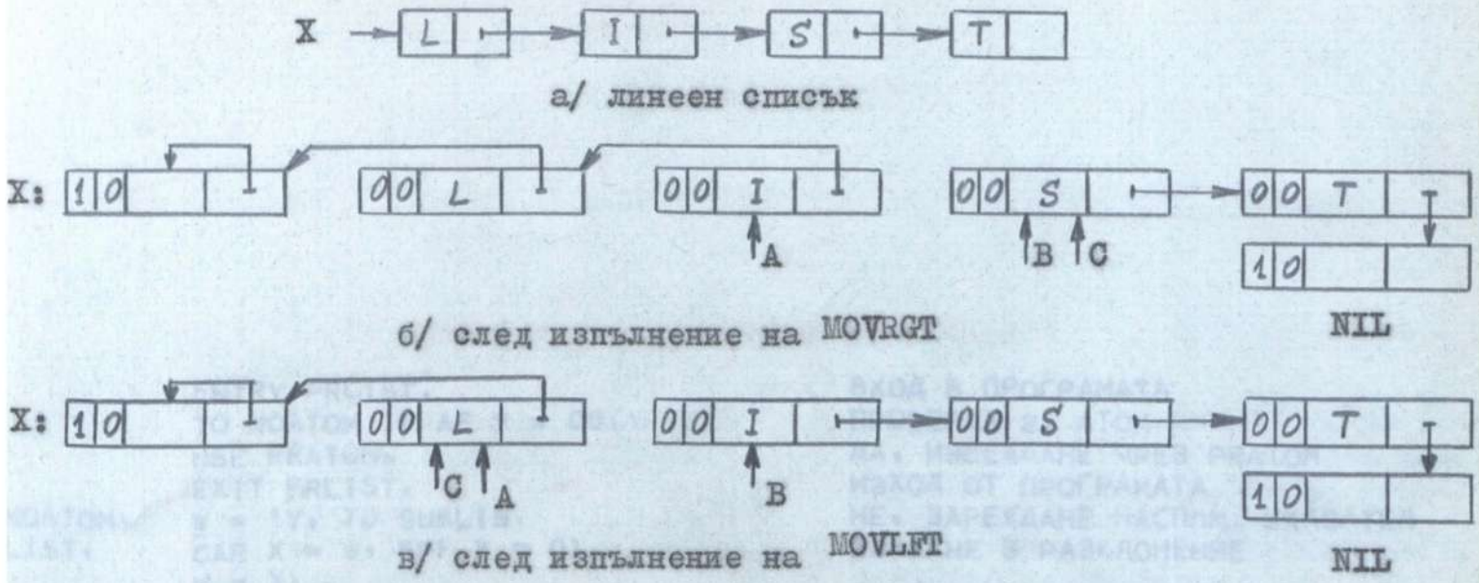
Фиг.3.17. Програма за извеждане на атом

```
          ENTRY PRLIST.                    ВХОД В ПОДПРОГРАМАТА
          TO NOATOM IF AF Y = 00.          ПРОВЕРКА ЗА АТОМ
          USE PRATOM.                      ДА, ИЗВЕЖДАНЕ ЧРЕЗ PRATOM
          EXIT PRLIST.                     ИЗХОД ОТ ПОДПРОГРАМАТА
NOATOM,   X = 'Y.                          НЕ, ЗАРЕЖДАНЕ НА СПОМ. УКАЗАТЕЛ
LIST,     PUSH DOWN S.                     ЗАПОМНЯНЕ НА СПОМ. УКАЗАТЕЛ
          CAR S = X.                           СТЕКА S S
          X = Y, LINE BUFFER = '(.         ОФОРМЯНЕ НА СПИСЪКА С ЛЯВА СКОБА
          TO NEXT IF (NIL) NE 00.          ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
NEXT,     Y = CAR X.                       ПРОВЕРКА НА СЛЕДВАЩИЯ CAR
          TO LIST IF AF Y = 00.            СПИСЪК ЛИ Е, ДА
          USE PRATOM.                      НЕ, ИЗВЕЖДАНЕ
ADVANCE,  TO ENDLIST IF CDR X = NIL.       ПРОВЕРКА ЗА КРАЙ НА СПИСЪК
          LINE BUFFER = ',.                ПРИ КРАЙ НА ЕЛЕМЕНТ, ЗАПЕТАЙКА
          TO STEP IF (NIL) NE 00.          ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
STEP,     X = CDR X, TO NEXT.              ПРИДВИЖВАНЕ КЪМ СЛЕДВАЩ ЕЛЕМЕНТ
ENDLIS,   LINE BUFFER = ').                ДЯСНА СКОБА ЗА КРАЙ НА СПИСЪК
          TO POPUP IF (NIL) NE 00.         ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
POPUP,    X = CAR S.                       ВЪЗОБНОВЯВАНЕ НА УКАЗАТЕЛЯ КЪМ
          POP UP S.                            НЕЗАВЪРШЕН СПИСЪК
          TO ADVANCE IF AF X = 00.         КРАЙ, НЕ
          EXIT PRLIST.                     ДА, ИЗХОД ОТ ПОДПРОГРАМАТА
```

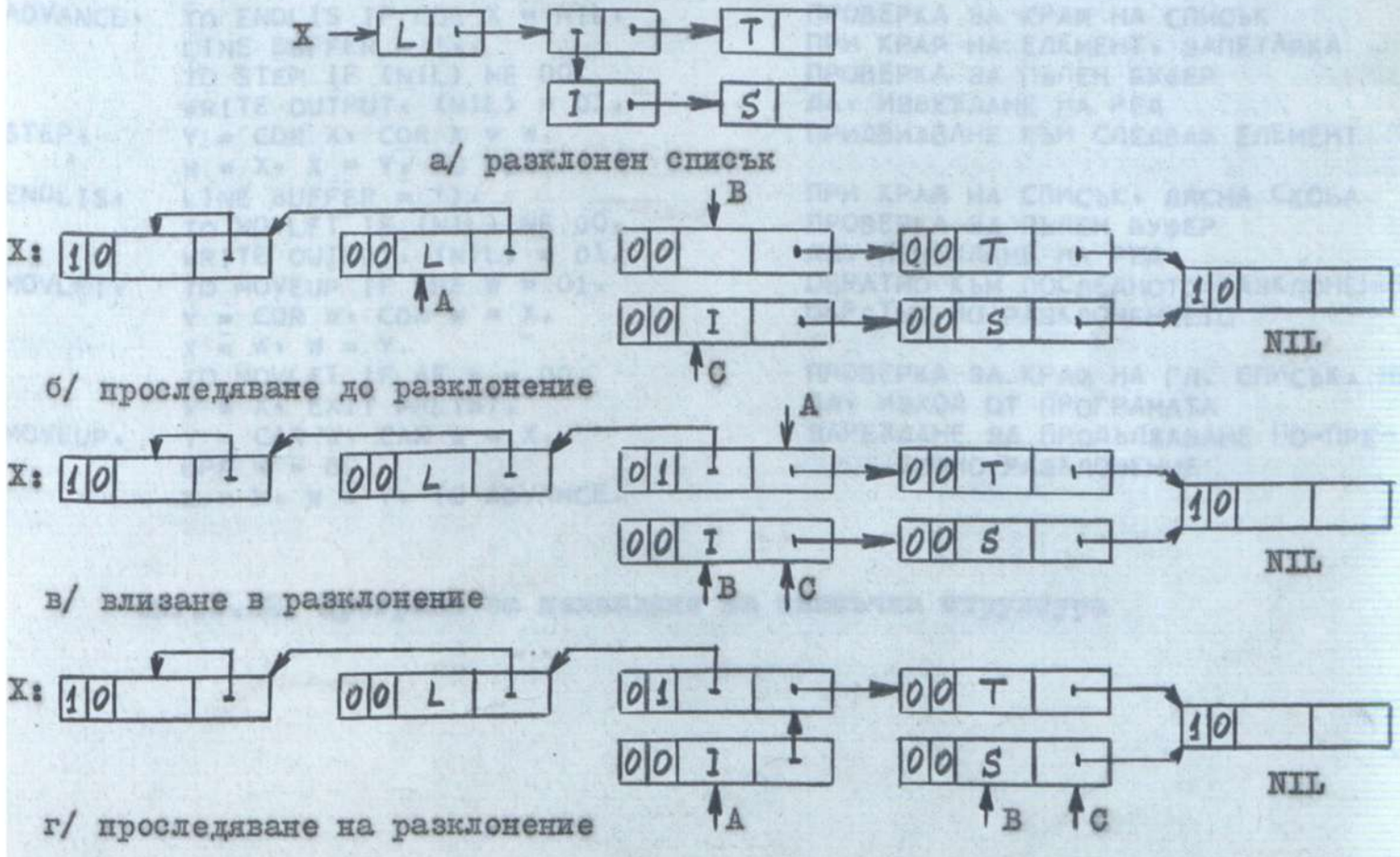
Фиг.3.18. Програма за извеждане на списъчна структура

```
          ENTRY PRLIST.                    ВХОД В ПРОГРАМАТА
          PUSH DOWN S.                     ЗАПАЗВАНЕ НА ТЕКУЩАТА СТОЙНОСТ НА S
          CAR S = GETOUT.                  ЗАРЕЖДАНЕ НА КРАЙНИЯ АДРЕС ЗА ВЪЗВР
          TO ENTER IF AF Y = 00.           ПРОВЕРКА ЗА АТОМ
          USE PRATOM.                      ДА, ИЗВЕЖДАНЕ ЧРЕЗ PRATOM
GETOUT,   POP UP S.                        ВЪЗОБНОВЯВАНЕ НА СПИСЪК S
          EXIT PRLIST.                     ИЗХОД ОТ ПРОГРАМАТА
LIST,     CAR S = ADVANCE.                 ЗАРЕЖДАНЕ НА АДРЕС ЗА ВЪЗВРАТ
ENTER,    PUSH DOWN S.                     ЗАПАЗВАНЕ НА АДРЕСА ЗА ВЪЗВРАТ
          CAR S = X, PUSH DOWN S.          ЗАПАЗВАНЕ НА СПОМ. УКАЗАТЕЛ
          X = Y, LINE BUFFER = '(.         ОФОРМЯНЕ НА СПИСЪКА С ЛЯВА СКОБА
          TO NEXT IF (NIL) NE 00.          ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
NEXT,     Y = CAR X.                       ПРОВЕРКА ДАЛИ СЛЕДВАЩИЯ ЕЛЕМЕНТ
          TO LIST IF AF Y = 00.                Е АТОМ
          USE PRATOM.                      ДА, ИЗВЕЖДАНЕ ЧРЕЗ PRATOM
ADVANCE,  TO ENDLIS IF CDR X = NIL.        ПРОВЕРКА ЗА КРАЙ НА СПИСЪК
          LINE BUFFER = ',.                ПРИ КРАЙ НА ЕЛЕМЕНТ, ЗАПЕТАЙКА
          TO STEP IF (NIL) NE 00.          ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
STEP,     X = CDR X, TO NEXT.              ПРИДВИЖВАНЕ КЪМ СЛЕДВАЩ ЕЛЕМЕНТ
ENDLIS,   LINE BUFFER = ').                ДЯСНА СКОБА ЗА КРАЙ НА СПИСЪК
          TO POPUP IF (NIL) NE 00.         ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
          WRITE OUTPUT, (NIL) = 01.        ДА, ИЗВЕЖДАНЕ НА РЕД
POPUP,    POP UP S, X = CAR S.             ВЪЗОВНОВЯВАНЕ НА СПОМ. УКАЗАТЕЛ
          POP UP S, TO CAR S.              ВРЪЩАНЕ В ИЗВИКВАЩАТА ПРОГРАМА
```

Фиг.3.19. Програма за извеждане на списъчна структура



Фиг.3.20. Движение в права и обратна посока за проследяване на линеен списък



Фиг.3.21. Проследяване на разклонен списък

```
        ENTRY PRLIST.                   ВХОД В ПРОГРАМАТА
        TO NOATOM IF AF Y = 00.         ПРОВЕРКА ЗА АТОМ
        USE PRATOM.                     ДА, ИЗВЕЖДАНЕ ЧРЕЗ PRATOM
        EXIT PRLIST.                    ИЗХОД ОТ ПРОГРАМАТА
NOATOM, W = 'Y, TO SUBLIS.              НЕ, ЗАРЕЖДАНЕ НАСПОМ. УКАЗАТЕЛ
LIST,   CAR X = W, BPF X = 01.          ВЛИЗАНЕ В РАЗКЛОНЕНИЕ
        W = X.
SUBLIS, X = Y, LINE BUFFER = '(.        ОФОРМЯНЕ НА СПИСЪКА С ЛЯВА СКОБА
        TO NEXT IF (NIL) NE 00.         ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
        WRITE OUTPUT, (NIL) = 01.       ДА, ИЗВЕЖДАНЕ НА РЕД
NEXT,   Y = CAR X.                      ПРОВЕРКА ДАЛИ СЛЕДВАЩИЯТ ЕЛЕМЕНТ
        TO LIST IF AF Y = 00.             Е АТОМ
        USE PRATOM.                     ДА, ИЗВЕЖДАНЕ ЧРЕЗ PRATOM
ADVANCE, TO ENDLIS IF CDR X = NIL.      ПРОВЕРКА ЗА КРАЙ НА СПИСЪК
        LINE BUFFER = ',.               ПРИ КРАЙ НА ЕЛЕМЕНТ, ЗАПЕТАЙКА
        TO STEP IF (NIL) NE 00.         ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
        WRITE OUTPUT, (NIL) = 01.       ДА, ИЗВЕЖДАНЕ НА РЕД
STEP,   Y = CDR X, CDR X = W.           ПРИДВИЖВАНЕ КЪМ СЛЕДВАЩ ЕЛЕМЕНТ
        W = X, X = Y, TO NEXT.
ENDLIS, LINE BUFFER = ').               ПРИ КРАЙ НА СПИСЪК, ДЯСНА СКОБА
        TO MOVLFT IF (NIL) NE 00.       ПРОВЕРКА ЗА ПЪЛЕН БУФЕР
        WRITE OUTPUT, (NIL) = 01.       ДА, ИЗВЕЖДАНЕ НА РЕД
MOVLFT, TO MOVEUP IF BPF W = 01.        ОБРАТНО КЪМ ПОСЛЕДНОТО РАЗКЛОНЕНИЕ
        Y = CDR W, CDR W = X.           ОБРАТНО ПО РАЗКЛОНЕНИЕТО
        X = W, W = Y.
        TO MOVLFT IF AF W = 00.         ПРОВЕРКА ЗА КРАЙ НА ГЛ. СПИСЪК, НЕ
        Y = X, EXIT PRLIST.             ДА, ИЗХОД ОТ ПРОГРАМАТА
MOVEUP, Y = CAR W, CAR W = X.           ЗАРЕЖДАНЕ ЗА ПРОДЪЛЖАВАНЕ ПО-ПРЕ-
        BPF W = 00.                       ДИШНО РАЗКЛОНЕНИЕ
        X = W, W = Y, TO ADVANCE.
```

Фиг.3.22. Програма за извеждане на списъчна структура

```
        ENTRY TRACER.                   ВХОД В ПРОГРАМАТА
        TO ENDTR IF CDR P1 LT (BOT).    ПРОВЕРКА ДАЛИ НАЧАЛНИЯТ УКАЗАТЕЛ Е
        TO ENDTR IF (TOP) LT CDR P1.        В РАМКИТЕ НА ОТДЕЛЕНАТА ПАМЕТ
        (P3) = CDR P1.                  ДА, ВЗЕМАНЕ НА АДРЕСИРАНИЯ ЕЛЕМЕНТ
        TO ENDTR IF AF P3 = 01.         ПРОВЕРКА ЗА МАРКА AF=1
        (P2) = (P1).                    НЯМА, ЗАПОЧВАНЕ НА ПРОСЛЕДЯВАНЕТО
FORWD,  CDR P1 = (P2).                  УКАЗАТЕЛ ЗА ВРЪЩАНЕ
MARK,   AF P3 = 01.                     МАРКИРАНЕ НА ЕЛЕМЕНТА
        (P2) = (P1), (P1) = (P3).       ДВИЖЕНИЕ ПО СПИСЪКА
        TO REV IF CDR P1 LT (BOT).      ПРОВЕРКА ДАЛИ CDR ПОЛЕТО УКАЗВА
        TO REV IF (TOP) LT CDR P1.          В РАМКИТЕ НА ПАМЕТТА
        (P3) = CDR P1.                  ДА, ВЗЕМАНЕ НА АДРЕСИРАНИЯ ЕЛЕМЕНТ
        TO FORWD IF AF P3 = 00.         ПРОВЕРКА ЗА МАРКА AF=0
REV,    TO CHKBR IF CAR P1 LT (BOT).    ПРОВЕРКА ДАЛИ CAR ПОЛЕТО УКАЗВА
        TO CHKBR IF (TOP) LT CAR P1.        В РАМКИТЕ НА ПАМЕТТА
        (P3) = CAR P1.                  ДА, ВЗЕМАНЕ НА АДРЕСИРАНИЯ ЕЛЕМЕНТ
        TO BRNCH IF AF P3 = 00.         НЯМА МАРКА, ИМА РАЗКЛОНЕНИЕ
CHKBR,  TO ENDBR IF BPF P2 = 01.        ИНАЧЕ, ПРОВЕРКА ЗА ОБРАТЕН УКАЗАТЕЛ
        (P3) = CDR P2, CDR P2 = (P1).   ВЪЗОБНОВЯВАНЕ НА УКАЗАТЕЛЯ ОТ CDR
        (P1) = (P2), (P2) = (P3).       ДВИЖЕНИЕ ОБРАТНО ПО СПИСЪКА
        TO REV IF (P1) NE (P2).         ПРОВЕРКА ЗА КРАЙ
ENDTR,  EXIT TRACER.                    ИЗХОД ОТ ПРОГРАМАТА
BRNCH,  CAR P1 = (P2), BPF P1 = 01,     ЗАРЕЖДАНЕ НА ОБРАТЕН УКАЗАТЕЛ В CAR
        TO MARK.                            И МАРКИРАНЕ НА РАЗКЛОНЕНИЕТО
ENDBR,  BPF P2 = 00.                    НУЛИРАНЕ НА BPF ПОЛЕТО
        (P3) = CAR P2, CAR P2 = (P1).   ВЪЗОБНОВЯВАНЕ НА УКАЗАТЕЛЯ ОТ CAR
        (P1) = (P2), (P2) = (P3).       ДВИЖЕНИЕ ОБРАТНО ПО СПИСЪКА
        TO CHKBR.
P1,     ELEMENT 00 00 00 00.            ЗАПАЗЕНА ОБЛАСТ ЗА ИЗПОЛЗУВАНИТЕ
P2,     ELEMENT 00 00 00 00.                ВРЕМЕННИ УКАЗАТЕЛИ ЗА
P3,     ELEMENT 00 00 00 00.                ПРОСЛЕДЯВАНЕТО
```

Фиг.3.23. Програмата TRACER за сканиране на списъците и отбелязване на елементите им

```
        ENTRY GETNEW.                   ВХОД В ПРОГРАМАТА
        TO ELOK IF (FREE) NE NIL.       ПРОВ. ЗА ИЗЧЕРПАН СП. СВ. ПАМ., НЕ
        (P1) = NIL.                     ДА, ВЛИЗАНЕ ВЪВ ВЕР. НА БАЗ. РЕГ.
SE@LS,  BPF P1 = 01, (P1) = CAR P1.     ДВИЖЕНИЕ КЪМ СЛЕДВАЩИЯ БАЗ. РЕГ.
        TO FAIL1 IF BPF P1 = 01.        ГРЕШКА ПРЕКЪСНАТА ВЕРИГА
        USE TRACER.                     ПРОСЛЕДЯВАНЕ НА СПИСЪКА
        TO SE@LS IF CAR P1 NE NIL.      КЪМ СЛЕДВАЩ БАЗ. РЕГ.
        (P1) = (BOT), (P2) = FREE.      ФОРМИРАНЕ НА НОВИЯ СП. СВОБ. ПАМЕТ
TSTFL,  TO CLRFL IF AF P1 = 01.         СВОБОДЕН ЛИ Е ТОЗИ ЕЛЕМЕНТ, НЕ
        CDR P2 = (P1), (P2) = (P1).     ДА, ПРИКАЧАНЕ
        TO ADVPT.
CLRFL,  AF P1 = 00.                     НУЛИРАНЕ НА AF ПОЛЕТО
ADVPT,  TO FIXUP IF (P1) = (TOP).       ПРОВЕРКА ЗА КРАЙ НА ПАМЕТТА
        INCR (P1), TO TSTFL.            НЕ, ПРОДЪЛЖИ
FIXUP,  TO FAIL2 IF (P2) = FREE.        ГРЕШКА НЯМА НЕИЗПОЛЗУВАНИ ЕЛЕМЕНТИ
        CDR P2 = NIL.                   ИНАЧЕ, КРАЙ НА СП. СВОБ. ПАМЕТ
        (P1) = NIL.                     ПОДГОТОВКА ЗА ВЪЗОБН. НА БАЗ. РЕГ.
FIXAT,  AF P1 = 01, BPF P1 = 00.        AF=1, BPF=0
        (P1) = CAR P1.                  ДВИЖЕНИЕ КЪМ СЛЕДВАЩ БАЗ. РЕГ.
        TO FIXAT IF (P1) NE NIL.        ВСИЧКИ СА ВЪЗОБНОВЕНИ, НЕ
ELOK,   RESULT = (FREE).                ВЗЕМАНЕ НА ЕЛЕМЕНТ ОТ СП. СВОБ.
        (FREE) = CDR FREE.                   ПАМ. И ПРИДВИЖВАНЕ НА УКАЗАТЕЛ
        EXIT GETNEW.                    ИЗХОД ОТ ПРОГРАМАТА
```

Фиг.3.24. Програмата GETNEW за автоматично поддържане на паметта

4. Функционален процесор за обработка на еднопосочни символни списъци HELP

4.1. СИНТАКСИС НА HELP

В основата си системата за програмиране на базата на функционалния процесор за обработка на еднопосочни символни списъци HELP е аналогична на известната и широко използувана система за програмиране LISP. Записът на потребителската програма при HELP е по-естествен, но са въведени някои ограничения в сравнение с LISP.

В много езици за програмиране е възприета разлика в записването на константите и променливите величини. Подобно решение е възприето и в HELP. Константите са атоми и се представят чрез низове от букви, цифри и звездички, първият символ на които е звездичка. Възможно е включването и на други символи освен изброените по-горе. Това се решава конкретно за съответна реализация. Променливите се дефинират по същия начин, като първият символ в низа не трябва да бъде звездичка. Символният низ, който удовлетворява условието за име на променлива се нарича идентификатор. Звездичките в константата или името на променливата се използуват за обозначаване на шпация, а шпациите в символните изрази на HELP се игнорират, което позволява свободен запис на изразите в изходната програма с оглед на подобряване на четливостта. На фиг.4.1 са показани примери на константи и променливи.

Програмата на HELP е множество от изрази за пресмятане. Входът представлява поредица от такива изрази, всеки от които завършва с ";". Стойностите им се пресмятат и извеждат /отпечатват/. Всяка изведена стойност се отпечатва на нов ред и завършва с ";". На фиг.4.2 е дадено формалното представяне на синтаксиса на HELP.

От представения синтаксис може да се заключи, че функционалният процесор за обработка на символна списъчна информация HELP обработва като основен тип данни символни величини, а максималната синтаксическа еденица е изразът. Освен това процесорът не разполага със средства за обработка на други типове данни и конвенционални процедурно-ориентирани възможности като оператори за преход, присвояване и други подобни, поради което е възможно той да бъде включен като подпрограма в някой процедурно-ориентиран процесор например ALGOL-60. В този случай в ALGOL-60 ще бъдат представени освен целите, реалните и логическите величини още и символни величини, обработката на които ще се извършва чрез пресмятане на символните изрази, които ще бъдат включени в синтаксиса на ALGOL-60 аналогично на аритметичните и логическите изрази.

4.2. СЕМАНТИКА НА HELP

Всеки от синтаксическите типове, представени на фиг.4.2 има стойност, която представлява указател. Например, стойността на атом е указател към базовия регистър за този атом.

Ако разгледаме дефиницията за първичен елемент, то една възможна стойност на първичен елемент е указател към базов регистър. Втора възможна стойност е указател към списък от изрази, стойностите на които от своя страна са указатели. Трета възможна стойност е указател към израз, заграден в ско-

би. Тъй като скобите се използуват само за указване на реда на действията при пресмятането на стойността, стойността на (E) е еквивалентна на стойността на E. Четвърта възможна стойност на първичен елемент е указател за изпълнение на функция, а пета възможна стойност - указател към аргумент на текуща функция. На фиг.4.3 е показана разликата между стойностите на E, (E) и E.

Логическият израз **not** LE е дефиниран само ако LE има стойност *TRUE или *FALSE, а **not** е логически оператор за отрицание.

Стойностите на $P_1=P_2$ и $P_1=P_2$ са дефинирани, ако стойностите на P_1 и P_2 са атоми. Стойността на $P_1=P_2$ ще бъде *TRUE ако P_1 и P_2 са идентични атоми и *FALSE ако P_1 и P_2 са различни атоми. Аналогично се дефинира стойността на $P_1=P_2$.

Простият израз въвежда логическите оператори за обединение **or** и за сечение **and**. В първите два случая простият израз не е определен, ако някой от операндите няма логическа стойност. Трета възможна стойност на прост израз е логически израз, а четвърта възможна стойност на логически израз е първичен елемент.

Функционален процесор само с прости изрази е аналогичен на конвенционален език за програмиране без възможности за условен преход. Изразът в HELP обезпечава условен преход чрез проверка на условието следващо **if**, в резултат на която се изпълнява простият израз следващ **then** или изразът следващ **else**. Може да се отбележи, че както условието, така и изразът следващ **then** са прости изрази и не могат да съдържат условие в себе си. Но тъй като всеки израз заграден в скоби е прост израз и скобите не променят стойността на израза, ограничение реално не съществува.

Представената реализация на HELP притежава пет основни функции, които са вградени в процесора. Те са:

CAR(E)

CDR(E)

CONS(E_1,E_2)

ATOM(E)

NULL(E)

Всички други функции трябва да бъдат дефинирани чрез израз със следната форма:

def $I_0(I_1,I_2,I_3, \ldots ,I_n):=E;$

I_0 е идентификатор, представляващ името на функцията, а $I_1,I_2 \ldots I_n$ са идентификатори, които се използуват като формални аргументи в дефиницията. При извикване на функцията за изпълнение тези формални аргументи се заместват с фактическите аргументи. За да има разлика между променливи и извиквания на функции за изпълнение, дори функции без аргументи трябва да притежават заграден в скоби списък на аргументи, макар и този списък да бъде празен. Идентификаторите $I_1,I_2, \ldots ,I_n$ от дефиницията на функция се наричат свързани променливи поради това, че са свързани с дефиницията на дадена функция. Те се изброяват в лявата страна на дефиницията и само тези, които са изброени в ляво, могат да участват в израза в дясната страна. Символът ":=" е оператор за присвояване. Резултатът от дефинирането на функция е процедура, която може да бъде приложена към определени аргументи за да се получат желани резултати.

Пример за дефиниране и използуване на функция е показан на фиг.4.4. Тази функция има за стойност втория елемент на аргумента,който трябва да бъде списък.

4.3. HELP ПРОЦЕСОР

Процесорът HELP може да се раздели на две главни части - компилатор и интерпретатор. Компилаторът чете подлежащите на обработка изрази и ги преобразува в списъчни структури, извършвайки синтаксическа проверка, след което интерпретаторът пресмята стойностите на тези изрази. Преобразуването на основните типове изрази в списъчни структури е показано на фиг.4.5. CAR полето на първия елемент на структурата съдържа адреса на процедурата, която пресмята стойността на тази структура-израз. Останалите полета на структурата указват подструктурите, които трябва да бъдат пресметнати, за да се пресметне структурата. Аналогично за подструктурите - CAR полето на първия елемент на подструктурата съдържа адреса на процедурата, която пресмята стойността на тази подструктура, а останалите полета на тази подструктура указват подподструктурите и т.н.

Пример за вътрешно представяне на израз на HELP е даден на фиг.4.6. Списъчната структура от фиг.4.6 съответства на израза

```
if ATOM(X) then X = < > else *FALSE;
```

Реализацията на процесора HELP е предназначена за електронната изчислителна машина FACOM 230-45S. Следователно, необходимо е процесорът HELP да се получи като асамблерова програма за тази машина. За тази цел се извършва транслиране на изходния WISP текст на процесора HELP в обектен FASP /асамблерите за машините FACOM се наричат FASP/ текст. Към този текст се прибавя входно-изходната управляваща програма WIOCS, написана също на FASP. За транслиране от WISP в FASP се използува спомагателният процесор за обработка на символни низове BASCOM и дефинициите на операторите на WISP чрез оператори на FASP, представени в масива WSPM.

За да се получи обектна машинна програма, FASP текстът на процесора HELP заедно с присъединената входно-изходно управляваща програма WIOCS се

асамблира, след което се извършва редактиране на връзките.

По-подробни сведения за описаните процеси на транслиране и асамблиране за получаване на процесора HELP като машинна програма са дадени в съответните раздели на пета глава.

В останалата част на настоящата глава се разглежда организацията и действието на процесора HELP, като за основа на разглеждането служи изходния WISP текст на процесора, представен в края на главата.

4.3.1. HELP интерпретатор

Интерпретаторът служи за пресмятане на стойностите на символните изрази, преобразувани от компилатора в съответни списъчни структури, показани на фиг.4.5. За всеки тип структура-израз съществува подпрограма, която пресмята стойността й. Дадена подпрограма служи за пресмятане само на съответстващата й структура, но в процеса на пресмятане може да се извърши обръщение към друга подпрограма или рекурсивно обръщение към същата подпрограма за да бъде пресметната съответна подструктура.

Крайният резултат от работата на подпрограма за пресмятане на символен израз ще бъде един и ще представлява стойността на този израз, но в процеса на пресмятане може да се използват няколко промеждутъчни резултата. Поради рекурсивното използуване на подпрограмите тези промеждутъчни резултати не могат да се запомнят във фиксирани елементи от паметта, а за тази цел се използува стек на резултатите. Ако някоя програма трябва да запомни резултат, тя го записва в стека. Никоя програма не може да извежда от стека без да е записала в него. Също поради рекурсивното използуване на подпрограмите, при извикване на подпрограма, в стек трябва да се записват адреса за връщане и указателите към подструктури. За запомняне на промеждутъчните резултати се използува стека R, за указателите към подструктури - стека S

и за адресите за връщане - стека T, а базовият регистър E указва структурата, която се пресмята.

На фиг.4.7 са представени блок-схемите и кодирането на подпрограмите за пресмятане на синтаксическите еденици A и P_1=P_2.

Подпрограмата за пресмятане на A е твърде проста. Адресът на атома се занася в стека R, след което се прави връщане в извикващата програма.

Подпрограмата за пресмятане на P_1=P_2 запомня указателя към подструктурата P_2 в стека S. Адресът за връщане се запомня в стека T, след което се пристъпва към пресмятане на подструктурата P_1, което се извършва чрез извикване на съответната подпрограма за пресмятане, адресът на която се намира в CAR полето на първия елемент на P_1. Ако стойността на P_1 не е атом, извършва се преход към индикация за грешка, а ако е атом - тази стойност се занася в стека R след което се пристъпва към пресмятане на P_2. От стека S се изважда указателят към подструктурата P_2, а адресът за връщане се запомня в стека T. Пресмятането на стойността на P_2 се извършва чрез извикване на подпрограмата за пресмятане, адресът на която се намира в CAR полето на първия елемент на P_2. Ако стойността на P_2 не е атом, извършва се преход към индикация за грешка, а ако тази стойност е атом, тя се сравнява със стойността на P_1 и при равенство на двете стойности крайният резултат е *TRUE, а в противен случай - *FALSE. Подпрограмата за пресмятане на P_1=P_2 завършва с изваждане на адреса за възврат към извикващата програма от стека T и извършване на този възврат.

Всички подпрограми за пресмятане на синтаксически еденици работят аналогично на представените по-горе две подпрограми, с изключение на подпрограмите за пресмятане на функции. Последните също трябва да позволяват рекурсивно използуване, което означава, че за записване на фактическите стойности на аргументите не може да се използуват фиксирани адреси от па-

метта, а е необходимо да се използува стек. Трябва да се обезпечи и механизъм за свързване на формалните аргументи от дефиницията на функцията със стойностите на съответните фактически аргументи за съответно извикване.

Пресмятането на функция се свежда до създаване на списъчна структура от атоми и стойностите на фактическите аргументи на функцията за съответно извикване. Тъй като не се позволява използуването на "свободни" аргументи, т.е. такива, които не са включени в списъка на формалните аргументи от дефиницията, може да се каже, че всички аргументи са локални за съответна функция. Схемата за свързване на формалните аргументи с фактически стойности изисква да се направят достъпни само тези аргументи, които са декларирани от функцията, изпълнявана в момента. Когато преди завършване на изпълнението на една функция се извика друга функция за изпълнение, текущите аргументи на първата функция трябва да се направят недостъпни чрез запомняне в стек и да се установи нов набор от фактически аргументи, съответствуващ на втората функция. Когато изпълнението на втората функция завърши, нейните аргументи се изтриват и достъпът до аргументите на първата функция се възобновява.

Фиг.4.8 показва вътрешното представяне на функция. Всички отнасяния към аргументите се представят чрез указатели към елементите на списъка на формалните аргументи от дефиницията.

Връзките между формалните аргументи и фактическите им стойности се установяват от списъка на връзките Q. Всеки член на Q се състои от елемент, чието CAR поле адресира формален аргумент от дефиницията, а CDR поле - фактическата стойност на този аргумент за съответно извикване на функцията. На фиг.4.9 е показан списъка Q за дефиницията

```
def REP(A,B) := if B=A then *C else A ;
```

и извикването

REP(*X,*Y);

заедно с дефиницията и извикването на функцията. Списъкът Q се използува от подпрограмата за намиране на фактическите стойности на аргументите. Кодирането на тази подпрограма е дадено на фиг.4.10. Тя преглежда списъка Q за елемент, чието CAR поле сочи формалния аргумент за заместване. Z е помощен указател за придвижване по списъка, а Y е временен указател към членовете на списъка. Сравняването на E и CAR Y се използува за откриване на съответствие на аргумента, а текущата стойност на този аргумент ще бъде указателят CDR Y.

Пресмятането на функция включва три етапа:

1/ Пресмятане на стойностите на фактическите аргументи от извикването на функцията.

2/ Свързване на тези пресметнати стойности с имената на формалните аргументи от дефиницията на функцията.

3/ Пресмятане на стойността на самата функция.

Подпрограмата за пресмятане на функции е показана на фиг.4.11. Най-напред тя проверява дали функцията е дефинирана, а след това - не е ли функция без аргументи. Ако има аргументи в стека R се поставя преграда, която служи за отделяне на пресметнатите стойности на аргументите на тази функция от намиращата се в стека друга информация. За преграда служи WISP атома F, който никога не може да служи за стойност на аргумент, защото HELP атомите трябва да започват със звездичка, което се проверява от компилатора. Изчислената стойност на поредния аргумент се запомня в стека Q, след което се пресмята израза, представляващ следващия аргумент. Тази стойност се запомня и процесът продължава до изчерпването на списъка на фактическите аргументи от извикването. Стекът S служи за запомняне на указателя към следващия фактически аргумент през времето, когато се пресмята предишният аргумент.

След като всички аргументи са пресметнати, указателят E трябва да сочи отново дефиницията на функцията за да се извърши свързването на пресметнатите стойности на фактическите аргументи със съответстващите им формални аргументи. В този момент стойностите на пресметнатите фактически аргументи са занесени в стека R в обратен ред. Списъкът на формалните аргументи от дефиницията също е подреден в обратен ред. Поради това списъкът на връзките Q може да бъде формиран веднага. Всеки член на този списък съдържа стойност в CDR полето и указател към формален аргумент в CAR полето на елемента за връзка. Построяването на списъка на връзките се извършва от подпрограмата от фиг.4.11 при етикет L32, която също проверява съответствието между броя на аргументите от дефиницията и извикването.

След пресмятането на функцията състоянието на R и S се възстановява. Състоянието на Q и T зависи от това, дали функцията има аргументи или не. Ако не е имало аргументи, направен е преход към L331 при което Q и T са останали непроменени. Ако е имало аргументи, списъкът Q се използува и преди изхода от подпрограмата неговото първоначално състояние трябва да се възстанови. Това се извършва при етикет L332. Стекът T се използува за запомняне на адресите за възврат.

Вградените функции не изискват списък на връзките, по което се различават от дефинираните от програмиста функции. Тяхното реализиране е аналогично на реализирането на подпрограмите за пресмятане на синтаксически елементи с използуване на стека на резултатите R. Дефинициите на тези функции не съдържат списък на формални аргументи, поради което CDR полето на дефиниращия елемент може да се използува за индициране на вградена функция. Докато при дефинираните от програмиста функции това поле съдържа NIL или указател към списъка на формалните аргументи, при дефинирането на вградените функции това поле съдържа по условие WISP атома F. Подпрограмата за прес-

мятане на стойността на вградената функция CONS е показана на фиг.4.12. Стекът T се променя и в случай на вградена функция. За да се направи възврат направо към програмата извикваща L3, най-горният излишен адрес за възврат към L3 трябва да се изтрие от стека T. Вградена функция, която се извиква без аргументи се отхвърля при етикет L331.

4.3.2. HELP компилатор

Компилаторът извършва лексически и синтаксически анализ на потребителската програма и преобразува символните изрази в списъчни структури. При лексическия анализ се проверява външното представяне на синтаксическите елементи във входния низ на потребителската програма. Например, лексическият анализатор отчита факта, че атомът представлява низ от букви, цифри и звездички, започващ със звездичка, а за синтаксическия анализатор атомът е просто указател към базов регистър от речника на атомите.

Лексическият анализ се извършва от програмата FINPUT, която се извиква за разпознаване на поредния синтаксически елемент. При изхода от FINPUT, базовият регистър Z съдържа номера на разпознатия синтаксически елемент, както е указано в табл.4.1.

В случай на атом, идентификатор или функция типът на разпознатия елемент не е достатъчен, а трябва да се знае кой атом, идентификатор или функция е това. За запомняне на атомите, идентификаторите и дефинициите на функции процесорът използува три речника:

- речник на атомите A,
- речник на променливите V,
- речник на функциите F.

След разпознаване на такъв елемент, базовият регистър Y ще сочи представянето на този елемент в съответния речник.

За тези основни символи от езика HELP, за които не съществува съответен знак в набора от знаци на реалната изчислителна машина, могат да се използуват запазени думи или комбинации от съществуващи знаци.

При откриване на грешна синтаксическа еденица или запазена дума, лексическият анализатор връща номер на синтаксически елемент 0. Когато синтаксическият анализатор открие синтаксически елемент 0, той прекратява обработката на съответния израз и преминава към пресмятане на следващия израз.

РАЗПОЗНАТ СИНТАКСИЧЕСКИ ЕЛЕМЕНТ	НОМЕР НА ЕЛЕМЕНТА (Z)	ЗАБЕЛЕЖКА
Неразпознат	0	
Атом	1	Y е указател към стойността на атома
Идентификатор	2	Y е указател към идентификатора
Функция	3	Y е указател към дефиницията на функцията
<	4	Възможно представяне '(
>	5	Възможно представяне ')
=	6	
(	7	
)	8	
;	9	Възможно представяне ',
≠	10	Запазена дума NE
,	11	
def	12	Запазена дума DEF
if	13	Запазена дума IF
then	14	Запазена дума THEN
else	15	Запазена дума ELSE
and	16	Запазена дума AND
or	17	Запазена дума OR
not	18	Запазена дума NOT
:=	19	Възможно представяне '=

Табл.4.1. Таблица на номерата на синтаксическите елементи

На фиг.4.13 е показана програмата за лексически анализ FINPUT, която се обръща към съответни подпрограми при определени ситуации. Тя използува програмата NEXTCH за записване на поредния знак, различен от шпация или край на реда, в елемента CHAR. В някои случаи FINPUT трябва да чете един знак след текущия синтаксически елемент и трябва да запази този знак до следващото извикване. При етикет SKIP, FINPUT разполага с първия знак на съответния пореден синтаксически елемент. Този знак се намира в CHAR и трябва да бъде изследван за да се определи номера на синтаксическия елемент. Атомът започва със звездичка, докато идентификаторът - с буква или цифра. Най-напред се прави проверка за атом. Проверката за идентификатор е означена условно чрез некоректния оператор

TO IDFN IF (CHAR) = LETORDIG.

За съответна реализация този оператор трябва да бъде заменен с група оператори, отчитащи конкретното вътрешно представяне на буквите и цифрите, обезпечено от входно-изходната управляваща програма WIOCS. Ако тези знаци са представени чрез последователни цели положителни трицифрени числа, броят на необходимите оператори се намалява. Това се вижда от фиг.4.14, на която са показани два конкретни начина за проверка дали знака е буква или цифра.

Ако първият знак на синтаксическия елемент е апостроф, прави се преход към етикет MULT за разпознаване на алтернативно представяне на основен символ от HELP чрез подходящ знак на реалната изчислителна машина, предшествуван от апостроф. След това чак се прави проверка за нормално представяне на основен символ.

ATOM, IDFN и MULT от фиг.4.13 са подпрограми за възприемането на атом, идентификатор /променлива, функция или запазена дума/ и алтернативно представяне на основен символ. Те от своя страна използуват програмите NEXTCH и READST, представени на фиг.4.15. Програмата READST се използува за форми-

ране на вътрешно представяне на атом или идентификатор чрез списък от букви, цифри и шпации /звездичките от атомите или идентификаторите на потребителската програма се заместват с шпации/.

Тъй като в HELP се използуват запазени думи за представянето на някои основни символи на езика, теоретическата постановка за пренебрегване на всички шпации в потребителската програма не е вече правилна, тъй като шпациите се използуват за разделяне на запазените думи. Това означава, че в атомите и идентификаторите не трябва да се включват шпации. READST определя дали шпацията е от значение и чрез елемента C управлява действието на NEXTCH. Ако в C е записан кода на шпация, NEXTCH пренебрегва прочетената шпация, а ако в C е записан неправилен код, NEXTCH разглежда всички прочетени знаци като значими.

Синтаксическият анализатор е получен чрез кодиране на синтаксическите формули от фиг.4.2. За всеки синтаксически тип има подпрограма за разпознаване, която просто търси синтаксическите елементи, които трябва да бъдат употребени, а списъкът, представляващ съответния израз, се създава в процеса на синтаксически анализ. Всяка синтаксическа подпрограма има два входа - единият се използува в случай на четене на следващ елемент, а другият - когато четене не се извършва. При изхода подпрограмата занася в CAR R указател към списъка, представляващ разпознатия синтаксически тип и чете синтаксическия елемент, който следва разпознатия тип.

На фиг.4.16 е показана подпрограмата за анализиране на прост израз. Двата входа в подпрограмата са RSE и SE. Списъкът T служи за запомняне на адресите за връщане. Тъй като формулата за прост израз е

SE ::= LE and SE LE or SE LE ,

подпрограмата трябва да открие логически израз, може би следван от and или or. След занасяне на адреса за възврат SE1 в стека T, подпрограмата прави

преход към подпрограмата за анализиране на логически израз. За отбелязване е, че в този случай се използува входът LE в подпрограмата за логически израз, при което не се чете синтаксически елемент. След връщането към SE1, указателят към списък, представляващ логически израз, е занесен в CAR R и е прочетен синтаксическия елемент, следващ този логически израз. Ако прочетеният елемент е нито **and** и нито **or**, стойността на логическия израз е стойност на простия израз и подпрограмата за разпознаване на прост израз прави връщане в извикващата програма. Стойността на простия израз е в CAR R.

Ако е прочетен елемент **and**, прави се преход към SE2, където стойността на логическия израз се запомня в стека R. Адресът L8 на интерпретиращата програма за **and** се записва в следващия елемент на R. При SE3 този адрес се запомня в стека R, а в стека T се занася нов адрес за възврат SE4, след което се прави рекурсивен вход чрез RSE в подпрограмата за синтаксически анализ на прост израз. Това се изисква от формулата за прост израз. След като синтаксическият тип следващ **and** бъде разпознат като прост израз, първоначалното изпълнение на подпрограмата за прост израз се възобновява от етикет SE4. Всички компоненти на простия израз са разпознати /LE, and, SE/ и върхът на стека на резултатите има съдържание, показано на фиг.4.17.а. Сега подпрограмата трябва да конструира списък, представляващ стандартното представяне на прост израз, показано на фиг.4.5 и дублирано на фиг.4.17.б. Това се извършва чрез разместване на най-горните три елемента в стека R и е показано на фиг.4.17.в. W и X се използуват като спомагателни указатели. След разместването, подпрограмата за разпознаване на прост израз връща управлението на извикващата програма. Стойността на разпознатия прост израз е в CAR R.

Всички подпрограми за синтаксически анализ работят аналогично на представената подпрограма за прост израз. Изключение прави само подпрограмата

за анализ на дефиниция на функция, представена на фиг.4.18. При етикет NEXTE подпрограмата е готова да обработи поредния израз от потребителската програма на HELP. Първият синтаксически елемент на този израз е прочетен и в стека на адресите за възврат T се занася адреса на подпрограмата EVAL, която управлява пресмятането на изразите.

Ако първият синтаксически елемент не е def, подпрограмата EX анализира израза и връща управлението на EVAL с указател към конструирания съответстващ на израза списък в CAR R и неразпознат завършващ синтаксически елемент, който трябва да е ";", а EVAL ще провери дали този завършващ елемент е правилен.

Ако първият синтаксически елемент е def, трябва да се създаде списъчна структура, съответстваща на дефиниция на функция. Тъй като в HELP е недопустимо функциите да се дефинират отново, т.е. да получават ново значение, прави се проверка в речника на функциите F дали съответна функция вече не е дефинирана, след което имената на новите функции се възприемат. Прави се сканиране на списъка на формалните аргументи, който може да е и празен. За всеки формален аргумент в списъка R се включва елемент, чието CAR поле съдържа адреса L4 на съответстващата подпрограма от интерпретатора /виж фиг.4.5/. След като всички аргументи се обработят, проверява се оператора ":=". В този момент CDR полето на най-горния елемент на стека R адресира списъка на аргументите, но CAR полето е неизползувано /виж фиг.4.19/. Подпрограмата занася адреса за връщане в стека T и прави преход към REX. REX компилира следващия оператора ":=" израз и връща управлението с указател в CAR R към списъчната структура, съответстваща на израза, а в Z - номера на следващия синтаксически елемент, който трябва да е ";". Това означава, че е получена желаната списъчна структура, съответстваща на прочетената дефиниция на функция /виж фиг.4.8 и фиг.4.19/.

За отбелязване е, че речникът на идентификаторите V се създава преди обработката на всеки израз и по този начин се осигурява еднозначност и локалност на аргументите за всяка дефиниция. Всяко дефиниция на функция създава свой собствен речник на идентификаторите, който се изтрива след завършване на обработката на дефиницията.

4.4. ПРОГРАМИРАНЕ НА HELP

В този раздел ще бъдат представени няколко примера за програмиране на HELP, засягащи основните концепции на функционалната обработка на списъци.

За удобство при разглеждането на примерите ще бъде използувано следното обозначаване на променливите:

- Ако променливата винаги е атом, тя се обозначава с имената A, B, C.
- Ако променливата винаги е списък, използуват се имената R, S, T.
- Ако променливата може да е атом или списък - с имената X, Y, Z.

Разбира се, трябва да се подчертае, че в определенията на HELP такова ограничение няма.

Поради възприетата в HELP запетайкова форма на обозначение на списъците, от особена важност е да се знае дали стойността на даден символен израз е празен списък "<>". Празен списък е просто списък, който няма елементи. Такъв списък може да се получи, например, когато се приложи функцията CDR(E) към списък, състоящ се от един елемент. В този случай при символичното представяне на списъците резултатът ще бъде атом NIL. Така че, празният списък при запетайковата форма е еквивалентен на атом NIL при символичната форма.

В HELP празният списък има свойствата на атом, т.е. ATOM(< >) има стойност *TRUE и < > може да бъде операнд в логическите изрази $P_1=P_2$ или

$P_1=P_2$. Но за разлика от другите атоми, обаче, празният списък може да бъде втори аргумент на вградената функция CONS.

Вградената в процесора основна функция NULL(X) има вида:

```
def NULL(X) := if ATOM(X) then X = < > else *FALSE ;
```

Тя има стойност *TRUE ако стойността на X е празен списък и *FALSE в противен случай. Условният израз в дефиницията на NULL(X) е необходим, защото логическият израз X =<>е недефиниран ако X не е атом, а NULL(X) трябва да бъде дефинирана за всякаква стойност на X.

При съставянето на дефиниции на функции може да се използува следният метод:

Предполага се, че за всички случаи с изключение на най-простия има готово решение. Тогава се прави решението за този най-прост случай и останалата работа се прехвърля към "готовото" решение. Но тъй като то не е направено, за неговото създаване се използува същият метод. Горното се повтаря до изчерпване на всички възможни ситуации.

За определяне дали един списък е линеен или разклонен нека бъде дефинирана функцията LINEAR(R), която има стойност *TRUE ако R е линеен списък и *FALSE ако R е разклонен списък.

Да разгледаме най-простия случай. Ако R е празен списък, той естествено няма подсписъци, което означава, че той е линеен:

```
def LINEAR(R) := if NULL(R) then *TRUE else .....
```

Сега може да се премине към по-общ случай. Ако CAR(R) не е атом, списъкът не е линеен, но ако CAR(R) е атом, това не означава, че списъкът е линеен. Във втория случай е необходимо да се провери следващия член на списъка. Тази проверка може да се извърши чрез рекурсивно извикване на същата функция:

```
def LINEAR(R) := if NULL(R) then *TRUE else
                 ATOM(CAR(R)) and LINEAR(CDR(R)) ;
```

Рекурсивното използуване на функциите е основна черта на процесора HELP. Рекурсивните функции се използуват вместо етикети и цикли в конвенционалните езици за програмиране и фактически без тях процесорът въобще не би могъл да се използува.

Необходимо е да се отбележи, че рекурсивната дефиниция трябва да съдържа условен израз, без който не може да се излезе от рекурсията. Последното е аналогично на цикъл без условен изход. Условието, при което рекурсивната функция не извиква себе си, се нарича условие за изход от рекурсия.

За функции, които обработват линейни списъци, условието за изход от рекурсията обикновено включва функцията NULL(X). Например:

```
def LAST(R) := if NULL(CDR(R)) then CAR(R) else LAST(CDR(R)) ;
```

Тази функция намира последния член на непразен списък. За отбелязване е, че LAST(< >) е недефинирана, защото CDR(< >) е недефинирана.

Прибавянето на нов член X в началото на списъка R се извършва чрез вградената функция CONS(X,R). Нека бъде съставена и функция за прибавяне на нов член X в края на списъка R.

```
def APPEND(X,R) := if NULL(R) then < X > else
                        CONS(CAR(R),APPEND(X,CDR(R))) ;
```

Условието за изход от рекурсията е отново празен списък.

За отбелязване е, че при обработка на разклонени списъци условието за изход от рекурсия много често е проверка за атом, т.е. функцията ATOM(X). Това проличава от следващите два примера.

Операторът = може да се използува за сравняване на два атома. За сравняване на два списъка е необходимо да бъде дефинирана функция. Най-простият случай е този, при който двата аргумента са атоми. При положение, че само единият аргумент е атом, стойността на функцията ще бъде *FALSE. Ако аргументите са списъци, необходимо е както CAR така и CDR от двата аргумента

да бъдат съответно равни.

```
def EQUAL(X,Y) := if ATOM(X) and ATOM(Y) then X = Y else
                  if ATOM(X) or ATOM(Y) then *FALSE else
                  EQUAL(CAR(X),CAR(Y)) and EQUAL(CDR(X),CDR(Y)) ;
```

Друга полезна функция е тази, която има стойност *TRUE когато атомът A е член на списъка R или на неговите подсписъци:

```
def AMONG(A,R) := if ATOM(R) then A = R else
                  AMONG(A,CAR(R)) or AMONG(A,CDR(R)) ;
```

Значително неудобство на функционалния процесор HELP е това, че той не разполага с памет за запомняне на резултатите от пресмятането на символните изрази. Стойностите на аргументите се въвеждат от входния низ на програмата, а стойностите на функциите се извеждат веднага след пресмятането им. Това налага за по-сложна задача да се дефинира твърде комплицирана функция, в която да бъдат включени всички пресмятания. В този случай се използува влагане на дефинирани функции в дефинициите на други функции, което значително подобрява условията за самостоятелното използуване на процесора HELP.

Като пример нека бъде съставена функцията CHECK за проверка на еднаквост на триъгълници в зависимост от три дадени елемента - страни и ъгли. Тези елементи се задават като списък, в който първо се записват еднаквите страни като атоми, а след тях - равните ъгли като списъци от образуващите ъгъла страни. Функцията ще се извиква с името си и списък на зададените елементи, напримерː

```
CHECK(<*A,*B,<*A,*B>>) ;
CHECK(<*AB,<*AB,*BC>,<*AC,*AB>>) ;
CHECK(<*A,*B,<*A,*C>>) ;
```

Стойността на функцията ще бъде *TRUE ако триъгълниците са еднакви и *FALSE в противен случай.

Нека бъдат дефинирани три спомагателни функции:

```
1/  def F(R) := CAR(R);                    - първи елемент на списъка
2/  def S(R) := CAR(CDR(R));               - втори елемент на списъка
3/  def T(R) := CAR(CDR(CDR(R)));          - трети елемент на списъка
```

Необходимо е да бъдат съставени функции на условията за равенство в различни случаи:

```
1/  def CH1(R) := ATOM(T(R));              - зададени три страни
2/  def CH2(R) := (F(R) = F(T(R)) and S(R) = S(T(R)))
                  or (F(R) = S(T(R)) and S(R) = F(T(R)));
                                           - зададени две страни и един ъгъл
3/  def CH3(R) := (F(R) = F(S(R)) or F(R) = S(S(R)))
                  and (F(R) = F(T(R)) or F(R) = S(T(R)));
                                           - зададени една страна и два ъгла
```

При това положение може да се дефинира главната функция:

```
def CHECK(R) := if ATOM(T(R)) then *TRUE else
                if not ATOM(F(R)) then *FALSE else
                if ATOM(S(R)) then CH2(R) else CH3(R);
```

Така съставената функция не прави проверка за повторение на един и същ елемент в списъка на зададените елементи. Освен това тя не проверява дали този списък съдържа три елемента. Ако елементите са по-малко от три ще се получи индикация за грешка, а ако са по-вече от три, останалите елементи ще се пренебрегнат. Освен това се оказва, че спомагателната функция CH1(R) не е необходима.

Примерът за проверка на еднаквост на триъгълници е показан на фиг.4.20.

Формалното символично диференциране на алгебрични изрази е класически пример за функционална обработка на символна списъчна информация. Структурата на алгебричните изрази е удобна за рекурсивно третиране, а правилата

за диференциране са добре определени:

$$\frac{d}{dx}(u+v) = \frac{du}{dx} + \frac{dv}{dx}$$

$$\frac{d}{dx}(u-v) = \frac{du}{dx} - \frac{dv}{dx}$$

$$\frac{d}{dx}(u*v) = \frac{du}{dx} * v + \frac{dv}{dx} * u$$

$$\frac{d}{dx}(u/v) = 1 / v / v * (\frac{du}{dx} * v - \frac{dv}{dx} * u)$$

За представяне на алгебричните изрази в удобна за пресмятане форма най-често се използува правата полска запис с предшествуване. При двуместните операции двата операнда и операторът образуват триплекс, който може да се представи чрез списък, първи член на който е операторът, втори член - първият операнд и трети член - вторият операнд. Например, на алгебричния израз A.X + B ще съответствува списъкът (*ADD,(*MULT,*A,*X),*B). Вместо операторите +, -, . и / се използуват съответно обозначенията ADD, SUB, MULT и DIV. Може да се отбележи, че е възможно използуването на ADD, SUB, MULT и DIV и за обозначаване на операнди, тъй като операторите и операндите заемат различни места в триплексите. Операторите заемат винаги първо място, а операндите - винаги второ и трето. Например, алгебричният израз MULT+ADD ще се представи чрез списъка (*ADD,*MULT,*ADD).

Главната функция за диференциране е DERIV(X,A), което трябва да се разбира "диференциране на X по отношение на A". Съставянето на дефиницията й ще започне с определяне на решение за най-простия случай и насочване към спомагателни функции при по-сложните случаи.

```
def DERIV(X,A) := if ATOM(X) then (if X=A then *1 else *0) else
                  if CAR(X) = *ADD then DADD(CDR(X),A) else
                  if CAR(X) = *SUB then DSUB(CDR(X),A) else
```

```
        if CAR(X) = *MULT then DMULT(CDR(X),A) else
        if CAR(X) = *DIV then DDIV(CDR(X),A) else ERROR( );
```

Производната на сума е сума от производните на събираемите, така че функцията DADD ще бъде дефинирана по следния начин:

```
def DADD(R,A) := <*ADD,DERIV(CAR(R),A),DERIV(SECOND(R),A)>;
```

където функцията SECOND има за стойност втория операнд от триплекса:

```
def SECOND(R) := CAR(CDR(R));
```

Но така дефинираната функция DADD притежава неудобството да оставя излишни нули /*0/, явяващи се като събираеми в производната. Проверката и избягването на тези излишни нули става за сметка на известно усложняване на функцията DADD и включването на спомагателната функция ISZERO.

```
def ISZERO(X) := if ATOM(X) then X = *0 else *FALSE;
```

В този случай DADD ще се дефинира така:

```
def DADD(R,A) := if ISZERO(DERIV(CAR(R),A)) then
                 DERIV(SECOND(R),A) else
                 if ISZERO(DERIV(SECOND(R),A)) then
                 DERIV(CAR(R),A) else
                 <*ADD,DERIV(CAR(R),A),DERIV(SECOND(R),A)>;
```

Производната на разлика е разликата на производната на първия операнд и производната на втория операнд. Ако производната на първия операнд е нула, тази стойност трябва да се запази поради това, че не е предвидена обработка на едноместен минусов оператор. Дефиницията на DSUB ще изглежда така:

```
def DSUB(R,A) := if ISZERO(DERIV(SECOND(R),A)) then
                 DERIV(CAR(R),A) else
                 <*SUB,DERIV(CAR(R),A),DERIV(SECOND(R),A)>;
```

При извличане на производната на произведение трябва да се имат предвид и излишните единици /*1/, явяващи се като множители в производната:

```
def ISONE(X) := if ATOM(X) then X = *1 else *FALSE;
def SUBPROD(X,Y,A) := if ISONE(DERIV(X,A)) then Y else
                      <*MULT,DERIV(X,A),Y>;
```

В крайна сметка функцията DMULT ще има вида:

```
def DMULT(R,A) := if ISZERO(DERIV(CAR(R),A)) then
                  (if ISZERO(DERIV(SECOND(R),A)) then *0 else
                   SUBPROD(SECOND(R),CAR(R),A)) else
                  if ISZERO(DERIV(SECOND(R),A)) then
                   SUBPROD(CAR(R),SECOND(R),A) else
        <*ADD,SUBPROD(CAR(R),SECOND(R),A),SUBPROD(SECOND(R),CAR(R),A)>;
```

За опростяване на записа, функцията DDIV използува и спомагателната функция MPL.

```
def MPL(R) := <*DIV,*1,<*MULT,SECOND(R),SECOND(R)>>;
```

Дефиницията на функцията DDIV има окончателен вид:

```
def DDIV(R,A) := if ISZERO(DERIV(SECOND(R),A)) then
                 (if ISZERO(DERIV(CAR(R),A)) then *0 else
          <*MULT,MPL(R),SUBPROD(CAR(R),SECOND(R),A)>) else
                 <*MULT,MPL(R),<*SUB,SUBPROD(CAR(R),SECOND(R),A),
                                 SUBPROD(SECOND(R),CAR(R),A)>>;
```

Ако искаме да получим първата производна на израза X по отношение на A, необходимо е да извикаме за изпълнение функцията

```
     DERIV(X,A);
```

за втората производна -

```
     DERIV(DERIV(X,A),B);
```

за третата производна -

```
     DERIV(DERIV(DERIV(X,A),B),C);
```

и т.н.

Примерът за символично диференциране на алгебрични изрази е показан в цялостен вид на фиг.4.21.

4.5. СРАВНЕНИЕ НА HELP С LISP

В този раздел ще бъде направено кратко сравнение на основните свойства на функционалните процесори за обработка на еднопосочни символни списъци HELP и LISP. LISP е създаден от проф. Джон МакКарти през 1960 година и намира значително по-голямо приложение от подобните процесори, предназначени за употреба при моделиране на изкуствен интелект и символичната математика.

LISP	HELP
LISP има голямо теоретично значение. От математична гледна точка той представлява стройна математическа структура, базираща се на пет първични операции - CAR, CDR, CONS, ATOM, EQ.	При HELP първичните операции са представени чрез вградените функции CAR, CDR, CONS, ATOM и NULL. За сравняване на атоми HELP използва синтаксичните конструкции $P_1=P_2$ или $P_1=P_2$, а за сравняване на списъци се налага създаването на непървична функция. Функцията NULL е необходима при HELP поради възприетата запетайкова форма на записване на списъците.
Основно достойнство на LISP с голямо теоретично значение е премахването на формалната разлика между данни и процедури, което е безусловно необходимо за система за програ-	При HELP е възприета по-удобна конвенционална форма за представяне на процедурите, което прави системата достъпна за по-широк кръг потребители, но външното представя-

миране, предназначена за приложение при моделиране на изкуствен интелект. Външното представяне на функциите в LISP е чрез списъчни структури. Тези функции обработват данни, представени също като списъчни структури. Поради това, че както функциите, така и данните се записват в една и съща форма, става възможно да се дефинира процедура за пресмятането на функция аналогично на процедура за пресмятане на данни, т.е. функциите обработват по един и същ начин данни и други функции.

Представяне на основните синтаксически типове в LISP:

(QUOTE,ATOM)

(LIST,e_1,e_2, ... ,e_n)

(I,e_1,e_2, ... ,e_n)

(EQ,e_1,e_2)

(COND,(e_{11},e_{21}),
(e_{12},e_{22}), ...
.....
(e_{1n},e_{2n}),(T,e_{nn})

В основата на програмирането на LISP лежи рекурсивното използуване на функциите. Освен петте основни функции на функциите се различава от това на данните. В паметта на процесора, обаче, както данните така и функциите се представят чрез списъчни структури.

Представяне на съответните основни синтаксически типове в HELP:

*ATOM

$\langle e_1,e_2,e_3, \dots ,e_n \rangle$

I(e_1,e_2, ... ,e_n)

e_1=e_2

if e_{11} then e_{21} else
if e_{12} then e_{22} else ...
.....
if e_{1n} then e_{2n} else e_{nn}

Програмирането на HELP също се базира на рекурсивното използуване на функциите. За разлика от LISP

ции, всички останали функции трябва да бъдат дефинирани. Дефинирането при LISP има три разновидности:

1/ дефиниране на функция за еднократна нерекурсивна употреба

2/ дефиниране на функция за еднократна рекурсивна употреба

3/ дефиниране на функция за многократна употреба.

в HELP има само един метод за дефиниране на неосновна функция, съответстващ на третата разновидност при LISP.

На фиг.4.20 е показана програмата от раздел 4.4 за проверка на еднаквост на триъгълници написана на HELP, а на фиг.4.22 - същата програма написана на LISP.

На фиг.4.21 е показана програмата от раздел 4.4 за символично диференциране на алгебрични изрази написана на HELP, а на фиг.4.23 - същата програма написана на LISP.

```
*THIS*IS*A*CONSTANT
*1975
*ИЗЧИСЛИТЕЛНА*ТЕХНИКА
```

а/ примери на константи

```
THIS*IS*A*VARIABLE
YEAR
X
СПЕЦИАЛНОСТ
```

б/ примери на променливи

Фиг.4.1. Примери на константи и променливи в HELP

$P ::= A \mid \langle E_1, E_2, \ldots, E_n \rangle \mid (E) \mid I(E_1, E_2, \ldots, E_n) \mid I$

$LE ::= \text{not } LE \mid P_1 = P_2 \mid P_1 = P_2 \mid P$

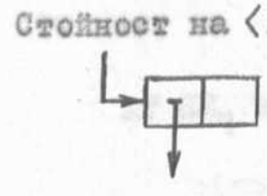
$SE ::= LE \text{ and } SE \mid LE \text{ or } SE \mid LE$

$E ::= \text{if } SE_1 \text{ then } SE_2 \text{ else } E \mid SE$

Обозначения:

$\langle\ \rangle$	-	списъчни скоби
A	-	атом
I	-	идентификатор
P	-	първичен елемент
LE	-	логически израз
SE	-	прост израз
E	-	израз

Фиг.4.2. Синтаксис на HELP



Стойност на E =
Стойност на (E)

Фиг.4.3. Стойности на изразите ⟨E⟩, (E) и E

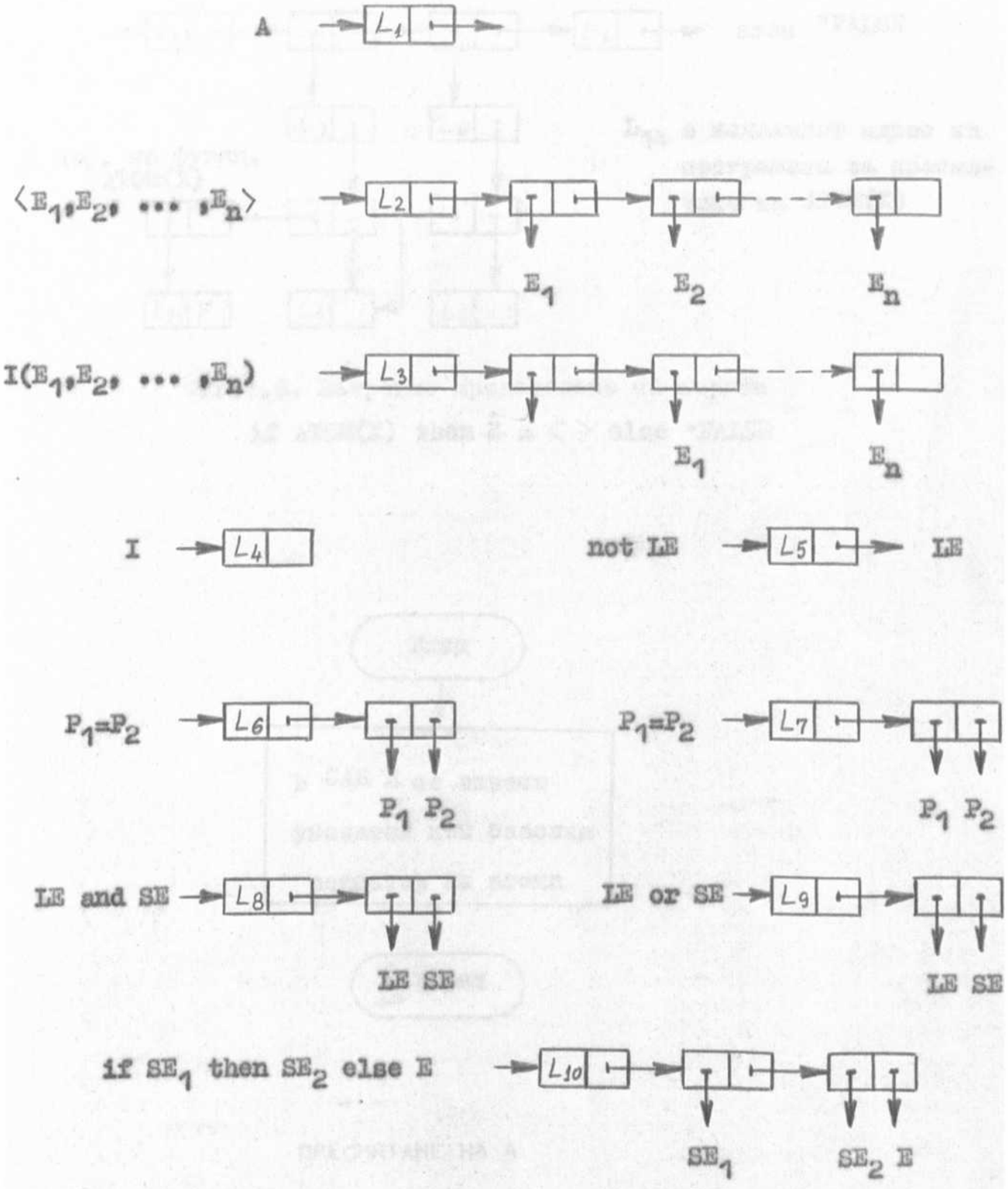
```
def SECOND(R) := CAR(CDR(R));
```

а/ дефиниране на функция

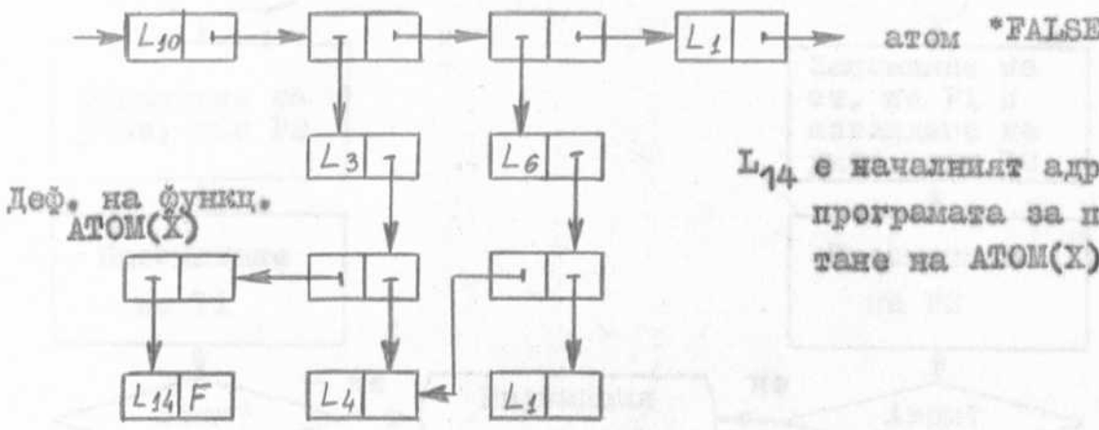
```
SECOND(⟨*A,*B,*C⟩);      - извикване
B;                       - резултат
```

б/ изпълнение на функция

Фиг.4.4. Използуване на функции в HELP



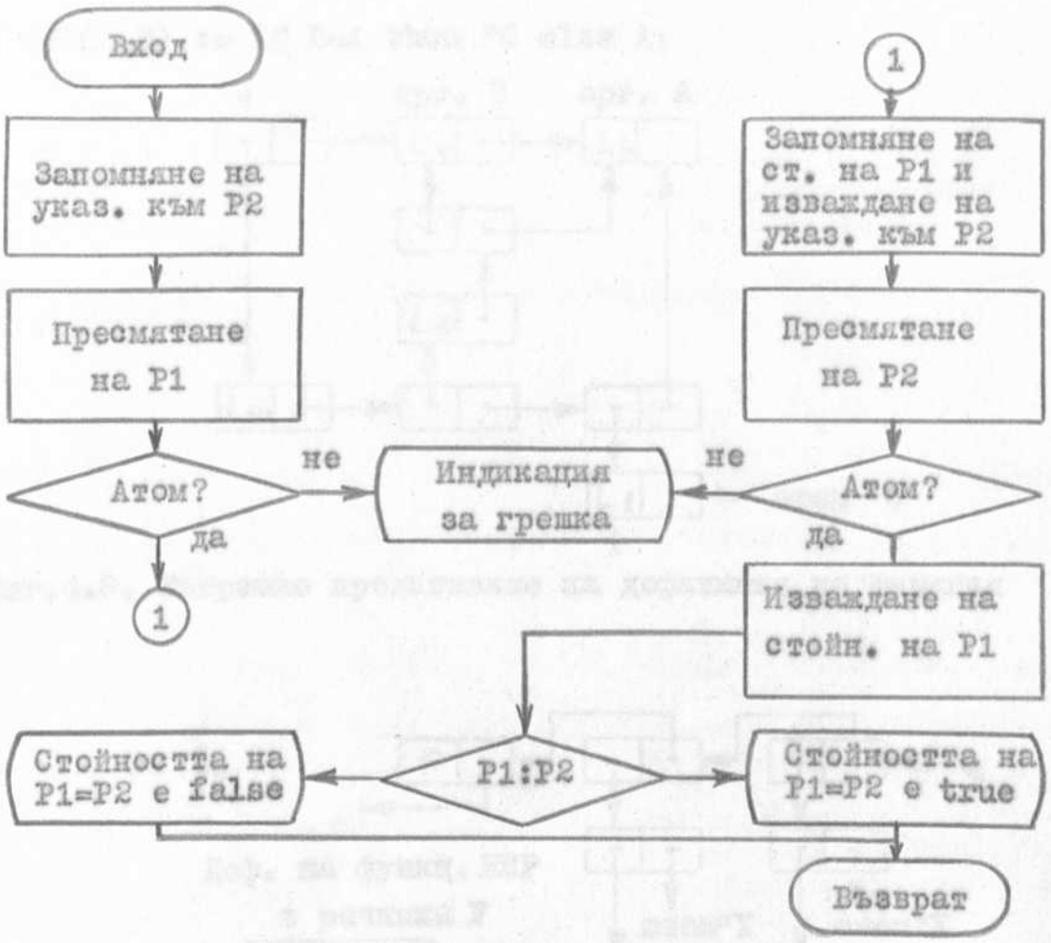
Фиг.4.5. Списъчни структури на изразите в HELP



Фиг.4.6. Вътрешно представяне на израза
if ATOM(X) then X = < > else *FALSE


```
L1.                    ПРЕСМЯТАНЕ НА А
,
          CAR R = CDR E.          ЗАНАСЯНЕ НА РЕЗУЛТАТА
          TO CAR T.               ВЪЗВРАТ
```

а/ Подпрограма за пресмятане на А.

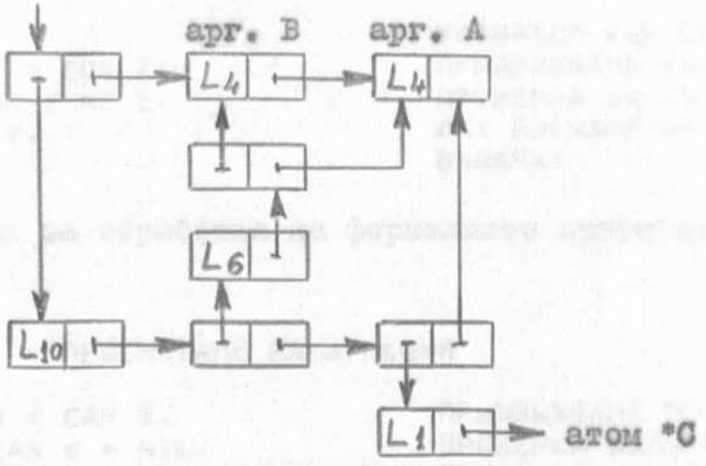



```
L6.                    ПРЕСМЯТАНЕ НА P1 = P2
.
          E = CDR E.          ПРИДВИЖВАНЕ ПО СТРУКТУРАТА
          PUSH DOWN S, CAR S = CDR E.   ЗАПОМНЯНЕ НА УКАЗАТБЛЯ КЪМ P2
          PUSH DOWN T, CAR T = L61.     ЗАПОМНЯНЕ НА АДРЕСА ЗА ВЪЗВРАТ
          E = CAR E, TO CAR E.          ПРЕСМЯТАНЕ НА P1
L61,      E = CAR R.            ПРОВЕРКА НА СТОЙНОСТТА НА P1
          TO FAIL IF AF E = 00.         ГРЕШКА АКО НЕ Е АТОМ
          PUSH DOWN R.          ЗАПОМНЯНЕ НА СТОЙНОСТТА НА P1
          E = CAR S, POP UP S.          ВЪЗОБНОВЯВАНЕ НА УКАЗАТЕЛЯ КЪМ P2
          CAR T = L62, TO CAR E.        АДР. ЗА ВЪЗВРАТ И ПРЕСМЯТАНЕ НА P2
L62,      E = CAR R.            ПРОВЕРКА НА СТОЙНОСТТА НА P2
          TO FAIL IF AF E = 00.         ГРЕШКА АКО НЕ Е АТОМ
          POP UP R.             ИЗВЛИЧАНЕ НА СТОЙНОСТТА НА P1
          TO L63 IF CAR R = E.          ПРЕХОД ПРИ РАВЕНСТВО
          CAR R = FALSE, TO L64.        РЕЗУЛТАТЪТ Е *FALSE
L63,      CAR R = TRUE.         РЕЗУЛТАТЪТ Е *TRUE
L64,      POP UP T, TO CAR T.           ВЪЗВРАТ
```

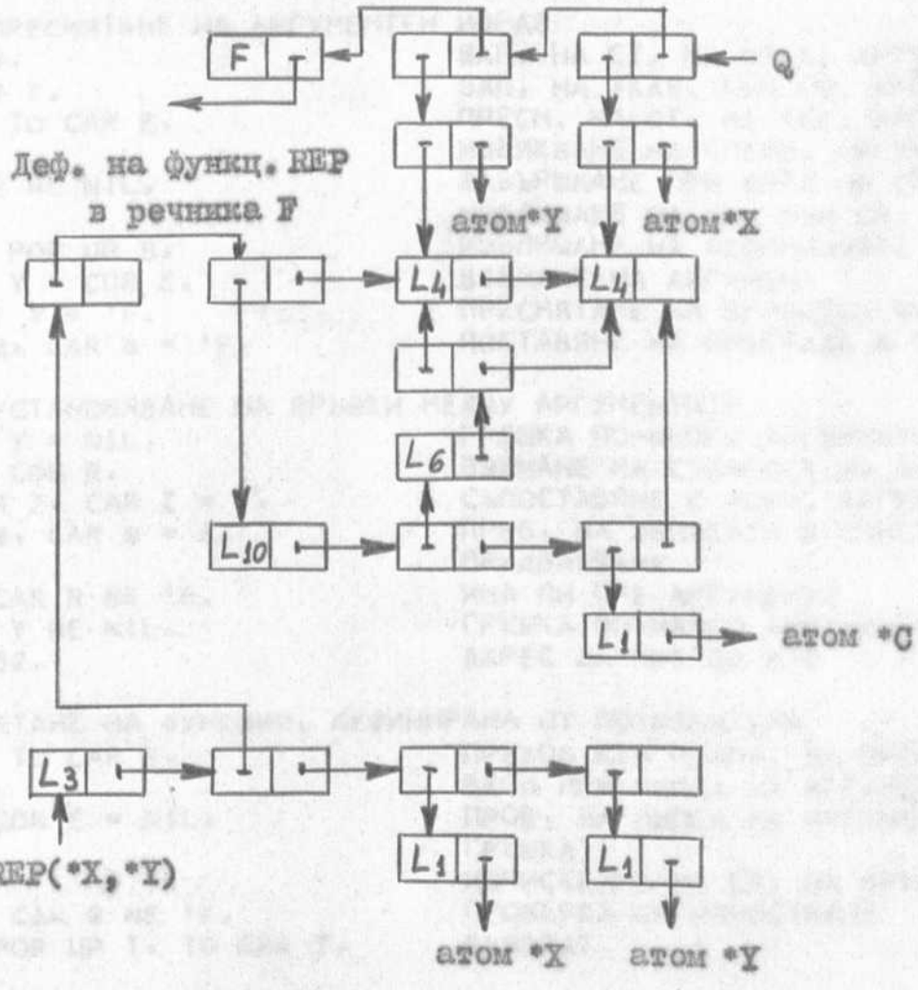
б/ Подпрограма за пресмятане на P1=P2.

Фиг.4.7. Подпрограми за пресмятане на синтаксическите единици A и P1=P2

```
def REP(A,B) := if B=A then *C else A;
```



Фиг.4.8. Вътрешно представяне на дефиниция на функция



Фиг.4.9. Дефиниция и извикване на функция и списък на връзките

```
L4.                    ОБРАБОТВАНЕ НА ФОРМАЛНИТЕ АРГУМЕНТИ
.
          Z = @.                                УКАЗАТЕЛ КЪМ СПИСЪКА НА ВРЪЗКИТЕ
L41,      Y = CAR Z, Z = CDR Z.                 ПРИДВИЖВАНЕ КЪМ СЛЕДВАЩИЯ ЕЛЕМЕНТ
          TO L41 IF CAR Y NE E.                 ПРОВЕРКА ЗА СЪОТВЕТСТВИЕ НА АРГ.
          CAR R = CDR Y.                        ДА, ВЗЕМАНЕ НА ФАКТИЧЕСКА СТОЙНОСТ
          TO CAR T.                             ВЪЗВРАТ
```

Фиг.4.10. Програма за обработка на формалните аргументи

```
L3.                       ПРЕСМЯТАНЕ НА ФУНКЦИЯ
.
          F = CDR E, W = CAR E.                 ПРИДВИЖВАНЕ ПО ФУНКЦИЯТА
          TO FAIL IF CAR W = NIL.               ПРОВЕРКА ДАЛИ Е ДЕФИНИРАНА
          F = CDR E, TO L331 IF E = NIL.        ПРИДВИЖВАНЕ ПО СПИСЪКА НА АРГ.
          PUSH DOWN S, CAR S = W.               ЗАПОМНЯНЕ НА ДЕФИНИЦИЯТА
          CAR R = 'F.                           ПОСТАВЯНЕ НА ПРЕГРАДА В R
          PUSH DOWN S.                          ЗАПОМНЯНЕ НА УКАЗ. КЪМ СЛ. ЕЛЕМЕНТ
          PUSH DOWN T, CAR T = L311.            ЗАПОМНЯНЕ НА АДРЕС ЗА ВЪЗВРАТ
.
L31.              ПРЕСМЯТАНЕ НА АРГУМЕНТЕН ИЗРАЗ
          PUSH DOWN R.                          ЗАП. НА СТ. НА ПРЕД. АРГУМЕНТ
          CAR S = CDR F.                        ЗАП. НА УКАЗ. КЪМ СЛ. АРГУМЕНТ
          F = CAR E, TO CAR E.                  ПРЕСМ. НА СТ. НА ТЕК. АРГУМЕНТ
L311,     E = CAR S.                            ИЗВИКВАНЕ НА СЛЕДВ. АРГУМЕНТ
          TO L31 IF E NE NIL.                   ЗАВЪРШВАНЕ ПРИ КРАЙ НА СПИСЪКА
          POP UP S.                             ИЗВЛИЧАНЕ НА УК. КЪМ СЛ. ЕЛЕМЕНТ
          W = CAR S, POP UP S.                  ИЗВЛИЧАНЕ НА ДЕФИНИЦИЯТА
          F = CAR W. Y = CDR E.                 ВЗЕМАНЕ НА АРГУМЕНТ
          TO CAR E IF Y = 'F.                   ПРЕСМЯТАНЕ НА ВГРАДЕНА ФУНКЦИЯ
          PUSH DOWN @, CAR @ = 'F.              ПОСТАВЯНЕ НА ПРЕГРАДА В СП. @
.
L32.              УСТАНОВЯВАНЕ НА ВРЪЗКИ МЕЖДУ АРГУМЕНТИТЕ
          TO FAIL IF Y = NIL.                   ГРЕШКА ПО-МНОГО АРГУМЕНТИ
          Z = R, R = CDR R.                     ВЗЕМАНЕ НА СТОЙНОСТ НА АРГУМЕНТ
          CDR Z = CAR Z, CAR Z = Y.             СЪПОСТАВЯНЕ С ФОРМ. АРГУМЕНТ
          PUSH DOWN @, CAR @ = Z.               ПРИБ. НА ДВОЙКАТА В СПИСЪКА @
          Y = CDR Y.                            ПРИДВИЖВАНЕ
          TO L32 IF CAR R NE 'F.                ИМА ЛИ ОЩЕ АРГУМЕНТИ
          TO FAIL IF Y NE NIL.                  ГРЕШКА ПО-МАЛКО АРГУМЕНТИ
          CAR T = L332.                         АДРЕС ЗА ПРЕХОД В Т
.
L33.          ПРЕСМЯТАНЕ НА ФУНКЦИЯ, ДЕФИНИРАНА ОТ ПОТРЕБИТЕЛЯ
          F = CAR E, TO CAR E.                  ПРЕХОД КЪМ ПОДПР. ЗА ПРЕСМЯТАНЕ
L331,     F = CAR W.                            ВХОД ПРИ ЛИПСА НА АРГУМЕНТИ
          TO L33 IF CDR E = NIL.                ПРОВ. ЗА ЛИПСА НА АРГУМЕНТИ
          TO FAIL.                              ГРЕШКА
L332,     POP UP @.                             ИЗЧИСТВАНЕ НА СП. НА ВРЪЗКИТЕ @
          TO L332 IF CAR @ NE 'F.               ПРОВЕРКА ЗА ИЗЧИСТВАНЕ
          POP UP @, POP UP T, TO CAR T.         ВЪЗВРАТ
```

Фиг.4.11. Програма за пресмятане на функция

```
L13.                ПРЕСМЯТАНЕ НА ВГРАДЕНАТА ФУНКЦИЯ CONS
.
        Z = CAR R, POP UP R.            ВЗЕМАНЕ НА ВТОРИЯ АРГ. НА CONS
        TO FAIL IF CAR R = 'F.          НЕПРАВИЛЕН БРОЙ АРГУМЕНТИ
        E = R, R = CDR R.               ВЗЕМАНЕ НА ПЪРВИЯ АРГ. В CAR ПОЛЕ
        TO FAIL IF CAR R NE 'F.         НЕПРАВИЛЕН БРОЙ АРГУМЕНТИ
        CDR E = Z.                      ПОСТАВЯНЕ НА ВТОРИЯ АРГ. В CDR ПОЛЕ
        CAR R = E.                      РЕЗУЛТАТА В СТЕКА НА РЕЗУЛТАТИТЕ R
        POP UP T.                       ИЗВ. НА ИЗЛИШНИЯ АДРЕС ЗА ВЪЗВРАТ
        TO CAR T IF Z = NIL.            ВЪЗВРАТ АКО ВТОРИЯ ЕЛЕМЕНТ Е
                                            ЛЕГАЛЕН (Т.Е. СПИСЪК)
        TO FAIL.                        ГРЕШКА
```

Фиг.4.12. Реализация на вградената функция CONS

```
        ENTRY FINPUT.                   ВХОД В ПРОГРАМАТА
        TO SKIP IF (CHAR) NE NIL.       ИМА ЛИ ПРЕДВАРИТЕЛНО ПРОЧЕТЕН ЗНАК
        USE NEXTCH.                     НЕ, ДА СЕ ПРОЧЕТЕ
SKIP,   TO ATOM IF (CHAR) = '*.         АТОМИТЕ ЗАПОЧВАТ С *
        TO IDEN IF (CHAR) = LETORDIG.   ИДЕНТИФИКАТОР
        TO MULT IF (CHAR) = '.          СИМВОЛ ОТ НЯКОЛКО ЗНАКА
        TO RPAR IF (CHAR) NE '(.
        Z = 07, TO NOCH.                ЛЯВА СКОБА
RPAR,   TO COMM IF (CHAR) NE ').
        Z = 08, TO NOCH.                ДЯСНА СКОБА
COMM,   TO EQUL IF (CHAR) NE ',.
        Z = 11, TO NOCH.                ЗАПЕТАЙКА
EQUL,   TO ERRC IF (CHAR) NE '=.
        Z = 06, TO NOCH.                РАВЕНСТВО
ERRC,   Z = 00.                         НЕРАЗПОЗНАТ СИМВОЛ
NOCH,   (CHAR) = NIL. EXIT FINPUT.      НЕ СЕ ЧЕТЕ ПРЕДВАРИТЕЛНО: ИЗХОД
CHAR,   ELEMENT 00 00 00 NIL.           БУФЕР
.
ATOM,   USE NEXTCH, USE READST.         ЧЕТЕНЕ НА ЗНАЦИ ОТ АТОМА
        D = 'A, M = 'X.                 ОБРЪЩЕНИЕ КЪМ РЕЧНИКА НА АТОМИТЕ
        USE LOOKUP.
        TO BAKA IF CDR Y NE NIL.        НАМЕРЕН ЛИ Е ТОЗИ АТОМ, ДА
        Z = NEW ELEMENT.                НЕ, ПРИКАЧИ НОВ РЕГИСТЪР
        CDR Y = Z, AF Z = 01.           ЗАРЕДИ ФЛАГ ЗА АТОМ
        X = NIL, CAR Z = CAR X.         ВКЛЮЧИ РЕГ. КЪМ ВЕРИГАТА НА РЕГ.
        CAR X = Z.                      ЗА КРАЙ НА ВЕРИГАТА
        CDR Z = S.                      ОБРАТЕН УКАЗАТЕЛ
BAKA,   Z = 01, EXIT FINPUT.            ВРЪЩАНЕ НА ИНДИКАТОР ЗА АТОМ
.
IDEN,   USE READST.                     ЧЕТЕНЕ НА НИЗ
        D = BASIC, M = NIL.             ОБРЪЩЕНИЕ КЪМ РЕЧНИКА НА ЗАП. ДУМИ
        USE LOOKUP.
        TO BSYM IF Y NE NIL.            НАМЕРЕНА ЛИ Е ТАЗИ ЗАП. ДУМА, ДА
        TO FUNC IF (CHAR) = '(.         НЕ, ФУНКЦИЯ ЛИ Е, ДА
        D = 'V, M = 'X.                 НЕ, ОБРЪЩЕНИЕ КЪМ РЕЧНИКА НА ПРОМ.
        USE LOOKUP.
        Z = 02, EXIT FINPUT.            ВРЪЩАНЕ НА КОД НА ПРОМЕНЛИВА
```

```
BSYM,     Z = CDR Y. EXIT FINPUT.          КОД ЗА ПЕЧАТ НА ДУМА
FUNC,     D = 'F, M = 'X.                  ОБРЪЩЕНИЕ КЪМ РЕЧНИКА НА ФУНКЦИИТЕ
          USE LOOKUP.
          TO BAKE IF CDR Y NE NIL.         ПРИ СТАРА ФУНКЦИЯ - НИЩО НЕ СЕ ИЗВ.
          Z = NEW ELEMENT.                 ИНАЧЕ, ЕЛЕМЕНТ ЗА ДЕФИНИРАНЕ СЕ
          CDR Y = Z.                           ПРИКАЧА В РЕЧНИКА
          CAR Z = NIL, CDR Z = S.
BAKF,     Z = 03, TO NOCH.                 ВРЪЩАНЕ НА ФУНКЦИЯ
,
MULT,     USE NEXTCH.                      ПРОВЕРКА НА ЗНАКА СЛЕД (')
          TO RBKT IF (CHAR) NE '(.
          Z = 04, TO NOCH.                 ЛЯВА СПИСЪЧНА СКОБА
RBKT,     TO SEMI IF (CHAR) NE ').
          Z = 05, TO NOCH.                 ДЯСНА СПИСЪЧНА СКОБА
SEMI,     TO ASGN IF (CHAR) NE ':.
          Z = 09, TO NOCH.                 ДВЕ ТОЧКИ
ASGN,     TO ERRC IF (CHAR) NE '=.
          Z = 19, TO NOCH.                 ПРИСВОЯВАНЕ
```

Фиг.4.13. Програма за определяне номера на синтаксическите елементи

```
          TO IDEN IF (CHAR) = 'A.
          TO IDEN IF (CHAR) = 'B.
          TO IDEN IF (CHAR) = 'C.
                  -- . --
                  -- . --
                  -- . --
          TO IDEN IF (CHAR) = '8.
          TO IDEN IF (CHAR) = '9.
```

а/ Произволни вътрешни кодове

```
          TO TEMP IF (CHAR) = 00.
          TO IDEN IF (CHAR) LT 37.
TEMP,
```

б/ Последователни вътрешни кодове /от 1 до 37/

Фиг.4.14. Конкретни реализации на оператора

```
          TO IDFN IF (CHAR) = LETORDIG.
```

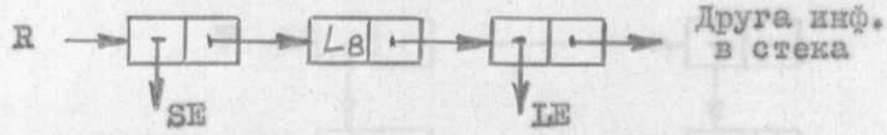
```
        ENTRY READST.                       ВХОД В ПРОГРАМАТА
        C = CHAR.                           ШПАЦИИТЕ ИМАТ ЗНАЧЕНИЕ
        Z = 'S, TO LOOP1.
STAR,   CAR Z = 1.                          ЗАМЕСТВАНЕ НА ЗВЕЗДИЧКИТЕ С ШПАЦИИ
LOOP,   USE NEXTCH.                         ЧЕТЕНЕ НА СЛЕДВ. ЗНАЧИМ ЗНАК
LOOP1,  CDR Z = NEW ELEMENT.                ПРИКАЧАНЕ НА НОВ ЕЛЕМЕНТ
        Z = CDR Z, CAR Z = (CHAR).          ВЗЕМАНЕ НА ЗНАКА
        TO LOOP IF (CHAR) = LETORDIG.       ПРОДЪЛЖАВАНЕ ПРИ БУКВА ИЛИ ЦИФРА
        TO STAR IF (CHAR) = '*.             ЗВЕЗДИЧКАТА Е ПОЗВОЛЕНА СЪЩО
        CAR Z = NIL, CDR Z = NIL.           ИНАЧЕ ИЗТРИВАНЕ НА ПОСЛЕДНИЯ ЕЛЕМ.
        C = ' .                             ШПАЦИИТЕ ВЕЧЕ НЯМАТ ЗНАЧЕНИЕ
        TO GOTCH IF (CHAR) NE ' .           ПРОВЕРКА ДАЛИ ЗНАКА ИМА ЗНАЧЕНИЕ
        USE NEXTCH.                         ВЗЕМАНЕ НА НОВ ЗНАК
GOTCH,  EXIT READST.                        ИЗХОД ОТ ПРОГРАМАТА

        ENTRY NEXTCH.                       ВХОД В ПРОГРАМАТА
GETCH,  (CHAR) = LINE BUFFER.               ВЪЗПР. НА СЛЕДВАЩИЯ ЗНАК
CKCHR,  TO GETCH IF (CHAR) = C.             ПРОВЕРКА ДАЛИ Е ЗНАЧИМ
        TO NEWL IF (CHAR) = NIL.            КЪМ НОВ РЕД АКО ЗНАКА Е CRLF
        EXIT NEXTCH.                        ИЗХОД ОТ ПРОГРАМАТА
NEWL,   READ INPUT.                         ЧЕТЕНЕ НА НОВ РЕД
        TO QUIT IF (INPUT) NE 00.           ИЗХОД ПРИ ГРЕШКА
        WRITE OUTPUT.                       ИЗВЕЖДАНЕ НА РЕД
        (CHAR) = ' , TO CKCHR.              CRLF ДЕЙСТВУВА КАТО ШПАЦИЯ
```

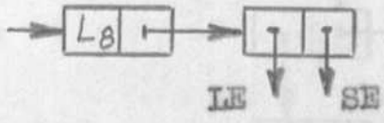
Фиг.4.15. Програми за въвеждане на атоми, функции и запазени думи

```
RSE,    USE FINPUT.                         ЧЕТЕНЕ НА СЛЕДВАЩ ЕЛЕМЕНТ
        PUSH DOWN T, CAR T = SE1.           ЗАРЕЖДАНЕ НА АДРЕС ЗА ВРЪЩАНЕ И
        TO LE.                                   ПРОВЕРКА ЗА ЛОГИЧЕСКИ ИЗРАЗ
SE1,    TO SE2 IF Z = 16.                   СЛЕДВА ЛИ 'AND', ДА
        TO SE5 IF Z NE 17.                  НЕ, СЛЕДВА ЛИ 'OR', НЕ
        PUSH DOWN R, CAR R = L9.            ДА, ТОВА Е LE 'OR' SE КОНСТР.
        TO SE3.
SE2,    PUSH DOWN R, CAR R = L8.            ТОВА Е LE 'AND' SE КОНСТР.
SE3,    PUSH DOWN R.                        ЗАПАЗВАНЕ НА ТЕКУЩИЯ СТЕК
        CAR T = SE4, TO RSE.                ПРОВЕРКА ЗА ПРОСТ ИЗРАЗ
SE4,    W = CDR R, X = CDR W.               РАЗМЕСТВАНЕ В СТЕКА
        CDR R = CDR X.                      ИЗВЛИЧАНЕ НА ДВА ЕЛЕМЕНТА
        CDR X = CAR R.                      ПРОСТ ИЗРАЗ
        CAR R = W.                          ОСТАВЯНЕ НА РЕЗУЛТАТА В СТЕКА
SE5,    POP UP T, TO CAR T.                 ВЪЗВРАТ
```

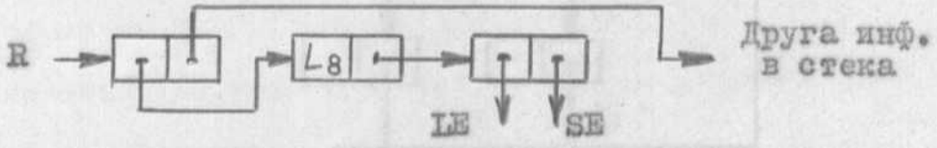
Фиг.4.16. Подпрограма за синтаксически анализ на прост израз



а/ стекът R след разпознаване на LE and SE



б/ стандартна форма на компилиран прост израз

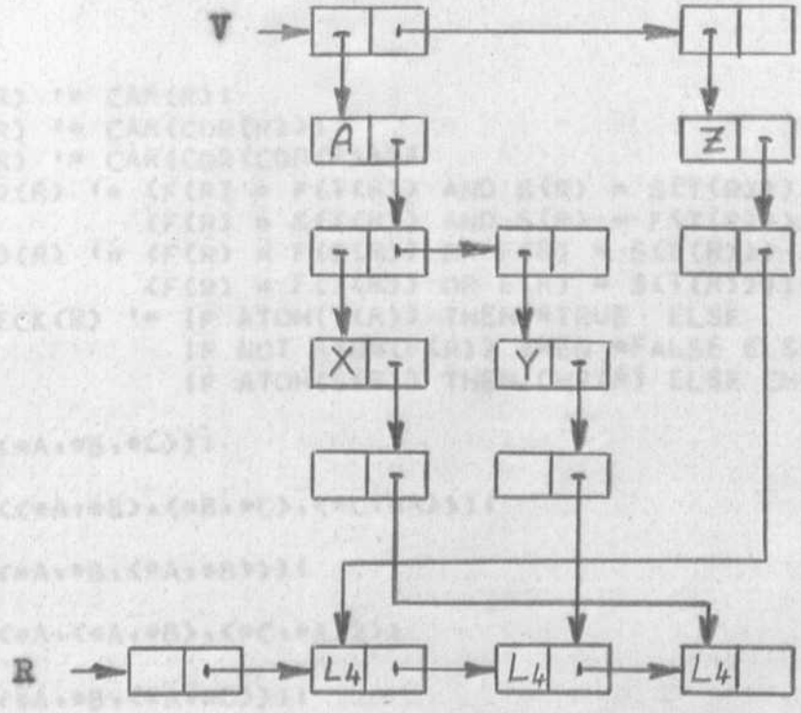


в/ стекът R след разместването

Фиг.4.17. Конструиране на списъка на стандартното представяне на прост израз

```
NEWR,    R = NEW ELEMENT.                    ПОРЕДЕН ЗНАК
NEXTE,   CAR R = NIL, CDR R = NIL.           ЗАПОЧВАНЕ НА НОВ ИЗРАЗ
         CAR T = EVAL, CDR T = NIL.
         V = NIL.                            НУЛИРАНЕ НА РЕЧНИКА НА ПРОМЕНЛИВИТЕ
         USE FINPUT.                         ПЪРВИ СИНТАКТИЧЕСКИ ЕЛЕМЕНТ
         TO EX IF Z NE 12.                   АКО НЕ Е 'DEF', ЧЕТЕНЕ НА ИЗРАЗ
         USE FINPUT, P = CDR Y.              ИНАЧЕ, ИМЕ НА ФУНКЦИЯ
         TO ERROR IF Z NE 03.                ГРЕШКА НЕ Е ФУНКЦИЯ
         TO ERROR IF CAR Y NE NIL.           ГРЕШКА ДВОЙНО ДЕФИНИРАНЕ
         USE FINPUT.                         НОВ СИНТАКТИЧЕСКИ ЕЛЕМЕНТ
BVARS,   TO ERROR IF Z NE 02.                ГРЕШКА НЕ Е ИДЕНТИФИКАТОР
         TO ERROR IF CDR Y NE NIL.           КОНТРОЛ НА НОВА ПРОМЕНЛИВА
         CAR R = L4, PUSH DOWN R.            ЗАРЕЖДАНЕ НА АДРЕСА НА ИНТЕРПР. ПР.
         CDR Y = CDR R.                      СТОЙНОСТ НА АРГУМЕНТ
         USE FINPUT.                         СЛЕДВАЩ СИНТАКТИЧЕСКИ ЕЛЕМЕНТ
         TO FIXDEF IF Z = 08.                ТРЯБВА ДА Е ДЯСНА СКОБА
         TO ERROR IF Z NE 11.                ИЛИ ЗАПЕТАЙКА, ИНАЧЕ ГРЕШКА
         USE FINPUT, TO BVARS.               СЛЕДВАЩ АРГУМЕНТ
FIXDEF,  USE FINPUT.                         ВЪЗПРИЕМАНЕ НА :=
         CAR T = FIXDF1.                     АДРЕС ЗА ВРЪЩАНЕ И
         TO REX IF Z = 19.                       ПРЕХОД ЗА ЗАВЪРШВАНЕ НА ИЗРАЗА
         TO ERROR.                           ИНАЧЕ, ГРЕШКА
FIXDF1,  TO ERROR IF Z NE 09.                ПРОВЕРКА ЗА ТОЧКА И ЗАПЕТАЙКА
         CAR P = R, TO NEWR.                 ЗАПАЗВАНЕ НА ФУНКЦИЯТА
```

Фиг.4.18. Дефиниране на функция



Фиг.4.19. Речник на идентификаторите след разпознаване на
def G(AX,AY,Z) :=

```
DEF F(R) '= CAR(R);
DEF S(R) '= CAR(CDR(R));
DEF T(R) '= CAR(CDR(CDR(R)));
DEF CH2(R) '= (F(R) = F(T(R)) AND S(R) = S(T(R))) OR
              (F(R) = S(T(R)) AND S(R) = F(T(R)));
DEF CH3(R) '= (F(R) = F(S(R)) OR F(R) = S(S(R))) AND
              (F(R) = F(T(R)) OR F(R) = S(T(R)));
DEF CHECK(R) '= IF ATOM(T(R)) THEN *TRUE  ELSE
                IF NOT ATOM(F(R)) THEN *FALSE ELSE
                IF ATOM(S(R)) THEN CH2(R) ELSE CH3(R) ;

CHECK((<*A,*B,*C>));

CHECK((<<*A,*B>,<*B,*C>,<*C,*A>>));

CHECK((<*A,*B,<*A,*B>>));

CHECK((<*A,<*A,*B>,<*C,*A>>));

CHECK((<*A,*B,<*A,*C>>));

CHECK((<*A,*B>));

CHECK((<*AB,*BC,<*AB,*BC>>));
```

Фиг.4.20. HELP програма за проверка на еднаквост на триъгълници

```
DEF SECOND(R)'= CAR(CDR(R));
DEF ISZERO(X)'= IF ATOM(X) THEN X = *0 ELSE *FALSE;
DEF ISONE(X)'= IF ATOM(X) THEN X = *1 ELSE *FALSE;
DEF SUBPROD(X,Y,A)'= IF ISONE(DERIV(X,A)) THEN Y ELSE <*MULT,DERIV(X,A),Y>;
DEF MPL(R)'= <*DIV,*1,<*MULT,SECOND(R),SECOND(R)>>;
DEF DADD(R,A)'= IF ISZERO(DERIV(CAR(R),A)) THEN DERIV(SECOND(R),A) ELSE
                IF ISZERO(DERIV(SECOND(R),A)) THEN DERIV(CAR(R),A) ELSE
                <*ADD,DERIV(CAR(R),A),DERIV(SECOND(R),A)>;
DEF DSUB(R,A)'= IF ISZERO(DERIV(SECOND(R),A)) THEN DERIV(CAR(R),A) ELSE
                <*SUB,DERIV(CAR(R),A),DERIV(SECOND(R),A)>;
DEF DMULT(R,A)'= IF ISZERO(DERIV(CAR(R),A)) THEN
                 (IF ISZERO(DERIV(SECOND(R),A)) THEN *0 ELSE
                     SUBPROD(SECOND(R),CAR(R),A)) ELSE
                 IF ISZERO(DERIV(SECOND(R),A)) THEN
                     SUBPROD(CAR(R),SECOND(R),A)  ELSE
        <*ADD,SUBPROD(CAR(R),SECOND(R),A),SUBPROD(SECOND(R),CAR(R),A)>;
DEF DDIV(R,A)'= IF ISZERO(DERIV(SECOND(R),A)) THEN
                (IF ISZERO(DERIV(CAR(R),A)) THEN *0 ELSE
                    <*MULT,MPL(R),SUBPROD(CAR(R),SECOND(R),A)>) ELSE
        <*MULT,MPL(R),<*SUB,SUBPROD(CAR(R),SECOND(R),A),
         SUBPROD(SECOND(R),CAR(R),A)>>;
DEF ERROR()'=<*ERROR,*ERROR,*ERROR>;
DEF DERIV(X,A)'= IF ATOM(X) THEN (IF X=A THEN *1 ELSE *0) ELSE
                 IF CAR(X)=*ADD THEN DADD(CDR(X),A) ELSE
                 IF CAR(X)=*SUB THEN DSUB(CDR(X),A) ELSE
                 IF CAR(X)=*MULT THEN DMULT(CDR(X),A) ELSE
                 IF CAR(X)=*DIV THEN DDIV(CDR(X),A) ELSE ERROR( );

DERIV(<*ADD,*A,<*MULT,<*ADD,*B,<*MULT,*C,*X>>,*X>>,*X);

DERIV(<*SUB,<*MULT,<*MULT,*2,*A>,*X>,<*DIV,*C,*X>>,*X);

DERIV(<*SUB,<*MULT,<*MULT,*2,*A>,*X>,<*DIV,*C,*X>>,*A);
```

Фиг.4.21. HELP програма за символично диференциране на алгебрични изрази

```
DEFINE ((
  (FF,(LAMBDA,(R),(CAR,R))),
  (SF,(LAMBDA,(R),(CAR,(CDR,R)))),
  (TF,(LAMBDA,(R),(CAR,(CDR,(CDR,R))))),
  (CH2,(LAMBDA,(R),(COND,((EQ,(FF,R),(FF,(TF,R))),
                          (COND,((EQ,(SF,R),(SF,(TF,R))),T),(T,F))),
                         ((EQ,(FF,R),(SF,(TF,R))),
                          (COND,((EQ,(SF,R),(FF,(TF,R))),T),(T,F))),(T,F)))),
  (CH3,(LAMBDA,(R),(COND,((EQ,(FF,R),(FF,(SF,R))),
                          (COND,((EQ,(FF,R),(FF,(TF,R))),T),
                                ((EQ,(FF,R),(SF,(TF,R))),T),(T,F)),
                         ((EQ,(FF,R),(SF,(SF,R))),
                          (COND,((EQ,(FF,R),(FF,(TF,R))),T),
                                ((EQ,(FF,R),(SF,(TF,R))),T),(T,F)),(T,F)))),
  (CHECK,(LAMBDA,(R),(COND,((QUOTE,(TF,R)),T),((QUOTE,(FF,R)),
        (COND,((QUOTE,(SF,R)),(CH2,R)),(T,(CH3,R)),(T,F))),(T,F))))    ))

CHECK((A,B,C))

CHECK(((A,B),(B,C),(C,A)))

CHECK((A,B,(A,B)))

CHECK((A,(A,B),(C,A)))

CHECK((A,B,(A,C)))

CHECK((AB,BC,(AB,BC)))

CHECK((A,B))
```

Фиг.4.22. LISP програма за проверка на еднаквост на триъгълници

```
DEFINE ((
  (SECOND,(LAMBDA,(R),(CAR,(CDR,R)))),
  (ISZERO,(LAMBDA,(X),(COND,((@UOTE,X),(E@,X,ZERO)),(T,F)))),
  (ISONE,(LAMBDA,(X),(COND,((@UOTE,X),(E@,X,ONE)),(T,F)))),
  (SUBPROD,(LAMBDA,(X,Y,A),(COND,((ISONE,(DERIV,X,A)),Y),(T,(LIST,MULT,
     (DERIV,X,A),Y))))),
  (MPL,(LAMBDA,(R),(LIST,DIV,ONE,(LIST,MULT,(SECOND,R),(SECOND,R))))),
  (DADD,(LAMBDA,(R,A),(COND,((ISZERO,(DERIV,(CAR,R),A)),(DERIV,(SECOND,R),A)),
                            ((ISZERO,(DERIV,(SECOND,R),A)),(DERIV,(CAR,R),A),
     (T,(LIST,ADD,(DERIV,(CAR,R),A),(DERIV,(SECOND,R),A))))))),
  (DSUB,(LAMBDA,(R,A),(COND,((ISZERO,(DERIV,(SECOND,R),A)),(DERIV,(CAR,R),A)),
     (T,(LIST,SUB,(DERIV,(CAR,R),A),(DERIV,(SECOND,R),A)))))),
  (DMULT,(LAMBDA,(R,A),(COND,((ISZERO,(DERIV,(CAR,R),A)),
     (COND,((ISZERO,(DERIV,(SECOND,R),A)),(@UOTE,ZERO)),
           (T,(SUBPROD,(SECOND,R),(CAR,R),A)))),
     (COND,((ISZERO,(DERIV,(SECOND,R),A)),(SUBPROD,(CAR,R),(SECOND,R),A)),
           (T,(SUBPROD,(CAR,R),(SECOND,R),A))),
     (T,(LIST,ADD,(SUBPROD,(CAR,R),(SECOND,R),A),(SUBPROD,(SECOND,R),(CAR,R),A))
     ))))),
  (DDIV,(LAMBDA,(R,A),(COND,((ISZERO,(DERIV,(SECOND,R),A)),
     (COND,((ISZERO,(DERIV,(CAR,R),A)),(@UOTE,ZERO)),
           (T,(LIST,MULT,(MPL,R),(SUBPROD,(CAR,R),(SECOND,R),A))))),
     (T,(LIST,MULT,(MPL,R),(LIST,SUB,(SUBPROD,(CAR,R),(SECOND,R),A),
           (SUBPROD,(SECOND,R),(CAR,R),A)))))))),
  (DERIV,(LAMBDA,(X,A),
     (COND,((@UOTE,X),(COND,((E@,X,A),(@UOTE,ONE)),(T,(@UOTE,ZERO)))),
           ((E@,(CAR,X),ADD),(DADD,(CDR,X),A)),
           ((E@,(CAR,X),SUB),(DSUB,(CDR,X),A)),
           ((E@,(CAR,X),MULT),(DMULT,(CDR,X),A)),
           ((E@,(CAR,X),DIV),(DDIV,(CDR,X),A)),
           (T,(@UOTE,ERROR))))                                  ))

DERIV((ADD,A,(MULT,(ADD,B,(MULT,C,X)),X)),X)

DERIV((SUB,(MULT,(MULT,2,A),X),(DIV,C,X)),X)

DERIV((SUB,(MULT,(MULT,2,A),X),(DIV,C,X)),A)
```

Фиг.4.23. LISP програма за символично диференциране на алгебрични изрази

HELP

```
        USE INITL.                                                  HELP0001
        A = NIL. F = NIL.                                           HELP0002
        READ CHAR. (NIL) = 01.                                      HELP0003
        C = ' .                                                     HELP0004
        (CHAR) = NIL. USE FINPUT.                                   HELP0005
        TO QUIT IF Z NE 01.                                         HELP0006
        (TRUE) = CDR Y.                                             HELP0007
        USE FINPUT.                                                 HELP0008
        TO QUIT IF Z NE 11.                                         HELP0009
        USE FINPUT.                                                 HELP0010
        TO QUIT IF Z NE 01.                                         HELP0011
        (FALSE) = CDR Y.                                            HELP0012
        USE FINPUT.                                                 HELP0013
        TO QUIT IF Z NE 11.                                         HELP0014
        USE FINPUT.                                                 HELP0015
        TO QUIT IF Z NE 03.                                         HELP0016
        X = NEW ELEMENT.                                            HELP0017
        CAR X = L11. CDR X = 'F.                                    HELP0018
        Y = CDR Y. CAR Y = X.                                       HELP0019
        USE FINPUT.                                                 HELP0020
        TO QUIT IF Z NE 03.                                         HELP0021
        X = NEW ELEMENT.                                            HELP0022
        CAR X = L12. CDR X = 'F.                                    HELP0023
        Y = CDR Y. CAR Y = X.                                       HELP0024
        USE FINPUT.                                                 HELP0025
        TO QUIT IF Z NE 03.                                         HELP0026
        X = NEW ELEMENT.                                            HELP0027
        CAR X = L13. CDR X = 'F.                                    HELP0028
        Y = CDR Y. CAR Y = X.                                       HELP0029
        USE FINPUT.                                                 HELP0030
        TO QUIT IF Z NE 03.                                         HELP0031
        X = NEW ELEMENT.                                            HELP0032
        CAR X = L14. CDR X = 'F.                                    HELP0033
        Y = CDR Y. CAR Y = X.                                       HELP0034
        USE FINPUT.                                                 HELP0035
        TO QUIT IF Z NE 03.                                         HELP0036
        X = NEW ELEMENT.                                            HELP0037
        CAR X = L15. CDR X = 'F.                                    HELP0038
        Y = CDR Y. CAR Y = X.                                       HELP0039
        USE FINPUT.                                                 HELP0040
        TO QUIT IF Z NE 09.                                         HELP0041
        @ = NEW ELEMENT. CDR @ = NIL.                               HELP0042
        T = NEW ELEMENT.                                            HELP0043
NEWR.   R = NEW ELEMENT.                                            HELP0044
NEXTE.  CAR R = NIL. CDR R = NIL.                                   HELP0045
        CAR T = EVAL. CDR T = NIL.                                  HELP0046
        V = NIL.                                                    HELP0047
        (NIL) = 01. LINE BUFFER = NIL.                              HELP0048
        WRITE OUTPUT.                                               HELP0049
        (CHAR) = NIL. USE FINPUT.                                   HELP0050
```

```
        TO EX IF Z NE 12.                                          HELP0051
        USE FINPUT.                                                HELP0052
        P = CDR Y.                                                 HELP0053
        TO ERR18 IF Z NE 03.                                       HELP0054
        TO ERR19 IF CAR Y NE NIL.                                  HELP0055
        USE FINPUT.                                                HELP0056
BVARS,  TO FIXDEF IF Z = 08.                                       HELP0057
        TO ERR20 IF Z NE 02.                                       HELP0058
        TO ERR21 IF CDR Y NE NIL.                                  HELP0059
        CAR R = L4, PUSH DOWN R.                                   HELP0060
        CDR Y = CDR R.                                             HELP0061
        USE FINPUT.                                                HELP0062
        TO FIXDEF IF Z = 08.                                       HELP0063
        TO ERR22 IF Z NE 11.                                       HELP0064
        USE FINPUT, TO BVARS.                                      HELP0065
ERR18,  1 = '1, 2 = '8, TO ERROR.                                  HELP0066
ERR19,  1 = '1, 2 = '9, TO ERROR.                                  HELP0067
ERR20,  1 = '2, 2 = '0, TO ERROR.                                  HELP0068
ERR21,  1 = '2, 2 = '1, TO ERROR.                                  HELP0069
ERR22,  1 = '2, 2 = '2, TO ERROR.                                  HELP0070
FIXDEF, USE FINPUT.                                                HELP0071
        CAR T = FIXDF1.                                            HELP0072
        TO REX IF Z = 19.                                          HELP0073
        1 = '2, 2 = '3, TO ERROR.                                  HELP0074
FIXDF1, TO L01 IF Z NE 09.                                         HELP0075
        CAR P = R.                                                 HELP0076
        (NIL) = 01.                                                HELP0077
        Y = P, USE PRATOM.                                         HELP0078
        LINE BUFFER = NIL, WRITE OUTPUT.                           HELP0079
        TO NEWR.                                                   HELP0080
.                                                                  HELP0081
.       ANALYZER FOR EXPRESSION.                                   HELP0082
.                                                                  HELP0083
REX,    USE FINPUT.                                                HELP0084
EX,     TO SE IF Z NE 13.                                          HELP0085
        PUSH DOWN T, CAR T = EX1.                                  HELP0086
        TO RSE.                                                    HELP0087
.                                                                  HELP0088
EX1,    PUSH DOWN R, CAR T = EX2.                                  HELP0089
        TO RSE IF Z = 14.                                          HELP0090
.                                                                  HELP0091
        1 = '1, 2 = '5, TO ERROR.                                  HELP0092
EX2,    PUSH DOWN R, CAR T = EX3.                                  HELP0093
        TO REX IF Z = 15.                                          HELP0094
.                                                                  HELP0095
        1 = '1, 2 = '6, TO ERROR.                                  HELP0096
EX3,    W = CDR R, X = CDR W.                                      HELP0097
        CDR R = CDR X.                                             HELP0098
        CDR W = CAR R, CDR X = W.                                  HELP0099
        PUSH DOWN X, CAR X = L10.                                  HELP0100
RETX,   CAR R = X.                                                 HELP0101
RET,    POP UP T, TO CAR T.                                        HELP0102
.                                                                  HELP0103
.       ANALYZER FOR SIMPLE EXPRESSION                             HELP0104
.                                                                  HELP0105
```

```
RSE,    USE FINPUT.
SE,     PUSH DOWN T, CAR T = SE1.                                    HELP0106
        TO LE.                                                       HELP0107
SE1,    TO SE2 IF Z = 16.                                            HELP0108
        TO RET IF Z NE 17.                                           HELP0109
        PUSH DOWN R, CAR R = L9.                                     HELP0110
        TO SE3.                                                      HELP0111
SE2,    PUSH DOWN R, CAR R = L8.                                     HELP0112
SE3,    PUSH DOWN R.                                                 HELP0113
        CAR T = SE4, TO RSE.                                         HELP0114
.                                                                    HELP0115
SE4,    W = CDR R, X = CDR W.                                        HELP0116
        CDR R = CDR X.                                               HELP0117
SE5,    CDR X = CAR R.                                               HELP0118
        CAR R = W, TO RET.                                           HELP0119
.                                                                    HELP0120
.       ANALYZER FOR LOGICAL EXPRESSION                              HELP0121
.                                                                    HELP0122
RLE,    USE FINPUT.                                                  HELP0123
LE,     PUSH DOWN T.                                                 HELP0124
        TO LE4 IF Z = 18.                                            HELP0125
        CAR T = LE1, TO PR.                                          HELP0126
LE1,    TO LE2 IF Z = 06.                                            HELP0127
        TO RET IF Z NE 10.                                           HELP0128
        PUSH DOWN R, CAR R = L7.                                     HELP0129
        TO LE3.                                                      HELP0130
LE2,    PUSH DOWN R, CAR R = L6.                                     HELP0131
LE3,    PUSH DOWN R.                                                 HELP0132
        CAR T = SE4, TO RPR.                                         HELP0133
.                                                                    HELP0134
LE4,    CAR T = LE5, TO RLE.                                         HELP0135
.                                                                    HELP0136
LE5,    X = NEW ELEMENT.                                             HELP0137
        CAR X = L5, CDR X = CAR R.                                   HELP0138
        TO RETX.                                                     HELP0139
.                                                                    HELP0140
.       ANALYZER FOR PRIMARY                                         HELP0141
.                                                                    HELP0142
RPR,    USE FINPUT.                                                  HELP0143
PR,     TO PR4 IF Z = 01.                                            HELP0144
        TO PR5 IF Z = 02.                                            HELP0145
        TO PR6 IF Z = 03.                                            HELP0146
        TO PR15 IF Z = 04.                                           HELP0147
        PUSH DOWN T, CAR T = PR1.                                    HELP0148
        TO REX IF Z = 07.                                            HELP0149
        1 = '1, 2 = '0, TO ERROR.                                    HELP0150
PR1,    TO PR2 IF Z = 08.                                            HELP0151
        1 = '1, 2 = '4, TO ERROR.                                    HELP0152
PR2,    POP UP T.                                                    HELP0153
PR3,    USE FINPUT, TO CAR T.                                        HELP0154
PR4,    X = NEW ELEMENT.                                             HELP0155
        CAR X = L1, CDR X = CDR Y.                                   HELP0156
        CAR R = X, TO PR3.                                           HELP0157
.                                                                    HELP0158
.       BOUND VARIABLE                                               HELP0159
                                                                     HELP0160
```

```
.                                                                    HELP0161
PR5.    Y = CDR Y, CAR R = Y.                                        HELP0162
        TO PR3 IF CAR Y = L4.                                        HELP0163
.                                                                    HELP0164
        1 = '1, 2 = '1, TO ERROR.                                    HELP0165
.                                                                    HELP0166
.         FUNCTION CALL                                              HELP0167
.                                                                    HELP0168
PR6.    CAR R = CDR Y, PUSH DOWN R.                                  HELP0169
        USE FINPUT.                                                  HELP0170
        TO PR14 IF Z = 08.                                           HELP0171
        PUSH DOWN T, CAR T = PR11.                                   HELP0172
.                                                                    HELP0173
.         ANALYZE A LIST OF EXPRESSIONS                              HELP0174
.                                                                    HELP0175
PR7.    CAR R = 'F, PUSH DOWN R.                                     HELP0176
        PUSH DOWN T, CAR T = PR8.                                    HELP0177
        TO EX.                                                       HELP0178
PR8.    TO PR9 IF Z NE 11.                                           HELP0179
        PUSH DOWN R, TO REX.                                         HELP0180
PR9.    X = NIL.                                                     HELP0181
PR10.   W = X, X = R, R = CDR R.                                     HELP0182
        CDR X = W.                                                   HELP0183
        TO PR10 IF CAR R NE 'F.                                      HELP0184
        PUSH DOWN X, TO RET.                                         HELP0185
.                                                                    HELP0186
.         COMPLETE A FUNCTION CALL                                   HELP0187
.                                                                    HELP0188
PR11.   POP UP T.                                                    HELP0189
        TO PR12 IF Z = 08.                                           HELP0190
        1 = '1, 2 = '2, TO ERROR.                                    HELP0191
PR14.   X = NEW ELEMENT.                                             HELP0192
        CDR X = NIL.                                                 HELP0193
PR12.   W = R, R = CDR R.                                            HELP0194
        CAR X = CAR R.                                               HELP0195
        CAR W = L3, CDR W = X.                                       HELP0196
        CAR R = W, TO PR3.                                           HELP0197
.                                                                    HELP0198
.         ANALYZE A BRACKETED LIST                                   HELP0199
.                                                                    HELP0200
PR15.   USE FINPUT.                                                  HELP0201
        TO PR18 IF Z = 05.                                           HELP0202
        PUSH DOWN T, CAR T = PR16.                                   HELP0203
        TO PR7.                                                      HELP0204
PR16.   POP UP T.                                                    HELP0205
        TO PR17 IF Z = 05.                                           HELP0206
        1 = '1, 2 = '3, TO ERROR.                                    HELP0207
PR18.   X = NEW ELEMENT.                                             HELP0208
        CDR X = NIL.                                                 HELP0209
PR17.   CAR X = L2, CAR R = X.                                       HELP0210
        TO PR3.                                                      HELP0211
.                                                                    HELP0212
.         EVALUATE AN EXPRESSION JUST COMPILED                       HELP0213
.                                                                    HELP0214
EVAL.   TO L01 IF Z NE 09.                                           HELP0215
```

```
        (NIL) = 01.                                                    HELP0216
        E = CAR R.                                                     HELP0217
        S = NEW ELEMENT.                                               HELP0218
        CAR S = NIL. CDR S = NIL.                                      HELP0219
        CAR T = L0. TO CAR E.                                          HELP0220
L0.                                                                    HELP0221
        Y = CAR R. USE PRLIST.                                         HELP0222
        LINE BUFFER = NIL.                                             HELP0223
        WRITE OUTPUT.                                                  HELP0224
        TO NEXTE.                                                      HELP0225
L01.    1 = '1. 2 = '7.                                                HELP0226
ERROR.  TO NONE IF (NIL) = 01.                                         HELP0227
        Y = (NIL). (NIL) = 01.                                         HELP0228
DASH.   LINE BUFFER = '-.                                              HELP0229
        TO DASH IF (NIL) LT Y.                                         HELP0230
NONE.   WRITE OUTPUT.                                                  HELP0231
        TO MESSG IF Z = 09.                                            HELP0232
        TO SKPS1.                                                      HELP0233
SKPS.   USE NEXTCH.                                                    HELP0234
SKPS1.  TO SKPS IF (CHAR) NE ''.                                       HELP0235
        USE NEXTCH.                                                    HELP0236
        TO SKPS IF (CHAR) NE '.                                        HELP0237
MESSG.  (NIL) = 01.                                                    HELP0238
        LINE BUFFER = 'E.                                              HELP0239
        LINE BUFFER = 'R.                                              HELP0240
        LINE BUFFER = 'R.                                              HELP0241
        LINE BUFFER = 'O.                                              HELP0242
        LINE BUFFER = 'R.                                              HELP0243
        LINE BUFFER = ' .                                              HELP0244
        LINE BUFFER = 1.                                               HELP0245
        LINE BUFFER = 2.                                               HELP0246
        LINE BUFFER = NIL.                                             HELP0247
        WRITE OUTPUT.                                                  HELP0248
        TO IFAIL IF 1 = ' .                                            HELP0249
        TO SKIPL.                                                      HELP0250
NXCALL. Y = CAR @. USE PRATOM.                                         HELP0251
        LINE BUFFER = '(.                                              HELP0252
        TO FSTRG IF (NIL) NE 00.                                       HELP0253
        WRITE OUTPUT. (NIL) = 01.                                      HELP0254
FSTRG.  POP UP @. Y = CAR @.                                           HELP0255
        TO PARG IF Y NE 'F.                                            HELP0256
        TO ENDAL.                                                      HELP0257
SEP.    LINE BUFFER = '.                                               HELP0258
        TO PARG IF (NIL) NE 00.                                        HELP0259
        WRITE OUTPUT. (NIL) = 01.                                      HELP0260
PARG.   Y = CDR Y. USE PRLIST.                                         HELP0261
        POP UP @. Y = CAR @.                                           HELP0262
        TO SEP IF Y NE 'F.                                             HELP0263
ENDAL.  POP UP @.                                                      HELP0264
        LINE BUFFER = ').                                              HELP0265
        LINE BUFFER = NIL.                                             HELP0266
        WRITE OUTPUT.                                                  HELP0267
IFAIL.  TO NXCALL IF CDR @ NE NIL.                                     HELP0268
SKIPL.  LINE BUFFER = NIL.                                             HELP0269
        WRITE OUTPUT. WRITE OUTPUT.                                    HELP0270
```

```
        TO NEXTE.                                                    HELP0271
FAIL,   1 = ' , TO MESSG.                                            HELP0272
.                                                                    HELP0273
L1.       EVALUATE  AN ATOM                                          HELP0274
.                                                                    HELP0275
        CAR R = CDR E.                                               HELP0276
        TO CAR T.                                                    HELP0277
.                                                                    HELP0278
L2.       EVALUATE A LIST                                            HELP0279
.                                                                    HELP0280
        TO L21 IF CDR E NE NIL.                                      HELP0281
        CAR R = NIL, TO CAR T.                                       HELP0282
L21,    PUSH DOWN T, CAR T = L22.                                    HELP0283
        TO C23.                                                      HELP0284
L22,    X = NIL.                                                     HELP0285
L23,    W = R, R = CDR R.                                            HELP0286
        CDR W = X, X = W.                                            HELP0287
        TO L23 IF CAR R NE 'F.                                       HELP0288
        TO RETX.                                                     HELP0289
.                                                                    HELP0290
L3.       EVALUATE A FUNCTION CALL                                   HELP0291
.                                                                    HELP0292
        E = CDR E, W = CAR E.                                        HELP0293
        TO L322 IF CAR W = NIL.                                      HELP0294
        PUSH DOWN T.                                                 HELP0295
        TO L331 IF CDR E = NIL.                                      HELP0296
        PUSH DOWN S, CAR S = W.                                      HELP0297
        CAR T = L31.                                                 HELP0298
.                                                                    HELP0299
.         COMMON CODE FOR EVALUATING A LIST  OF EXPRESSIONS          HELP0300
.                                                                    HELP0301
C23.                                                                 HELP0302
        E = CDR E.                                                   HELP0303
        CAR R = 'F.                                                  HELP0304
        PUSH DOWN S.                                                 HELP0305
        PUSH DOWN T, CAR T = C232.                                   HELP0306
C231,   PUSH DOWN R.                                                 HELP0307
        CAR S = CDR E.                                               HELP0308
ECARE,  E = CAR E, TO CAR E.                                         HELP0309
C232,   E = CAR S.                                                   HELP0310
        TO C231 IF E NE NIL.                                         HELP0311
        POP UP S, TO RET.                                            HELP0312
.                                                                    HELP0313
L31.      RETURN FROM EVALUATING ARGUMENTS                           HELP0314
.                                                                    HELP0315
        W = CAR S, POP UP S.                                         HELP0316
        E = CAR W, Y = CDR E.                                        HELP0317
        TO CAR E IF Y = 'F.                                          HELP0318
        PUSH DOWN @, CAR @ = 'F.                                     HELP0319
.                                                                    HELP0320
L32.      SET UP NEW ASSOCIATIONS FOR THE BOUND VARIABLES            HELP0321
.                                                                    HELP0322
        TO L321 IF Y = NIL.                                          HELP0323
        Z = R, R = CDR R.                                            HELP0324
        CDR Z = CAR Z, CAR Z = Y.                                    HELP0325
```

```
        PUSH DOWN @, CAR @ = Z.                                          HELP0326
        Y = CDR Y.                                                       HELP0327
        TO L32 IF CAR R NE 'F.                                           HELP0328
        TO L321 IF Y NE NIL.                                             HELP0329
.                                                                        HELP0330
L33.    EVALUATE A USER-DEFINED FUNCTION                                 HELP0331
.                                                                        HELP0332
        PUSH DOWN @, CAR @ = W.                                          HELP0333
        CAR T = L332, TO ECARE.                                          HELP0334
L331,   E = CAR W.                                                       HELP0335
        PUSH DOWN @, CAR @ = 'F.                                         HELP0336
        TO L33 IF CDR E = NIL.                                           HELP0337
L321,   PUSH DOWN @, CAR @ = W.                                          HELP0338
        Z = '0, TO FAIL.                                                 HELP0339
L322,   PUSH DOWN @, CAR @ = 'F.                                         HELP0340
        TO L321.                                                         HELP0341
L332,   POP UP @, TO L332 IF CAR @ NE 'F.                                HELP0342
        POP UP @, TO RET.                                                HELP0343
.                                                                        HELP0344
L4.     EVALUATE A BOUND VARIABLE                                        HELP0345
.                                                                        HELP0346
        Z = @.                                                           HELP0347
L41,    Z = CDR Z, Y = CAR Z.                                            HELP0348
        TO L41 IF CAR Y NE E.                                            HELP0349
        CAR R = CDR Y.                                                   HELP0350
        TO CAR T.                                                        HELP0351
.                                                                        HELP0352
L5.     NEGATE A LOGICAL EXPRESSION                                      HELP0353
.                                                                        HELP0354
        PUSH DOWN T, CAR T = L51.                                        HELP0355
ECDRE,  E = CDR E, TO CAR E.                                             HELP0356
L51,    TO L53 IF CAR R = (FALSE).                                       HELP0357
        TO L52 IF CAR R = (TRUE).                                        HELP0358
        Z = '6, TO FAIL.                                                 HELP0359
L52,    CAR R = (FALSE), TO RET.                                         HELP0360
L53,    CAR R = (TRUE), TO RET.                                          HELP0361
.                                                                        HELP0362
L6.     EVALUATE P1 = P2                                                 HELP0363
.                                                                        HELP0364
        PUSH DOWN T, CAR T = L61.                                        HELP0365
.                                                                        HELP0366
.       COMMON CODE FOR THE EVALUATION OF TWO PRIMARIES                  HELP0367
.                                                                        HELP0368
C67.                                                                     HELP0369
        E = CDR E.                                                       HELP0370
        PUSH DOWN S, CAR S = CDR E.                                      HELP0371
        PUSH DOWN T, CAR T = C671.                                       HELP0372
        TO ECARE.                                                        HELP0373
C671,   E = CAR R.                                                       HELP0374
        TO FAIL4 IF AF E = 00.                                           HELP0375
        PUSH DOWN R.                                                     HELP0376
        E = CAR S, POP UP S.                                             HELP0377
        CAR T = C672, TO CAR E.                                          HELP0378
FAIL4,  Z = '4, TO FAIL.                                                 HELP0379
C672,   E = CAR R, POP UP R.                                             HELP0380
```

```
        TO RET IF AF E = 01.                                    HELP0381
        2 = '5, TO FAIL.                                        HELP0382
.                                                               HELP0383
.       CONTINUE PROCESSING P1 = P2.                            HELP0384
.                                                               HELP0385
L61,    TO L53 IF CAR R = E, TO L52.                            HELP0386
.                                                               HELP0387
L7.     EVALUATE P1 NE P2                                       HELP0388
.                                                               HELP0389
        PUSH DOWN T, CAR T = L71.                               HELP0390
        TO C67.                                                 HELP0391
L71,    TO L53 IF CAR R NE E, TO L52.                           HELP0392
.                                                               HELP0393
L8.     EVALUATE LE AND SE                                      HELP0394
.                                                               HELP0395
        PUSH DOWN T, CAR T = L81.                               HELP0396
        TO L101.                                                HELP0397
L81,    E = CAR S, POP UP S.                                    HELP0398
        TO RET IF CAR R = (FALSE).                              HELP0399
        TO L92 IF CAR R NE (TRUE).                              HELP0400
.                                                               HELP0401
L82.    VALUE OF EXPRESSION IS VALUE OF SE                      HELP0402
.                                                               HELP0403
        CAR T = L83, TO CAR E.                                  HELP0404
L83,    TO RET IF CAR R = (TRUE).                               HELP0405
        TO RET IF CAR R = (FALSE).                              HELP0406
        2 = '8, TO FAIL.                                        HELP0407
.                                                               HELP0408
L9.     EVALUATE LE OR SE                                       HELP0409
.                                                               HELP0410
        PUSH DOWN T, CAR T = L91.                               HELP0411
        TO L101.                                                HELP0412
L91,    E = CAR S, POP UP S.                                    HELP0413
        TO RET IF CAR R = (TRUE).                               HELP0414
        TO L82 IF CAR R = (FALSE).                              HELP0415
.                                                               HELP0416
L92,    2 = '7, TO FAIL.                                        HELP0417
.                                                               HELP0418
L10.    EVALUATE AN EXPRESSION                                  HELP0419
.                                                               HELP0420
        PUSH DOWN T, CAR T = L102.                              HELP0421
L101,   E = CDR E.                                              HELP0422
        PUSH DOWN S, CAR S = CDR E.                             HELP0423
        TO ECARE.                                               HELP0424
L102,   E = CAR S, POP UP S.                                    HELP0425
        POP UP T.                                               HELP0426
        TO ECARE IF CAR R = (TRUE).                             HELP0427
        TO ECDRE IF CAR R = (FALSE).                            HELP0428
        2 = '9, TO FAIL.                                        HELP0429
.                                                               HELP0430
L11.    EVALUATE THE BUILT-IN FUNCTION CAR                      HELP0431
.                                                               HELP0432
        Z = CAR R, POP UP R.                                    HELP0433
        TO L111 IF CAR R NE 'F.                                 HELP0434
        CAR R = CAR Z.                                          HELP0435
```

```
        TO RET IF AF Z = 00.                                    HELP0436
L111,   2 = '1, TO FAIL.                                        HELP0437
.                                                               HELP0438
L12.      EVALUATE THE BUILT-IN FUNCTION CDR                    HELP0439
.                                                               HELP0440
        Z = CAR R, POP UP R.                                    HELP0441
        TO L121 IF CAR R NE 'F.                                 HELP0442
        CAR R = CDR Z.                                          HELP0443
        TO RET IF AF Z = 00.                                    HELP0444
L121,   2 = '2, TO FAIL.                                        HELP0445
.                                                               HELP0446
L13.      EVALUATE THE BUILT-IN FUNCTION CONS                   HELP0447
.                                                               HELP0448
        Z = CAR R, POP UP R.                                    HELP0449
        TO L131 IF CAR R = 'F.                                  HELP0450
        E = R, R = CDR R.                                       HELP0451
        TO L131 IF CAR R NE 'F.                                 HELP0452
        CDR E = Z.                                              HELP0453
        CAR R = E.                                              HELP0454
        TO RET IF Z = NIL.                                      HELP0455
        TO RET IF AF Z = 00.                                    HELP0456
L131,   2 = '3, TO FAIL.                                        HELP0457
.                                                               HELP0458
L14.      EVALUATE THE BUILT-IN FUNCTION ATOM                   HELP0459
.                                                               HELP0460
        Z = CAR R, POP UP R.                                    HELP0461
        TO L321 IF CAR R NE 'F.                                 HELP0462
        TO L142 IF AF Z = 00.                                   HELP0463
L141,   CAR R = (TRUE), TO RET.                                 HELP0464
L142,   CAR R = (FALSE), TO RET.                                HELP0465
.                                                               HELP0466
L15.      EVALUATE THE BUILT-IN FUNCTION NULL                   HELP0467
.                                                               HELP0468
        Z = CAR R, POP UP R.                                    HELP0469
        TO L321 IF CAR R NE 'F.                                 HELP0470
        TO L141 IF Z = NIL, TO L142.                            HELP0471
.                                                               HELP0472
.                                                               HELP0473
.                                                               HELP0474
        ENTRY FINPUT.                                           HELP0475
        TO SKIP IF (CHAR) NE NIL.                               HELP0476
        USE NEXTCH.                                             HELP0477
SKIP,   TO ATOM IF (CHAR) = '*.                                 HELP0478
        TO IDFN IF (CHAR) = LETORDIG.                           HELP0479
        TO MULT IF (CHAR) = '*.                                 HELP0480
        TO SEMIC IF (CHAR) = ';.                                HELP1480
        TO LBKTC IF (CHAR) = '<.                                HELP2480
        TO RBKTC IF (CHAR) = '>.                                HELP3480
        TO RPAR IF (CHAR) NE '(.                                HELP0481
        Z = 07, TO NOCH.                                        HELP0482
RPAR,   TO COMM IF (CHAR) NE ').                                HELP0483
        Z = 08, TO NOCH.                                        HELP0484
COMM,   TO EQUL IF (CHAR) NE ',.                                HELP0485
        Z = 11, TO NOCH.                                        HELP0486
EQUL,   TO ERRC IF (CHAR) NE '=.                                HELP0487
```

```
        Z = 06, TO NOCH.                                        HELP0488
ERRC,   Z = 00.                                                 HELP0489
NOCH,   (CHAR) = NIL, EXIT FINPUT.                              HELP0490
ATOM,   USE NEXTCH, USE READST.                                 HELP0491
        D = 'A, M = 'X.                                         HELP0492
        USE LOOKUP.                                             HELP0493
        TO BAKA IF CDR Y NE NIL.                                HELP0494
        Z = NEW ELEMENT, CDR Y = Z.                             HELP0495
        AF Z = 01.                                              HELP0496
        X = NIL, CAR Z = CAR X.                                 HELP0497
        CAR X = Z.                                              HELP0498
        CDR Z = S.                                              HELP0499
BAKA,   Z = 01, EXIT FINPUT.                                    HELP0500
IDFN,   USE READST.                                             HELP0501
        D = BASIC, M = NIL.                                     HELP0502
        USE LOOKUP.                                             HELP0503
        TO BSYM IF Y NE NIL.                                    HELP0504
        TO FUNC IF (CHAR) = '(.                                 HELP0505
        D = 'V, M = 'X.                                         HELP0506
        USE LOOKUP.                                             HELP0507
        Z = 02, EXIT FINPUT.                                    HELP0508
BSYM,   Z = CDR Y, EXIT FINPUT.                                 HELP0509
FUNC,   D = 'F, M = 'X.                                         HELP0510
        USE LOOKUP.                                             HELP0511
        TO BAKF IF CDR Y NE NIL.                                HELP0512
        Z = NEW ELEMENT, CDR Y = Z.                             HELP0513
        CAR Z = NIL, CDR Z = S.                                 HELP0514
BAKF,   Z = 03, TO NOCH.                                        HELP0515
MULT,   USE NEXTCH.                                             HELP0516
        TO RBKT IF (CHAR) NE '(.                                HELP0517
LBKTC,  Z = 04, TO NOCH.                                        HELP0518
RBKT,   TO SEMI IF (CHAR) NE ').                                HELP0519
RBKTC,  Z = 05, TO NOCH.                                        HELP0520
SEMI,   TO ASGN IF (CHAR) NE ',.                                HELP0521
SEMIC,  Z = 09, TO NOCH.                                        HELP0522
ASGN,   TO ERRC IF (CHAR) NE '=.                                HELP0523
ASGNC,  Z = 19, TO NOCH.                                        HELP0524
.                                                               HELP0525
.                                                               HELP0526
.                                                               HELP0527
        ENTRY LOOKUP.                                           HELP0528
        Z = S.                                                  HELP0529
NEXT,   TO ABSENT IF CDR D = NIL.                               HELP0530
        D = CDR D, Y = CAR D.                                   HELP0531
CHECK,  TO NEXT IF CAR Y NE CAR Z.                              HELP0532
        TO BACK IF CAR Y = NIL.                                 HELP0533
STEP,   Y = CDR Y, Z = CDR Z.                                   HELP0534
        TO CKSUB IF CAR Y NE CAR Z.                             HELP0535
        TO STEP IF CAR Y NE NIL.                                HELP0536
        EXIT LOOKUP.                                            HELP0537
CKSUB,  D = Y, Y = CAR D.                                       HELP0538
        TO CHECK IF AF Y = 00.                                  HELP0539
        TO NONSRT IF M = NIL.                                   HELP0540
        Y = NEW ELEMENT.                                        HELP0541
        CAR Y = CAR D, CDR Y = CDR D.                           HELP0542
```

```
        CAR D = Y.                                              HELP0543
INSERT, CDR D = NEW ELEMENT.                                    HELP0544
        D = CDR D, CDR D = NIL.                                 HELP0545
        CAR D = Z, Y = 'Z.                                      HELP0546
TOEND,  Y = CDR Y.                                              HELP0547
        TO TOEND IF CAR Y NE NIL.                               HELP0548
BACK,   EXIT LOOKUP.                                            HELP0549
ABSENT, TO INSERT IF M NE NIL.                                  HELP0550
NONSRT, Y = NIL, EXIT LOOKUP.                                   HELP0551
.                                                               HELP0552
.                                                               HELP0553
.                                                               HELP0554
        ENTRY READST.                                           HELP0555
        C = CHAR.                                               HELP0556
        Z = 'S, TO LOOP1.                                       HELP0557
STAR,   CAR Z = ' .                                             HELP0558
LOOP,   USE NEXTCH.                                             HELP0559
LOOP1,  CDR Z = NEW ELEMENT.                                    HELP0560
        Z = CDR Z, CAR Z = (CHAR).                              HELP0561
        TO LOOP IF (CHAR) = LETORDIG.                           HELP0562
        TO STAR IF (CHAR) = '*.                                 HELP0563
        CAR Z = NIL, CDR Z = NIL.                               HELP0564
        C = ' .                                                 HELP0565
        TO GOTCH IF (CHAR) NE ' .                               HELP0566
        USE NEXTCH.                                             HELP0567
GOTCH,  EXIT READST.                                            HELP0568
.                                                               HELP0569
.                                                               HELP0570
.                                                               HELP0571
        ENTRY NEXTCH.                                           HELP0572
GETCH,  (CHAR) = LINE BUFFER.                                   HELP0573
CKCHR,  TO GETCH IF (CHAR) = C.                                 HELP0574
        TO NEWL IF (CHAR) = NIL.                                HELP0575
        EXIT NEXTCH.                                            HELP0576
NEWL,   READ INPUT.                                             HELP0577
        TO QUIT IF (INPUT) NE 00.                               HELP0578
        WRITE OUTPUT.                                           HELP0579
        (CHAR) = ' , TO CKCHR.                                  HELP0580
QUIT,   STOP.                                                   HELP0581
.                                                               HELP0582
.                                                               HELP0583
.                                                               HELP0584
        ENTRY PRLIST.                                           HELP0585
        TO NOATOM IF AF Y = 00.                                 HELP0586
        USE PRATOM.                                             HELP0587
        EXIT PRLIST.                                            HELP0588
NOATOM, X = 'Y.                                                 HELP0589
LIST,   PUSH DOWN S.                                            HELP0590
        CAR S = X.                                              HELP0591
        X = Y, LINE BUFFER = '(.                                HELP0592
        TO NEXOUT IF (NIL) NE 00.                               HELP0593
        WRITE OUTPUT, (NIL) = 01.                               HELP0594
NEXOUT, Y = CAR X.                                              HELP0595
        TO LIST IF AF Y = 00.                                   HELP0596
        USE PRATOM.                                             HELP0597
```

```
ADV,    TO ENDLIS IF CDR X = NIL.                                  HELP0598
        LINE BUFFER = ',.                                          HELP0599
        TO STEP1 IF (NIL) NE 00.                                   HELP0600
        WRITE OUTPUT. (NIL) = 01.                                  HELP0601
STEP1,  X = CDR X. TO NEXOUT.                                      HELP0602
ENDLIS,LINE BUFFER = ').                                           HELP0603
        TO POPUP IF (NIL) NE 00.                                   HELP0604
        WRITE OUTPUT. (NIL) = 01.                                  HELP0605
POPUP,  X = CAR S.                                                 HELP0606
        POP UP S.                                                  HELP0607
        TO ADV IF AF X = 00.                                       HELP0608
        EXIT PRLIST.                                               HELP0609
.                                                                  HELP0610
.                                                                  HELP0611
.                                                                  HELP0612
        ENTRY PRATOM.                                              HELP0613
        TO ADV1 IF Y NE NIL.                                       HELP0614
        Y = NILAT. TO NOTEND.                                      HELP0615
ADV1,   Y = CDR Y.                                                 HELP0616
CHECK1,TO NOTEND IF CAR Y NE NIL.                                  HELP0617
        EXIT PRATOM.                                               HELP0618
NOTEND,Z = CAR Y.                                                  HELP0619
        TO PRINT IF AF Z = 01.                                     HELP0620
        Y = Z. TO CHECK1.                                          HELP0621
PRINT,  LINE BUFFER = Z.                                           HELP0622
        TO ADV1 IF (NIL) NE 00.                                    HELP0623
        WRITE OUTPUT.                                              HELP0624
        (NIL) = 01. TO ADV1.                                       HELP0625
.                                                                  HELP0626
.           DATA ELEMENTS                                          HELP0627
.                                                                  HELP0628
CHAR,   ELEMENT 00 00 01 NIL.                                      HELP0629
INPUT,  ELEMENT 00 00 01 00.                                       HELP0630
OUTPUT,ELEMENT 00 00 04 00.                                        HELP0631
FALSE,  ELEMENT 00 00 00 00.                                       HELP0632
TRUE,   ELEMENT 00 00 00 00.                                       HELP0633
BASIC,  ELEMENT 01 00 00 B1.                                       HELP0634
B1,     ELEMENT 00 00 B11 B2.                                      HELP0635
B2,     ELEMENT 00 00 B21 B3.                                      HELP0636
B3,     ELEMENT 00 00 B31 B4.                                      HELP0637
B4,     ELEMENT 00 00 B41 B5.                                      HELP0638
B5,     ELEMENT 00 00 B51 B6.                                      HELP0639
B6,     ELEMENT 00 00 B61 B7.                                      HELP0640
B7,     ELEMENT 00 00 B71 NIL.                                     HELP0641
B11,    ELEMENT 00 00 'A  B12.                                     HELP0642
B12,    ELEMENT 00 00 'N  B13.                                     HELP0643
B13,    ELEMENT 00 00 'D  B14.                                     HELP0644
B14,    ELEMENT 00 00 NIL 16.                                      HELP0645
B21,    ELEMENT 00 00 'D  B22.                                     HELP0646
B22,    ELEMENT 00 00 'E  B23.                                     HELP0647
B23,    ELEMENT 00 00 'F  B24.                                     HELP0648
B24,    ELEMENT 00 00 NIL 12.                                      HELP0649
B31,    ELEMENT 00 00 'E  B32.                                     HELP0650
B32,    ELEMENT 00 00 'L B33.                                      HELP0651
B33,    ELEMENT 00 00 'S B34.                                      HELP0652
```

```
B34.    ELEMENT 00 00 'E B35.                                    HELP0653
B35.    ELEMENT 00 00 NIL 15.                                    HELP0654
B41.    ELEMENT 00 00 'I B42.                                    HELP0655
B42.    ELEMENT 00 00 'F B43.                                    HELP0656
B43.    ELEMENT 00 00 NIL 13.                                    HELP0657
B51.    ELEMENT 00 00 'N B52.                                    HELP0658
B52.    ELEMENT 00 00 B521 B53.                                  HELP0659
B53.    ELEMENT 00 00 B531 NIL.                                  HELP0660
B521.   ELEMENT 00 00 'E B522.                                   HELP0661
B522.   ELEMENT 00 00 NIL 10.                                    HELP0662
B531.   ELEMENT 00 00 'O B532.                                   HELP0663
B532.   ELEMENT 00 00 'T B533.                                   HELP0664
B533.   ELEMENT 00 00 NIL 18.                                    HELP0665
B61.    ELEMENT 00 00 'O B62.                                    HELP0666
B62.    ELEMENT 00 00 'R B63.                                    HELP0667
B63.    ELEMENT 00 00 NIL 17.                                    HELP0668
B71.    ELEMENT 00 00 'T B72.                                    HELP0669
B72.    ELEMENT 00 00 'H B73.                                    HELP0670
B73.    ELEMENT 00 00 'E B74.                                    HELP0671
B74.    ELEMENT 00 00 'N B75.                                    HELP0672
B75.    ELEMENT 00 00 NIL 14.                                    HELP0673
NILAT.  ELEMENT 00 00 '( N1.                                     HELP0674
N1.     ELEMENT 00 00 ') N2.                                     HELP0675
N2.     ELEMENT 00 00 NIL NIL.                                   HELP0676
.                                                                HELP0677
.                                                                HELP0678
.                                                                HELP0680
        END PROGRAM.                                             HELP0681
```

DYNL

```
        ENTRY INITL.                                        DYNL0001
        A = 8.                                              DYNL0002
LINK,   B = A. INCR A.                                      DYNL0003
        CDR B = A.                                          DYNL0004
        AF B = 00, BPF B = 00.                              DYNL0005
        TO LINK IF A LT 9.                                  DYNL0006
        TO BTOP IF A NE 9.                                  DYNL0007
.                                                           DYNL0008
        AF A = 00, BPF A = 00.                              DYNL0009
        B = A.                                              DYNL0010
BTOP,   CDR B = NIL.                                        DYNL0011
        (FREE) = 8.                                         DYNL0012
        (BOT) = 8, (TOP) = B.                               DYNL0013
        EXIT INITL.                                         DYNL0014
FREE,   ELEMENT 00 00 00 NIL.                               DYNL0015
BOT,    ELEMENT 00 00 00 00.                                DYNL0016
TOP,    ELEMENT 00 00 00 00.                                DYNL0017
        ENTRY TRACER.                                       DYNL0018
        TO ENDTR IF CDR P1 LT (BOT).                        DYNL0019
        TO ENDTR IF (TOP) LT CDR P1.                        DYNL0020
        (P3) = CDR P1.                                      DYNL0021
        TO ENDTR IF AF P3 = 01.                             DYNL0022
        (P2) = (P1).                                        DYNL0023
FORWD,  CDR P1 = (P2).                                      DYNL0024
MARK,   AF P3 = 01.                                         DYNL0025
        (P2) = (P1), (P1) = (P3).                           DYNL0026
        TO REV IF CDR P1 LT (BOT).                          DYNL0027
        TO REV IF (TOP) LT CDR P1.                          DYNL0028
        (P3) = CDR P1.                                      DYNL0029
        TO FORWD IF AF P3 = 00.                             DYNL0030
REV,    TO CHKBR IF CAR P1 LT (BOT).                        DYNL0031
        TO CHKBR IF (TOP) LT CAR P1.                        DYNL0032
        (P3) = CAR P1.                                      DYNL0033
        TO BRNCH IF AF P3 = 00.                             DYNL0034
CHKBR,  TO ENDBR IF BPF P2 = 01.                            DYNL0035
        (P3) = CDR P2, CDR P2 = (P1).                       DYNL0036
        (P1) = (P2), (P2) = (P3).                           DYNL0037
        TO REV IF (P1) NE (P2).                             DYNL0038
ENDTR,  EXIT TRACER.                                        DYNL0039
BRNCH,  CAR P1 = (P2), BPF P1 = 01.                         DYNL0040
        TO MARK.                                            DYNL0041
ENDBR,  BPF P2 = 00.                                        DYNL0042
        (P3) = CAR P2, CAR P2 = (P1).                       DYNL0043
        (P1) = (P2), (P2) = (P3).                           DYNL0044
        TO CHKBR.                                           DYNL0045
P1,     ELEMENT 00 00 00 00.                                DYNL0046
P2,     ELEMENT 00 00 00 00.                                DYNL0047
P3,     ELEMENT 00 00 00 00.                                DYNL0048
        ENTRY GETNEW.                                       DYNL0049
        TO ELOK IF (FREE) NE NIL.                           DYNL0050
```

```
      TO FAIL3 IF (BOT) = (TOP).                                          DYNL0051
      (P1) = NIL.                                                         DYNL0052
SEQLS,BPF P1 = 01, (P1) = CAR P1.                                         DYNL0053
      TO FAIL1 IF BPF P1 = 01.                                            DYNL0054
      USE TRACER.                                                         DYNL0055
      TO SEQLS IF CAR P1 NE NIL.                                          DYNL0056
      (P1) = (BOT), (P2) = FREE.                                          DYNL0057
TSTFL,TO CLRFL IF AF P1 = 01.                                             DYNL0058
      CDR P2 = (P1), (P2) = (P1).                                         DYNL0059
      TO ADVPT.                                                           DYNL0060
CLRFL,AF P1 = 00.                                                         DYNL0061
ADVPT,TO FIXUP IF (P1) = (TOP).                                           DYNL0062
      INCR (P1), TO TSTFL.                                                DYNL0063
FIXUP,TO FAIL2 IF (P2) = FREE.                                            DYNL0064
      CDR P2 = NIL.                                                       DYNL0065
      (P1) = NIL.                                                         DYNL0066
FIXAT,AF P1 = 01, BPF P1 = 00.                                            DYNL0067
      (P1) = CAR P1.                                                      DYNL0068
      TO FIXAT IF (P1) NE NIL.                                            DYNL0069
ELOK, (P1) = (FREE).                                                      DYNL0070
      (FREE) = CDR FREE.                                                  DYNL0071
      EXIT GETNEW.                                                        DYNL0072
FAIL1,(NIL) = 06, LINE BUFFER = '1.  CORRUPTED CAR CHAIN.                 DYNL0073
      TO FAILI.                                                           DYNL0074
FAIL2,(NIL) = 06, LINE BUFFER = '2.  NO GARBAGE COLLECTED.                DYNL0075
      TO FAILI.                                                           DYNL0076
FAIL3,(NIL) = 06, LINE BUFFER = '3.  MEMORY HAS NOT BEEN INITIALIZED.     DYNL0077
FAILI,LINE BUFFER = NIL, (NIL) = 01.                                      DYNL0078
      LINE BUFFER = 'F.                                                   DYNL0079
      LINE BUFFER = 'A.                                                   DYNL0080
      LINE BUFFER = 'I.                                                   DYNL0081
      LINE BUFFER = 'L.                                                   DYNL0082
      LINE BUFFER = ' .                                                   DYNL0083
      WRITE OUTPT, STOP.                                                  DYNL0084
.                                                                         DYNL0085
.  OUTPUT CHANNEL SPECIFIER.                                              DYNL0086
.                                                                         DYNL0087
OUTPT, ELEMENT 00 00 04 00.                                               DYNL0088
      END DYNALC.                                                         DYNL0089
```

5. Етапи на реализацията на функционалния процесор HELP

5.1. ВХОДНО-ИЗХОДНА УПРАВЛЯВАЩА СИСТЕМА IOCS

Реализацията на входно-изходните операции е значително по-трудна от реализацията на останалите операции на предложените абстрактни машини. Опитът на редица специалисти в областта на програмирането показва, че за предпочитане е изолирането на абстрактните машини от конкретните конфигурации на реалните изчислителни машини чрез обработване на входно-изходните операции от специален модул. В представената разработка този модул улеснява съществено създаването на процесорите SIMCOM и BASCOM и е наречен Входно-изходна управляваща система - IOCS. С малки изменения, тази управляваща входа и изхода програма се използува и от процесора HELP.

Целта на IOCS е да обезпечи канали за обмен на данни. Всеки канал представлява връзка между логическо устройство на разположение на програмата-процесор и съответно физическо устройство от конфигурацията на реалната изчислителна машина. IOCS осигурява на програмиста достъп до неограничен брой логически устройства. Освен това, IOCS е достатъчно гъвкава и позволява включването на всевъзможни съвременни физически входно-изходни устройства.

Всички входно-изходни операции на процесорите се дефинират за логическите устройства - номер на устройството, желана операция и формат и разположение на данните в оперативната памет. Тази управляваща информация, обаче, е

недостатъчна за извършване на операцията. Необходимо е да се знае как се извършва операцията на съответното физическо устройство - адрес на устройството, формат на записа, буфери, управляващи флагове и т.н. Тази допълнителна управляваща информация се занася в IOCS чрез съответни подпрограми.

Съществува голямо разнообразие от входно-изходни устройства, но операциите, извършвани от тях, принадлежат към следващите категории:

1/ Четене - прехвърляне на данни от устройството към паметта,

2/ Запис - прехвърляне на данни от паметта към устройството,

3/ Управление - операции, при които не се прехвърлят данни.

В представената конкретна реализация операциите за отваряне на масиви са обособени отделно и представляват четвърта категория:

4/ Подготовка - подготовка на устройството за работа.

В главната програма на IOCS се дефинират логическите устройства. Дефиницията на логическо устройство трябва да включва шест типа указания:

- текущо състояние на логическото устройство,
- как това логическо устройство изпълнява операциите от изброените четири категории,
- кое е съответното физическо устройство.

IOCS проверява текущото състояние на логическото устройство, задейства и управлява работата на съответното физическо устройство, проверява завършването на операцията и обновява информацията за състоянието на логическото устройство. Краят на всяка входно-изходна операция може да бъде:

1/ нормален,

2/ откриване на EOF или EOM,

3/ откриване на неправилна операция за съответното устройство,

4/ откриване на специфични ненормални ситуации за различните устройства.

Представената IOCS открива само първите три състояния за край на операцията.

Когато за дадено логическо устройство в конфигурацията на изчислителната машина не е включено съответно физическо устройство, в дефиницията за начално състояние на логическото устройство се записва нула. Това означава, че каналът не е осъществен.

Функцията IOOP действува като супервизор в IOCS. Всяко извикване на IOCS за изпълнение на входно-изходна операция по принцип е еквивалентно на оператора на FORTRAN

JRES = IOOP(JOP,JDEV,JBA,JP1,JP2)

където JRES - резултат от изпълнението на операцията
JOP - код на операцията
JDEV - номер на логическото устройство
JBA - базов адрес на масива на данните
JP1, JP2 - указатели към масива

Подпрограмите IREAD, IWRIT, IOPEN и ICNTL изпълняват съответно операциите за четене, запис, подготовка и управление. Те отчитат характеристиките на съответните физически входно-изходни устройства. Тяхното действие се разбира от поясненията на съответните места в текстовете на подпрограмите.

Функцията IFEOF осигурява проверка за EOF /край на масив/ при операция за четене. Ако съответният език за програмиране позволява включване на проверката за EOF в операторите за четене, функцията IFEOF не е необходима.

Функцията ICVCI служи за преобразуване на кода на символите на конкретна изчислителна машина в цели положителни десетични трицифрени числа, представляващи вътрешното представяне на тези символи в процесорите SIMCOM, BASCOM и HELP. Функцията ICVIC служи за извършване на обратното преобразуване. От скоростта на преобразуване зависи в голяма степен скоростта на работа на процесорите, поради което е желателно тези две функции да бъдат кодирани в асамблеров код за съответната машина. Необходимо е вътрешните кодове на циф-

рите от 0 до 9 включително да се представят чрез последователни увеличаващи се числа. Символът CRLF /връщане на каретката и нов ред/ управлява въвеждането или извеждането на редове с променлива дължина за съответни устройства /пишуща машина, четец и перфоратор на лента/. Вътрешният код на този символ е произволно цяло отрицателно число, еквивалентно на NIL от езика WISP.

Текстът на входно-изходната управляваща система IOCS, използувана при реализацията за електронната изчислителна машина FACOM 230-45S, е показан в раздел 5.6 чрез масива IOCS.

Следва текстът на входно-изходната управляваща система IOCS, подготвен на ФОРЗ2 за електронната изчислителна машина МИНСК-32 и придружен с коментари.

F-ПРОГРАМА: MAINT


```
РЕД                     ОПЕРАТОР

001           ROUTINE MAIN
002     C
003     C ГЛАВНАТА ПРОГРАМА НА ВХОДНО-ИЗХОДНАТА УПРАВЛЯВАЩА СИСТЕМА ЗАРЕЖДА
004     C УПРАВЛЯВАЩИТЕ МАСИВИ ЗА КАНАЛИТЕ И ЗАТВАРЯ КАНАЛИТЕ В КРАЯ НА
005     C РАБОТАТА. УПРАВЛЯВАЩИТЕ МАСИВИ ИМАТ СЛЕДНАТА СТРУКТУРА:
006     C
007     C   JCHAN(KCH+1) = УПРАВЛЯВАЩ КОД ЗА ЧЕТЕНЕ
008     C   JCHAN(KCH+2) = УПРАВЛЯВАЩ КОД ЗА ЗАПИС
009     C   JCHAN(KCH+3) = УПРАВЛЯВАЩ КОД ЗА ОТВАРЯНЕ
010     C   JCHAN(KCH+4) = УПРАВЛЯВАЩ КОД ЗА УПРАВЛЕНИЕ
011     C   JCHAN(KCH+5) = НОМЕР НА УСТРОЙСТВОТО - FN
012     C   JCHAN(KCH+6) = НАЧАЛНО СЪСТОЯНИЕ НА КАНАЛА
013     C   KCH=KCH+6
014     C
015     C ЗНАЧЕНИЯ НА КОДОВЕТЕ:
016     C
017     C   ЧЕТЕНЕ       0 - ЧЕТЕНЕТО Е НЕПОЗВОЛЕНО
018     C                1 - ЧЕТЕНЕ НА РЕД ОТ 80 ЗНАКА, ПРЕОБРАЗУВАНЕ НА ВСЕКИ
019     C                    ЗНАК В ЦЯЛО И ЗАПИСВАНЕ В МАСИВА НА
020     C                    ПОТРЕБИТЕЛЯ
021     C
022     C   ЗАПИС        0 - ЗАПИСЪТ Е НЕПОЗВОЛЕН
023     C                1 - ЗАПИС НА РЕД ОТ 80 ЗНАКА СЛЕД ПРЕОБРАЗУВАНЕ НА
024     C                    ЦЕЛИТЕ ЧИСЛА В ЗНАЦИ
025     C                2 - ЗАПИС С УПРАВЛЯВАЩ СИМВОЛ ЗА НОВ РЕД
026     C                3 - ЗАПИС С УПРАВЛЯВАЩ СИМВОЛ ЗА НОВА СТРАНИЦА
027     C
028     C   ОТВАРЯНЕ     0 - НЕ СЕ ИЗИСКВАТ ДЕЙСТВИЯ ЗА ОТВАРЯНЕ НА МАСИВА
029     C                1 - УСТРОЙСТВОТО СЕ ПРЕНАВИВА ПРИ ВСЯКО ОТВАРЯНЕ
030     C
031     C   УПРАВЛЕНИЕ   0 - НЯМА УПРАВЛЯВАЩИ ОПЕРАЦИИ
032     C                1 - END OF FILE CONTROL
033     C                2 - END OF MEDIUM CONTROL
034     C
035     C   СЪСТОЯНИЕ НА КАНАЛА
036     C                0 - НЯМА ТОЗИ КАНАЛ
037     C                1 - КАНАЛЪТ Е ЗАТВОРЕН
038     C                2 - КАНАЛЪТ Е ОТВОРЕН ЗА ЧЕТЕНЕ
039     C                3 - КАНАЛЪТ Е ОТВОРЕН ЗА ЗАПИС
040     C                4 - КАНАЛЪТ Е ОТВОРЕН ЗА ЧЕТЕНЕ И ЗАПИС
041     C                5 - ОТКРИТО Е СЪСТОЯНИЕ EOF ИЛИ EOM
042     C
043     C ПРЕДСТАВЕНАТА ВХОДНО-ИЗХОДНА УПРАВЛЯВАЩА СИСТЕМА ПОЗВОЛЯВА ВКЛЮЧВА
044     C НЕ НА НОВИ КАНАЛИ ИЛИ ИЗКЛЮЧВАНЕ НА КАНАЛИ ДА СЕ ИЗВЬРШВА МНОГО
045     C ЛЕСНО ЧРЕЗ УПРАВЛЯВАЩИТЕ МАСИВИ.
046     C
047           DIMENSION JCHAN(30)
048           COMMON MAXCH,JCHAN
049           EXTERNAL PROGR
050           EXTERNAL IOOPS
051           KCH=0
052     C
053     C ВХ-ИЗХ КАНАЛ 1   ЧЕТЕЦ НА КАРТИ
054           JCHAN(KCH+1)=1
055           JCHAN(KCH+2)=0
```

ИСТ РЕД ОПЕРАТОР

```
056           JCHAN(KCH+3)=0
057           JCHAN(KCH+4)=1
058           JCHAN(KCH+5)=1
059           JCHAN(KCH+6)=1
060           KCH=KCH+6
061     C
062     C ВХ-ИЗХ КАНАЛ 2   МАГНИТНА ЛЕНТА
063           JCHAN(KCH+1)=2
064           JCHAN(KCH+2)=4
065           JCHAN(KCH+3)=1
066           JCHAN(KCH+4)=2
067           JCHAN(KCH+5)=7
068           JCHAN(KCH+6)=1
069           KCH=KCH+6
070     C
071     C ВХ-ИЗХ КАНАЛ 3   ПЕРФОРАТОР НА КАРТИ
072           JCHAN(KCH+1)=0
073           JCHAN(KCH+2)=1
074           JCHAN(KCH+3)=0
075           JCHAN(KCH+4)=0
076           JCHAN(KCH+5)=3
077           JCHAN(KCH+6)=1
078           KCH=KCH+6
079     C
080     C ВХ-ИЗХ КАНАЛ 4   ШИРОК ПЕЧАТ
081           JCHAN(KCH+1)=0
082           JCHAN(KCH+2)=3
083           JCHAN(KCH+3)=0
084           JCHAN(KCH+4)=0
085           JCHAN(KCH+5)=6
086           JCHAN(KCH+6)=1
087           KCH=KCH+6
088     C
089     C ВХ-ИЗХ КАНАЛ 5   МАГНИТНА ЛЕНТА
090           JCHAN(KCH+1)=2
091           JCHAN(KCH+2)=4
092           JCHAN(KCH+3)=1
093           JCHAN(KCH+4)=2
094           JCHAN(KCH+5)=8
095           JCHAN(KCH+6)=1
096           KCH=KCH+6
097     C
098     C ЗАРЕЖДАНЕ НА БРОЯ НА КАНАЛИТЕ
099     C
100           MAXCH=KCH/6
101     C
102     C ИЗВИКВАНЕ НА ПОТРЕБИТЕЛСКАТА ПРОГРАМА. ВСИЧКИ ПОТРЕБИТЕЛСКИ ПРОГРА-
103     C МИ СЕ КОМПИЛИРАТ КАТО ПОДПРОГРАМИ С ИМЕ PROGR. ТОВА МОЖЕ ДА БЪДЕ
104     C SIMC,STG2,BASC ИЛИ ТЕСТ ПРОГРАМА.
105     C
106           CALL PROGR (ПАРАМ)
107     C
108     C ЗАТЪВАРЯНЕ НА ВСИЧКИ МАСИВИ
109     C
110           DO 1 K=1,MAXCH
```

F-ПРОГРАМА: MAINT

ЮСТ РЕД ОПЕРАТОР

```
111          CALL IOOPS (0,K,JCHAN,1,1,IOOP)
112          KCH=IOOP
113        1 CONTINUE
114    C
115    C НОРМАЛЕН КРАЙ НА ПРОГРАМАТА
116    C
117          STOP
118          END
```

F-ПРОГРАМА: IOOST


```
ИСТ РЕД                    ОПЕРАТОР

00  001          SUBROUTINE IOOPS(JOP,JCH,JBA,JP1,JP2,IOOP)
    002    C
    003    C          ПОДПРОГРАМАТА IOOPS ЗАМЕСТВА ФУНКЦИЯТА IOOP И ПОДПРОГРАМИТЕ
    004    C          IREAD, IWRIT, IOPEN, ICNTL И IFFOF. ПО ТОЗИ НАЧИН СЕ СПЕС-
    005    C          ТЯВА ОПЕРАТИВНА ПАМЕТ - ОКОЛО 3000 МАШИННИ ДУМИ. ОСВЕН ТОВА
    006    C          Е НЕОБХОДИМО ФУНКЦИЯТА IOOP ДА ВР ЩА И ДРУГА СТОЙНОСТ (JP2),
    007    C          КОЕТО НЕ Е ПОЗВОЛЕНО В В ФОР32 И ПОРАДИ ТОВА ТРЯБВА ДА СЕ
    008    C          ИЗПОЛЗУВА ПОДПРОГРАМА.
    009    C
    010    C          ПОДПРОГРАМАТА IOOPS УПРАВЛЯВА И ИЗП ЛНЯВА ВХОДНО-ИЗХОДНИТЕ
    011    C          ОПЕРАЦИИ
    012    C          JOP Е КОД НА ОПЕРАЦИЯТА
    013    C          JCH Е НОМЕР НА КАНАЛА
    014    C          JBA Е БАЗОВ АДРЕС НА БУФЕРА
    015    C          JP1 И JP2 СА УКАЗАТЕЛИ, СВ РЗАНИ С БУФЕРА
    016    C
    017    C          СТОЙНОСТТА НА IOOP ОТРАЗЯВА ЗАВ РШЕКА НА ОПЕРАЦИЯТА
    018    C               IOOP = 0  НОРМАЛЕН ЗАВ РШЕК
    019    C               IOOP = 1  ОТКРИТ Е EOF ИЛИ EOM
    020    C               IOOP = 2  НЕПРАВИЛНА ОПЕРАЦИЯ ЗА С ОТВЕТНИЯ КАНАЛ
    021    C
    022          DIMENSION JCHAN(30),JBA(1),JTB(80),LINE(127)
    023          EXTERNAL ICVCI,ICVIC
    024          COMMON MAXCH,JCHAN
    025    C
    026    C          ПРОВЕРКА ЗА ЛЕГАЛНОСТ НА НОМЕРА НА КАНАЛА
    027          IF (JCH) 407,405,401
    028      401 IF (MAXCH-JCH) 407,451,451
    029    C
    030    C          ИЗЧИСЛЯВАНЕ НА АДРЕСА НА УПРАВЛ. ИНФ. ЗА С ОТВЕТНИЯ КАНАЛ
    031      451 KCH=(JCH-1)*6
    032    C
    033    C          ПОДГОТОВКА ЗА ИЗЧИСЛЯЕМ GOTO В ЗАВИСИМОСТ ОТ С СТ. НА КАНАЛА
    034          K=JCHAN(KCH+6)
    035          IF (K) 452,407,452
    036    C
    037    C          ПРЕХОД В ЗАВИСИМОСТ ОТ ТИПА НА ОПЕРАЦИЯТА
    038      452 IF (JOP) 410,402,420
    039    C
    040    C                    УПРАВЛЕНИЕ
    041    C
    042    C          АКО Е НЕОБХОДИМО СЕ ИЗВ РШВА ЗАТВАРЯНЕ НА МАСИВА
    043      402 IF (JCHAN(KCH+6)-1) 453,407,453
    044      453 IF (JCHAN(KCH+4)) 454,403,454
    045      454 IF (JCHAN(KCH+4)+1-JCHAN(KCH+6)) 503,551,503
    046      551 JDEV=JCHAN(KCH+5)
    047          IF (JCHAN(KCH+4)-2) 501,502,501
    048      501 READ(JDEV,100) JTB
    049          IF (JTB(1)-171798691184) 552,503,552
    050      552 GOTO 501
    051      502 ENDFILE JDEV
    052      503 JCHAN(KCH+6)=1
    053          IOOP=0
    054          RETURN
    055    C
```

```
ЛИСТ РЕД                    ОПЕРАТОР

056   C          ОБРАБОТКА НА ИСКАНЕ ЗА ЗАТВАРЯНЕ
057     403 IF (JP1-JP2) 407,455,407
058     455 JCHAN(KCH+6)=1
059   C
060   C          ВР ЩАНЕ ПРИ УСПЕШНО ЗАВ РШВАНЕ НА ОПЕРАЦИЯТА
061     404 IOOP=0
062         RETURN
063   C
064   C          ЗА КАНАЛ 0 - ЗАПИС И ЗАТВАРЯНЕ И EOF ПРИ ЧЕТЕНЕ СЕ ИГНОРИРАТ
065     405 IF (JOP) 406,406,404
066   C
067   C          ВР ЩАНЕ ПРИ ОТКРИТ EOF
068     406 IOOP=1
069         RETURN
070   C
071   C          ВР ЩАНЕ ПРИ НЕПРАВИЛНА ОПЕРАЦИЯ
072     407 IOOP=2
073         RETURN
074   C
075   C                ЧЕТЕНЕ
076   C
077   C          ПРЕХОД В ЗАВИСИМОСТ ОТ С СТОЯНИЕТО НА КАНАЛА
078     410 GOTO (411,412,407,412,406), K
079   C
080   C          НЕПОЗВОЛЕНО ЧЕТЕНЕ
081     411 IF (JCHAN(KCH+1)) 456,407,456
082     456 JCHAN(KCH+6)=2
083   C
084   C          ОТВАРЯНЕ НА МАСИВА
085         IF (JCHAN(KCH+3)) 457,412,457
086     457 JDEV=JCHAN(KCH+5)
087         REWIND JDEV
088   C
089   C          ПРЕХВ РЛЯНЕ НА ЕДИН ЗАПИС С ФОРМАТ A1
090     412 IOOP=0
091   C
092   C          ДЕФИНИРАНЕ НА Д ЛЖИНАТА НА РЕДА
093         LNLNG=80
094   C
095   C          НОМЕР НА УСТРОЙСТВОТО
096         JDEV=JCHAN(KCH+5)
097   C
098   C          ЗАНАСЯНЕ НА СТОЙНОСТ НА ПАРАМЕТ РА JP2
099         JP2=JP1+LNLNG-1
100   C
101   C          ЧЕТЕНЕ НА ЕДИН ЗАПИС
102         READ(JDEV,100) (JBA(K),K=JP1,JP2)
103   C
104   C          ПРОВЕРКА ЗА EOF
105         IF (JBA(JP1)-171798691184) 651,602,651
106   C
107   C          ПРЕКОДИРАНЕ ОТ ФОРМАТ A1 В ЦЯЛО ПОЛОЖИТЕЛНО ЧИСЛО
108     651 DO 800 K=JP1,JP2
109         JBA(K)=ICVCI(JBA(K))
110     800 CONTINUE
```

```
      C
      C        ЗАРЕЖДАНЕ НА УКАЗАТЕЛЯ JP2 И НОРМАЛНО ВРЪЩАНЕ
            JP2=JP2+1
            RETURN
      C
      C        ВРЪЩАНЕ С ИНДИКАЦИЯ ЗА EOF
        602 IOOP=1
            JCHAN(KCH+6)=5
            RETURN
      C
      C                 ЗАПИС
      C
      C        ПРЕХОД В ЗАВИСИМОСТ ОТ С СТОЯНИЕТО НА КАНАЛА
        420 GOTO (421,407,422,422,406), K
      C
      C        НЕПОЗВОЛЕН ЗАПИС
        421 IF (JCHAN(KCH+2)) 459,407,459
        459 JCHAN(KCH+6)=3
      C
      C        ОТВАРЯНЕ НА МАСИВА
            IF (JCHAN(KCH+3)) 460,422,460
        460 JDEV=JCHAN(KCH+5)
            REWIND JDEV
      C
      C        ПРЕХВЪРЛЯНЕ НА ЕДИН ЗАПИС С ФОРМАТ A1
        422 IOOP=0
      C
      C        ДЕФИНИРАНЕ НА ДЪЛЖИНАТА НА РЕДОВЕТЕ
            LNLNG=80
            KMAX=127
      C
      C        ДЕФИНИРАНЕ НА ШПАЦИЯ
            KFILL=-66571993088
      C
      C        КОД НА ОПЕРАЦИЯТА И НОМЕР НА УСТРОЙСТВОТО
            JHOW=JCHAN(KCH+2)
            JDEV=JCHAN(KCH+5)
      C
      C        КОГАТО JP1=JP2 СЕ ИЗВЕЖДА ПРАЗЕН РЕД
            IF (JP1-JP2) 751,702,751
      C
      C        ПРЕКОДИРАНЕ ОТ ЦЯЛО ПОЛОЖИТЕЛНО ЧИСЛО В В ФОРМАТ A1
        751 K1=JP1
            DO 900 K=1,KMAX
            LINE(K)=ICVIC(JBA(K1))
            K1=K1+1
            IF (K1-JP2) 900,703,900
        900 CONTINUE
            K=KMAX
            GOTO 703
      C
      C        ПОДГОТОВКА ЗА ИЗВЕЖДАНЕ НА ПРАЗЕН РЕД
        702 LINE(1)=KFILL
            K=1
        703 GOTO (710,720,730), JHOW
```

F-ПРОГРАМА: IOOST

ист ред ОПЕРАТОР

```
      C
      C        ДОП ЛВАНЕ НА РЕДА С ШПАЦИИ ЗА СР
  710 IF (K-LNLNG) 752,752,712
  752 I=K+1
      DO 711 K=I,LNLNG
  711 LINE(K)=KFILL
      C
      C        ПЕРФОРИРАНЕ НА КАРТА НА СР
  712 WRITE(JDEV,100) (LINE(K),K=1,LNLNG)
      RETURN
      C
      C        ОТПЕЧАТВАНЕ НА РЕД НА LP ИЛИ TYP
  720 WRITE(JDEV,101) (LINE(I),I=1,K)
      RETURN
      C
      C        НОВА СТРАНИЦА И НУЛИРАНЕ НА КОДА ЗА НОВА СТРАНИЦА
  730 WRITE(JDEV,102) (LINE(I),I=1,K)
      JCHAN(KCH+2)=2
      RETURN
      C
      C        ФОРМАТИ
  100 FORMAT (80A1)
  101 FORMAT (127A1)
  102 FORMAT (1H1,127A1)
      C
      C        КРАЙ НА ПОДПРОГРАМАТА
      END
```

ССК МИНСК-32

РП -ICVCI

СИМП-ССКСI

```
**          -70 00 0 0000 0 1000
        ОБЛАСТЬ     1                    001 010        БАЗ      0
0 000000     00 00 0 0000 0 0000         001 020        НОП
**          -70 00 0 0040 0 1000
0 000001-0 000003                        001 030        РЗВ      3
0 000004    -05 00 0 0003 0 0001         001 040        ЧСП      0,3;1
0 000005    -17 00 1 0001 0 0000         001 050        ЗУ       1
0 000006     61 00 0 0011 3 0000         001 060        ЛСД      +136В;3,0
0 000007    -30 00 0 0010 2 0000         001 070        З        2,0
0 000010    -21 00 0 0000 0 0001         001 080        ВЫХ      0,0;1
0 000011     00 00 0 0000 0 0136      *                 КЧ       +136В
```

ССК МИНСК-32

РП -ICVIC

СИМП-ССКIC

```
**          -70 00 0 0000 0 1000
        ОБЛАСТЬ     1                   001 010     БАЗ     0
0 000000    00 00 0 0000 0 0000         001 020     НОП
**          -70 00 0 0040 0 1000
0 000001-0 000003                       001 030     РЗВ     3
0 000004    -05 00 0 0003 0 0001        001 040     ЧСП     0,3;1
0 000005    -17 00 1 0001 0 0000        001 050     ЗУ      1
0 000006     61 00 0 0011 3 0000        001 060     ЛСД     +36В;3,0
0 000007    -30 00 0 0010 2 0000        001 070     З       2,0
0 000010    -21 00 0 0000 0 0001        001 080     ВЫХ     0,0;1
0 000011    00 00 0 0000 0 0036     *               КЧ      +36В
```

5.2. ПРОСТ КОМПИЛАТОР SIMCOM

Процесорът, върху който се базира първият етап на реализацията, се нарича Прост компилатор - SIMCOM. За да може да се реализира практически на всякаква електронна изчислителна машина, трябва да е възможно неговото кодиране на елементарен асамблеров код. Изходът му може да бъде също асамблеров код.

Възможен е и друг подход. Ако SIMCOM бъде написан на FORTRAN, той ще може да се използува без изменения на всяка изчислителна машина, снабдена с транслатор от FORTRAN. Освен това, изходът му може да бъде също фортранова програма. Тази възможност е възприета в представената реализация.

Простият компилатор в основата си е програма за обработка на символни низове чрез подбиране по образец и заместване. Подбирането по образец се използува за разпознаване на изходните оператори на езика на абстрактната машина FLBM, а заместването - за конструиране на оператори на обектен език. Двата процеса са ограничени до минимални възможности, тъй като процесорът SIMCOM трябва да се програмира ръчно. Това ограничение се отразява върху възможностите на езика FLUB.

Простият компилатор SIMCOM изисква наличието на входно-изходната управляваща система IOCS, която служи за управляване на въвеждането на изходните символни низове от входно устройство с последователна организация и достъп и извеждането на обектните символни низове на подобно изходно устройство. В простия компилатор всеки символ се представя във вътрешен код, който е цяло положително десетично трицифрено число. Наборът от символи се определя от реализацията, като е възможно да се добавят нови или да се премахват някои от използуваните символи, за което са предназначени функциите ICVCI и ICVIC на входно-изходната управляваща система.

Изходни низове за SIMCOM са операторите от езика FLUB на абстрактната

машина FLBM. Всеки оператор трябва да указва операцията за изпълнение и операндите за съответното изпълнение. Дадена операция може да се изпълни много пъти, но операндите за всяко изпълнение ще бъдат различни. Шаблоните с които борави SIMCOM включват фиксирани низове /за сравняване на операциите/ и елементи, които специфицират структури /за сравняване на операндите/. Единичният символ е най-прост низ с фиксирана дължина и всеки низ с фиксирана дължина различна от единица може да се изрази чрез сцепление на единични символи. За да се достигне желаната простота на SIMCOM, за шаблон е приет единичен символ, т.е. структурните елементи могат да бъдат само единични символи.

Необходими са четири символа, които имат специално значение за дефинирането на операциите на абстрактната машина чрез масива FLBM. Първите два символа се използуват при операциите за подбиране по образец. Единият служи за указване на края на изходните оператори, а другият - за указване на мястото на операндите в тези оператори. Първият се нарича флаг за край на изходен оператор, а вторият - флаг за параметър. При реализацията се използуват съответно символите "." и "*", но те могат да се променят чрез промяна в първа и втора колонки на първата карта на масива FLBM.

След извършване на разпознаването на изходния оператор, необходимо е да се извърши конструиране на обектните оператори. За управление на този процес се използуват другите два символа със специално предназначение. Това са флаговете за указване на край на обектен оператор и за място на параметър в такъв оператор. При реализацията се използуват съответно символите "@" и "#". Те също могат да бъдат променяни чрез промяна в трета и четвърта колонки на първата карта на масива FLBM. За представената реализация тази карта съдържа:

.'@'0

В пета колонка е записан символът нула.

Ограниченията, които SIMCOM налага върху езика FLUB са следните:

1/ Всеки изходен оператор заема един ред /карта/ и започва от първа позиция на реда. Операторите трябва да завършват със символа за край на изходен оператор и да не съдържат излишни шпации.

2/ Операндите не могат да се състоят от по-вече от един символ и трябва да адресират базовите регистри на абстрактната машина.

3/ Етикетите се дефинират чрез специален оператор и трябва да се състоят от две десетични цифри, поне едната от които да не е нула.

На фиг.5.1 е показан пълният набор оператори на езика FLUB, като местата за операнди са обозначени с апостроф.

За да бъде обяснена обработката на параметрите, възприети са следните съкращения за записване на формата и заместването им:

e - символ за място на параметър в обектен ред /четвърти символ на първата карта на масива FLBM/.

d - цифра между 1 и 9 включително /номер на параметър/.

l - цифра между 1 и 9 включително /трансформация на параметър/.

Простият компилатор SIMCOM обезпечава два начина за заместване на параметър в конструирания обектен ред:

- Заместване на параметър по име

Формат: ed0

Действие: Трите символа на формата от конструирания обектен ред се заместват с името на параметъра d от подлежащия на обработка изходен ред.

Примери: фиг.5.2.1, фиг.5.2.2.

- Заместване на параметър по стойност

Формат: ed1

Действие: Трите символа на формата от конструирания обектен ред се заместват с трицифреното десетично число, което е вътрешното представяне на този параметър.

Примери: фиг.5.2.3, фиг.5.2.4.

Необходимостта от генератор на символи в SIMCOM произтича от фиг.5.2.4. Параметър 0 служи за генериране на трицифрено десетично число. Първоначално зареденатa стойност е 100. Извикването на параметър '0m /m е число от 0 до 9 включително/ занася сумата на m и текущата стойност на генератора в конструирания ред на мястото на формата '0m. Стойността на генератора не се променя до пълното извеждане на обектните оператори на съответната дефиниция, след което тази стойност се увеличава с $m_{max}+1$.

Въпреки ограничени, възможностите на SIMCOM за обработка на параметри и вградения генератор на символи позволяват сравнително леко описване на езика FLUB чрез команди на език от тип асамблер или чрез оператори на език от по-високо ниво. За предстaвената реализация за електронната изчислителна машина FACOM 230-45S като език на вграждане е използуван FORTRAN, а дефинициите на операторите на FLUB чрез оператори на FORTRAN са представени в раздел 5.6 чрез масива FLBM. Първата карта /запис/ на този масив съдържа специалните символи, след които следват картите /записите/ на дефинициите. Всяка дефиниция започва с шаблон на изходен оператор, следван от шаблоните на заместващите го обектни оператори и завършва с карта /запис/ на един символ за край на обектен ред в първа позиция.

Необходимо е да се отбележи, че при съставянето на дефинициите е възможно допускането на грешки, някои от които се откриват много трудно. Това налага да бъдат предвидени средства за проверка на съставените дефиниции. За проверката на дефинициите на операторите на езика FLUB, т.е. моделирането на абстрактната машина FLBM, са предвидени два етапа, отразени в параграфи 5.5.3 и 5.5.4. При всеки от тях се използува масив с програма на езика FLUB за извършване на тестването и масив със специално подготвени за тази програма данни. Програмата се компилира най-напред от FLUB на FORTRAN, а след това

от FORTRAN в машинна форма. Получената машинна програма се изпълнява с подготвените данни и от съобщенията, които се отпечатват, може да се прецени къде да се търси грешката или дали кодирането на дефинициите е правилно.

Работата на простия компилатор SIMCOM протича в следната последователност:

- Прочитане на масива FLBM и запомняне в паметта на специалните символи от първата карта и следващите дефиниции на операторите на изходния език чрез оператори на обектния език.
- След това, операторите на изходната програма се въвеждат /четат/ и обработват един по един.
- Всеки прочетен оператор се сравнява с шаблоните на изходните оператори от дефинициите и при откриване на съответствие се извършва занасяне на параметрите от обработвания изходен оператор на съответните места в шаблоните на обектните оператори от дефиницията.
- Така получените обектни оператори се извеждат /перфорират/, след което се преминава към обработка на следващия оператор от изходната програма.

Следователно, дефинициите на операциите на абстрактната машина, следвани от изходната програма, кодирана на езика на абстрактната машина, представляват данни за простия компилатор SIMCOM. Програмата се отделя от дефинициите чрез перфокарта /запис/ с два последователни флага за край на обектен оператор в първа и втора колонки /позиции/.

Ако не бъде открито съответствие между поредния обработван изходен оператор на програмата и някой от шаблоните на дефинициите, изходният оператор се извежда без промяна. Това действие на простия компилатор се използува не само за откриване на грешно написани изходни оператори, но и като странично ефект за директно въвеждане на оператори в получаваната обектна програма. Всички редове от масива FLBM, следващи картата с два последователни фла-

га за край на дефинирането, се копират директно в обектната програма.

Простият компилатор дава индикация и прекратява работата си при следните ненормални ситуации:

STOP 10 - Четенето е непозволено за канал 1 или е открит край на масива при опит за четене от този канал.

STOP 11 - Открит е край на масив за канал 1 при опит за четене на шаблон на изходен оператор.

STOP 12 - Открит е край на масив за канал 1 при опит за четене на шаблон на обектен оператор.

STOP 20 - Изчерпване на паметта при опит за запомняне на шаблон на изходен оператор.

STOP 21 - Изчерпване на паметта при опит за запомняне на шаблон на обектен оператор.

STOP 22 - Изчерпване на паметта при опит за запомняне на изходен оператор.

STOP 23 - Изчерпване на паметта при опит за преобразуване на символ за параметър в низ от цифри.

STOP 10 индицира грешка в някоя изпълнителна подпрограма или грешно занасяне на характеристиките на устройството за канал 1 в главната програма на входно-изходната управляваща система IOCS.

STOP 11 и STOP 12 обикновено са следствие на неправилно указани флагове в първата карта на масива на дефинициите. Трябва да се провери и завършването на дефинициите с два флага за край на шаблон на обектен ред. Обикновена грешка е завършване на последната дефиниция с един флаг за край на нов ред и след това два флага на следващия нов ред. При това положение двата флага няма да спрат фазата на дефиниране.

STOP 20 до STOP 23 се дължат на недостиг на оперативна памет. Необходимо е да се увеличи работния масив LIST, за да се обезпечи допълнително памет. В този случай е необходимо да се корегира и стойността на KMAX, която осигурява работа в границите на дефинираната чрез масив LIST памет.

Всеки елемент от паметта на абстрактната машина FLBM се състои от три полета - FLG, VAL и PTR. FLG полето е еквивалентно по значение на AF и BPF полетата, VAL - на CAR полето и PTR - на CDR полето на елемент от паметта на абстрактната машина WSPM, представена в трета глава на дисертацията. При използването на FORTRAN като език на вграждане всяко поле има дължина на цяло число, вследствие на което се получава неефективно използуване на оперативната памет на реалната изчислителна машина. Оптимизиране, обаче, не е необходимо, тъй като простият компилатор SIMCOM се използува само за създаването на аналогичния по предназначение, но с много по-големи възможности основен компилатор BASCOM.

Текстът на простия компилатор SIMCOM, използуван при реализацията за електронната изчислителна машина FACOM 230-45S, е показан в раздел 5.6 чрез масива SIMC.

Следва текстът на простия компилатор, подготвен на ФОР32 за електронната изчислителна машина МИНСК-32 и придружен с коментари. В този вариант се наложи прибавянето на втори генератор на символи. За тази цел параметър 9 беше преобразуван в генератор, но с възможност да се използува също и като параметър.

ЛИСТ РЕД ОПЕРАТОР

```
000 001          SUBROUTINE PROGR (ПАРАМ)
    002   C
    003   C        ИЗПОЛЗУВА СЕ РАБОТЕН МАСИВ LIST
    004   C
    005          DIMENSION LIST(12000),MIST(12000)
    006          EXTERNAL IOOPS
    007          EQUIVALENCE (LIST(1),MIST(1))
    008   C
    009   C        УСТАНОВЯВАНЕ НА ГРАНИЦА НА МАСИВА ЗА ДА СЕ КОНТРОЛИРА
    010   C        ПРЕПЪЛВАНЕТО МУ
    011   C
    012          KMAX=12000-80
    013   C
    014   C        МАСИВЪТ LIST СЕ ИЗПОЛЗУВА ПО СЛЕДНИЯ НАЧИН
    015   C             LIST(1) = ЗНАК ЗА КРАЙ НА РЕД ОТ ИЗХОДНАТА ПРОГРАМА
    016   C             LIST(2) = ФЛАГ ЗА ПАРАМЕТЪР В ИЗХОДНАТА ПРОГРАМА
    017   C             LIST(3) = ЗНАК ЗА КРАЙ НА КОДОВ РЕД ОТ ДЕФИНИЦИЯ
    018   C             LIST(4) = ФЛАГ ЗА ПАРАМЕТЪР В ДЕФИНИЦИЯ
    019   C             LIST(5) = ЗНАК НУЛА
    020   C             LIST(6) = ОСНОВЕН ГЕНЕРАТОР НА СИМВОЛИ (ПАРАМЕТЪР 0)
    021   C             LIST(7) = МЯСТО ЗА ПАРАМЕТЪР 0
    022   C             LIST(8) =                    1
    023   C             LIST(9) =                    2
    024   C             LIST(10) =                   3
    025   C             LIST(11) =                   4
    026   C             LIST(12) =                   5
    027   C             LIST(13) =                   6
    028   C             LIST(14) =                   7
    029   C             LIST(15) =                   8
    030   C             LIST(16) =                   9
    031   C             LIST(17) = СПОМАГАТЕЛЕН ГЕНЕРАТОР НА СИМВОЛИ (ПАРАМ 9)
    032   C          ОТ LIST(18) ДО LIST(12000) = МАКРОДЕФИНИЦИИ И РЕДОВЕ НА
    033   C                                       ИЗХОДНАТА И ОБЕКТНАТА ПРОГР.
    034   C
    035   C        ПРОМЕНЛИВИТЕ I,J,K,L,M,N СЕ ИЗПОЛЗУВАТ КАТО УКАЗАТЕЛИ И
    036   C        СПОМАГАТЕЛНИ ВЕЛИЧИНИ
    037   C
    038   C        ВСЯКА МАКРОДЕФИНИЦИЯ СЕ ЗАПИСВА В РАБОТНИЯ МАСИВ LIST КАТО
    039   C        БЛОК. АКО LIST(M) Е ПЪРВИЯТ ЕЛЕМЕНТ НА БЛОКА, ТОГАВА
    040   C             LIST(M) = ИНДЕКС ОТ КЪДЕТО ЗАПОЧВА СЛЕДВАЩАТА ДЕФИНИЦИЯ
    041   C             LIST(M+1) = НАЙ-ГОЛЕМИЯТ МОДИФИКАТОР НА ПАРАМ 0 СПОРЕД
    042   C                         СЪОТВЕТНАТА ДЕФИНИЦИЯ. АКО ПАРАМ 0 НЕ СЕ
    043   C                         ИЗПОЛЗУВА LIST(M+1)=-1. СТОЙНОСТТА НА ОСН.
    044   C                         ГЕН. ЩЕ СЕ УВЕЛИЧИ С (LIST(M+1)+1) СЛЕД
    045   C                         ПЪЛНОТО СКАНИРАНЕ НА ДЕФИНИЦИЯТА
    046   C             LIST(M+2) = НАЙ-ГОЛЕМИЯТ МОДИФИКАТОР НА ПАРАМ 9, АНАЛО-
    047   C                         ГИЧНО НА LIST(M+1)
    048   C             LIST(M+3) = ПЪРВИ ЗНАК НА ШАБЛОНА. ФЛАГЪТ ЗА КРАЙ НА
    049   C                         РЕДА СЕ ЗАМЕСТВА С -1. СЛЕДВАТ КОДОВИТЕ
    050   C                         РЕДОВЕ ОТ ДЕФИНИЦИЯТА, ВСЕКИ ЗАВЪРШВАЩ С -1
    051   C
    052   C        В КОДОВИТЕ РЕДОВЕ ОТ ДЕФИНИЦИЯТА ВСЯКО ИЗВИКВАНЕ НА ПАРАМЕ-
    053   C        ТЪР СЕ ПРЕДСТАВЯ ЧРЕЗ ТРИ ЕЛЕМЕНТА ОТ МАСИВА LIST. ПЪРВИЯТ
    054   C        СЪДЪРЖА -2, ВТОРИЯТ ПРЕДСТАВЛЯВА ИНДЕКСА НА МЯСТОТО ЗА СЪОТ-
    055   C        ВЕТНИЯ ПАРАМЕТЪР, А ТРЕТИЯТ Е ЦИФРАТА-МОДИФИКАТОР НА ПАРАМЕ-
```

```
056   C          ТЬРА, НАПРИМЕР ↑41 ЩЕ СЕ ЗАПИШЕ:
057   C               -2
058   C               +11
059   C               +1
060   C
061   C          ЧЕТЕНЕ НА КАРТАТА С ФЛАГОВЕ И СПЕЦИАЛНИ ЗНАЦИ
062   C
063        CALL IOOPS(-1,1,LIST,1,I,IOOP)
064        IF (IOOP) 50,51,50
065     50 PAUSE 1
066        RETURN
067   C
068   C          ЗАНАСЯНЕ НА НАЧАЛНА СТОЙНОСТ В ОСН. ГЕН.
069   C
070     51 MIST(6)=100
071   C
072   C          ЗАНАСЯНЕ НА НАЧАЛНА СТОЙНОСТ В СПОМ. ГЕН.
073   C
074        MIST(17)=200
075   C
076   C          ИНДЕКС ЗА ВЬЗПРИЕМАНЕ НА ПЬРВАТА ДЕФИНИЦИЯ
077   C
078        K=19
079   C
080   C          НАЧАЛНО ЗАРЕЖДАНЕ НА МОДИФИКАТОРИТЕ ЗА ПАРАМ 0 И 9 (ГЕНЕР)
081   C
082      1 MIST(K)=-1
083        MIST(K+1)=-1
084   C
085   C          ЧЕТЕНЕ НА ШАБЛОНА НА МАКРОДЕФИНИЦИЯТА
086   C
087        CALL IOOPS(-1,1,LIST,K+2,I,IOOP)
088        IF (IOOP) 52,53,52
089     52 PAUSE 2
090        RETURN
091   C
092   C          ПРОВЕРКА ЗА ПРЕПЬЛВАНЕ НА МАСИВА LIST
093   C
094     53 IF (I-KMAX) 55,55,54
095     54 PAUSE 4
096        RETURN
097   C
098   C          ГАРАНТИРАНЕ НА ЗНАК ЗА КРАЙ НА ИЗХАДЕН РЕД В КРАЯ НА ШАБЛОНА
099   C
100     55 MIST(I)=LIST(1)
101   C
102   C          СКАНИРАНЕ НА ШАБЛОНА ЗА ЗНАК ЗА КРАЙ НА ИЗХОДЕН РЕД
103   C
104        I=K+1
105      2 I=I+1
106        IF (LIST(I)-LIST(1)) 2,13,2
107   C
108   C          ОБРАБОТКА НА КОДИРАНИТЕ РЕДОВЕ ОТ ДЕФИНИЦИИТЕ, ТЕ СЕ ЧЕТАТ
109   C          ПРИ ЕТИКЕТ 13. ПРИ ЕТИКЕТ 10 РЕДЬТ Е ПРОЧЕТЕН И ТРЯБВА ДА СЕ
110   C          СКАНИРА ЗА ФЛАГОВЕ
```

```
111   C
112      10 I=I+1
113   C
114   C        ПРИ КРАЙ НА РЕД GOTO 12
115   C
116        IF (LIST(I)-LIST(3)) 11,12,11
117   C
118   C        ПРИ ФЛАГ ЗА ПАРАМ GOTO 56 ИНАЧЕ GOTO 10
119   C
120      11 IF (LIST(I)-LIST(4)) 10,56,10
121   C
122   C        ЗАМЕСТВАНЕ НА ФЛАГА ЗА ПАРАМЕТЪР
123   C
124      56 MIST(I)=-2
125         MIST(I+1)=LIST(I+1)-LIST(5)+7
126         I=I+2
127         MIST(I)=LIST(I)-LIST(5)
128   C
129   C        ПРОВЕРКА ЗА ПАРАМ 0
130   C
131         IF (LIST(I-1)-7) 60,57,60
132      57 IF (LIST(K)-LIST(I)) 58,59,59
133      58 MIST(K)=LIST(I)
134      59 GOTO 10
135   C
136   C        ПРОВЕРКА ЗА ПАРАМ 9
137   C
138      60 IF (LIST(I-1)-16) 10,61,10
139      61 IF (LIST(K+1)-LIST(I)) 611,612,612
140     611 MIST(K+1)=LIST(I)
141     612 GOTO 10
142   C
143   C        ОТКРИТ Е ФЛАГ ЗА КРАЙ НА КОДОВ РЕД
144   C
145      12 MIST(I)=-1
146   C
147   C        ЧЕТЕНЕ НА СЛЕДВАЩ РЕД ОТ ДЕФИНИЦИЯТА
148   C
149      13 I=I+1
150         CALL IOOPS(-1,1,LIST,I,J,IOOP)
151         IF (IOOP) 62,63,62
152      62 PAUSE 3
153         RETURN
154   C
155   C        ПРОВЕРКА ЗА ПРЕПЪЛВАНЕ НА ПАМЕТТА
156   C
157      63 IF (J-KMAX) 65,65,64
158      64 PAUSE 5
159         RETURN
160   C
161   C        ГАРАНТИРАНЕ НА ФЛАГ ЗА КРАЙ НА РЕДА
162   C
163      65 MIST(J)=LIST(3)
164   C
165   C        АКО ФЛАГЪТ ЗА КРАЙ НА РЕДА НЕ Е В ПЪРВА ПОЗИЦИЯ, РЕДЪТ Е
```

```
C        ВАЛИДЕН
C
      IF (LIST(I)-LIST(3)) 11,66,11
C
C        В ПРОТИВЕН СЛУЧАЙ КРАЙ НА МАКРОДЕФИНИЦИЯ. ЗАРЕЖДАНЕ НА
C        ИНДЕКСА ЗА СЛЕДВАЩАТА МАКРОДЕФИНИЦИЯ
C
   66 MIST(K-1)=I
      K=I+1
C
C        АКО РЕДЪТ НЕ СЕ СЪСТОИ ОТ ДВА ПОСЛЕДОВАТЕЛНИ ФЛАГА ЗА КРАЙ,
C        GOTO 1 ЗА ВЪЗПРИЕМАНЕ НА СЛЕДВАЩАТА МАКРОДЕФИНИЦИЯ
C
      IF (K-J) 67,67,1
   67 IF (LIST(K)-LIST(3)) 1,20,1
C
C        ЧЕТЕНЕ НА РЕД ОТ ИЗХОДНАТА ПРОГРАМА
C
   20 CALL IOOPS(-1,1,LIST,I,N,IOOP)
      IF (IOOP) 68,69,68
   68 RETURN
   69 MIST(N)=LIST(1)
C
C        ИНДЕКС НА ПЪРВАТА МАКРОДЕФИНИЦИЯ ЗА СРАВНЯВАНЕ С РЕДА ОТ
C        ИЗХОДНАТА ПРОГРАМА
C
      M=18
   30 L=8
      J=M+2
C
C        СРАВНЯВАНЕ НА ЗНАК ПО ЗНАК
C
      DO 33 K=I,N
      J=J+1
C
C        АКО ТЕКУЩИЯТ ЗНАК ОТ ШАБЛОНА Е ФЛАГ ЗА ПАРАМЕТЪР GOTO 32
C
      IF (LIST(J)-LIST(2)) 70,32,70
C
C        ПРИ НЕСЪВПАДЕНИЕ НА ЗНАЦИТЕ GOTO 31
C
   70 IF (LIST(J)-LIST(K)) 31,71,31
C
C        ПРИ ФЛАГ ЗА КРАЙ НА ИЗХАДЕН РЕД GOTO 40 ИНАЧЕ ПРОДЪЛЖИ
C        СКАНИРАНЕТО - GOTO 33
C
   71 IF (LIST(J)-LIST(1)) 33,40,33
C
C        НЕСЪВПАДЕНИЕ - СЛЕДВАЩА ДЕФИНИЦИЯ
C
   31 M=LIST(M)
      IF (M-I) 30,72,72
   72 CALL IOOPS(1,3,LIST,I,N,IOOP)
      IF (IOOP) 73,74,73
   73 PAUSE 6
```

```
      RETURN
   74 GOTO 20
C
C        ФЛАГ ЗА ПАРАМЕТЬР ОТКРИТ В ШАБЛОНА, АКО СЬОТВ. ВХОДЕН ЗНАК E
C        ФЛАГ ЗА КРАЙ НА РЕДА - НЕСЬВПАДЕНИЕ GOTO 31
C
   32 IF (LIST(K)-LIST(1)) 75,31,75
C
C        ЗАНАСЯНЕ НА ПАРАМЕТЬРА
C
   75 MIST(L)=LIST(K)
      L=L+1
   33 CONTINUE
C
C        КОНСТРУИРАНЕ НА ОБЕКТЕН РЕД. ВХОД В ПРОГРАМАТА ПРИ ЕТИКЕТ 40
C
C        ПРЕХВЬРЛЯНЕ НА ЕДИН ЗНАК В КОНСТРУИРАНИЯ РЕД
C
   41 MIST(K)=LIST(J)
C
C        ПРИДВИЖВАНЕ НА ИНДЕКСА НА КОНСТРУИРАНИЯ РЕД
C
   42 K=K+1
C
C        ПРИДВИЖВАНЕ НА ИНДЕКСА НА КОДОВИЯ РЕД
C
   43 J=J+1
C
C        GOTO 48 ПРИ ИЗЧЕРПАНА МАКРОДЕФИНИЦИЯ
C
      IF (LIST(M)-J) 76,48,76
C
C        ИЗВИКВАНЕ НА ПАРАМЕТЬР - GOTO 44
C        КРАЙ НА РЕД - GOTO 47
C
   76 IF (LIST(J)+1) 44,47,41
C
C        ОБРАБОТКА НА ПАРАМЕТЬР
C
   44 L=LIST(J+1)
      J=J+2
C
C        ПАРАМ 0 - GOTO 45
C
      IF (L-7) 77,45,77
C
C        ПАРАМ 9 - GOTO 49
C
   77 IF (L-16) 78,49,78
C
C        МОДИФИКАТОР РАЗЛИЧЕН ОТ 0 - GOTO 46
C
   78 IF (LIST(J)) 46,79,46
C
C        КОПИРАНЕ НА ПАРАМЕТЬРА В КОНСТРУИРАНИЯ РЕД
```

ЛИСТ РЕД ОПЕРАТОР

```
276   C
277       79 MIST(K)=LIST(L)
278          GOTO 42
279   C
280   C          ПРИ ПАРАМЕТЪР 9
281   C
282       49 MIST(16)=LIST(J)+LIST(17)
283          GOTO 46
284   C
285   C          ПРИ ПАРАМЕТЪР 0
286   C
287       45 MIST(7)=LIST(J)+LIST(6)
288   C
289   C          ПРЕВРЪЩАНЕ НА ПАРАМЕТЪР В ТРИЦИФРЕНО ЧИСЛО
290   C
291       46 MIST(K)=LIST(L)/100
292          N=LIST(L)/10
293          MIST(K+1)=N-LIST(K)*10+LIST(5)
294          MIST(K+2)=LIST(L)-N*10+LIST(5)
295          MIST(K)=LIST(K)+LIST(5)
296   C
297   C          ПРИДВИЖВАНЕ НА ИНДЕКСА
298   C
299          K=K+3
300          GOTO 43
301   C
302   C          ИЗВЕЖДАНЕ ПРИ КРАЙ НА РЕД
303   C
304       47 CALL IOOPS(1,3,LIST,I,K,IOOP)
305          IF (IOOP) 80,40,80
306       80 PAUSE 7
307          RETURN
308   C
309   C          ВХОД В ПРОГРАМАТА ЗА КОНСТРУИРАНЕ НА РЕД
310   C
311       40 K=I
312          GOTO 43
313   C
314   C          УВЕЛИЧАВАНЕ НА ГЕНЕРАТОРИТЕ
315   C
316       48 MIST(6)=LIST(M+1)+LIST(6)+1
317          MIST(17)=LIST(M+2)+LIST(17)+1
318          GOTO 20
319   C
320   C          КРАЙ НА SIMC
321   C
322          END
```

```
FLG ' = '.
VAL ' = PTR '.
PTR ' = VAL '.
GET ' = '.
STO ' = '.
```

а/ операции за прехвърляне на данни

```
VAL ' = ' + '.
VAL ' = ' - '.
PTR ' = ' + '.
PTR ' = ' - '.
PTR ' = ' * '.
PTR ' = ' / '.
```

б/ аритметични операции с цели числа

```
TO ''.
TO '' IF FLG ' = '.
TO '' IF FLG ' NE '.
TO '' IF VAL ' = '.
TO '' IF VAL ' NE '.
TO '' IF PTR ' = '.
TO '' IF PTR ' NE '.
TO '' IF PTR ' GE '.
TO '' BY '.
RETURN BY '.
STOP.
```

в/ управляващи операции

```
READ NEXT '.
WRITE NEXT '.
VAL ' = CHAR.
CHAR = VAL '.
REWIND '.
```

г/ входно-изходни операции

```
LOC ''.
END PROGRAM.
```

д/ псевдо-операции

Фиг.5.1. Операции на абстрактната изчислителна машина FLBM

```
1. МАКРОДЕФИНИЦИЯ: |ЗАГЛАВИЕ ''''''.
                   |          *** '10'20'30'40'50'60'***C
                   |@
                   |
          ВХОД:    |ЗАГЛАВИЕ ДОКЛАД.
                   |
         ИЗХОД:    |          *** ДОКЛАД ***
                   |
2. МАКРОДЕФИНИЦИЯ: |''' ' = ' ' '.
                   |     J'10'40=J'10'50'60J'10'70@
                   |@
                   |
          ВХОД:    |VAL A = B + C.
                   |
         ИЗХОД:    |     JVA=JVB+JVC
                   |
          ВХОД:    |PTR F = F + C.
                   |
         ИЗХОД:    |     JPF=JPF+JPC
                   |
3. МАКРОДЕФИНИЦИЯ: |FLUB КОДОВЕ НА ',',',.
                   |'10 /'11/@
                   |'20 /'21/@
                   |'30 /'31/@
                   |@
                   |
          ВХОД:    |FLUB КОДОВЕ НА A,B,X.
                   |
         ИЗХОД:    |A /011/
                   |B /012/
                   |X /034/
                   |
4. МАКРОДЕФИНИЦИЯ: |TO '' IF FLG ' = '.
                   |     IF (JF'30-JF'40) '01,'10'20,'01@
                   |  '01 CONTINUE@
                   |@
                   |
          ВХОД:    |TO 25 IF FLG M = N.
       ГЕНЕРАТОР:  |125
                   |
         ИЗХОД:    |     IF (JFM-JFN) 126,25,126
                   |  126 CONTINUE
       ГЕНЕРАТОР:  |127
```

Фиг.5.2. Примери на макродефиниции, обработка на параметри и действия на простия компилатор SIMCOM

5.3. ОСНОВЕН КОМПИЛАТОР BASCOM

За създаването на Основния компилатор BASCOM се използува една разработка на подобен компилатор наречен STAGE2 /представен в раздел 5.6 чрез масива STG2/, възможностите на който се разширяват, за да може да компилира програми, написани на езика WISP. Разширението на STAGE2 се налага поради следните особености на WISP:

1/ На един ред може да се запишат по-вече от един оператор.

2/ В операторите могат да бъдат включени излишни шпации.

3/ Операторите могат да съдържат небалансирани скоби.

Тези особености на езика WISP изискват включването в компилатора STAGE2 на подпрограми /представени в раздел 5.6 чрез масива WSED/, които четат изходните редове и след като премахнат излишните шпации и модифицират небалансираните скоби, подават операторите един по един за по-нататъшна обработка. Следователно, основният компилатор BASCOM се състои структурно от две части - компилатора STAGE2 и добавените чрез масива WSED подпрограми.

WSED отпечатва всеки ред от изходната програма така както е прочетен и по този начин обезпечава на програмиста листинг от изходната програма. След това изходният ред се сканира, премахват се излишните шпации и се разделят отделните оператори. Всеки едносимволен непосредствен операнд, предшествуван от апостроф, се заменя с трицифрения си десетичен еквивалент, за което се използува съответствието, определено от входно-изходната управляваща система IOCS. Чрез преобразуването на едносимволните непосредствени операнди се отстраняват и небалансираните скоби.

Подобно на SIMCOM, STAGE2 е процесор, който преобразува символни низове съобразно набор от дефиниции, съставени от програмиста. Той използува подбиране по образец за разпознаване на изходните оператори и изолиране на параметрите, като за шаблони се използуват единични символи и балансирани

елементи. Това означава, че операндите от езика на абстрактната машина могат да бъдат с произволна дължина и могат да използуват скоби с произволна дълбочина.

Символите, които имат специално значение за компилатора STAGE2 са 12, докато при SIMCOM бяха само 5. Първата карта на масива на дефинициите /наречена карта на флаговете/ определя конкретните знаци за представянето на тези символи чрез първите си 12 колонки. За представената реализация тази карта съдържа:

.'@'0 (+-*/)

Възможностите на процесора STAGE2 по отношение на обработката на символни низове и компилиране се характеризират от начините за трансформиране на параметрите и предвидените други действия на процесора. За да бъде обяснено действието на процесора, възприети са следните съкращения в дефинициите на подлежащите на компилиране оператори на изходния език, които служат за записване на формата и заместването на параметрите и изразяване на другите действия на процесора:

c - произволен символ, но различен от символ за край на обектен ред /трети символ от картата на флаговете/ или символ CRLF /връщане на каретката и нов ред/

d - произволна цифра между 1 и 9 включително

e - символ за място на параметър в обектен ред /четвърти символ от картата на флаговете/

m,n - произволни цифри между 0 и 9 включително.

Компилаторът STAGE2 открива грешки от четири типа:

1/ Неправилен аритметичен израз.

2/ Неправилна трансформация на параметър или формат на действие.

3/ Грешки при входно-изходна операция.

4/ Недостатъчно памет за завършване на работата.

Индикацията за тези грешки е отпечатване на съответно съобщение

********** EXPR ERROR или

********** CONV ERROR или

********** IOCH ERROR или

********** FULL ERROR , след което следва обратно сканиране и отпечатване на обектните редове в обратна последователност до причинилия грешката изходен ред включително.

IOCH ERROR и FULL ERROR са грешки, след които работата на компилатора не може да продължи.

Ще бъдат дадени определения и обяснения на форматите на записване и работата на процесора STAGE2 при обработка на параметрите и изпълнението на предвидените действия. Към всяко определение ще бъдат приложени примери, които спомагат за изясняване на определенията.

- Заместване на параметър по име в конструирания ред

Формат: ed0

Действие: Трите символа на формата от конструирания обектен ред се заместват с името на параметъра d от подлежащия на обработка изходен ред, т.е. заместване на параметър по име.

Примери: фиг.5.3.1, фиг.5.3.2.

- Заместване на параметър по стойност в конструирания ред

Формат: ed1

Действие: Името на параметъра d от подлежащия на обработка изходен ред се използува за адресиране на паметта. Получената стойност на параметъра замества трите символа на формата в конструирания обектен ред, т.е. заместване на параметър по стойност.

Съдържанието на паметта не се изменя. Ако стойността на параметъра е нула или ако параметърът е недефиниран, нищо не се занася в конструира-

ния ред.

Примери: фиг.5.3.3.

Формат: ed2

Действие: Името на параметъра d от подлежащия на обработка изходен ред се използува за адресиране на паметта. Получената стойност на параметъра замества трите символа на формата в конструирания обектен ред, т.е. заместване на параметър по стойност.

Ако стойността на параметъра е нула, нищо не се занася в конструирания ред. Ако параметърът е недефиниран, той се дефинира като се използува текущата стойност на генератора на цели числа, след което стойността на генератора се увеличава с единица. Така дефинираната стойност на параметъра замества трите символа на формата в конструирания обектен ред.

Примери: фиг.5.3.4.

- Заместване на следващ параметъра символ в конструирания ред

Формат: ed3

Действие: Трите символа на формата в конструирания обектен ред се заместват от символа, който непосредствено следва параметъра d във входния низ. Ако параметърът d е непосредствено преди края на входния низ, нищо не се занася в конструирания обектен ред.

Примери: фиг.5.3.5.

- Заместване по стойност на разглеждан като аритметичен израз параметър в конструирания ред

Формат: ed4

Действие: Трите символа на формата в конструирания обектен ред се заместват с низ от цифри, възможно предшествуван от минусов знак, представляващи стойността на параметъра d, разглеждан като аритметичен израз. Ако параметърът d е неправилен аритметичен израз, отпечатва се съответно съобще-

ние за грешка.

Примери: фиг.5.3.6.

- Заместване на параметър с дължината на името му

Формат: ed5

Действие: Трите символа на формата в конструирания обектен ред се заместват с цяло число, равно на броя на символите в името на параметъра d. Ако параметърът d е нулев низ, символите на формата се заместват с нула.

Примери: фиг.5.3.7.

- Заместване на параметър с конструиран ред

Формат: ed6

Действие: Конструираният ред става параметър d. Елементът, непосредствено следващ ed6, се игнорира и сканирането продължава от следващия елемент на макродефиницията.

Използуването на ed6 трябва да става предпазливо. Стойността на параметъра d се променя невъзвратимо чрез това действие. Ако действието се използува в итерация /виж ed7 и eF7/, стойността на параметъра d ще бъде дефинирана само през време на текущата итерация. Стойността на този параметър не ще бъде достъпна след следващата итерация, нито пък след завършване на итерациите.

Примери: фиг.5.3.8.

- Контекстно-управлявани итерации

Формат: ed7

Действие: Трите символа на формата задействуват контекстно-управлявана итерация за първо изпълнение. Контекстно-управляваната итерация започва със запазване на текущия параметър d и дефиниране на символите които следват ed7 в същия ред като разграничаващи символи. Ако конструираният ред е за-

граден в скоби, външните скоби се отстраняват. Итерациите се повтарят чрез присвояване на параметър d най-дългия балансиран низ, който започва в началото на конструирания ред и не съдържа незаградени в скоби разграничаващи символи. Този низ и разграничаващите символи, които го следват, се изтриват от конструирания ред. Ако конструираният ред е нулев, искането за итерация се игнорира и итерациите завършват. След всяко изпълнение на итерация, сканирането за следващата итерация започва от кодовия ред на дефиницията, който следва кодовия ред съдържащ ed7 и започващ итерациите.

Примери: фиг.5.3.9.

- Преобразуване на символ в цяло число

Формат: ed8

Действие: Трите символа на формата се заместват с цяло число, представляващо еквивалента на параметъра d според съответната функция за прекодиране от входно-изходната управляваща система. Параметърът d трябва да бъде единичен символ /знак/. Прекодирането е обикновено машинно зависимо. Ако параметърът d не е единичен символ, отпечатва се съответно съобщение за грешка.

Примери: фиг.5.3.10.

- Завършване на обработката

Формат: eF0

Действие: Процесорът завършва работа веднага след разпознаването на символите на формата eF0. Конструираният ред не се извежда, нито се подлага на сканиране.

Примери: фиг.5.3.11.

- Извеждане на конструирания ред

Формат: eF1m

Действие: Ако конструираният ред не е нулев, той се извежда чрез канал m без да бъде по-вече сканиран от процесора. Четирите символа на формата трябва да бъдат записани в края на шаблона на конструирания ред от макродефиницията. Всеки символ следващ eF1m се разглежда като искане за операция пренавиване за съответния канал /вик eF1mc/.

Ако конструираният ред е нулев, eF1m се разглежда като искане за форматен изход. Четирите символа на формата трябва да бъдат в края на шаблона на конструирания ред. Следващият ред за сканиране се разглежда като шаблон, указващ формата на изхода. Всеки низ от цифри в този ред дефинира поле, в което съответният параметър се занася. Например, низ от единици ще дефинира поле, в което ще се занесе първият параметър. Ако фактическият параметър е по-дълъг от дефинираното поле, извършва се отрязване от дясно на символи от параметъра. Ако шаблонът е по-дълъг от параметъра, той се запълня с шпации, следващи параметъра. Символите от шаблона, които не са цифри, се занасят без изменение в обектния ред. Ако два низа от еднакви цифри са разделени от други символи, те се разглеждат като две различни полета, в които съответният параметър се занася независимо за всяко поле.

Ако конструираният ред е нулев и няма по-вече редове от макродефиницията за сканиране, появява се съобщение за съответна грешка. Извършва се обратен преглед и отпечатване и обработката на макродефиницията завършва.

Указанието за изходен канал може да липсва в приложения шаблон. В този случай се използува канал 3.

Примери: фиг.5.3.12, фиг.5.3.13, фиг.5.3.14, фиг.5.3.15.

Формат: eF1mc

Действие: Аналогично на действието на eF1m с допълнително пренавиване на канал m преди извеждане на съответния ред. Елементът, непосредствено след-

ващ eF1mc се игнорира и сканирането продължава от следващия елемент, подлежащ на обработка.

Примери: фиг.5.3.16.

- Смяна на входно-изходните канали и копиране на текст

Формат: meF2n

Действие: Каналът m става текущ входен канал. Ако параметър 1 не е нулев, редовете на текста се копират от входния канал на канал n, като копирането продължава до реда, който има начален подниз, равен на параметър 1. Този ред се игнорира и текущият входен канал се позиционира за четене на следващия ред. Ако няма по-вече редове за четене, операцията по копирането на текста завършва с действия по затваряне на масива на входния канал.

Ако стойността на параметър 1 е нулева, никакво копиране на текст не се извършва, но канал m става текущ входен канал.

Обозначенията за m и/или n могат да се пропуснат. Когато символите се намират в началото на реда, текущият входен канал не се изменя. Когато символите eF2 са поставени в края на реда, текстът за копиране се направлява към канал 3. Във всеки случай, когато параметър 1 не е нулев, копиране на текст ще се извърши.

Символите meF2n трябва да бъдат сами на ред от макродефиницията. Позицията на eF2 в съответствие с началото и края на реда определя кои възможности да бъдат взети. Ако във формата са включени допълнителни символи, те се разглеждат като искане за пренавиване /вик mceF2nc/.

Примери: фиг.5.3.17, фиг.5.3.18.

Формат: mceF2nc

Действие: Аналогично на действието на meF2n с тази разлика, че двата канала се пренавиват преди извършване на операцията по копиране на текста. Елементът, следващ mceF2nc се игнорира и сканирането продължава от следващия

елемент, подлежащ на обработка.

Примери: фиг.5.3.19.

- Запасяне на информация в паметта

Формат: eF3

Действие: Параметър 1 се използува за адресиране на паметта. Параметър 2 е новата стойност, която се запася в паметта. Старата стойност се загубва.

Елементът, непосредствено следващ eF3, се игнорира и сканирането продължава от следващия елемент.

Примери: фиг.5.3.20.

- Безусловно зареждане на брояча за прескачане на редове

Формат: eF4

Действие: Параметър 1 се разглежда като аритметичен израз, стойността му се изчислява и резултатът се запася в брояча за прескачане /пропускане/ на редове. Ако параметър 1 не е аритметичен израз, отпечатва се съответно съобщение за грешка.

Елементът, непосредствено следващ eF4, се игнорира и сканирането се възобновява от следващия елемент.

Примери: фиг.5.3.21.

- Зареждане на брояча за прескачане на редове в зависимост от резултата от сравняването на два низа

Формат: eF5k

Действие: Параметрите 1 и 2 се сравняват за еднаквост. Ако k=0 и параметрите 1 и 2 са еднакви /идентични/, в брояча за прескачане /пропускане/ на редове се запася стойността на параметър 3, разглеждан като аритметичен израз. Ако k=1 и параметрите 1 и 2 не са еднакви, в брояча за прескачане на редове се запася стойността на параметър 3. При всички други условия в брояча не се запася нищо.

Елементът, непосредствено следващ eF5k, се игнорира и сканирането се възобновява от следващия елемент.

Параметрите 1 и 2 могат да бъдат произволни низове, но параметър 3 трябва да бъде аритметичен израз. В противен случай се отпечатва съответно съобщение за грешка.

Примери: фиг.5.3.22, фиг.5.3.23.

- Зареждане на брояча за прескачане на редове в зависимост от стойностите на два израза

Формат: eF6k

Действие: Параметри 1 и 2 се изчисляват като аритметични изрази и резултатите се сравняват. В зависимост от символа к и отношението на стойностите на параметрите 1 и 2, в брояча за прескачане на редове се зарежда стойността на параметър 3. Условията за зареждане на брояча са:

к = - и параметър 1 < параметър 2

к = 0 и параметър 1 = параметър 2

к = 1 и параметър 1 ≠ параметър 2

к = + и параметър 1 > параметър 2

Ако някой от параметрите не е аритметичен израз, отпечатва се съответно съобщение за грешка.

Примери: фиг.5.3.24, фиг.5.3.25.

- Итерации, управлявани по брой

Фориат: eF7

Действие: Конструираният ред се изчислява като аритметичен израз и стойността му се занася в брояча на итерации. Ако стойността не е нула, започва изпълнението на управлявани по брой итерации и се извършва първата итерация. Ако стойността е нула, никаква операция не се извършва. Във всеки случай елементът следващ eF7 се игнорира и сканирането продължава от след-

вящия елемент.

Управляваните по брой итерации започват със запазване на стойността на брояча на итерации, като при всяко изпълнение на итерация стойността на брояча се намалява с единица. Ако стойността на брояча не е отрицателна, при всяко изпълнение на итерация сканирането се възобновява от реда, следващ този, съдържащ символите на формата eF7.

Ако конструираният ред не е правилен аритметичен израз, отпечатва се съответно съобщение за грешка.

Примери: фиг.5.3.26.

- Изпълнение на итерация

Формат: eF8

Действие: Изпълнява се веднаж съответната текуща итерация, независимо дали е контекстно управлявана или управлявана по брой. Ако няма започнати итерации, никаква операция не се извършва. Елементът, непосредствено следващ eF8, се игнорира и сканирането се възобновява от следващия елемент.

Текущата итерация завършва, ако символите на формата eF8 се намират в началото на реда, когато се извършва прескачане на редове. При това е възможно завършване на итерациите преждевременно чрез прескачане на последните eF8 до край на итерациите, като е необходимо символите на формата да бъдат в началото на реда. Ако е необходимо да се прескочат всичките итерации /за разлика от прескачането на някоя от тях/ допълнително внимание трябва да бъде отделено за правилното кодиране. През време на процеса на прескачане, стойността на брояча на итерации се запазва. Тази стойност се увеличава с единица за всяко eF7, което се среща в началото на ред и се намалява с единица за всяко eF8, което се среща в началото на ред. Текущите итерации завършват, само ако eF8 бъде открито в началото на ред, когато броя на започнатите итерации е нула. Нека се има предвид, че ако

eF7 се случи в началото на ред, тогава конструираният ред е нулев и броячът на итерации ще получи стойност нула. Извикване на управлявана по брой итерация при стойност на брояча на итерации нула не предизвиква никаква операция.

Примери за използуване на eF8 могат да бъдат намерени при управлявани по брой и контекстно управлявани итерации. Разбирането на примерите за прескачане на цял цикъл изисква разбиране на условното прескачане на редове в зависимост от параметри от тип произволни низове, както и двата вида итерации.

Примери: фиг.5.3.27.

- Изход от текуща макродефиниция

Формат: eF9

Действие: Сканирането на текущия кодиран ред завършва веднага и управлението се връща към извикващата макродефиниция. Конструираният ред се изоставя. Това действие на процесора може да се използува при прескачане на редове в зависимост от условие, за да се избегне нуждата от точно указване на броя на редовете за прескачане. В този случай ще се прескочат всички редове до края на макродефиницията.

Примери: фиг.5.3.28.

- Генериране на обратен преглед

Формат: eFE

Действие: Отпечатва се съобщение за грешка при прекодиране чрез канал 4, следвано от стойността на конструирания ред. Следва обратен преглед, който отпечатва извикването на съответната макродефиниция, извикването на макродефиницията, която генерира предното извикване и т.н. Последният ред на обратния преглед е изходния оператор на компилираната програма. Елементът, непосредствено следващ eFE, се игнорира и сканирането продължава

от следващия елемент. Конструираният ред не се изменя и символите eFE не се копират в него.

Служи и за обратен преглед при откриване на грешка, като в този случай се включва автоматично в действие от процесора.

Примери: фиг.5.3.29.

До тук бяха разгледани функциите на двете съставни части на основния компилатор BASCOM - подпрограмите от масива WSED и компилатора STAGE2.

Масивът STG2, който е изходния текст на компилатора STAGE2, представлява коректна програма на езика FLUB, поради което простият компилатор SIMCOM, като използува масива на дефинициите FLBM, може да извърши преобразуването й във фортранова програма. Масивът WSED, обаче, не представлява коректна програма на езика FLUB. Той съдържа непосредствени операнди, а също и етикети, изразени чрез низове от символи. Компилирането на оператори, съдържащи такива операнди, е извън възможностите на простия компилатор. Ето защо основният компилатор BASCOM, изходният текст на който се получава чрез обединение на масивите STG2 и WSED, се създава на два етапа.

Най-напред, като се използува простият компилатор SIMCOM и масивът на дефинициите FLBM, извършва се преобразуване на изходния FLUB текст на компилатора STAGE2 /масив STG2/ във фортранова програма. Тази фортранова програма се транслира, за да се получи компилатора STAGE2 като машинна програма.

Като се имат предвид описаните възможности на този компилатор, създава се нов масив от дефиниции - BASM, в който освен възможностите на масива на дефинициите FLBM са включени и възможности за компилиране на непосредствени операнди и многосимволни етикети. В представената реализация за електронната изчислителна машина FACOM 230-45S като език на вграждане за основния компилатор BASCOM е избран FORTRAN, следователно, необходимо е масивът на дефинициите BASM, показан в раздел 5.6, да определя заместването на изходните опе-

ратори чрез оператори на FORTRAN.

От масивите STG2 и WSED се подготвя един общ масив, представляващ изходния текст на основния компилатор BASCOM. Този текст се преобразува, като за целта се използуват компилаторът STAGE2 и масивът на дефинициите BASM. Получената фортранова програма се транслира, за да се получи основният компилатор BASCOM като машинна програма.

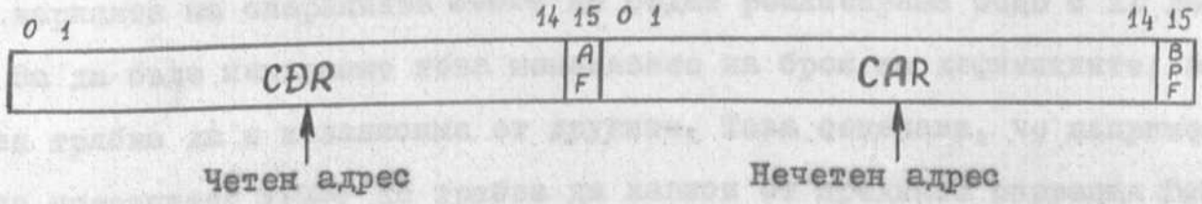
Трябва да се подчертае, че основният компилатор BASCOM е процесор за обработка на символни низове и има самостоятелно значение и област на приложение. В основата му са залегнали принципите за подбиране по образец и заместване, характерни за езика SNOBOL, макар и с известни ограничения. Този процесор от компилиращ тип може да служи за транслиране на програми написани на дефиниран от потребителя специализиран език за програмиране в програми на реализиран на изчислителната машина конвенционален език за програмиране или асамблер. Осигуряването на възможности за използуване на дефинирани от потребителите езици е една от насоките на развитие на системното програмиране.

Ако основният компилатор BASCOM намира широко приложение, необходимо е той да бъде оптимизиран. За извършването на оптимизацията се използува неоптимизираният BASCOM. Може да се отбележи само, че процесът на оптимизация е принципно аналогичен на процеса на създаване на процесора HELP.

В представената разработка, създаденият основен компилатор BASCOM се използува за преобразуване на изходния текст на процесора HELP от езика WISP на езика на асамблера FASP на електронната изчислителна машина FACOM 230-45S. За да стане възможно това, необходимо е да бъде създаден масивът на дефинициите на операторите на езика WISP чрез команди на FASP, т.е. абстрактната машина WSPM да бъде моделирана на асамблера на реалната машина.

Най-напред е необходимо да бъде определено моделирането на елементите от паметта на абстрактната машина. Тъй като машинната дума на FACOM 230-45S

се състои от 16 бита /2 байта/, за един елемент от паметта на абстрактната машина ще бъдат използувани две последователни машинни думи, първата от които има четен адрес. При четенето или записа на тези елементи ще се използува едновременно адресиране на двете машинни думи. При това адресиране най-младшият бит на адреса не се използува, поради което първата дума трябва да има четен адрес. Тази особеност позволява най-рационално използуване на оперативната памет чрез следното разположение на полетата на списъчния елемент в двете последователни машинни думи:



В CDR и CAR полетата може да се запише адрес с размерност 15 бита и да бъдат адресирани 32768 елемента от паметта на абстрактната машина, като фактически се адресират 65536 думи от паметта на реалната изчислителна машина FACOM 230-45S.

За реализацията на AF и BPF полетата се използуват ненужните при двойната адресация най-младши битове на двете машинни думи.

Тъй като при индиректното адресиране се използува съдържанието на CDR полето, то трябва да бъде разположено в първата дума.

Абстрактната машина WSPM, описана в трета глава на дисертацията, изпълнява осем операции, операндите на които могат да бъдат модифицирани по шест начина. Например, първият операнд на оператора за присвояване може да използува директно или четирите типа индиректно адресиране, а вторият операнд може да бъде освен това и непосредствен. Тъй като модифицирането на двата операнда е независимо, различните варианти само на операцията присвояване са 30. Такъв начин за кодиране на дефинициите е явно неприемлив поради изключително големия им брой.

За да се намали броят на дефинициите, необходими за реализацията на езика WISP, възможно е по-вечето от операциите на абстрактната машина WSPM да бъдат разложени на последователности от по-прости операции. Например, присвояването може да се представи чрез операция за вземане на стойността на втория операнд /fetch op2/, следвана от операция за занасяне на тази стойност на адреса на първия операнд /store op1/. По този начин различните случаи на модификация на операндите се разглеждат поотделно, а не в комбинации. За присвояването, шестте възможни варианта на операцията fetch и петте възможни варианта на операцията store ще бъдат реализирани общо с 11 дефиниции.

За да бъде изпълнимо това намаляване на броя на дефинициите, всяка дефиниция трябва да е независима от другите. Това означава, че например кодирането на операцията store не трябва да зависи от предната операция fetch. При това, всички операции fetch трябва да занасят резултата от изпълнението си на определено място и в определен формат. Разбира се, това в известна степен намалява ефективността на изпълнението на някои операции.

В табл.5.1 е показано разлагането на операторите на езика WISP на съставни оператори, които имат само по един операнд. Това разлагане е машинно-независимо и съществува възможност за продължаването му по отношение на операндите, защото всеки операнд е цяло число, макар и модифицирано по един от шестте начина. Създадени са дефиниции, които определят вида на модификатора и го свързват с операционния код от табл.5.1, за да формират комбиниран операционен код, както е показано в табл.5.2. В последната таблица валидните комбинирани операционни кодове са отбелязани с "да". Например, FETIM е валиден код, докато STOIM е невалиден. Изборът на валидните кодове зависи както от обективната постановка на задачата, така също и от субективното разбиране на реализатора. Например, преходите по AF и BPF са легални операции на WSPM, но при тяхното изпълнение винаги ще се извършва преход на адреси 0 или 1.

Това в някои случаи може да е полезно, но ако реализаторът схваща тези операции като недобра практика, той може да разглежда съответните комбинирани кодове като невалидни и да не ги реализира.

ОПЕРАЦИИ	СЪСТАВНИ ОПЕРАЦИИ
INCR op1.	INC op1
TO op1.	JMP op1
op1 = op2.	FET op2 STO op1
IO op1 ON op2.	FET op1 IOP op2
TO op1 IF op2 = op3.	FET op2 CMP op3 JEQ op1
TO op1 IF op2 NE op3.	FET op2 CMP op3 JNE op1
TO op1 IF op2 LT op3.	FET op2 CMP op3 JLT op1
ELEMENT op1 op2 op3 op4.	DAF op1 DBP op2 DCA op3 DCD op4
ENTRY op1.	LEN op1
USE op1.	USE op1
EXIT op1.	XIT op1
Label	LOC label

Табл.5.1. Разлагане на операторите на WISP на съставни оператори

ОПЕРАЦИЯ	МОДИФИКАТОР					
	IM	DR	AF	BP	CA	CD
JMP	да	да			да	да
INC		да			да	да
FET	да	да	да	да	да	да
STO		да	да	да	да	да
IOP	да	да			да	да
CMP	да	да	да	да	да	да
JEQ	да	да			да	да
JNE	да	да			да	да
JLT	да	да			да	да
DAF	да					
DBP	да					
DCA	да					
DCD	да					
LEN	да					
USE	да					
XIT	да					
LOC	да					

Табл.5.2. Валидни операционни кодове на съставните операции

В представения в раздел 5.6 масив на дефинициите WSPM, първата карта съдържа специалните символи, използувани при кодирането. Следват макродефинициите, които до картата с идентификационен номер WSPM0240 са машинно-независими и осъществяват разлагането на основните, допълнителните, разширените и псевдо операциите на езика WISP на съставни еднооперандни микрооперации, които от своя страна се разлагат на възможните варианти според модифициране-

то на операндите. От картата с идентификационен номер WSPM0240 започват дефинициите на всички позволени варианти на микрооперациите. На подбирането на команди от асамблера на реалната машина при кодирането на дефинициите няма да се спирам, защото то предполага познаване на електронната изчислителна машина FACOM 230-45S. Може да се отбележи само, че нейната логическа структура, команди и операционна система са аналогични на логическата структура, командите и операционната система на IBM/360.

При кодирането на дефинициите на позволените варианти на микрооперациите се оказа, че обектните редове на много от тях се състоят от по няколко команди на асамблера на FACOM 230-45S. Освен това, дефинициите на различните варианти на някои микрооперации се различават само по първия или по първия и последния обектни редове. Това позволи да бъде намален обемът на получаваната обектна програма, като в дефинициите се запазят само първият или първият и последният обектни редове, между които се вмъкне команда за преход към оформената като подпрограма еднаква част. Например, вариантите STODR и STOCD на микрооперацията STO могат да се реализират с помощта на следните дефиниции:

```
STODR @.
   LD  '10,,6'F1@
   TR  4,6'F1@
   HRD 1,,4'F1@
   TR  4,3'F1@
   HLD 1,,4'F1@
   TR  6,4'F1@
   STD '10,,6'F1@
@
STOCD @.
   LD  ('10),,6'F1@
   TR  4,6'F1@
   HRD 1,,4'F1@
   TR  4,3'F1@
   HLD 1,,4'F1@
```

```
    TR  6,4'F1@
    STD ('10),,6'F1@
  @
```

Всяко срещане на STODR ор или STOCD ор ще се замества в обектната програма с последователности от по седем команди на асамблера. Пет от командите са еднакви в двата случая. Ето защо в представената реализация тези дефиниции бяха заменени със следните:

```
STODR @.
   LD  '10,,6'F1@
   B   STODRCD'F1@
   STD '10,,6'F1@
@
STOCD @.
   LD  ('10),,6'F1@
   B   STODRCD'F1@
   STD ('10),,6'F1@
@
```

В този случай всяко срещане на STODR ор или STOCD ор ще се замества в обектната програма с последователности от по три команди на асамблера. Разбира се, трябва да бъде обезпечена подпрограмата:

```
STODRCD  ST  RETADR,,0
         TR  4,6
         HRD 1,,4
         TR  4,3
         HLD 1,,4
         TR  6,4
         B   (RETADR)
```

Чрез използването на тази идея беше намален обемът на използваната от процесора HELP оперативна памет с около 35% вследствие на намаления брой команди на главната програма, но времето за изпълнение се увеличи с около 10% поради включените допълнителни преходи към подпрограми и връщания в главната програма.

Подпрограмите за подлежащите на подобно третиране дефиниции са включени в използуваната от процесора HELP входно-изходна управляваща система WIOCS, представена в раздел 5.6 чрез масива WIOCS. Тези подпрограми имат идентификационни номера на картите от WSPSR001 до WSPSR130.

Ще използувам случая да разясня някои подробности на входно-изходната управляваща система WIOCS, която представлява разширение на описаната в раздел 5.1 входно-изходна управляваща система IOCS, използувана от процесорите SIMCOM, STAGE2 и BASCOM.

Входно-изходните операции на абстрактната машина WSPM за предаване на редове между буфера в паметта и съответното входно-изходно устройство могат да се изпълняват от входно-изходната управляваща система IOCS. Необходимо е обаче, към IOCS да бъдат добавени възможности за занасяне на резултата от изпълнението на операцията в CDR полето на спецификатора на операцията. Може би е необходимо да се припомни, че вторият операнд на входно-изходните оператори за предаване на редове е адресът на спецификатора на операцията. В CAR полето на този спецификатор е записан номерът на устройството, вземащо участие в операцията, а резултатът от изпълнението на операцията се записва в CDR полето.

Всички операции за предаване на редове използуват входно-изходния буфер, така че само два аргумента на функцията IOOP са променливи, като освен това могат да се изчисляват през време на изпълнението. Тези аргументи са кодът на операцията и логическият номер на устройството.

Фиг.5.4, фиг.5.5 и фиг.5.6 представляват блокови схеми на програмата, която обработва операциите, свързани с буфера и с извикването на функцията IOOP от входно-изходната управляваща система IOCS. Входът в тази програма, която фактически представлява входно-изходната управляваща система WIOCS, се извършва с два аргумента - стойността на първия операнд на входно-изходната операция и адреса, представен от втория операнд. Функцията IOOP се извиква

след определяне на операцията, а останалите четири фактически аргумента са CAR op2, LB, 1 и LIMIT. LB е базовият адрес на буфера, а LIMIT е индексът на първата позиция на буфера след полезната информация в него.

При реализацията на абстрактната машина WSPM всеки символ трябва да бъде представен чрез адреса на съответстващия му базов регистър. Целите числа, които входно-изходната управляваща система IOCS обезпечава като вътрешни кодове на символите, обикновено не са удобни поради две причини:

- Ако за реализацията на един елемент от паметта на абстрактната машина са необходими n адреса от паметта на реалната машина, реалните адреси на последователни базови регистри се различават с n, докато целите числа, отговарящи на последователни символи, обикновено се различават с 1.
- Целите числа, които са вътрешното представяне на символите, обикновено са малки /трицифрени десетични числа/, но малките адреси от паметта на реалната изчислителна машина може да се използват от операционната система и да не са достъпни за базови регистри.

И така, символите трябва да бъдат превърнати в адреси на съответните базови регистри. Ако базовите регистри образуват блок от последователни думи в паметта на реалната машина, като за всеки регистър се използват n думи, тогава адресът на съответния базов регистър може лесно да се изчисли от цялото число, получено като вътрешен код от входно-изходната управляваща система IOCS. Това преизчисляване може да се извърши или при четенето и записването на редове или при прехвърлянето на единични символи.

Изчисляването, използувано за преобразуване на символите в адреси на базови регистри, трябва да бъде включено и в дефинициите за адресиране на базовите регистри. Най-често това адресиране има формата BASE+c*n, където BASE е базов адрес на блока на базовите регистри, c е вътрешното представяне на съответния символ според IOCS, а n е броят думи от паметта на реалната

машина за реализиране на един елемент от паметта на абстрактната машина.

Базовите регистри трябва да бъдат оформени във верига чрез CAR полетата си, което се изисква от подпрограмите за събиране на неизползуваните списъчни елементи. Най-лесно това се извършва чрез използуване на псевдо-команди за формиране на базовите регистри заедно с необходимата информация, занесена в съответните им полета. Възможно е това действие да се включи във входно-изходната управляваща система WIOCS. В този случай базовият регистър NIL трябва да има свое собствено име, тъй като той се адресира директно както от входно-изходната управляваща система, така и от главната програма.

За да бъде входно-изходната управляваща система WIOCS по-ефективна, в представената реализация тя е прекодирана на асамблера на изчислителната машина FACOM 230-45S и съобразена с входно-изходните характеристики на процесора HELP. Тъй като е възприето той да използува само две входно-изходни устройства - вход от карти и широк печат като изход, само тези устройства бяха включени в WIOCS.

Изходният текст на входно-изходната управляваща система WIOCS започва с указване на обема на използуваната от процесора HELP свободна памет чрез дефиниране на област. След това се дефинира блока на базовите регистри, началният адрес на който е BASE. Базовите регистри се дефинират като константи, чрез което се занася необходимата информация в тях и се извършва свързването им във верига посредством CAR полетата. Следват две таблици за директно прекодиране на символите в адреси на базовите регистри и обратно. Началните адреси на тези таблици са съответно TABCI и TABIC. При етикети CRBUF и LPBUF за всяко едно от използуваните устройства се осигурява буфер в оперативната памет. С етикет LBUF се дефинира WISP буферът, необходим на процесора HELP. Следва кодиране на функциите на WIOCS за четене и запис на редове. Подпрограмите, които са добавени според фиг.5.4,5.5,5.6 са представени в края на текста.

```
1. МАКРОДЕФИНИЦИЯ: |ИМЕ НА '.
                   |ИМЕТО Е '10@
                   |@
                   |
           ВХОД:   |ИМЕ НА ПРИМЕР.
                   |
         ИЗХОД:    |ИМЕТО Е ПРИМЕР
                   |
2. МАКРОДЕФИНИЦИЯ: |' = ' + '.
                   |  L   '20@
                   |  ADD '30@
                   |  ST  '10@
                   |@
                   |
           ВХОД:   |AB = AD + DB.
                   |
         ИЗХОД:    |  L   AD
                   |  ADD DB
                   |  ST  AB
                   |
3. МАКРОДЕФИНИЦИЯ: |СТОЙНОСТ НА '.
                   |СТОЙНОСТТА Е '11@
                   |@
                   |
           ВХОД:   |СТОЙНОСТ НА ПРИМЕР.
  ПАМЕТ (ПРИМЕР):  |A532-F
                   |
         ИЗХОД:    |СТОЙНОСТТА Е A532-F
                   |
           ВХОД:   |СТОЙНОСТ НА NAME.
          NAME:    |
                   |
         ИЗХОД:    |СТОЙНОСТТА Е
                   |
4. МАКРОДЕФИНИЦИЯ: |' = ' + '.
                   |  L   VAR+'22@
                   |  ADD VAR+'32@
                   |  ST  VAR+'12@
                   |@
                   |
           ВХОД:   |AB = AD + DB.
     ПАМЕТ (AD):   |1
     ПАМЕТ (DB):   |4
      ГЕНЕРАТОР:   |12
                   |
         ИЗХОД:    |  L   VAR+1
                   |  ADD VAR+4
                   |  ST  VAR+12
      ГЕНЕРАТОР:   |13
```

```
5. МАКРОДЕФИНИЦИЯ: |'ПРИМЕР'.
                   |СИМВОЛЪТ СЛЕДВАЩ ПАР1 Е ('13)@
                   |СИМВОЛЪТ СЛЕДВАЩ ПАР2 Е ('23)@
                   |@
                   |
          ВХОД:    |ПРОСТПРИМЕРЗАРАЗЛАГАНЕ.
                   |
        ИЗХОД:     |СИМВОЛЪТ СЛЕДВАЩ ПАР1 Е (П)
                   |СИМВОЛЪТ СЛЕДВАЩ ПАР2 Е ()
                   |
6. МАКРОДЕФИНИЦИЯ: |EVALUATE '.
                   |THE VALUE IS '14@
                   |@
                   |
          ВХОД:    |EVALUATE (3+15)/6.
                   |
        ИЗХОД:     |THE VALUE IS 3
                   |
          ВХОД:    |EVALUATE 12-15.
                   |
        ИЗХОД:     |THE VALUE IS -3
                   |
          ВХОД:    |EVALUATE ЯБЪЛКА+КРУША*ДОМАТ+4.
 ПАМЕТ (ЯБЪЛКА):   |-21
  ПАМЕТ (КРУША):   |7
  ПАМЕТ (ДОМАТ):   |5
                   |
        ИЗХОД:     |THE VALUE IS 18
                   |
          ВХОД:    |EVALUATE ЗЕЛЕНЧУК.
ПАМЕТ (ЗЕЛЕНЧУК):  |РЯПА
                   |
        ИЗХОД:     |********** EXPR ERROR
                   |THE VALUE IS
                   |EVALUATE ЗЕЛЕНЧУК
                   |
7. МАКРОДЕФИНИЦИЯ: |HOW LONG IS '.
                   |THE GIVEN STRING IS '15 CHAR LONG@
                   |@
                   |
          ВХОД:    |HOW LONG IS GEORGE.
                   |
        ИЗХОД:     |THE GIVEN STRING IS 6 CHAR LONG
                   |
          ВХОД:    |HOW LONG IS.
                   |
        ИЗХОД:     |THE GIVEN STRING IS 0 CHAR LONG
```

```
 8. МАКРОДЕФИНИЦИЯ: !ПРИМЕР ' '.
                    !'10'26@
                    !'10 '20@
                    !@
                    !
              ВХОД: !ПРИМЕР ABC XYZ.
                    !
            ИЗХОД: !ABC ABC
                    !
 9. МАКРОДЕФИНИЦИЯ: !РАЗЛОЖИ СПИСЪКА '.
                    !'10'17.@
                    !'10@
                    !'F8@
                    !@
                    !
              ВХОД: !РАЗЛОЖИ СПИСЪКА A,B,C,D.
                    !
            ИЗХОД: !A
                    !B
                    !C
                    !D
                    !
              ВХОД: !РАЗЛОЖИ СПИСЪКА A,(B,C),D.
                    !
            ИЗХОД: !A
                    !(B,C)
                    !D
                    !
              ВХОД: !РАЗЛОЖИ СПИСЪКА ((A,(B,C),D)).
                    !
            ИЗХОД: !(A,(B,C),D)
                    !
10.МАКРОДЕФИНИЦИЯ: !НАМЕРИ ЕКВИВАЛЕНТА НА '.
                    !ЕКВИВАЛЕНТЪТ Е '18@
                    !@
                    !
              ВХОД: !НАМЕРИ ЕКВИВАЛЕНТА НА A.
                    !
            ИЗХОД: !ЕКВИВАЛЕНТЪТ Е 011
                    !
              ВХОД: !НАМЕРИ ЕКВИВАЛЕНТА НА ABC.
                    !
            ИЗХОД: !********** CONV ERROR
                    !ЕКВИВАЛЕНТЪТ Е
                    !НАМЕРИ ЕКВИВАЛЕНТА НА ABC
                    !
11.МАКРОДЕФИНИЦИЯ: !END.
                    !'F0@
                    !@
                    !
              ВХОД: !END.
                    !
            ИЗХОД: !
```

```
12.МАКРОДЕФИНИЦИЯ: |ОТПЕЧАТАЙ '.
                   |          '10'F14@
                   |@
                   |
          ВХОД:    |ОТПЕЧАТАЙ ДОКЛАД.
                   |
         ИЗХОД:    |          ДОКЛАД
                   |
13.МАКРОДЕФИНИЦИЯ: |ТАВ ',',',',.
                   |'F14@
                   |111111 222222 333333@
                   |@
                   |
          ВХОД:    |ТАВ ПЪРВИ,ВТОРИ,ПОСЛЕДЕН,
                   |
         ИЗХОД:    |ПЪРВИ  ВТОРИ  ПОСЛЕД
                   |
14.МАКРОДЕФИНИЦИЯ: |ОТПЕЧАТАЙ ДВА ПЪТИ ',',.
                   |'F14@
                   |111 222 111 222@
                   |@
                   |
          ВХОД:    |ОТПЕЧАТАЙ ДВА ПЪТИ ABCDE,XYZ.
                   |
         ИЗХОД:    |ABC XYZ ABC XYZ
                   |
15.МАКРОДЕФИНИЦИЯ: |LOC '.
                   |'F1@
                   | 11111 EQU *@
                   |@
                   |
          ВХОД:    |LOC A.
                   |
         ИЗХОД:    | A     EQU *
                   |
          ВХОД:    |LOC SIN.
                   |
         ИЗХОД:    | SIN   EQU *
                   |
16.МАКРОДЕФИНИЦИЯ: |START CHANNEL '.
                   |НАЧАЛО 'F15R@
                   |@
                   |
          ВХОД:    |START CHANNEL 5.
                   |
         ИЗХОД:    |
                   |НАЧАЛО
```

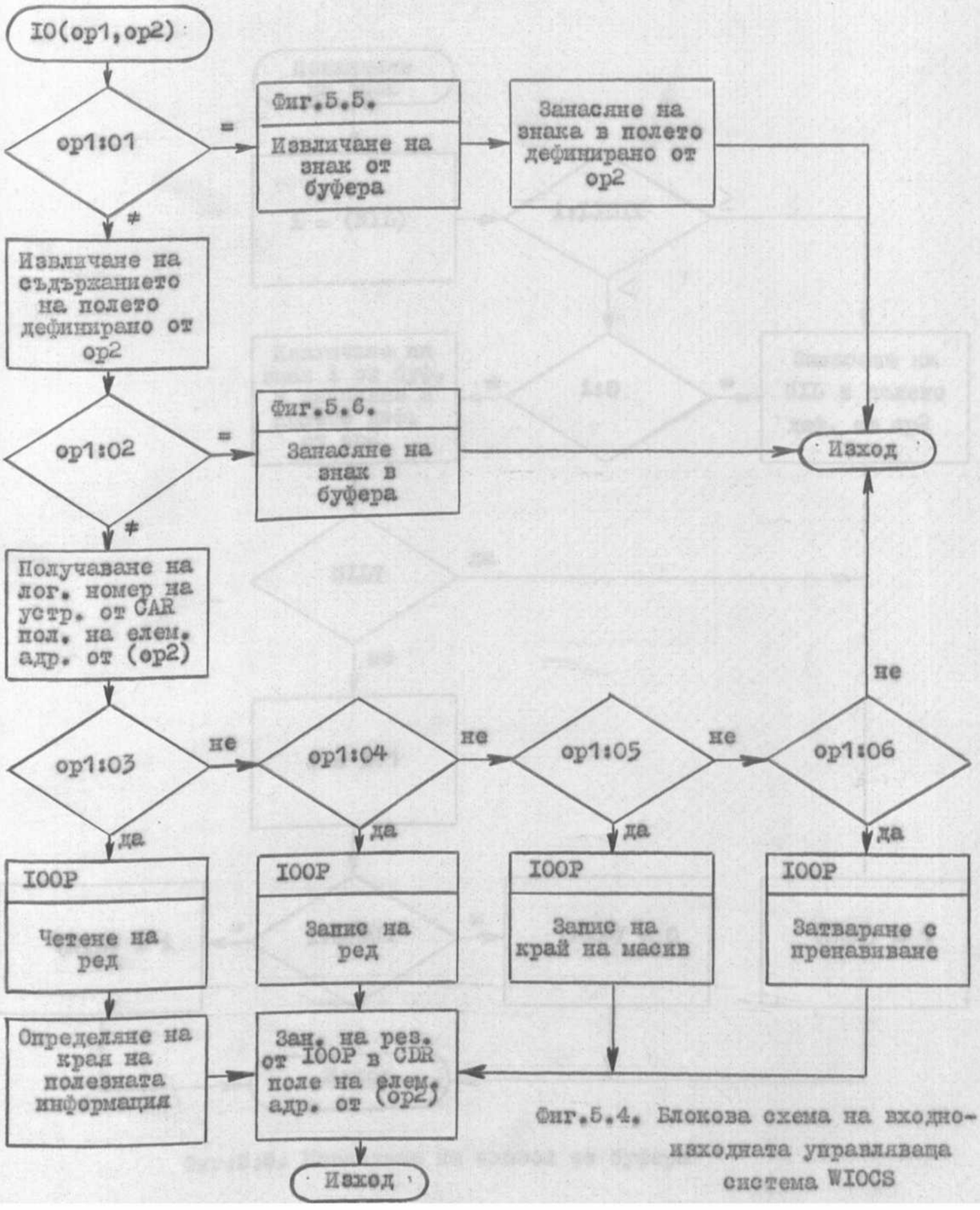
```
17.МАКРОДЕФИНИЦИЯ: !DELETE TO '.
                   !'F25@
                   !@
                   !
          ВХОД:    !DELETE TO END COMMENT.
                   !THE @UICK BROWN FOX JUMPED OVER THE LAZY DOG.S BACK.
                   !1234567890 TIMES
                   !END COMMENT IS AN INTIAL  STRING OF THIS LINE
                   !NEXT VALID OUTPUT.
                   !
         ИЗХОД:    !NEXT VALID OUTPUT.
                   !
18.МАКРОДЕФИНИЦИЯ: !PRINT TO ' FROM CHANNEL '.
                   !'20'F24@
                   !@
                   !
          ВХОД:    !PRINT TO EOF FROM CHANNEL 6.
    КАНАЛ № 6:     !ВСИЧКИ РЕДОВЕ ДО КРАЯ НА МАСИВА
                   !ЩЕ БЪДАТ ОТПЕЧАТАНИ
                   !
                   !
         ИЗХОД:    !ВСИЧКИ РЕДОВЕ ДО КРАЯ НА МАСИВА
                   !ЩЕ БЪДАТ ОТПЕЧАТАНИ
                   !
                   !
19.МАКРОДЕФИНИЦИЯ: !COPY TO 5 REWOUND UNTIL '.
                   !'F25R@
                   !@
                   !
          ВХОД:    !COPY TO 5 REWOUND UNTIL END DATA.
                   !РАЗНИ ТЕКСТОВЕ
                   !АКО END DATA НЕ Е В НАЧАЛОТО НА РЕДА
                   !КОПИРАНЕТО НЯМА ДА СПРЕ
                   !END DATA Е В НАЧАЛОТО НА РЕДА
                   !
         ИЗХОД:    !
                   !РАЗНИ ТЕКСТОВЕ
                   !АКО END DATA НЕ Е В НАЧАЛОТО НА РЕДА
                   !КОПИРАНЕТО НЯМА ДА СПРЕ
                   !
20.МАКРОДЕФИНИЦИЯ: !' E@U '.
                   !'F3@
                   !@
                   !
          ВХОД:    !КНИГА E@U МАЛКИЯТ ПРИНЦ.
                   !
         ИЗХОД:    !
   ПАМЕТ (КНИГА):  !МАЛКИЯТ ПРИНЦ
```

```
21.МАКРОДЕФИНИЦИЯ: !SKIP '.
                   !'F4@
                   !@
                   !
         ВХОД:     !SKIP 3.
                   !
        ИЗХОД:     !
                   !
         ВХОД:     !SKIP N+7.'
   ПАМЕТ (N):      !5
                   !
        ИЗХОД:     !
                   !
         ВХОД:     !SKIP VAR-4.
 ПАМЕТ (VAR):      !ИМЕ
                   !
        ИЗХОД:     !********** EXPR ERROR
                   !SKIP VAR-4
                   !
22.МАКРОДЕФИНИЦИЯ: !IF ' = ' SKIP '.
                   !'F50@
                   !@
                   !
         ВХОД:     !IF ТОВА = ОНОВА SKIP 4.
                   !
        ИЗХОД:     !
                   !
         ВХОД:     !IF АБВГ = АБВГ SKIP 17.
                   !
        ИЗХОД:     !
                   !
23.МАКРОДЕФИНИЦИЯ: !IF ' NE ' SKIP '.
                   !'F51@
                   !@
                   !
         ВХОД:     !IF ТОВА NE ОНОВА SKIP 4.
                   !
        ИЗХОД:     !
                   !
24.МАКРОДЕФИНИЦИЯ: !IF ' LT ' SKIP '.
                   !'F6-@
                   !@
                   !
         ВХОД:     !IF 14 LT X+2 SKIP 9.
   ПАМЕТ (X):      !23
                   !
        ИЗХОД:     !
```

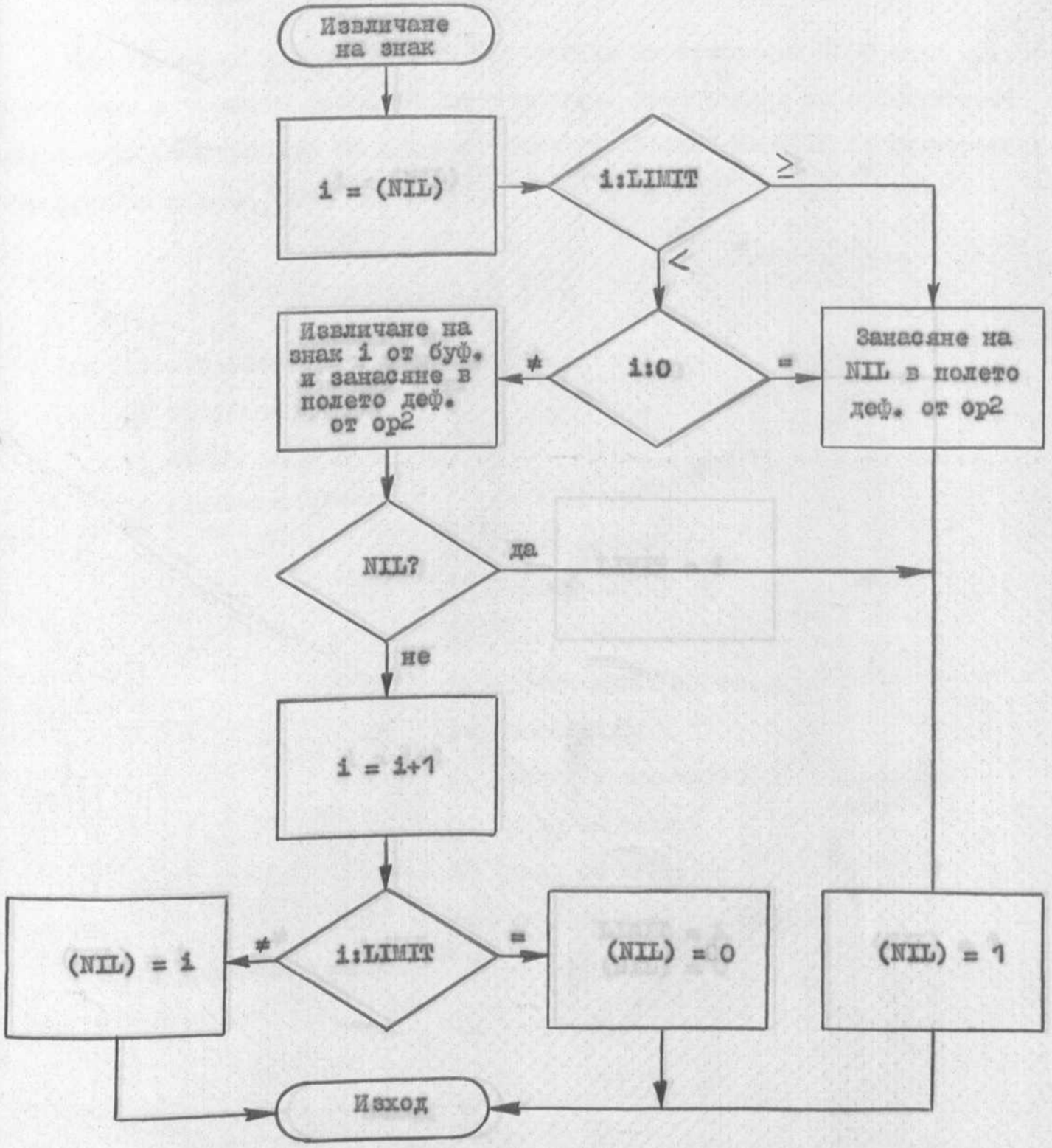
```
25.МАКРОДЕФИНИЦИЯ: !IF ' LE ' SKIP '.
                   !IF '10 LT '20 SKIP '30+1@
                   !'F60@
                   !@
                   !
          ВХОД:    !IF 1234 LE 4321 SKIP 2.
                   !
        ИЗХОД:     !
                   !
          ВХОД:    !IF 1234 LE 1234 SKIP 3.
                   !
        ИЗХОД:     !
                   !
          ВХОД:    !IF K LE L SKIP M.
     ПАМЕТ (K):    !3
     ПАМЕТ (L):    !4
     ПАМЕТ (M):    !LOC X
                   !
        ИЗХОД:     !********** EXPR ERROR
                   !IF K LE L SKIP M
                   !
                   !
26.МАКРОДЕФИНИЦИЯ: !ИЗВЕДИ ' РЕДА ЗАПОЧВАЩИ С '.
                   !'10'F7@
                   !'20@
                   !'F8@
                   !@
                   !
          ВХОД:    !ИЗВЕДИ 3 РЕДА ЗАПОЧВАЩИ С *.
                   !
        ИЗХОД:     !*
                   !*
                   !*
                   !
          ВХОД:    !ИЗВЕДИ N-3 РЕДА ЗАПОЧВАЩИ С #.
     ПАМЕТ (N):    !5
                   !
        ИЗХОД:     !#
                   !#
                   !
          ВХОД:    !ИЗВЕДИ M-2 РЕДА ЗАПОЧВАЩИ С C.
     ПАМЕТ (M):    !2
                   !
        ИЗХОД:     !C
```

```
27.МАКРОДЕФИНИЦИЯ: !SPLIT IF SPACE FOLLOW'.
                   !'10'17 ,@
                   !IF '13 = , SKIP 5@
                   !'F7@
                   !'10'27@
                   !'20@
                   !'F8@
                   !SKIP 1@
                   !'10@
                   !'F8@
                   !@
                   !
          ВХОД:    !SPLIT IF SPACE FOLLOW ABC,XYZ JONES, 14.
                   !
         ИЗХОД:    !ABC
                   !X
                   !Y
                   !Z
                   !JONES
                   !1
                   !4
                   !
28.МАКРОДЕФИНИЦИЯ: !ПРИМЕР.
                   !НАПУСНИ ТАЗИ ДЕФИНИЦИЯ 'F9@
                   !ТОЗИ РЕД НИКОГА НЯМА ДА БЪДЕ СКАНИРАН@
                   !@
                   !
          ВХОД:    !ПРИМЕР.
                   !
         ИЗХОД:    !
                   !
                   !
29.МАКРОДЕФИНИЦИЯ: !NUMBER '.
                   !'10'271234567890@
                   !IF '20 =  SKIP 1@
                   !НЕПРАВИЛЕН ПАРАМЕТЪР 'FE 'F9@
                   !'F8@
                   !VAR EQU ='10@
                   !@
                   !
          ВХОД:    !NUMBER 12345.
                   !
         ИЗХОД:    !
   ПАМЕТ (VAR):    !=12345
                   !
          ВХОД:    !NUMBER XYZ.
                   !
         ИЗХОД:    !********** CONV ERROR
                   !НЕПРАВИЛЕН ПАРАМЕТЪР
                   !NUMBER XYZ
   ПАМЕТ (VAR):    !
```

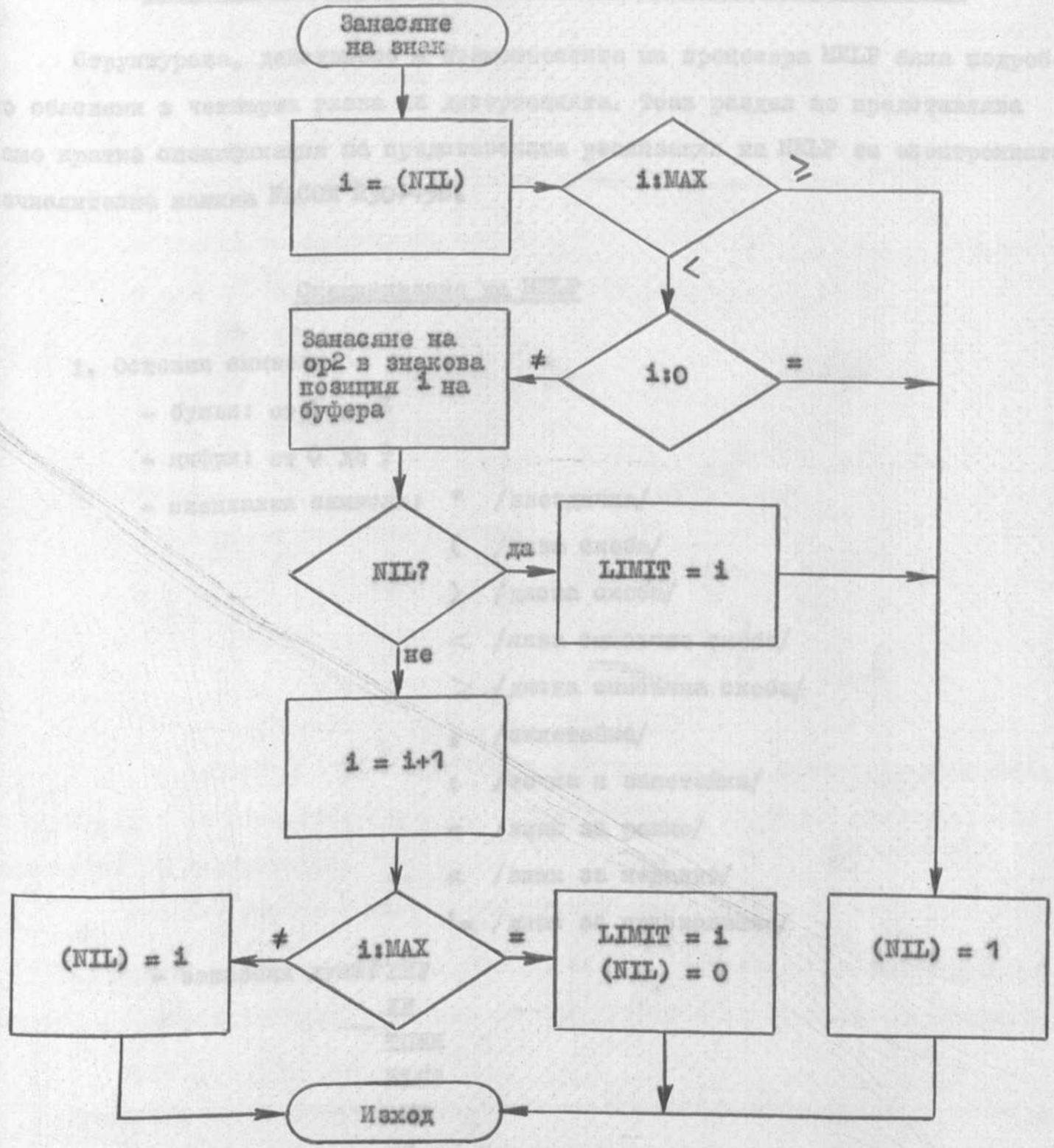
Фиг.5.3. Примери на макродефиниции, обработка на параметри и действия на компилатора BASCOM



Фиг.5.4. Блокова схема на входно-изходната управляваща система WIOCS



Фиг.5.5. Извличане на символ от буфера



Фиг.5.6. Занасяне на символ в буфера

5.4. ПРОЦЕСОР ЗА ОБРАБОТКА НА СИМВОЛНА СПИСЪЧНА ИНФОРМАЦИЯ HELP

Структурата, действието и възможностите на процесора HELP бяха подробно обяснени в четвърта глава на дисертацията. Този раздел ще представлява само кратка спецификация на представената реализация на HELP за електронната изчислителна машина FACOM 230-45S.

Спецификация на HELP

1. Основни символи:
   - букви: от A до Z
   - цифри: от 0 до 9
   - специални символи:
     - * /звездичка/
     - (/лява скоба/
     -) /дясна скоба/
     - < /лява списъчна скоба/
     - > /дясна списъчна скоба/
     - , /запетайка/
     - ; /точка и запетайка/
     - = /знак за равно/
     - ≠ /знак за неравно/
     - '= /знак за присвояване/
   - запазени думи:
     - DEF
     - IF
     - THEN
     - ELSE
     - AND
     - OR
     - NOT

2. Константи: низове от букви, цифри и звездички, започващи със звездичка.

3. Идентификатори: низове от букви, цифри и звездички, незапочващи със звездичка.

4. Програма: поредица от символни изрази за пресмятане.

5. Синтаксис на символните изрази:

$P ::= A \mid \langle E_1, E_2, \ldots, E_n \rangle \mid (E) \mid I(E_1, E_2, \ldots, E_n) \mid I$

LE ::= NOT LE | $P_1 = P_2$ | $P_1 = P_2$ | P

SE ::= LE AND SE | LE OR SE | LE

E ::= IF SE THEN SE ELSE E | SE

6. Функции:

- основни вградени функции: CAR(R)
  CDR(R)
  CONS(X,R)
  ATOM(X)
  NULL(X)

- дефинирани от потребителя функции: всички използувани функции /с изключение на основните вградени функции/ трябва да бъдат предварително дефинирани от програмиста:

DEF име-на-функция (списък-на-формални-аргументи) := дефиниращ-израз ;

7. Грешки в потребителската програма: При препълване на паметта изпълнението на потребителската програма се прекратява и се отпечатва съобщение

FAIL 2

Препълването на паметта обикновено се предизвиква от функция, която не може да излезе от цикъл на рекурсивно използуване. Изразът, който причинява грешката, е последният отпечатан израз и той трябва да бъде внимателно про-

верен. Разбира се, възможно е тази грешка да се дължи на недостатъчно памет за изчисляване на даден израз, ако потребителската програма е много голяма.

Грешките при компилиране се дължат на синтаксически неправилни изрази в потребителската програма. Текущият входен ред се отпечатва, като всички сканирани позиции се подчертават с чертички. По този начин се локализира мястото на грешката. След това процесорът HELP отпечатва допълнителните редове на грешния израз /ако такива съществуват/ и номера на грешката.

Грешките при интерпретиране се откриват в процеса на пресмятане на изразите. След отпечатване на номера на грешката, процесорът HELP отпечатва извикванията на функции за изпълнение, участвуващи в израза. Последното извикване се отпечатва първо, следва по-предното извикване и т.н. За всяко извикване се отпечатва както името на функцията, така и фактическите аргументи за това извикване. Всяко извикване се отпечатва на нов ред.

В табл.5.3 са специфицирани грешките при компилиране, а в табл.5.4 - грешките при интерпретиране на символните изрази на потребителската програма.

ГРЕШКИ ПРИ КОМПИЛИРАНЕ		
Код	Значение	Обяснение
10	Неправилен първичен елемент	Сканираната конструкция трябва да бъде първичен елемент, но той няма съответна правилна форма.
11	Недекларирай идентификатор	Идентификатор, който не е име на функция, може да се употреби само като формален аргумент в дефиниция на функция.
12	Неправилен разделител	Фактическите аргументи при извикване на функция трябва да бъдат разделени със запетайки, а изброяването им да завършва с дясна скоба.
13	Неправилен разделител	Изразите в даден списък трябва да бъдат разделени със запетайки, а списъкът да завършва с дясна списъчна скоба.

Код	Значение	Обяснение
14	Пропусната скоба	Разпознат е първичен елемент с форма (E), но завършващата дясна скоба е пропусната.
15	Пропуснато then	Разпознато е условие в израз, но то не е следвано от THEN.
16	Пропуснато else	Разпознат е израз, в който са употребени IF и THEN , но е пропуснато ELSE.
17	Пропуснат символ ;	Разпознат е израз, който обаче не завършва със символа ";".
18	Пропусната функция	Разпознат е символът DEF , но следващата конструкция не е идентификатор, следван от формални аргументи, заградени в скоби.
19	Многократна дефиниция	Дефиниция на функция със същото име вече съществува.
20	Неправилен формален аргумент	Очаква се формален аргумент, но конструкцията не е идентификатор.
21	Дублиран формален аргумент	Текущият формален аргумент вече е използуван веднаж в същата дефиниция.
22	Неправилен разделител	Формалните аргументи трябва да бъдат разделени със запетайки и изброяването им да завършва с дясна скоба.
23	Пропуснат символ '=	Разпознат е идентификатор на функция следващ DEF , но той не е следван от символа ":=".

Табл.5.3. Грешки при компилиране

ГРЕШКИ ПРИ ИНТЕРПРЕТИРАНЕ		
Код	Значение	Обяснение
0	Несъответствие между броя на формалните и фактическите аргументи или недефинирана функция	Броят на фактическите аргументи при извикването на функция, дефинирана от програмиста, не съответствува на броя на формалните аргументи от дефиницията или липсва съответна дефиниция.
1	Неправилен аргумент на функцията CAR	Вградената функция CAR не е извикана със само един аргумент или аргументът е атом.
2	Неправилен аргумент на функцията CDR	Вградената функция CDR не е извикана със само един аргумент или аргументът е атом.
3	Неправилен аргумент на функцията CONS	Вградената функция CONS не е извикана с два аргумента или вторият аргумент не е списък.
4	Сравняване на списъци	Първият операнд на $P_1=P_2$ или $P_1=P_2$ е списък.
5	Сравняване на списъци	Вторият операнд на $P_1=P_2$ или $P_1=P_2$ е списък.
6	Стойността не е логическа	Операндът на NOT LE не е логическа величина.
7	Стойността не е логическа	Първият операнд на LE AND SE или LE OR SE не е логическа величина.
8	Стойността не е логическа	Вторият операнд на LE AND SE или LE OR SE не е логическа величина.
9	Стойността не е логическа	Стойността на условието на израз не е логическа величина.

Табл.5.4. Грешки при интерпретиране

Извикването на процесора HELP се извършва чрез следната последователност от управляващи карти за операционната система на електронната изчислителна машина FACOM 230-45S:

```
$ JOB HELP
$ EX HELP,LIB=HELP,VOL=0006
$ FD LIST=PK,VOL=0001,CYL=1,SOUT=A
$ FD UIN=*
*TRUE,*FALSE,CAR(CDR(CONS(ATOM(NULL(;          /специална карта/
```

/карти на потребителската програма/

```
$ FIN
```

5.5. БЛОКОВИ СХЕМИ НА ЕТАПИТЕ НА РЕАЛИЗАЦИЯТА

Този раздел съдържа блоковите схеми на етапите на реализацията на процесора за обработка на символна списъчна информация HELP за електронната изчислителна машина FACOM 230-45S. Трябва да се поясни, че при всички блокови схеми активните програми са показани в горната част, а масивите с данни - в долната част на чертежите.

Листингите на етапите на реализацията могат да се намерят в приложение В.

5.5.1. Проверка на входно-изходната управляваща система IOCS

За проверка на входно-изходната управляваща система IOCS са подготвени два масива - IOCP и IOCT. Масивът IOCP представлява програма за извършване на проверката, а IOCT е масив на данните за тази проверка. На фиг.5.7 е показана блок-схемата на процеса на проверка. При правилно кодиране на входно-изходната управляваща система IOCS чрез канали 3 и 4 се извеждат двата масива, показани на фиг.5.8.

5.5.2. Проверка на простия компилатор SIMCOM

След проверката на входно-изходната управляваща система се извършва проверка на простия компилатор SIMCOM. Използува се масивът с данни за тестване SIMT. Блок-схемата на етапа на проверка на SIMCOM е показана на фиг.5.9. При правилна работа на простия компилатор SIMCOM чрез канал 3 се извежда /перфорира/ масивът, показан на фиг.5.10. В този случай полученият в машинна форма прост компилатор SIMCOM се използува в следващите етапи на реализацията.

5.5.3. Първа проверка за грешки в моделирането на абстрактната машина FLBM

Масивът FLBM представлява набора от дефиниции, които процесорът SIMCOM използува за преобразуване на операторите от езика FLUB в оператори на FORTRAN. Масивите FLT1 и FLD1 служат за извършване на първата проверка на кодирането на дефинициите. Блок-схемата на този етап е показана на фиг.5.11. При правилно кодиране чрез канал 4 се извежда /отпечатва/ масив, в който всички редове са номерирани. При откриване на грешки се отпечатват съответни насочващи съобщения. Тези грешки трябва да се отстранят и етапът да се повтори.

5.5.4. Втора проверка за грешки в моделирането на абстрактната машина FLBM

Втората проверка за грешки в моделирането на абстрактната машина FLBM се извършва след успешното завършване на първата проверка. Използуват се масивите FLT2 и FLD2. Блок-схемата на този етап е показана на фиг.5.12. При правилно кодиране на дефинициите чрез канали 3 и 4 се извеждат два масива, състоящи се от номерирани редове. При откриване на грешки се отпечатват съответни насочващи съобщения. Тези грешки трябва да се отстранят и етапът да се повтори.

5.5.5. Проверка и получаване на процесора STAGE2 в обектна форма

Масивът STG2 е текстът на процесора STAGE2 на езика FLUB. Като се използуват простият компилатор SIMCOM и проверените вече дефиниции от масива FLBM, извършва се преобразуването на изходния текст на процесора STAGE2 във фортранова програма. Тази фортранова програма се транслира и процесорът

STAGE2 се получава в обектна машинна форма. Проверката на процесора STAGE2 се извършва като се използува масивът с данни ST2T. Блок-схемата на този етап е показана на фиг.5.13. Чрез канали 3 и 4 се извеждат два масива, от съдържанието на които може да се прецени дали процесорът STAGE2 работи правилно.

5.5.6. Получаване на основния компилатор BASCOM в обектна форма

Масивите STG2 и WSED се обединяват съгласно указанията в съответните карти на WSED. Полученият обединен масив представлява изходния текст на основния компилатор BASCOM. Като се използуват процесорът STAGE2 и дефинициите от масива BASM /масивът BASM е разширение на масива FLEM поради особеностите на WSED/, извършва се преобразуване на изходния текст на BASCOM във фортранова програма. Тази фортранова програма се транслира и основният компилатор BASCOM се получава в обектна машинна форма. Блок-схемата на този етап е показана на фиг.5.14.

5.5.7. Проверка за грешки в моделирането на абстрактната машина WSPM

Масивът WSPM е наборът от дефиниции, които основният компилатор BASCOM използува за преобразуване на операторите на езика WISP в оператори на езика на асамблера на изчислителната машина FACOM 230-45S. Масивът WSPT представлява програма, а масивът WSPD - данни за извършване проверката на кодирането на дефинициите. Блок-схемата на тази проверка е дадена на фиг.5.15. При правилно кодиране на дефинициите се извежда /отпечатва/ масив, в който всички редове са номерирани. При откриване на грешки се отпечатват съответни насочващи съобщения. Тези грешки трябва да се отстранят и етапът да се повтори.

5.5.8. Получаване на процесора HELP в обектна форма

Масивите HELP и DYNL се обединяват. Обединеният масив представлява изходният текст на процесора HELP на езика WISP. Като се използуват основният компилатор BASCOM и дефинициите от масива WSPM, извършва се преобразуване на изходния текст на процесора HELP в асамблерова програма. Тази програма се асамблира и процесорът HELP се получава в обектна машинна форма. Блок-схемата на този етап е показана на фиг.5.16.

Процесорът HELP представлява желаната реализация на процесор за обработка на символна списъчна информация.

Фиг.5.7. Блок-схема на проверката на входно-изходната управляваща система IOCS

```
MOBILE PROGRAMMING SYSTEM I/O TEST - CHANNEL 3 OUTPUT                    IOCT0001
```

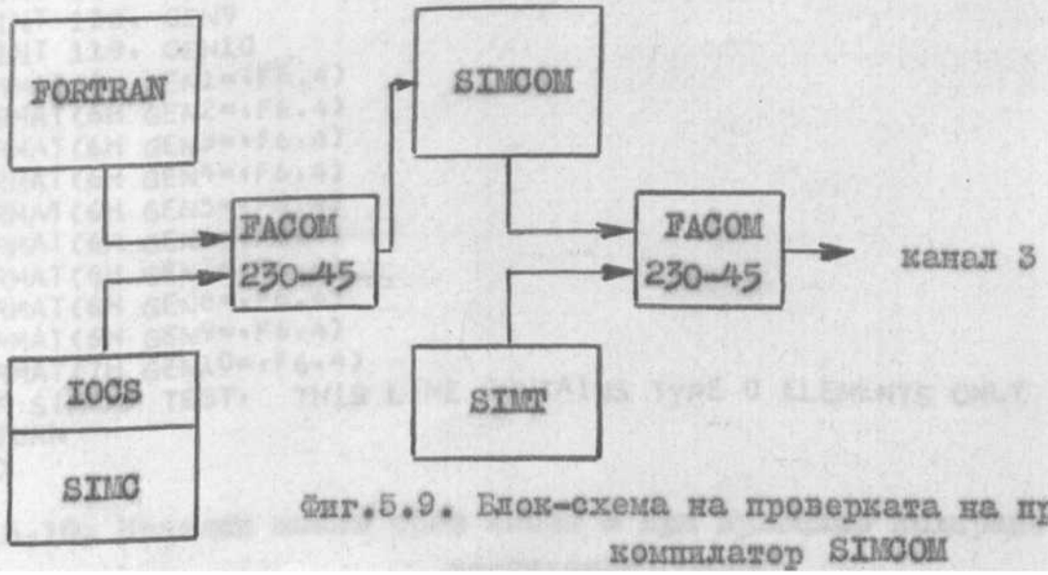
а/ изведен масив чрез канал 3

```
MOBILE PROGRAMMING SYSTEM I/O TEST - CHANNEL 4 OUTPUT                    IOCT0001

1)  IF THIS APPEARS TWICE, THE CHANNEL IS NOT BEING CLEARED PROPERLY.    IOCT0002
2)
3)  IF THIS LINE IS OK, THEN TWO BUFFER AREAS DO NOT INTERACT.           IOCT0003
4)  IF THIS LINE IS OK, THEN THE BASE OF THE ARRAY CAN BE OFFSET.        IOCT0004
```

б/ изведен масив чрез канал 4

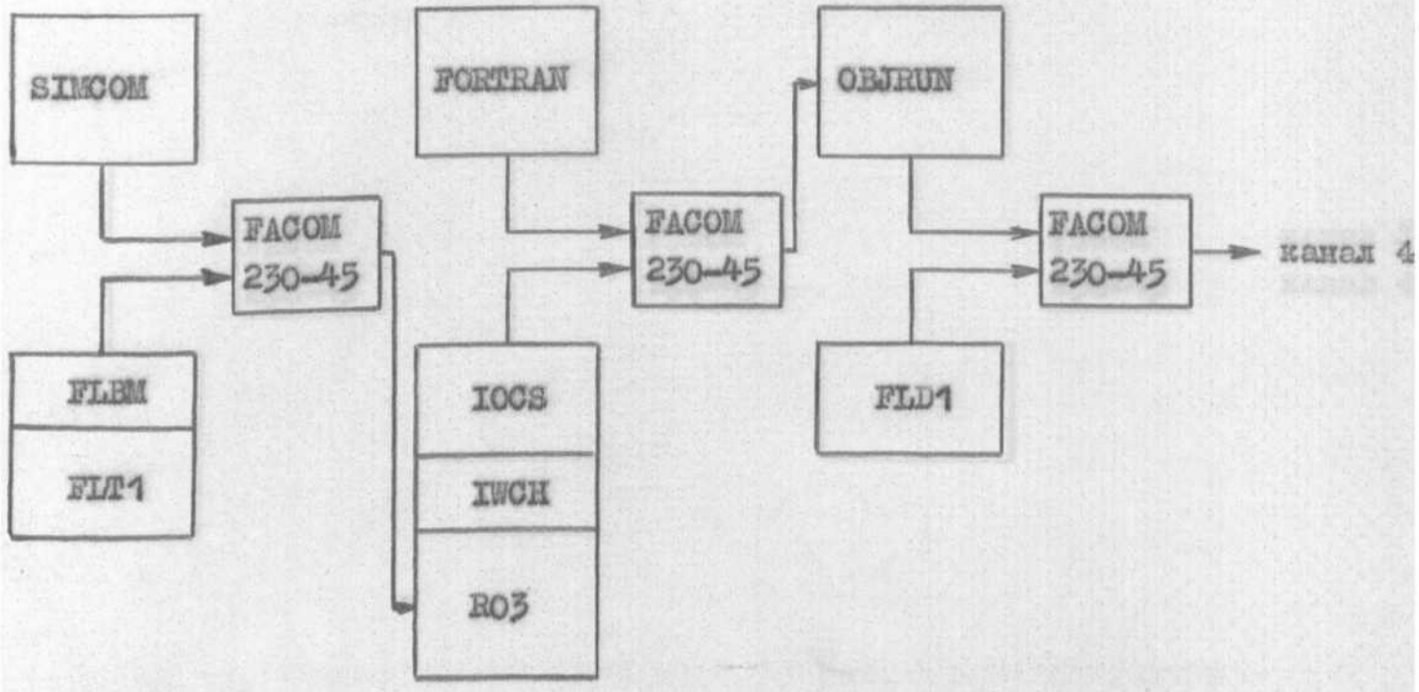
Фиг.5.8. Изведени масиви при правилно кодиране на входно-изходната управляваща система IOCS



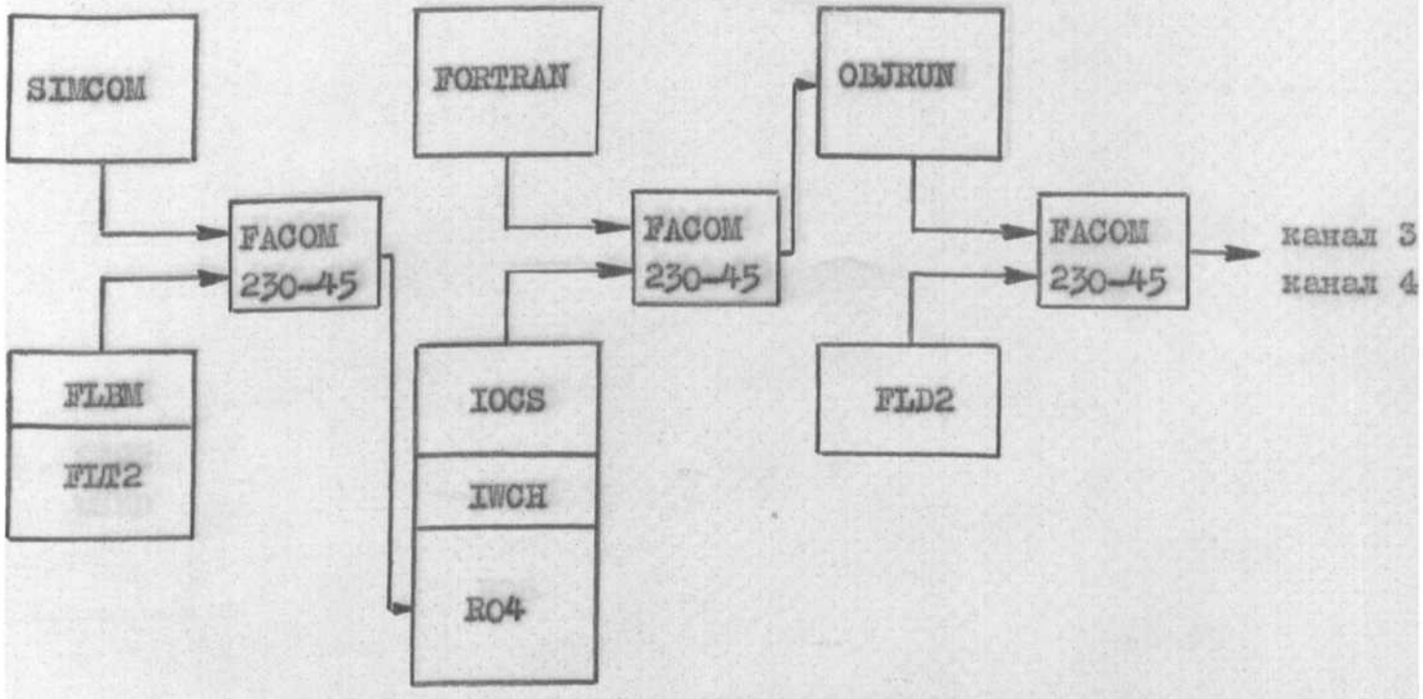
Фиг.5.9. Блок-схема на проверката на простия компилатор SIMCOM

```
      SUBROUTINE PROGR                                                  SIMT0041
C  THIS TEST DATA CHECKS OUT THE WORKINGS OF SIMCMP. THE FIRST THREE    SIMT0042
C  LINES WILL NOT BE MATCHED, AND SHOULD BE OUTPUT IN THEIR ENTIRETY.   SIMT0043
      PRINT 100, GEN1
      PRINT 101, GEN2
      PRINT 102, GEN3
      PRINT 103, GEN4
      PRINT 104, GEN5
      PRINT 105, GEN6
      PRINT 106, GEN7
      PRINT 107, GEN8
      PRINT 108, GEN9
      PRINT 109, GEN10
  100 FORMAT(6H GEN1=,F6.4)
  101 FORMAT(6H GEN2=,F6.4)
  102 FORMAT(6H GEN3=,F6.4)
  103 FORMAT(6H GEN4=,F6.4)
  104 FORMAT(6H GEN5=,F6.4)
  105 FORMAT(6H GEN6=,F6.4)
  106 FORMAT(6H GEN7=,F6.4)
  107 FORMAT(6H GEN8=,F6.4)
  108 FORMAT(6H GEN9=,F6.4)
  109 FORMAT(7H GEN10=,F6.4)
      NINE=1+2+3+4+5+6+7+8+9
      TESTC=A
      TYPE1=001
C  THIS CODE BODY LINE HAS NO TERMINATOR. IT SHOULD HAVE 80 CHARACTERS. SIMT0039
      TESTC=0
      TYPE1=027
      PRINT 110, GEN1
      PRINT 111, GEN2
      PRINT 112, GEN3
      PRINT 113, GEN4
      PRINT 114, GEN5
      PRINT 115, GEN6
      PRINT 116, GEN7
      PRINT 117, GEN8
      PRINT 118, GEN9
      PRINT 119, GEN10
  110 FORMAT(6H GEN1=,F6.4)
  111 FORMAT(6H GEN2=,F6.4)
  112 FORMAT(6H GEN3=,F6.4)
  113 FORMAT(6H GEN4=,F6.4)
  114 FORMAT(6H GEN5=,F6.4)
  115 FORMAT(6H GEN6=,F6.4)
  116 FORMAT(6H GEN7=,F6.4)
  117 FORMAT(6H GEN8=,F6.4)
  118 FORMAT(6H GEN9=,F6.4)
  119 FORMAT(7H GEN10=,F6.4)
C  END OF SIMCMP TEST.  THIS LINE CONTAINS TYPE 0 ELEMENTS ONLY
      RETURN
      END
```

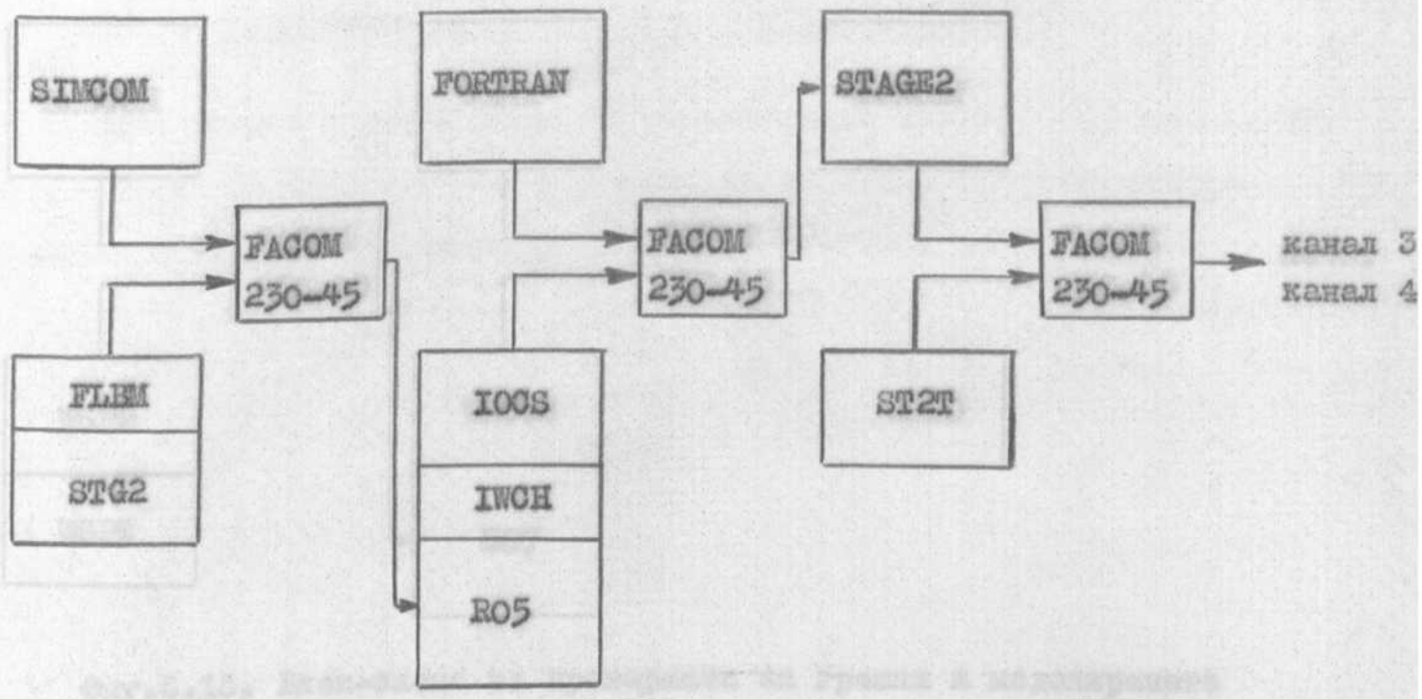
Фиг.5.10. Изведен масив чрез канал 3 при правилно кодиране на простия компилатор SIMCOM



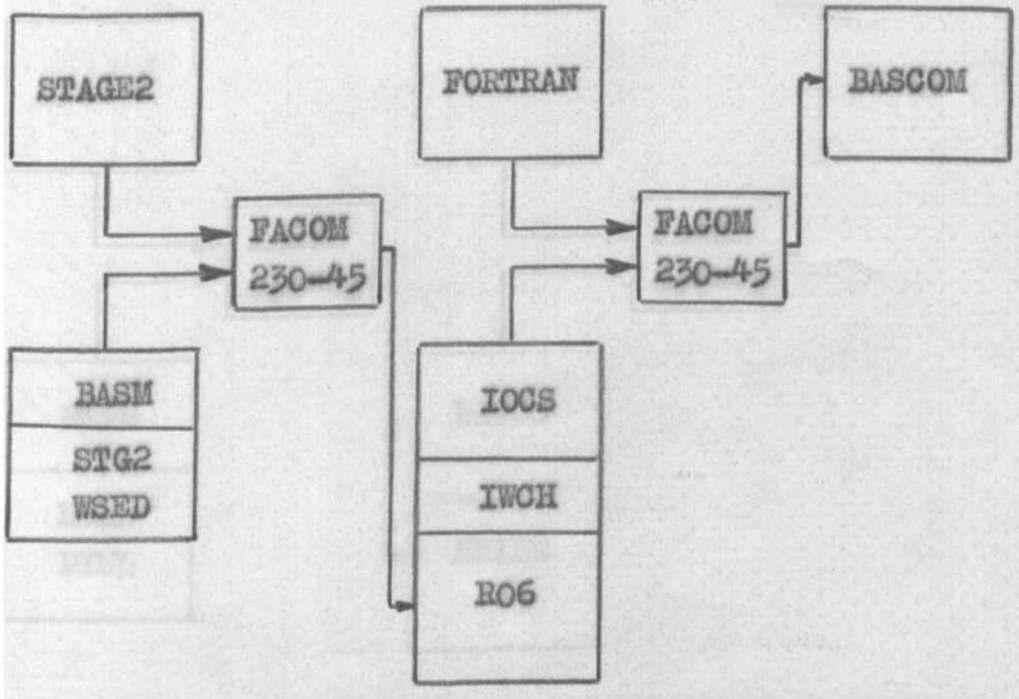
Фиг.5.11. Блок-схема на първата проверка за грешки в моделирането на абстрактната машина FLBM



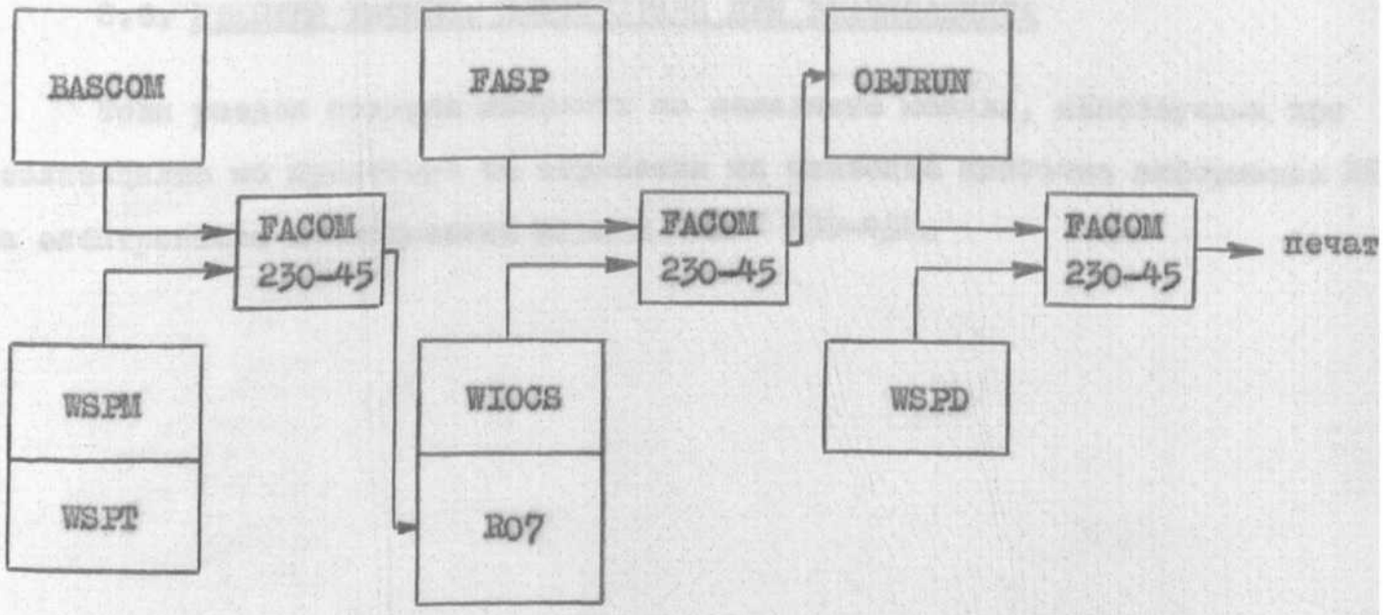
Фиг.5.12. Блок-схема на втората проверка за грешки в моделирането на абстрактната машина FLBM



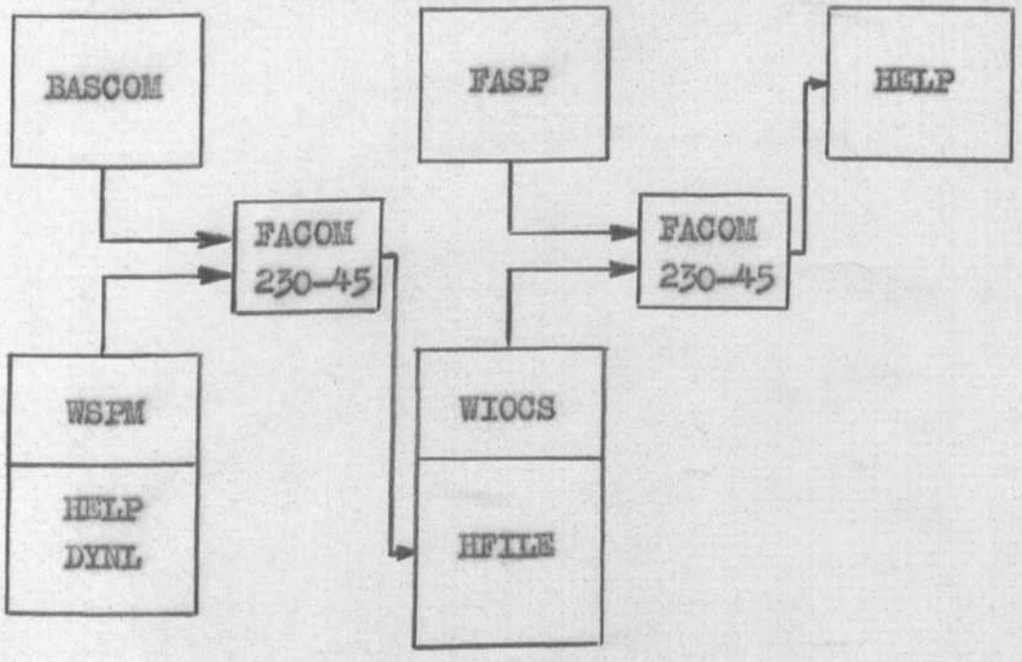
Фиг.5.13. Блок-схема на проверката и получаването на процесора STAGE2 в обектна форма



Фиг.5.14. Блок-схема на получаването на основния компилатор BASCOM в обектна форма



Фиг.5.15. Блок-схема на проверката за грешки в моделирането на абстрактната машина WSPM



Фиг.5.16. Блок-схема на получаването на процесора HELP в обектна форма

5.6. ИЗХОДНИ МАСИВИ, ИЗПОЛЗУВАНИ ПРИ РЕАЛИЗАЦИЯТА

Този раздел съдържа листинги на изходните масиви, използувани при реализацията на процесора за обработка на символна списъчна информация HELP за електронната изчислителна машина FACOM 230-45S.

IOCS

```
      DIMENSION JCHAN(24)                                          IOCS0001
      COMMON MAXCH,JCHAN                                           IOCS0002
      KCH=0                                                        IOCS0003
      JCHAN(KCH+1)=1                                               IOCS0004
      JCHAN(KCH+2)=0                                               IOCS0005
      JCHAN(KCH+3)=0                                               IOCS0006
      JCHAN(KCH+4)=1                                               IOCS0007
      JCHAN(KCH+5)=5                                               IOCS0008
      JCHAN(KCH+6)=1                                               IOCS0009
      KCH=KCH+6                                                    IOCS0010
      JCHAN(KCH+1)=1                                               IOCS0011
      JCHAN(KCH+2)=1                                               IOCS0012
      JCHAN(KCH+3)=1                                               IOCS0013
      JCHAN(KCH+4)=2                                               IOCS0014
      JCHAN(KCH+5)=4                                               IOCS0015
      JCHAN(KCH+6)=1                                               IOCS0016
      KCH=KCH+6                                                    IOCS0017
      JCHAN(KCH+1)=0                                               IOCS0018
      JCHAN(KCH+2)=1                                               IOCS0019
      JCHAN(KCH+3)=0                                               IOCS0020
      JCHAN(KCH+4)=0                                               IOCS0021
      JCHAN(KCH+5)=7                                               IOCS0022
      JCHAN(KCH+6)=1                                               IOCS0023
      KCH=KCH+6                                                    IOCS0024
      JCHAN(KCH+1)=0                                               IOCS0025
      JCHAN(KCH+2)=3                                               IOCS0026
      JCHAN(KCH+3)=0                                               IOCS0027
      JCHAN(KCH+4)=0                                               IOCS0028
      JCHAN(KCH+5)=6                                               IOCS0029
      JCHAN(KCH+6)=1                                               IOCS0030
      KCH=KCH+6                                                    IOCS0031
      MAXCH=KCH/6                                                  IOCS0032
      CALL PROGR                                                   IOCS0033
      DO 1 K=1,MAXCH                                               IOCS0034
    1 KCH=IOOP(0,K,JCHAN,1,1)                                      IOCS0035
      STOP                                                         IOCS0036
      END                                                          IOCS0037
```

```
      FUNCTION IOOP(JOP,JCH,JBA,JP1,JP2)                                IOCSOP01
      DIMENSION JCHAN(36)                                               IOCSOP02
      DIMENSION JBA(1)                                                  IOCSOP03
      COMMON MAXCH,JCHAN                                                IOCSOP04
      IF (JCH) 7,5,1                                                    IOCSOP05
    1 IF (MAXCH .LT. JCH) GO TO 7                                       IOCSOP06
      KCH=(JCH-1)*6                                                     IOCSOP07
      K=JCHAN(KCH+6)                                                    IOCSOP08
      IF (K .EQ. 0) GO TO 7                                             IOCSOP09
      IF (JOP) 10,2,20                                                  IOCSOP10
    2 IF (JCHAN(KCH+6) .EQ. 1) GO TO 7                                  IOCSOP11
      IF (JCHAN(KCH+4) .EQ. 0) GO TO 3                                  IOCSOP12
      CALL ICNTL(KCH,JBA,JP1,JP2,IOOP)                                  IOCSOP13
      RETURN                                                            IOCSOP14
    3 IF (JP1 .NE. JP2) GO TO 7                                         IOCSOP15
      JCHAN(KCH+6)=1                                                    IOCSOP16
    4 IOOP=0                                                            IOCSOP17
      RETURN                                                            IOCSOP18
    5 IF (JOP .GE. 0) GO TO 4                                           IOCSOP19
    6 IOOP=1                                                            IOCSOP20
      RETURN                                                            IOCSOP21
    7 IOOP=2                                                            IOCSOP22
      RETURN                                                            IOCSOP23
   10 GO TO (11,12,7,12,6),K                                            IOCSOP24
   11 IF (JCHAN(KCH+1) .EQ. 0) GO TO 7                                  IOCSOP25
      JCHAN(KCH+6)=2                                                    IOCSOP26
      IF (JCHAN(KCH+3) .EQ. 0) GO TO 12                                 IOCSOP27
      CALL IOPEN(-JOP,KCH,JBA,JP1,JP2,IOOP)                             IOCSOP28
      IF (IOOP .NE. 0) RETURN                                           IOCSOP29
   12 CALL IREAD(-JOP,KCH,JBA,JP1,JP2,IOOP)                             IOCSOP30
      GO TO 30                                                          IOCSOP31
   20 GO TO (21,7,22,22,6),K                                            IOCSOP32
   21 IF (JCHAN(KCH+2) .EQ. 0) GO TO 7                                  IOCSOP33
      JCHAN(KCH+6)=3                                                    IOCSOP34
      IF (JCHAN(KCH+3) .EQ. 0) GO TO 22                                 IOCSOP35
      CALL IOPEN(JOP,KCH,JBA,JP1,JP2,IOOP)                              IOCSOP36
      IF (IOOP .NE. 0) RETURN                                           IOCSOP37
   22 CALL IWRIT(JOP,KCH,JBA,JP1,JP2,IOOP)                              IOCSOP38
   30 IF (IOOP .EQ. 1) JCHAN(KCH+6)=5                                   IOCSOP39
      RETURN                                                            IOCSOP40
      END                                                               IOCSOP41
```

```
      SUBROUTINE IREAD(JOP,KCH,JBA,JP1,JP2,JRES)                        IOCSRD01
      DIMENSION JBA(30000),JCHAN(36)                                    IOCSRD02
      COMMON MAXCH,JCHAN                                                IOCSRD03
      DATA LNLNG/80/                                                    IOCSRD04
      JRES=0                                                            IOCSRD05
      JDEV=JCHAN(KCH+5)                                                 IOCSRD06
      JP2=JP1-1+LNLNG                                                   IOCSRD07
      READ (JDEV,100,END=2) (JBA(K),K=JP1,JP2)                          IOCSRD08
      DO 1 K=JP1,JP2                                                    IOCSRD09
    1 JBA(K)=ICVCI(JBA(K))                                              IOCSRD10
      JP2=JP2+1                                                         IOCSRD11
      RETURN                                                            IOCSRD12
    2 JRES=1                                                            IOCSRD13
      RETURN                                                            IOCSRD14
  100 FORMAT (80A1)                                                     IOCSRD15
      END                                                               IOCSRD16


      SUBROUTINE IWRIT(JOP,KCH,JBA,JP1,JP2,JRES)                        IOCSWR01
      DIMENSION JBA(30000),LINE(136),JCHAN(36)                          IOCSWR02
      COMMON MAXCH,JCHAN                                                IOCSWR03
      DATA LNLNG,KMAX/80,136/                                           IOCSWR04
      DATA KFILL/1H /                                                   IOCSWR05
      JRES=0                                                            IOCSWR06
      JHOW=JCHAN(KCH+2)                                                 IOCSWR07
      JDEV=JCHAN(KCH+5)                                                 IOCSWR08
      IF (JP1 .EQ. JP2) GO TO 2                                         IOCSWR09
      K1=JP1                                                            IOCSWR10
      DO 1 K=1,KMAX                                                     IOCSWR11
      LINE(K)=ICVIC(JBA(K1))                                            IOCSWR12
      K1=K1+1                                                           IOCSWR13
      IF (K1 .EQ. JP2) GO TO 3                                          IOCSWR14
    1 CONTINUE                                                          IOCSWR15
      K=KMAX                                                            IOCSWR16
      GO TO 3                                                           IOCSWR17
    2 LINE(1)=KFILL                                                     IOCSWR18
      K=1                                                               IOCSWR19
    3 GO TO (10,20,30),JHOW                                             IOCSWR20
   10 IF (K .GE. LNLNG) GO TO 12                                        IOCSWR21
      I=K+1                                                             IOCSWR22
      DO 11 K=I,LNLNG                                                   IOCSWR23
   11 LINE(K)=KFILL                                                     IOCSWR24
   12 WRITE (JDEV,100) (LINE(K),K=1,LNLNG)                              IOCSWR25
      RETURN                                                            IOCSWR26
   20 WRITE (JDEV,101) (LINE(I),I=1,K)                                  IOCSWR27
      RETURN                                                            IOCSWR28
   30 WRITE (JDEV,102) (LINE(I),I=1,K)                                  IOCSWR29
      JCHAN(KCH+2)=2                                                    IOCSWR30
      RETURN                                                            IOCSWR31
  100 FORMAT (80A1)                                                     IOCSWR32
  101 FORMAT (1H ,136A1)                                                IOCSWR33
  102 FORMAT (1H1,136A1)                                                IOCSWR34
      END                                                               IOCSWR35
```

```
      SUBROUTINE IOPEN(JOP,KCH,JBA,JP1,JP2,JRES)                        IOCSON01
      DIMENSION JBA(30000),JCHAN(36)                                    IOCSON02
      COMMON MAXCH,JCHAN                                                IOCSON03
      JDEV=JCHAN(KCH+5)                                                 IOCSON04
      REWIND JDEV                                                       IOCSON05
      JRES=0                                                            IOCSON06
      RETURN                                                            IOCSON07
      END                                                               IOCSON08


      SUBROUTINE ICNTL(KCH,JBA,JP1,JP2,JRES)                            IOCSCN01
      DIMENSION JBA(30000),JCHAN(36),JTB(80)                            IOCSCN02
      COMMON MAXCH,JCHAN                                                IOCSCN03
      IF (JCHAN(KCH+4)+1 .NE. JCHAN(KCH+6)) GO TO 3                     IOCSCN04
      JDEV=JCHAN(KCH+5)                                                 IOCSCN05
      IF (JCHAN(KCH+4) .EQ. 2) GO TO 2                                  IOCSCN06
    1 READ (JDEV,100,END=3) JTB                                         IOCSCN07
      GO TO 1                                                           IOCSCN08
    2 ENDFILE JDEV                                                      IOCSCN09
    3 JCHAN(KCH+6)=1                                                    IOCSCN10
      JRES=0                                                            IOCSCN11
      RETURN                                                            IOCSCN12
  100 FORMAT (80A1)                                                     IOCSCN13
      END                                                               IOCSCN14


      FUNCTION ICVCI(JCHAR)                                             IOCSCI01
      DIMENSION JCTCI(64)                                               IOCSCI02
      INTEGER JCTCI                                                     IOCSCI03
      DATA JCTCI                                                        IOCSCI04
     1/ 0,1,2,3,4,5,6,7,8,9,37,38,39,40,41,42,                         IOCSCI05
     2 43,10,11,12,13,14,15,16,17,18,44,45,46,47,48,49,                 IOCSCI06
     3 50,51,19,20,21,22,23,24,25,26,52,53,54,55,56,57,                 IOCSCI07
     4 27,28,29,30,31,32,33,34,35,36,58,59,60,61,62,63/                 IOCSCI08
      IF (JCHAR) 1,2,3                                                  IOCSCI09
    1 J1=JCHAR+32704                                                    IOCSCI10
      IF (J1) 4,4,5                                                     IOCSCI11
    4 ICVCI=J1+64                                                       IOCSCI12
      RETURN                                                            IOCSCI13
    5 J1=J1/256                                                         IOCSCI14
      J2=J1-63                                                          IOCSCI15
      IF (J2) 4,4,6                                                     IOCSCI16
    6 ICVCI=JCTCI(J2)                                                   IOCSCI17
      RETURN                                                            IOCSCI18
    2 ICVCI=-1                                                          IOCSCI19
      RETURN                                                            IOCSCI20
    3 J2=JCHAR/256-63                                                   IOCSCI21
      IF (J2) 2,2,6                                                     IOCSCI22
      END                                                               IOCSCI23
```

```
      FUNCTION ICVIC(INT)                                                    IOCSIC01
      DIMENSION JCTIC(63)                                                    IOCSIC02
      INTEGER JCTIC                                                          IOCSIC03
      DATA JCTIC                                                             IOCSIC04
     1/-16064,-15808,-15552,-15296,-15040,-14784,-14528,-14272,-14016,       IOCSIC05
     2 -11968,-11712,-11452,-11200,-10944,-10688,-10432,-10176, -9920,       IOCSIC06
     3  -7616, -7360, -7104, -6848, -6592, -6336, -6080, -5824, -4032,       IOCSIC07
     4  -3776, -3520, -3264, -3008, -2752, -2496, -2240, -1984, -1728,       IOCSIC08
     5  19008, 19264, 19520, 19776, 20032, 20288, 20544, 23104, 23360,       IOCSIC09
     6  23616, 23872, 24128, 24384, 24640, 24896, 27200, 27456, 27712,       IOCSIC10
     7  27968, 28224, 28480, 31296, 31552, 31808, 32064, 32320, 32576/       IOCSIC11
      IF (INT) 1,2,3                                                         IOCSIC12
    1 ICVIC=32808                                                            IOCSIC13
      RETURN                                                                 IOCSIC14
    2 ICVIC=16448                                                            IOCSIC15
      RETURN                                                                 IOCSIC16
    3 IF (INT-63) 4,4,5                                                      IOCSIC17
    4 ICVIC=JCTIC(INT)                                                       IOCSIC18
      RETURN                                                                 IOCSIC19
    5 ICVIC=(INT-64)*256-32704                                               IOCSIC20
      RETURN                                                                 IOCSIC21
      END                                                                    IOCSIC22
```

IOCP

```
      SUBROUTINE PROGR                                                  IOCP0001
      DIMENSION LIST(160)                                               IOCP0002
      IF (IOOP(-1,1,LIST,1,I) .NE. 0) STOP 11                           IOCP0003
      IF (IOOP(1,3,LIST,1,I) .NE. 0) STOP 31                            IOCP0004
      LIST(46)=LIST(46)+1                                               IOCP0005
      IF (IOOP(1,4,LIST,1,I) .NE. 0) STOP 41                            IOCP0006
      IF (IOOP(1,4,LIST,1,1) .NE. 0) STOP 42                            IOCP0007
      IF (IOOP(-1,1,LIST,1,I) .NE. 0) STOP 12                           IOCP0008
      IF (IOOP(1,4,LIST,1,I) .NE. 0) STOP 43                            IOCP0009
      LIST(1)=LIST(1)+1                                                 IOCP0010
      IF (IOOP(1,4,LIST,1,3) .NE. 0) STOP 44                            IOCP0011
      IF (IOOP(-1,1,LIST,1,I) .NE. 0) STOP 13                           IOCP0012
      IF (IOOP(-1,1,LIST,I,J) .NE. 0) STOP 14                           IOCP0013
      IF (IOOP(1,4,LIST,1,I) .NE. 0) STOP 45                            IOCP0014
      IF (IOOP(1,4,LIST(I),1,J-I+1) .NE. 0) STOP 46                     IOCP0015
      IF (IOOP(-1,1,LIST,I,J) .NE. 1) STOP 15                           IOCP0016
      RETURN                                                            IOCP0017
      END                                                               IOCP0018
```

IOCT

```
MODULE PROGRAMMING SYSTEM I/O TEST - CHANNEL 3 OUTPUT                    IOCT0001
1)  IF THIS APPEARS TWICE, THE CHANNEL IS NOT BEING CLEARED PROPERLY.    IOCT0002
3)  IF THIS LINE IS OK, THEN TWO BUFFER AREAS DO NOT INTERACT.           IOCT0003
4)  IF THIS LINE IS OK, THEN THE BASE OF THE ARRAY CAN BE OFFSET.        IOCT0004
```

IWCH

```
      SUBROUTINE IWRCH(JCHAR,JBUFF,JINDX,JFLAG,JLENG)                    IWCH0001
      DIMENSION JBUFF(81)                                                IWCH0002
      IF (JCHAR .LT. 0) GO TO 1                                          IWCH0003
      IF (80 .LT. JINDX) GO TO 1                                         IWCH0004
      JBUFF(JINDX)=JCHAR                                                 IWCH0005
      JINDX=JINDX+1                                                      IWCH0006
      JFLAG=0                                                            IWCH0007
      RETURN                                                             IWCH0008
    1 JBUFF(JINDX)=-1                                                    IWCH0009
      JLENG=JINDX                                                        IWCH0010
      JINDX=1                                                            IWCH0011
      JFLAG=1                                                            IWCH0012
      RETURN                                                             IWCH0013
      END                                                                IWCH0014
```

SIMC

```
      SUBROUTINE PROGR                                                  SIMC0001
      DIMENSION LIST(12000)                                             SIMC0002
      KMAX=12000-80                                                     SIMC0003
      IF (IOOP(-1,1,LIST,1,I) .NE. 0) STOP 10                           SIMC0004
      LIST(6)=100                                                       SIMC0005
      K=18                                                              SIMC0006
    1 LIST(K)=-1                                                        SIMC0007
      IF (IOOP(-1,1,LIST(2),K,I) .NE. 0) STOP 11                        SIMC0008
      IF (I .GE. KMAX) STOP 20                                          SIMC0009
      LIST(I+1)=LIST(1)                                                 SIMC0010
      I=K                                                               SIMC0011
    2 I=I+1                                                             SIMC0012
      IF (LIST(I)-LIST(1)) 2,13,2                                       SIMC0013
   10 I=I+1                                                             SIMC0014
      IF (LIST(I) .EQ. LIST(3)) GO TO 12                                SIMC0015
   11 IF (LIST(I) .NE. LIST(4)) GO TO 10                                SIMC0016
      LIST(I)=-2                                                        SIMC0017
      LIST(I+1)=LIST(I+1)-LIST(5)+7                                     SIMC0018
      I=I+2                                                             SIMC0019
      LIST(I)=LIST(I)-LIST(5)                                           SIMC0020
      IF (LIST(I-1) .NE. 7) GO TO 10                                    SIMC0021
      IF (LIST(K) .LT. LIST(I)) LIST(K)=LIST(I)                         SIMC0022
      GO TO 10                                                          SIMC0023
   12 LIST(I)=-1                                                        SIMC0024
                                                                        SIMC0025
   13 I=I+1                                                             SIMC0026
      IF (IOOP(-1,1,LIST,I,J) .NE. 0) STOP 12                           SIMC0027
      IF (J .GE. KMAX) STOP 21                                          SIMC0028
      LIST(J)=LIST(3)                                                   SIMC0029
      IF (LIST(I) .NE. LIST(3)) GO TO 11                                SIMC0030
      LIST(K-1)=I                                                       SIMC0031
      K=I+1                                                             SIMC0032
      IF (K .GE. J) GO TO 1                                             SIMC0033
      IF (LIST(K) .NE. LIST(3)) GO TO 1                                 SIMC0034
   20 IF (IOOP(-1,1,LIST,I,N) .NE. 0) RETURN                            SIMC0035
      LIST(N)=LIST(1)                                                   SIMC0036
      M=17                                                              SIMC0037
   30 L=8                                                               SIMC0038
      J=M+1                                                             SIMC0039
      DO 33 K=I,N                                                       SIMC0040
      J=J+1                                                             SIMC0041
      IF (LIST(J) .EQ. LIST(2)) GO TO 32                                SIMC0042
      IF (LIST(J) .NE. LIST(K)) GO TO 31                                SIMC0043
      IF (LIST(J)-LIST(1)) 33,40,33                                     SIMC0044
   31 M=LIST(M)                                                         SIMC0045
      IF (M .LT. I) GO TO 30                                            SIMC0046
      IF (IOOP(1,3,LIST,I,N) .NE. 0) STOP 30                            SIMC0047
      GO TO 20                                                          SIMC0048
   32 IF (LIST(K) .EQ. LIST(1)) GO TO 31                                SIMC0049
      LIST(L)=LIST(K)                                                   SIMC0050
      L=L+1
```

```
33 CONTINUE                                                          SIMC0051
41 LIST(K)=LIST(J)                                                   SIMC0052
42 K=K+1                                                             SIMC0053
43 J=J+1                                                             SIMC0054
   IF (LIST(M) .EQ. J) GO TO 48                                      SIMC0055
   IF (LIST(J)+1) 44,47,41                                           SIMC0056
44 L=LIST(J+1)                                                       SIMC0057
   J=J+2                                                             SIMC0058
   IF (L .EQ. 7) GO TO 45                                            SIMC0059
   IF (LIST(J) .NE. 0) GO TO 46                                      SIMC0060
   LIST(K)=LIST(L)                                                   SIMC0061
   GO TO 42                                                          SIMC0062
45 LIST(7)=LIST(J)+LIST(6)                                           SIMC0063
46 LIST(K)=LIST(L)/100                                               SIMC0064
   N=LIST(L)/10                                                      SIMC0065
   LIST(K+1)=N-LIST(K)*10+LIST(5)                                    SIMC0066
   LIST(K+2)=LIST(L)-N*10+LIST(5)                                    SIMC0067
   LIST(K)=LIST(K)+LIST(5)                                           SIMC0068
   K=K+3                                                             SIMC0069
   GO TO 43                                                          SIMC0070
47 IF (IOOP(1,3,LIST,I,K) .NE. 0) STOP 31                            SIMC0071
40 K=I                                                               SIMC0072
   GO TO 43                                                          SIMC0073
48 LIST(6)=LIST(M+1)+LIST(6)+1                                       SIMC0074
   GO TO 20                                                          SIMC0075
   END                                                               SIMC0076
```

SIMT

```
.'@'0                                                                    SIMT0001
END PROGRAM.                                                             SIMT0002
C  END OF SIMCMP TEST. THIS LINE CONTAINS TYPE 0 ELEMENTS ONLY@           SIMT0003
      RETURN@                                                            SIMT0004
      END@                                                               SIMT0005
@                                                                        SIMT0006
TYPE 2 '.                                                                SIMT0007
      TESTC='10@                                                         SIMT0008
      TYPE1='11@                                                         SIMT0009
@                                                                        SIMT0010
TEST PARAMETER 0.                                                        SIMT0011
      PRINT '00, GEN1@                                                   SIMT0012
      PRINT '01, GEN2@                                                   SIMT0013
      PRINT '02, GEN3@                                                   SIMT0014
      PRINT '03, GEN4@                                                   SIMT0015
      PRINT '04, GEN5@                                                   SIMT0016
      PRINT '05, GEN6@                                                   SIMT0017
      PRINT '06, GEN7@                                                   SIMT0018
      PRINT '07, GEN8@                                                   SIMT0019
      PRINT '08, GEN9@                                                   SIMT0020
      PRINT '09, GEN10@                                                  SIMT0021
  '00 FORMAT(6H GEN1=,F6.4)@                                             SIMT0022
  '01 FORMAT(6H GEN2=,F6.4)@                                             SIMT0023
  '02 FORMAT(6H GEN3=,F6.4)@                                             SIMT0024
  '03 FORMAT(6H GEN4=,F6.4)@                                             SIMT0025
  '04 FORMAT(6H GEN5=,F6.4)@                                             SIMT0026
  '05 FORMAT(6H GEN6=,F6.4)@                                             SIMT0027
  '06 FORMAT(6H GEN7=,F6.4)@                                             SIMT0028
  '07 FORMAT(6H GEN8=,F6.4)@                                             SIMT0029
  '08 FORMAT(6H GEN9=,F6.4)@                                             SIMT0030
  '09 FORMAT(7H GEN10=,F6.4)@                                            SIMT0031
@                                                                        SIMT0032
NULL MACRO.                                                              SIMT0033
@                                                                        SIMT0034
NINE PARAMETERS '''''''''.                                               SIMT0035
      NINE='10+'20+'30+'40+'50+'60+'70+'80+'90@                          SIMT0036
@                                                                        SIMT0037
CODE BODY LINE WITH NO TERMINATOR.                                       SIMT0038
C  THIS CODE BODY LINE HAS NO TERMINATOR. IT SHOULD HAVE 80 CHARACTERS.  SIMT0039
@@                                                                       SIMT0040
      SUBROUTINE PROGR                                                   SIMT0041
C  THIS TEST DATA CHECKS OUT THE WORKINGS OF SIMCMP.  THE FIRST THREE    SIMT0042
C  LINES WILL NOT BE MATCHED, AND SHOULD BE OUTPUT IN THEIR ENTIRETY.    SIMT0043
TEST PARAMETER 0.                                                        SIMT0044
NULL MACRO.                                                              SIMT0045
NINE PARAMETERS 123456789.                                               SIMT0046
TYPE 2 A.                                                                SIMT0047
CODE BODY LINE WITH NO TERMINATOR.                                       SIMT0048
TYPE 2 0.                                                                SIMT0049
TEST PARAMETER 0.                                                        SIMT0050
END PROGRAM.                                                             SIMT0051
```

FLBM

```
.'@'0                                                                  FLBM0001
FLG ' = ',                                                             FLBM0002
      JF'10=JF'20@                                                     FLBM0003
@                                                                      FLBM0004
''' ' = ' ' ',                                                         FLBM0005
      J'10'40=J'10'50'60J'10'70@                                       FLBM0006
@                                                                      FLBM0007
''' ' = ''' ',                                                         FLBM0008
      J'10'40=J'50'80@                                                 FLBM0009
@                                                                      FLBM0010
GET ' = ',                                                             FLBM0011
      JF'10=L(JP'20)@                                                  FLBM0012
      JV'10=L(JP'20+1)@                                                FLBM0013
      JP'10=L(JP'20+2)@                                                FLBM0014
@                                                                      FLBM0015
STO ' = ',                                                             FLBM0016
      L(JP'10)=JF'20@                                                  FLBM0017
      L(JP'10+1)=JV'20@                                                FLBM0018
      L(JP'10+2)=JP'20@                                                FLBM0019
@                                                                      FLBM0020
TO '' IF ''' ' = ',                                                    FLBM0021
      IF (J'30'60,E@,J'30'70) GO TO '10'20@                            FLBM0022
@                                                                      FLBM0023
TO '' IF ''' ' '' ',                                                   FLBM0024
      IF (J'30'60,'70'80,J'30'90) GO TO '10'20@                        FLBM0025
@                                                                      FLBM0026
TO '' BY ',                                                            FLBM0027
      JP'30='00-99@                                                    FLBM0028
      GO TO '10'20@                                                    FLBM0029
  '00 CONTINUE@                                                        FLBM0030
@                                                                      FLBM0031
RETURN BY ',                                                           FLBM0032
C       USE THE FOLLOWING CARD WHEN TRANSLATING FLT1                   FLBM0033
      GO TO (100), JP'10@                                              FLBM0034
C       USE THE FOLLOWING CARD WHEN TRANSLATING FLT2                   FLBM0035
      GO TO (100,101,102,103), JP'10@                                  FLBM0036
C       USE THE FOLLOWING TWO CARDS WHEN TRANSLATING STAGE2            FLBM0037
      GO TO (100,101,102,103,104,105,106,107,108,109,110,111,112,@     FLBM0038
     1113,114,115,116,117), JP'10@                                     FLBM0039
@                                                                      FLBM0040
TO '',                                                                 FLBM0041
      GO TO '10'20@                                                    FLBM0042
@                                                                      FLBM0043
STOP,                                                                  FLBM0044
      RETURN@                                                          FLBM0045
@                                                                      FLBM0046
VAL ' = CHAR,                                                          FLBM0047
      JV'10=LB(LBR)@                                                   FLBM0048
      LBR=LBR+1@                                                       FLBM0049
@                                                                      FLBM0050
```

```
CHAR = VAL '.                                                          FLBM0051
      CALL IWRCH(JV'10,LB,LBW,JF'10,LBL)@                              FLBM0052
@                                                                      FLBM0053
READ NEXT '.                                                           FLBM0054
      JF'10=IOOP(-1,JV'10,LB,1,LBL)@                                   FLBM0055
      LB(LBL)=-1@                                                      FLBM0056
      LBR=1@                                                           FLBM0057
@                                                                      FLBM0058
WRITE NEXT '.                                                          FLBM0059
      JF'10=IOOP(1,JV'10,LB,1,LBL)@                                    FLBM0060
      LBW=1@                                                           FLBM0061
@                                                                      FLBM0062
REWIND '.                                                              FLBM0063
      JF'10=IOOP(0,JV'10,LB,1,1)@                                      FLBM0064
      JF'10=0@                                                         FLBM0065
@                                                                      FLBM0066
LOC ''.                                                                FLBM0067
   '10'20 CONTINUE@                                                    FLBM0068
@                                                                      FLBM0069
MESSAGE ''''; TO '.                                                    FLBM0070
      MB(11)='11@                                                      FLBM0071
      MB(12)='21@                                                      FLBM0072
      MB(13)='31@                                                      FLBM0073
      MB(14)='41@                                                      FLBM0074
      JF'50=IOOP(1,JV'50,MB,1,21)@                                     FLBM0075
@                                                                      FLBM0076
END PROGRAM.                                                           FLBM0077
      END@                                                             FLBM0078
@                                                                      FLBM0079
MESSAGE ''''''''.                                                      FLBM0080
      DO 1000 J=1,9@                                                   FLBM0081
 1000 MB(J)='11@                                                       FLBM0082
      MB(10)='21@                                                      FLBM0083
      MB(15)='21@                                                      FLBM0084
      MB(16)='31@                                                      FLBM0085
      MB(17)='41@                                                      FLBM0086
      MB(18)='51@                                                      FLBM0087
      MB(19)='61@                                                      FLBM0088
      MB(20)='71@                                                      FLBM0089
@@                                                                     FLBM0090
      SUBROUTINE PROGR                                                 FLBM0091
      DIMENSION LB(81),MB(20),L(12000)                                 FLBM0092
      COMMON MAXCH,JCHAN(24)                                           FLBM0093
      JP9=12000                                                        FLBM0094
      JF0=0                                                            FLBM0095
      JF1=1                                                            FLBM0096
      JF2=2                                                            FLBM0097
      JF3=3                                                            FLBM0098
      JV0=0                                                            FLBM0099
      JV1=1                                                            FLBM0100
      JV2=2                                                            FLBM0101
      JV3=3                                                            FLBM0102
      JV4=4                                                            FLBM0103
      JV5=5                                                            FLBM0104
      JV6=6                                                            FLBM0105
```

```
      JV7=7                                                          FLBM0106
      JV8=8                                                          FLBM0107
      JV9=9                                                          FLBM0108
      JP0=0                                                          FLBM0109
      JP1=1                                                          FLBM0110
      JP2=2                                                          FLBM0111
      JP3=3                                                          FLBM0112
      JP5=10                                                         FLBM0113
      JP7=3                                                          FLBM0114
      JP8=1                                                          FLBM0115
      LBL=1                                                          FLBM0116
      LBR=1                                                          FLBM0117
      LBW=1                                                          FLBM0118
      JP9=JP8+(JP9/JP7-1)*JP7                                        FLBM0119
MESSAGE * ERROR.                                                     FLBM0120
```

FLT1

```
VAL B = 1 + 0.                                                    FLT10001
READ NEXT B.                                                      FLT10002
VAL W = 4 + 0.                                                    FLT10003
WRITE NEXT W.                                                     FLT10004
VAL B = 1 + 0.                                                    FLT10005
READ NEXT B.                                                      FLT10006
VAL W = 4 + 0.                                                    FLT10007
WRITE NEXT W.                                                     FLT10008
VAL B = 1 + 0.                                                    FLT10009
READ NEXT B.                                                      FLT10010
TO 01.                                                            FLT10011
VAL W = 4 + 0.                                                    FLT10012
WRITE NEXT W.                                                     FLT10013
LOC 01.                                                           FLT10014
VAL B = 1 + 0.                                                    FLT10015
READ NEXT B.                                                      FLT10016
TO 02 IF FLG 1 = 1.                                               FLT10017
VAL W = 4 + 0.                                                    FLT10018
WRITE NEXT W.                                                     FLT10019
LOC 02.                                                           FLT10020
VAL B = 1 + 0.                                                    FLT10021
READ NEXT B.                                                      FLT10022
TO 03 IF FLG 1 = 2.                                               FLT10023
TO 04.                                                            FLT10024
LOC 03.                                                           FLT10025
VAL W = 4 + 0.                                                    FLT10026
WRITE NEXT W.                                                     FLT10027
LOC 04.                                                           FLT10028
VAL B = 1 + 0.                                                    FLT10029
READ NEXT B.                                                      FLT10030
TO 05 IF FLG 1 NE 2.                                              FLT10031
VAL W = 4 + 0.                                                    FLT10032
WRITE NEXT W.                                                     FLT10033
LOC 05.                                                           FLT10034
VAL B = 1 + 0.                                                    FLT10035
READ NEXT B.                                                      FLT10036
TO 06 IF FLG 1 NE 1.                                              FLT10037
TO 07.                                                            FLT10038
LOC 06.                                                           FLT10039
VAL W = 4 + 0.                                                    FLT10040
WRITE NEXT W.                                                     FLT10041
LOC 07.                                                           FLT10042
VAL B = 1 + 0.                                                    FLT10043
READ NEXT B.                                                      FLT10044
TO 08 IF VAL 1 = 1.                                               FLT10045
VAL W = 4 + 0.                                                    FLT10046
WRITE NEXT W.                                                     FLT10047
LOC 08.                                                           FLT10048
VAL B = 1 + 0.                                                    FLT10049
READ NEXT B.                                                      FLT10050
```

```
TO 09 IF VAL 1 = 2.                                                    FLT10051
TO 10.                                                                 FLT10052
LOC 09.                                                                FLT10053
VAL W = 4 + 0.                                                         FLT10054
WRITE NEXT W.                                                          FLT10055
LOC 10.                                                                FLT10056
VAL B = 1 + 0.                                                         FLT10057
READ NEXT B.                                                           FLT10058
TO 11 IF VAL 1 NE 2.                                                   FLT10059
VAL W = 4 + 0.                                                         FLT10060
WRITE NEXT W.                                                          FLT10061
LOC 11.                                                                FLT10062
VAL B = 1 + 0.                                                         FLT10063
READ NEXT B.                                                           FLT10064
TO 12 IF VAL 1 NE 1.                                                   FLT10065
TO 13.                                                                 FLT10066
LOC 12.                                                                FLT10067
VAL W = 4 + 0.                                                         FLT10068
WRITE NEXT W.                                                          FLT10069
LOC 13.                                                                FLT10070
VAL B = 1 + 0.                                                         FLT10071
READ NEXT B.                                                           FLT10072
TO 14 IF PTR 1 = 1.                                                    FLT10073
VAL W = 4 + 0.                                                         FLT10074
WRITE NEXT W.                                                          FLT10075
LOC 14.                                                                FLT10076
VAL B = 1 + 0.                                                         FLT10077
READ NEXT B.                                                           FLT10078
TO 15 IF PTR 1 = 2.                                                    FLT10079
TO 16.                                                                 FLT10080
LOC 15.                                                                FLT10081
VAL W = 4 + 0.                                                         FLT10082
WRITE NEXT W.                                                          FLT10083
LOC 16.                                                                FLT10084
VAL B = 1 + 0.                                                         FLT10085
READ NEXT B.                                                           FLT10086
TO 17 IF PTR 1 NE 2.                                                   FLT10087
VAL W = 4 + 0.                                                         FLT10088
WRITE NEXT W.                                                          FLT10089
LOC 17.                                                                FLT10090
VAL B = 1 + 0.                                                         FLT10091
READ NEXT B.                                                           FLT10092
TO 18 IF PTR 1 NE 1.                                                   FLT10093
TO 19.                                                                 FLT10094
LOC 18.                                                                FLT10095
VAL W = 4 + 0.                                                         FLT10096
WRITE NEXT W.                                                          FLT10097
LOC 19.                                                                FLT10098
VAL B = 1 + 0.                                                         FLT10099
READ NEXT B.                                                           FLT10100
TO 20 IF PTR 3 GE 1.                                                   FLT10101
VAL W = 4 + 0.                                                         FLT10102
WRITE NEXT W.                                                          FLT10103
LOC 20.                                                                FLT10104
VAL B = 1 + 0.                                                         FLT10105
```

```
READ NEXT B.                                                    FLT10106
TO 21 IF PTR 1 GE 1.                                            FLT10107
VAL W = 4 + 0.                                                  FLT10108
WRITE NEXT W.                                                   FLT10109
LOC 21.                                                         FLT10110
VAL B = 1 + 0.                                                  FLT10111
READ NEXT B.                                                    FLT10112
TO 22 IF PTR 1 GE 2.                                            FLT10113
TO 23.                                                          FLT10114
LOC 22.                                                         FLT10115
VAL W = 4 + 0.                                                  FLT10116
WRITE NEXT W.                                                   FLT10117
LOC 23.                                                         FLT10118
VAL B = 1 + 0.                                                  FLT10119
READ NEXT B.                                                    FLT10120
FLG A = 1.                                                      FLT10121
TO 24 IF FLG A = 1.                                             FLT10122
VAL W = 4 + 0.                                                  FLT10123
WRITE NEXT W.                                                   FLT10124
LOC 24.                                                         FLT10125
VAL B = 1 + 0.                                                  FLT10126
READ NEXT B.                                                    FLT10127
VAL A = PTR 3.                                                  FLT10128
TO 25 IF VAL A = 3.                                             FLT10129
VAL W = 4 + 0.                                                  FLT10130
WRITE NEXT W.                                                   FLT10131
LOC 25.                                                         FLT10132
VAL B = 1 + 0.                                                  FLT10133
READ NEXT B.                                                    FLT10134
PTR A = VAL 3.                                                  FLT10135
TO 26 IF PTR A = 3.                                             FLT10136
VAL W = 4 + 0.                                                  FLT10137
WRITE NEXT W.                                                   FLT10138
LOC 26.                                                         FLT10139
VAL B = 1 + 0.                                                  FLT10140
READ NEXT B.                                                    FLT10141
FLG C = 0.                                                      FLT10142
PTR C = VAL 0.                                                  FLT10143
VAL C = PTR 2.                                                  FLT10144
TO 27 IF PTR C = 0.                                             FLT10145
VAL W = 4 + 0.                                                  FLT10146
WRITE NEXT W.                                                   FLT10147
LOC 27.                                                         FLT10148
VAL B = 1 + 0.                                                  FLT10149
READ NEXT B.                                                    FLT10150
TO 28 IF FLG C = 0.                                             FLT10151
VAL W = 4 + 0.                                                  FLT10152
WRITE NEXT W.                                                   FLT10153
LOC 28.                                                         FLT10154
VAL B = 1 + 0.                                                  FLT10155
READ NEXT B.                                                    FLT10156
VAL C = PTR 0.                                                  FLT10157
FLG C = 0.                                                      FLT10158
PTR C = VAL 0.                                                  FLT10159
TO 29 IF VAL C = 0.                                             FLT10160
```

```
VAL W = 4 + 0.                                                  FLT10161
WRITE NEXT W.                                                   FLT10162
LOC 29.                                                         FLT10163
VAL B = 1 + 0.                                                  FLT10164
READ NEXT B.                                                    FLT10165
TO 30 IF FLG C = 0.                                             FLT10166
VAL W = 4 + 0.                                                  FLT10167
WRITE NEXT W.                                                   FLT10168
LOC 30.                                                         FLT10169
VAL B = 1 + 0.                                                  FLT10170
READ NEXT B.                                                    FLT10171
VAL D = PTR 0.                                                  FLT10172
PTR D = VAL 0.                                                  FLT10173
FLG D = 3.                                                      FLT10174
TO 31 IF PTR D = 0.                                             FLT10175
VAL W = 4 + 0.                                                  FLT10176
WRITE NEXT W.                                                   FLT10177
LOC 31.                                                         FLT10178
VAL B = 1 + 0.                                                  FLT10179
READ NEXT B.                                                    FLT10180
TO 32 IF VAL D = 0.                                             FLT10181
VAL W = 4 + 0.                                                  FLT10182
WRITE NEXT W.                                                   FLT10183
LOC 32.                                                         FLT10184
VAL B = 1 + 0.                                                  FLT10185
READ NEXT B.                                                    FLT10186
VAL W = 4 + 0.                                                  FLT10187
WRITE NEXT W.                                                   FLT10188
VAL B = 1 + 0.                                                  FLT10189
READ NEXT B.                                                    FLT10190
VAL W = 4 + 0.                                                  FLT10191
WRITE NEXT W.                                                   FLT10192
VAL B = 1 + 0.                                                  FLT10193
READ NEXT B.                                                    FLT10194
FLG E = 0.                                                      FLT10195
PTR E = VAL 0.                                                  FLT10196
VAL E = 1 + 3.                                                  FLT10197
TO 33 IF VAL E = 4.                                             FLT10198
VAL W = 4 + 0.                                                  FLT10199
WRITE NEXT W.                                                   FLT10200
LOC 33.                                                         FLT10201
VAL B = 1 + 0.                                                  FLT10202
READ NEXT B.                                                    FLT10203
TO 34 IF FLG E = 0.                                             FLT10204
VAL W = 4 + 0.                                                  FLT10205
WRITE NEXT W.                                                   FLT10206
LOC 34.                                                         FLT10207
VAL B = 1 + 0.                                                  FLT10208
READ NEXT B.                                                    FLT10209
TO 35 IF PTR E = 0.                                             FLT10210
VAL W = 4 + 0.                                                  FLT10211
WRITE NEXT W.                                                   FLT10212
LOC 35.                                                         FLT10213
VAL B = 1 + 0.                                                  FLT10214
READ NEXT B.                                                    FLT10215
```

```
VAL W = 4 + 0.                                                    FLT10216
WRITE NEXT W.                                                     FLT10217
VAL B = 1 + 0.                                                    FLT10218
READ NEXT B.                                                      FLT10219
VAL C = CHAR.                                                     FLT10220
VAL D = CHAR.                                                     FLT10221
VAL E = CHAR.                                                     FLT10222
VAL E = CHAR.                                                     FLT10223
VAL F = CHAR.                                                     FLT10224
CHAR = VAL C.                                                     FLT10225
CHAR = VAL D.                                                     FLT10226
CHAR = VAL E.                                                     FLT10227
CHAR = VAL F.                                                     FLT10228
VAL G = F + 0.                                                    FLT10229
CHAR = VAL G.                                                     FLT10230
CHAR = VAL E.                                                     FLT10231
VAL G = F + 1.                                                    FLT10232
CHAR = VAL G.                                                     FLT10233
CHAR = VAL E.                                                     FLT10234
VAL G = F + 2.                                                    FLT10235
CHAR = VAL G.                                                     FLT10236
CHAR = VAL E.                                                     FLT10237
VAL G = F + 3.                                                    FLT10238
CHAR = VAL G.                                                     FLT10239
CHAR = VAL E.                                                     FLT10240
VAL G = F + 4.                                                    FLT10241
CHAR = VAL G.                                                     FLT10242
CHAR = VAL E.                                                     FLT10243
VAL G = F + 5.                                                    FLT10244
CHAR = VAL G.                                                     FLT10245
CHAR = VAL E.                                                     FLT10246
VAL G = F + 6.                                                    FLT10247
CHAR = VAL G.                                                     FLT10248
CHAR = VAL E.                                                     FLT10249
VAL G = F + 7.                                                    FLT10250
CHAR = VAL G.                                                     FLT10251
CHAR = VAL E.                                                     FLT10252
VAL G = F + 8.                                                    FLT10253
CHAR = VAL G.                                                     FLT10254
CHAR = VAL E.                                                     FLT10255
VAL G = F + 9.                                                    FLT10256
CHAR = VAL G.                                                     FLT10257
VAL W = 4 + 0.                                                    FLT10258
WRITE NEXT W.                                                     FLT10259
VAL B = 1 + 0.                                                    FLT10260
READ NEXT B.                                                      FLT10261
VAL C = CHAR.                                                     FLT10262
VAL D = CHAR.                                                     FLT10263
CHAR = VAL C.                                                     FLT10264
CHAR = VAL D.                                                     FLT10265
CHAR = VAL E.                                                     FLT10266
CHAR = VAL E.                                                     FLT10267
VAL H = PTR 0.                                                    FLT10268
VAL G = F + H.                                                    FLT10269
CHAR = VAL G.                                                     FLT10270
```

```
CHAR = VAL E.                                                    FLT10271
VAL H = PTR 1.                                                   FLT10272
VAL G = F + H.                                                   FLT10273
CHAR = VAL G.                                                    FLT10274
CHAR = VAL E.                                                    FLT10275
VAL H = PTR 2.                                                   FLT10276
VAL G = F + H.                                                   FLT10277
CHAR = VAL G.                                                    FLT10278
CHAR = VAL E.                                                    FLT10279
VAL H = PTR 3.                                                   FLT10280
VAL G = F + H.                                                   FLT10281
CHAR = VAL G.                                                    FLT10282
VAL W = 4 + 0.                                                   FLT10283
WRITE NEXT W.                                                    FLT10284
VAL B = 1 + 0.                                                   FLT10285
READ NEXT B.                                                     FLT10286
FLG F = 0.                                                       FLT10287
PTR F = VAL 0.                                                   FLT10288
VAL F = 3 - 1.                                                   FLT10289
TO 36 IF VAL F = 2.                                              FLT10290
VAL W = 4 + 0.                                                   FLT10291
WRITE NEXT W.                                                    FLT10292
LOC 36.                                                          FLT10293
VAL B = 1 + 0.                                                   FLT10294
READ NEXT B.                                                     FLT10295
TO 37 IF FLG F = 0.                                              FLT10296
VAL W = 4 + 0.                                                   FLT10297
WRITE NEXT W.                                                    FLT10298
LOC 37.                                                          FLT10299
VAL B = 1 + 0.                                                   FLT10300
READ NEXT B.                                                     FLT10301
TO 38 IF PTR F = 0.                                              FLT10302
VAL W = 4 + 0.                                                   FLT10303
WRITE NEXT W.                                                    FLT10304
LOC 38.                                                          FLT10305
VAL B = 1 + 0.                                                   FLT10306
READ NEXT B.                                                     FLT10307
FLG G = 0.                                                       FLT10308
PTR G = VAL 0.                                                   FLT10309
VAL G = 1 - 3.                                                   FLT10310
VAL H = G + 4.                                                   FLT10311
TO 39 IF VAL H = 2.                                              FLT10312
VAL W = 4 + 0.                                                   FLT10313
WRITE NEXT W.                                                    FLT10314
LOC 39.                                                          FLT10315
VAL B = 1 + 0.                                                   FLT10316
READ NEXT B.                                                     FLT10317
TO 40 IF FLG G = 0.                                              FLT10318
VAL W = 4 + 0.                                                   FLT10319
WRITE NEXT W.                                                    FLT10320
LOC 40.                                                          FLT10321
VAL B = 1 + 0.                                                   FLT10322
READ NEXT B.                                                     FLT10323
TO 41 IF PTR G = 0.                                              FLT10324
VAL W = 4 + 0.                                                   FLT10325
```

```
WRITE NEXT W.                                                    FLT10326
LOC 41.                                                          FLT10327
VAL B = 1 + 0.                                                   FLT10328
READ NEXT B.                                                     FLT10329
VAL H = G + 1.                                                   FLT10330
VAL I = 0 - 1.                                                   FLT10331
TO 42 IF VAL H = I.                                              FLT10332
VAL W = 4 + 0.                                                   FLT10333
WRITE NEXT W.                                                    FLT10334
LOC 42.                                                          FLT10335
VAL B = 1 + 0.                                                   FLT10336
READ NEXT B.                                                     FLT10337
VAL J = I + I.                                                   FLT10338
TO 43 IF VAL J = G.                                              FLT10339
VAL W = 4 + 0.                                                   FLT10340
WRITE NEXT W.                                                    FLT10341
LOC 43.                                                          FLT10342
VAL B = 1 + 0.                                                   FLT10343
READ NEXT B.                                                     FLT10344
VAL K = H - 1.                                                   FLT10345
TO 44 IF VAL K = G.                                              FLT10346
VAL W = 4 + 0.                                                   FLT10347
WRITE NEXT W.                                                    FLT10348
LOC 44.                                                          FLT10349
VAL B = 1 + 0.                                                   FLT10350
READ NEXT B.                                                     FLT10351
VAL L = 1 - G.                                                   FLT10352
TO 45 IF VAL L = 3.                                              FLT10353
VAL W = 4 + 0.                                                   FLT10354
WRITE NEXT W.                                                    FLT10355
LOC 45.                                                          FLT10356
VAL B = 1 + 0.                                                   FLT10357
READ NEXT B.                                                     FLT10358
FLG M = 0.                                                       FLT10359
VAL M = PTR 0.                                                   FLT10360
PTR M = 1 + 2.                                                   FLT10361
TO 46 IF PTR M = 3.                                              FLT10362
VAL W = 4 + 0.                                                   FLT10363
WRITE NEXT W.                                                    FLT10364
LOC 46.                                                          FLT10365
VAL B = 1 + 0.                                                   FLT10366
READ NEXT B.                                                     FLT10367
TO 47 IF FLG M = 0.                                              FLT10368
VAL W = 4 + 0.                                                   FLT10369
WRITE NEXT W.                                                    FLT10370
LOC 47.                                                          FLT10371
VAL B = 1 + 0.                                                   FLT10372
READ NEXT B.                                                     FLT10373
TO 48 IF VAL M = 0.                                              FLT10374
VAL W = 4 + 0.                                                   FLT10375
WRITE NEXT W.                                                    FLT10376
LOC 48.                                                          FLT10377
VAL B = 1 + 0.                                                   FLT10378
READ NEXT B.                                                     FLT10379
FLG N = 0.                                                       FLT10380
```

```
VAL N = PTR 0.                                                  FLT10381
PTR N = 3 - 2.                                                  FLT10382
TO 49 IF PTR N = 1.                                             FLT10383
VAL W = 4 + 0.                                                  FLT10384
WRITE NEXT W.                                                   FLT10385
LOC 49.                                                         FLT10386
VAL B = 1 + 0.                                                  FLT10387
READ NEXT B.                                                    FLT10388
TO 50 IF FLG N = 0.                                             FLT10389
VAL W = 4 + 0.                                                  FLT10390
WRITE NEXT W.                                                   FLT10391
LOC 50.                                                         FLT10392
VAL B = 1 + 0.                                                  FLT10393
READ NEXT B.                                                    FLT10394
TO 51 IF VAL N = 0.                                             FLT10395
VAL W = 4 + 0.                                                  FLT10396
WRITE NEXT W.                                                   FLT10397
LOC 51.                                                         FLT10398
VAL B = 1 + 0.                                                  FLT10399
READ NEXT B.                                                    FLT10400
PTR O = VAL 5.                                                  FLT10401
PTR P = VAL 4.                                                  FLT10402
PTR P = 0 - P.                                                  FLT10403
FLG O = 0.                                                      FLT10404
VAL O = PTR 0.                                                  FLT10405
PTR O = 2 - O.                                                  FLT10406
PTR P = P + 1.                                                  FLT10407
TO 52 IF PTR O = P.                                             FLT10408
VAL W = 4 + 0.                                                  FLT10409
WRITE NEXT W.                                                   FLT10410
LOC 52.                                                         FLT10411
VAL B = 1 + 0.                                                  FLT10412
READ NEXT B.                                                    FLT10413
TO 53 IF FLG O = 0.                                             FLT10414
VAL W = 4 + 0.                                                  FLT10415
WRITE NEXT W.                                                   FLT10416
LOC 53.                                                         FLT10417
VAL B = 1 + 0.                                                  FLT10418
READ NEXT B.                                                    FLT10419
TO 54 IF VAL O = 0.                                             FLT10420
VAL W = 4 + 0.                                                  FLT10421
WRITE NEXT W.                                                   FLT10422
LOC 54.                                                         FLT10423
VAL B = 1 + 0.                                                  FLT10424
READ NEXT B.                                                    FLT10425
PTR Q = 0 - 1.                                                  FLT10426
PTR R = P + 2.                                                  FLT10427
TO 55 IF PTR R = Q.                                             FLT10428
VAL W = 4 + 0.                                                  FLT10429
WRITE NEXT W.                                                   FLT10430
LOC 55.                                                         FLT10431
VAL B = 1 + 0.                                                  FLT10432
READ NEXT B.                                                    FLT10433
PTR S = R + Q.                                                  FLT10434
PTR S = S + Q.                                                  FLT10435
```

```
TO 56 IF PTR S = P.                                    FLT10436
VAL W = 4 + 0.                                         FLT10437
WRITE NEXT W.                                          FLT10438
LOC 56.                                                FLT10439
VAL B = 1 + 0.                                         FLT10440
READ NEXT B.                                           FLT10441
PTR T = R - 2.                                         FLT10442
TO 57 IF PTR T = P.                                    FLT10443
VAL W = 4 + 0.                                         FLT10444
WRITE NEXT W.                                          FLT10445
LOC 57.                                                FLT10446
VAL B = 1 + 0.                                         FLT10447
READ NEXT B.                                           FLT10448
PTR T = 2 - R.                                         FLT10449
TO 58 IF PTR T = 3.                                    FLT10450
VAL W = 4 + 0.                                         FLT10451
WRITE NEXT W.                                          FLT10452
LOC 58.                                                FLT10453
VAL B = 1 + 0.                                         FLT10454
READ NEXT B.                                           FLT10455
FLG U = 0.                                             FLT10456
PTR U = VAL 0.                                         FLT10457
VAL U = PTR 0.                                         FLT10458
TO 59 IF VAL U = H.                                    FLT10459
VAL W = 4 + 0.                                         FLT10460
WRITE NEXT W.                                          FLT10461
LOC 59.                                                FLT10462
VAL B = 1 + 0.                                         FLT10463
READ NEXT B.                                           FLT10464
TO 60 IF FLG U = 0.                                    FLT10465
VAL W = 4 + 0.                                         FLT10466
WRITE NEXT W.                                          FLT10467
LOC 60.                                                FLT10468
VAL B = 1 + 0.                                         FLT10469
READ NEXT B.                                           FLT10470
TO 61 IF PTR U = 0.                                    FLT10471
VAL W = 4 + 0.                                         FLT10472
WRITE NEXT W.                                          FLT10473
LOC 61.                                                FLT10474
VAL B = 1 + 0.                                         FLT10475
READ NEXT B.                                           FLT10476
FLG V = 0.                                             FLT10477
VAL V = PTR 0.                                         FLT10478
PTR V = VAL I.                                         FLT10479
TO 62 IF PTR V = 0.                                    FLT10480
VAL W = 4 + 0.                                         FLT10481
WRITE NEXT W.                                          FLT10482
LOC 62.                                                FLT10483
VAL B = 1 + 0.                                         FLT10484
READ NEXT B.                                           FLT10485
TO 63 IF FLG V = 0.                                    FLT10486
VAL W = 4 + 0.                                         FLT10487
WRITE NEXT W.                                          FLT10488
LOC 63.                                                FLT10489
VAL B = 1 + 0.                                         FLT10490
```

```
READ NEXT B.                                                    FLT10491
TO 64 IF VAL V = 0.                                             FLT10492
VAL W = 4 + 0.                                                  FLT10493
WRITE NEXT W.                                                   FLT10494
LOC 64.                                                         FLT10495
VAL B = 1 + 0.                                                  FLT10496
READ NEXT B.                                                    FLT10497
VAL X = PTR 0.                                                  FLT10498
FLG X = 0.                                                      FLT10499
PTR D = VAL 9.                                                  FLT10500
PTR X = 3 * 3.                                                  FLT10501
TO 65 IF PTR X = D.                                             FLT10502
VAL W = 4 + 0.                                                  FLT10503
WRITE NEXT W.                                                   FLT10504
LOC 65.                                                         FLT10505
VAL B = 1 + 0.                                                  FLT10506
READ NEXT B.                                                    FLT10507
TO 66 IF FLG X = 0.                                             FLT10508
VAL W = 4 + 0.                                                  FLT10509
WRITE NEXT W.                                                   FLT10510
LOC 66.                                                         FLT10511
VAL B = 1 + 0.                                                  FLT10512
READ NEXT B.                                                    FLT10513
TO 67 IF VAL X = 0.                                             FLT10514
VAL W = 4 + 0.                                                  FLT10515
WRITE NEXT W.                                                   FLT10516
LOC 67.                                                         FLT10517
VAL B = 1 + 0.                                                  FLT10518
READ NEXT B.                                                    FLT10519
PTR X = P * S.                                                  FLT10520
TO 68 IF PTR X = D.                                             FLT10521
VAL W = 4 + 0.                                                  FLT10522
WRITE NEXT W.                                                   FLT10523
LOC 68.                                                         FLT10524
VAL B = 1 + 0.                                                  FLT10525
READ NEXT B.                                                    FLT10526
PTR D = 0 - D.                                                  FLT10527
FLG Y = 0.                                                      FLT10528
VAL Y = PTR 0.                                                  FLT10529
PTR Y = P * 3.                                                  FLT10530
TO 69 IF PTR Y = D.                                             FLT10531
VAL W = 4 + 0.                                                  FLT10532
WRITE NEXT W.                                                   FLT10533
LOC 69.                                                         FLT10534
VAL B = 1 + 0.                                                  FLT10535
READ NEXT B.                                                    FLT10536
TO 70 IF FLG Y = 0.                                             FLT10537
VAL W = 4 + 0.                                                  FLT10538
WRITE NEXT W.                                                   FLT10539
LOC 70.                                                         FLT10540
VAL B = 1 + 0.                                                  FLT10541
READ NEXT B.                                                    FLT10542
TO 71 IF VAL Y = 0.                                             FLT10543
VAL W = 4 + 0.                                                  FLT10544
WRITE NEXT W.                                                   FLT10545
```

```
LOC 71.                                                    FLT10546
VAL B = 1 + 0.                                             FLT10547
READ NEXT B.                                               FLT10548
PTR Y = 3 * P.                                             FLT10549
TO 72 IF PTR Y = D.                                        FLT10550
VAL W = 4 + 0.                                             FLT10551
WRITE NEXT W.                                              FLT10552
LOC 72.                                                    FLT10553
VAL B = 1 + 0.                                             FLT10554
READ NEXT B.                                               FLT10555
PTR D = VAL 9.                                             FLT10556
FLG Z = 0.                                                 FLT10557
VAL Z = PTR 0.                                             FLT10558
PTR Z = D / 3.                                             FLT10559
TO 73 IF PTR Z = 3.                                        FLT10560
VAL W = 4 + 0.                                             FLT10561
WRITE NEXT W.                                              FLT10562
LOC 73.                                                    FLT10563
VAL B = 1 + 0.                                             FLT10564
READ NEXT B.                                               FLT10565
TO 74 IF FLG Z = 0.                                        FLT10566
VAL W = 4 + 0.                                             FLT10567
WRITE NEXT W.                                              FLT10568
LOC 74.                                                    FLT10569
VAL B = 1 + 0.                                             FLT10570
READ NEXT B.                                               FLT10571
TO 75 IF VAL Z = 0.                                        FLT10572
VAL W = 4 + 0.                                             FLT10573
WRITE NEXT W.                                              FLT10574
LOC 75.                                                    FLT10575
VAL B = 1 + 0.                                             FLT10576
READ NEXT B.                                               FLT10577
PTR D = 0 - D.                                             FLT10578
PTR Z = D / S.                                             FLT10579
TO 76 IF PTR Z = 3.                                        FLT10580
VAL W = 4 + 0.                                             FLT10581
WRITE NEXT W.                                              FLT10582
LOC 76.                                                    FLT10583
VAL B = 1 + 0.                                             FLT10584
READ NEXT B.                                               FLT10585
FLG Z = 0.                                                 FLT10586
VAL Z = PTR 0.                                             FLT10587
PTR Z = D / 3.                                             FLT10588
TO 77 IF PTR Z = S.                                        FLT10589
VAL W = 4 + 0.                                             FLT10590
WRITE NEXT W.                                              FLT10591
LOC 77.                                                    FLT10592
VAL B = 1 + 0.                                             FLT10593
READ NEXT B.                                               FLT10594
TO 78 IF FLG Z = 0.                                        FLT10595
VAL W = 4 + 0.                                             FLT10596
WRITE NEXT W.                                              FLT10597
LOC 78.                                                    FLT10598
VAL B = 1 + 0.                                             FLT10599
READ NEXT B.                                               FLT10560
```

```
TO 79 IF VAL Z = 0.                                    FLT10601
VAL W = 4 + 0.                                         FLT10602
WRITE NEXT W.                                          FLT10603
LOC 79.                                                FLT10604
VAL B = 1 + 0.                                         FLT10605
READ NEXT B.                                           FLT10606
PTR D = VAL 9.                                         FLT10607
PTR Z = D / S.                                         FLT10608
TO 80 IF PTR Z = S.                                    FLT10609
VAL W = 4 + 0.                                         FLT10610
WRITE NEXT W.                                          FLT10611
LOC 80.                                                FLT10612
VAL B = 1 + 0.                                         FLT10613
READ NEXT B.                                           FLT10614
PTR E = VAL 5.                                         FLT10615
FLG Z = 0.                                             FLT10616
VAL Z = PTR 0.                                         FLT10617
PTR Z = D / E.                                         FLT10618
TO 81 IF PTR Z = 1.                                    FLT10619
VAL W = 4 + 0.                                         FLT10620
WRITE NEXT W.                                          FLT10621
LOC 81.                                                FLT10622
VAL B = 1 + 0.                                         FLT10623
READ NEXT B.                                           FLT10624
TO 82 IF FLG Z = 0.                                    FLT10625
VAL W = 4 + 0.                                         FLT10626
WRITE NEXT W.                                          FLT10627
LOC 82.                                                FLT10628
VAL B = 1 + 0.                                         FLT10629
READ NEXT B.                                           FLT10630
TO 83 IF VAL Z = 0.                                    FLT10631
VAL W = 4 + 0.                                         FLT10632
WRITE NEXT W.                                          FLT10633
LOC 83.                                                FLT10634
VAL B = 1 + 0.                                         FLT10635
READ NEXT B.                                           FLT10636
PTR E = 0 - E.                                         FLT10637
FLG Z = 0.                                             FLT10638
VAL Z = PTR 0.                                         FLT10639
PTR Z = D / E.                                         FLT10640
TO 84 IF PTR Z = 0.                                    FLT10641
VAL W = 4 + 0.                                         FLT10642
WRITE NEXT W.                                          FLT10643
LOC 84.                                                FLT10644
VAL B = 1 + 0.                                         FLT10645
READ NEXT B.                                           FLT10646
TO 85 IF FLG Z = 0.                                    FLT10647
VAL W = 4 + 0.                                         FLT10648
WRITE NEXT W.                                          FLT10649
LOC 85.                                                FLT10650
VAL B = 1 + 0.                                         FLT10651
READ NEXT B.                                           FLT10652
TO 86 IF VAL Z = 0.                                    FLT10653
VAL W = 4 + 0.                                         FLT10654
WRITE NEXT W.                                          FLT10655
```

```
LOC 86.                                                    FLT10656
VAL B = 1 + 0.                                             FLT10657
READ NEXT B.                                               FLT10658
VAL F = 0 - 6.                                             FLT10659
TO 87 IF VAL F = 6.                                        FLT10660
TO 88.                                                     FLT10661
LOC 87.                                                    FLT10662
VAL W = 4 + 0.                                             FLT10663
WRITE NEXT W.                                              FLT10664
LOC 88.                                                    FLT10665
VAL B = 1 + 0.                                             FLT10666
READ NEXT B.                                               FLT10667
TO 89 IF VAL F NE 8.                                       FLT10668
VAL W = 4 + 0.                                             FLT10669
WRITE NEXT W.                                              FLT10670
LOC 89.                                                    FLT10671
VAL B = 1 + 0.                                             FLT10672
READ NEXT B.                                               FLT10673
TO 90 IF PTR R = 1.                                        FLT10674
TO 91.                                                     FLT10675
LOC 90.                                                    FLT10676
VAL W = 4 + 0.                                             FLT10677
WRITE NEXT W.                                              FLT10678
LOC 91.                                                    FLT10679
VAL B = 1 + 0.                                             FLT10680
READ NEXT B.                                               FLT10681
TO 92 IF PTR S NE 1.                                       FLT10682
VAL W = 4 + 0.                                             FLT10683
WRITE NEXT W.                                              FLT10684
LOC 92.                                                    FLT10685
VAL B = 1 + 0.                                             FLT10686
READ NEXT B.                                               FLT10687
TO 93 IF PTR 2 GE 0.                                       FLT10688
VAL W = 4 + 0.                                             FLT10689
WRITE NEXT W.                                              FLT10690
LOC 93.                                                    FLT10691
VAL B = 1 + 0.                                             FLT10692
READ NEXT B.                                               FLT10693
TO 94 IF PTR S GE 3.                                       FLT10694
TO 95.                                                     FLT10695
LOC 94.                                                    FLT10696
VAL W = 4 + 0.                                             FLT10697
WRITE NEXT W.                                              FLT10698
LOC 95.                                                    FLT10699
VAL B = 1 + 0.                                             FLT10700
READ NEXT B.                                               FLT10701
TO 96 IF PTR R GE S.                                       FLT10702
VAL W = 4 + 0.                                             FLT10703
WRITE NEXT W.                                              FLT10704
LOC 96.                                                    FLT10705
VAL B = 1 + 0.                                             FLT10706
READ NEXT B.                                               FLT10707
TO 97 IF PTR S GE R.                                       FLT10708
TO 98.                                                     FLT10709
LOC 97.                                                    FLT10710
```

```
VAL W = 4 + 0.                                                  FLT10711
WRITE NEXT W.                                                   FLT10712
LOC 98.                                                         FLT10713
VAL B = 1 + 0.                                                  FLT10714
READ NEXT B.                                                    FLT10715
VAL W = 4 + 0.                                                  FLT10716
WRITE NEXT W.                                                   FLT10717
VAL B = 1 + 0.                                                  FLT10718
READ NEXT B.                                                    FLT10719
VAL W = 4 + 0.                                                  FLT10720
WRITE NEXT W.                                                   FLT10721
TO 99 BY I.                                                     FLT10722
VAL W = 4 + 0.                                                  FLT10723
WRITE NEXT W.                                                   FLT10724
STOP.                                                           FLT10725
LOC 99.                                                         FLT10726
VAL B = 1 + 0.                                                  FLT10727
READ NEXT B.                                                    FLT10728
VAL W = 4 + 0.                                                  FLT10729
WRITE NEXT W.                                                   FLT10730
VAL B = 1 + 0.                                                  FLT10731
READ NEXT B.                                                    FLT10732
RETURN BY I.                                                    FLT10733
STOP.                                                           FLT10734
END PROGRAM.                                                    FLT10735
```

FLD1

```
1.  IF FLUB IS CORRECT, OUTPUT CONSISTS ONLY OF NUMBERED LINES.                FLD10001
2.  FIRST TWO LINES RELY ON READ NEXT *, WRITE NEXT * AND VAL * = 1 + 0        FLD10002
    TO ** FAILED.                                                              FLD10003
    TO ** IF FLG * = * FALS ON EQUAL.                                          FLD10004
    TO ** IF FLG * = * TRANSFERS ON UNEQUAL.                                   FLD10005
    TO ** IF FLG * NE * FAILS ON UNEQUAL.                                      FLD10006
    TO ** IF FLG * NE *TRANSFERS ON EQUAL.                                     FLD10007
    TO ** IF VAL * = * FAILS ON EQUAL.                                         FLD10008
    TO ** IF VAL * = * TRANSFERS ON UNEQUAL.                                   FLD10009
    TO ** IF VAL * NE * FAILS ON UNEQUAL.                                      FLD10010
    TO ** IF VAL * NE * TRANSFERS ON EQUAL.                                    FLD10011
    TO ** IF PTR * = * FAILS ON EQUAL.                                         FLD10012
    TO ** IF PTR * = = TRANSFERS ON UNEQUAL.                                   FLD10013
    TO ** IF PTR * NE * FAILS ON UNEQUAL.                                      FLD10014
    TO ** IF PTR * NE * TRANSFERS ON EQUAL.                                    FLD10015
    TO ** IF PTR * GE * FAILS ON GREATER.                                      FLD10016
    TO ** IF PTR * GE * FAILS ON EQUAL.                                        FLD10017
    TO ** IF PTR * GE * TRANSFERS ON LESS.                                     FLD10018
    FLG * = * FAILS.                                                           FLD10019
    VAL * = PTR * FAILS.                                                       FLD10020
    PTR * = VAL * FAILS.                                                       FLD10021
    VAL * = PTR * CHANGED PTR FIELD.                                           FLD10022
    VAL * = PTR * CHANGED FLG FIELD.                                           FLD10023
    PTR * = VAL * CHANGED VAL FIELD.                                           FLD10024
    PTR * = VAL * CHANGED FLG FIELD.                                           FLD10025
    FLG * = * CHANGED PTR FIELD.                                               FLD10026
    FLG * = * CHANGED VAL FIELD.                                               FLD10027
3.  BASIC CONDITIONALS AND ASSIGNMENTS (VAL TO PTR, PTR TO VAL AND FLG         FLD10028
4.  TO FLG) HAVE BEEN CHECKED WITH POSITIVE OPERANDS.                          FLD10029
    VAL * = * + * FAILS FOR ALL +.                                             FLD10030
    VAL * = * + * CHANGED FLG FIELD ALL +.                                     FLD10031
    VAL * = * + * CHANGED PTR FIELD ALL +.                                     FLD10032
5.  THE NEXT TWO CARD NUMBERS TEST VAL * = CHAR AND CHAR = VAL *.              FLD10033
6.  0 X X X X X X X X X ARE THE CONTENTS OF VAL 0 - VAL 9.                     FLD10034
7.  X X X X ARE THE CONTENTS OF PTR 0 - PTR 3.                                 FLD10035
    VAL * = * - * FAILS ALL +.                                                 FLD10036
    VAL * = * - * CHANGED FLG FIELD ALL +.                                     FLD10037
    VAL * = * - * CHANGED PTR FIELD ALL +.                                     FLD10038
    VAL * = * - * FAILS RES. - OR VAL * = * + * FAILS OP1 -.                   FLD10039
    VAL * = * - * CHANGED FLG FIELD RESULT -.                                  FLD10040
    VAL * = * - * CHANGED PTR FIELD RESULT -.                                  FLD10041
    VAL * = * + * FAILS OP1 - OP2 + RESULT -.                                  FLD10042
    VAL * = * + * FAILS OP1 - OP2 -.                                           FLD10043
    VAL * = * - * FAILS OP1 - OP2 +.                                           FLD10044
    VAL * = * - * FAILS OP1 + OP2 -.                                           FLD10045
    PTR * = * + * FAILS ALL +.                                                 FLD10046
    PTR * = * + * CHANGED FLG FIELD.                                           FLD10047
    PTR * = * + * CHANGED VAL FIELD.                                           FLD10048
    PTR * = * - * FAILS ALL +.                                                 FLD10049
    PTR * = * - * CHANGED FLG FIELD ALL +.                                     FLD10050
```

```
    PTR * = * - * CHANGED VAL FIELD ALL +.                                    FLD10051
    PTR * = * - * FAILS RES. - OR PTR * = * + * FAILS OP1 -.                  FLD10052
    PTR * = * - * CHANGED FLG FIELD  RESULT -.                                FLD10053
    PTR * = * - * CHANGED VAL FIELD  RESULT -.                                FLD10054
    PTR * = * + * FAILS  OP1 - OP2 + RESULT -.                                FLD10055
    PTR * = * + * FAILS  OP1 - OP2 -.                                         FLD10056
    PTR * = * - * FAILS  OP1 - OP2 +.                                         FLD10057
    PTR * = * - * FAILS  OP1 + OP2 -.                                         FLD10058
    VAL * = PTR * FAILS ON -.                                                 FLD10059
    VAL * = PTR * CHANGED FLG ON -.                                           FLD10060
    VAL * = PTR * CHANGED PTR ON -.                                           FLD10061
    PTR * = VAL * FAILS ON -.                                                 FLD10062
    PTR * = VAL * CHANGED FLG ON -.                                           FLD10063
    PTR * = VAL * CHANGED VAL ON -.                                           FLD10064
    PTR * = * * * FAILS BOTH +.                                               FLD10065
    PTR * = * * * CHANGED FLG FIELD  BOTH +.                                  FLD10066
    PTR * = * * * CHANGED VAL FIELD  BOTH +.                                  FLD10067
    PTR * = * * * FAILS BOTH -.                                               FLD10068
    PTR * = * * * FAILS  OP1 - OP2 +.                                         FLD10069
    PTR * = * * * CHANGED FLG FIELD  OP1 - OP2 +.                             FLD10070
    PTR * = * * * CHANGED VAL FIELD  OP1 - OP2 +.                             FLD10071
    PTR * = * * * FAILS  OP1 + OP2 -.                                         FLD10072
    PTR * = * / * FAILS  BOTH + INTEGRAL RESULT.                              FLD10073
    PTR * = * / * CHANGED FLG FIELD  BOTH + INTEGRAL RESULT.                  FLD10074
    PTR * = * / * CHANGED VAL FIELD  BOTH + INTEGRAL RESULT.                  FLD10075
    PTR * = * / * FAILS  BOTH - INTEGRAL RESULT.                              FLD10076
    PTR * = * / * FAILS  OP1 - OP2 + INTEGRAL RESULT.                         FLD10077
    PTR * = * / * CHANGED FLG FIELD  OP1 - OP2 + INTEGRAL RESULT.             FLD10078
       PTR * = * / * CHANGED VAL FIELD  OP1 - OP2 +.                          FLD10079
    PTR * = * / * FAILS OP1 + OP2 - INTEGRAL RESULT.                          FLD10080
    PTR * = * / * FAILS  BOTH + NON-INTEGRAL @UOTIENT.                        FLD10081
    PTR * = * / * CHANGED FLG FIELD BOTH + NON-INTEGRAL @UOTIENT.             FLD10082
    PTR * = * / * CHANGEE VAL FIELD BOTH + NON-INTEGRAL @UOTIENT.             FLD10083
    PTR * = * / * FAILS  OP1 + OP2 - NON-INTEGRAL @UOTIENT.                   FLD10084
    PTR * = * / * CHANGED FLG FIELD  OP1 + OP2 - NON-INTEGRAL @UOTIENT.       FLD10085
    PTR * = * / * CHANGED VAL FIELD                                           FLD10086
    TO ** IF VAL * = * TRANSFERS ON OP1 = -OP2.                               FLD10087
    TO ** IF VAL * NE * FAILS ON UNE@UAL, OPPOSITE SIGN.                      FLD10088
    TO ** IF PTR * = * TRANSFERS ON OP1 = -OP2.                               FLD10089
    TO ** IF PTR * NE * FAILS ON UNE@UAL, OPPOSITE SIGN.                      FLD10090
    TO ** IF PTR * GE * FAILS ON OP1 + OP2 - ABS OP1 GT ABS OP2.              FLD10091
    TO ** IF PTR * GE * TRANSFERS ON OP1 - OP2 + ABS OP1 GT ABS OP2.          FLD10092
    TO ** IF PTR * GE * FAILS ON OP1 - OP2 - OP1 GT OP2.                      FLD10093
    TO ** IF PTR * GE * TRANSFERS ON OP1 - OP2 OP2 GT OP1.                    FLD10094
8.  FURTHER CONDITIONAL AND REGISTER ARITHMETIC CHECKING IS COMPLETE.          FLD10095
9.  IF LINE 10 IS MISSING AND 9 DUPLICATED, TO ** BY * FAILED.                FLD10096
10. IF LINE 11 IS MISSING, RETURN BY * FAILED.                                FLD10097
11. CHECK OF TO AND RETURN COMPLETED.                                         FLD10098
```

FLT2

```
VAL B = 1 + 0.                                                    ;6 20001
READ NEXT B.                                                      ;6 20002
VAL W = 4 + 0.                                                    ;6 20003
WRITE NEXT W.                                                     ;6 20004
VAL B = 1 + 0.                                                    ;6 20005
READ NEXT B.                                                      ;6 20006
VAL W = 4 + 0.                                                    ;6 20007
WRITE NEXT W.                                                     ;6 20008
VAL B = 1 + 0.                                                    ;6 20009
READ NEXT B.                                                      ;6 20010
VAL W = 4 + 0.                                                    ;6 20011
WRITE NEXT W.                                                     ;6 20012
VAL B = 1 + 0.                                                    ;6 20013
READ NEXT B.                                                      ;6 20014
VAL W = 4 + 0.                                                    ;6 20015
WRITE NEXT W.                                                     ;6 20016
VAL B = 1 + 0.                                                    ;6 20017
READ NEXT B.                                                      ;6 20018
VAL W = 3 + 0.                                                    ;6 20019
WRITE NEXT W.                                                     ;6 20020
VAL B = 1 + 0.                                                    ;6 20021
READ NEXT B.                                                      ;6 20022
VAL W = 4 + 0.                                                    ;6 20023
WRITE NEXT W.                                                     ;6 20024
VAL W = 4 + 0.                                                    ;6 20025
MESSAGE CONV TO W.                                                ;6 20026
VAL B = 1 + 0.                                                    ;6 20027
READ NEXT B.                                                      ;6 20028
VAL W = 4 + 0.                                                    ;6 20029
WRITE NEXT W.                                                     ;6 20030
VAL W = 4 + 0.                                                    ;6 20031
MESSAGE EXPR TO W.                                                ;6 20032
VAL B = 1 + 0.                                                    ;6 20033
READ NEXT B.                                                      ;6 20034
VAL W = 4 + 0.                                                    ;6 20035
WRITE NEXT W.                                                     ;6 20036
VAL W = 4 + 0.                                                    ;6 20037
MESSAGE FULL TO W.                                                ;6 20038
VAL B = 1 + 0.                                                    ;6 20039
READ NEXT B.                                                      ;6 20040
VAL W = 4 + 0.                                                    ;6 20041
WRITE NEXT W.                                                     ;6 20042
FLG X = 3.                                                        ;6 20043
VAL X = 4 + 0.                                                    ;6 20044
PTR X = 2 + 0.                                                    ;6 20045
MESSAGE IOCH TO X.                                                ;6 20046
VAL B = 1 + 0.                                                    ;6 20047
READ NEXT B.                                                      ;6 20048
TO 65 IF VAL X = 4.                                               ;6 20049
VAL W = 4 + 0.                                                    ;6 20050
```

```
WRITE NEXT W.                                                    FLT20051
LOC 65.                                                          FLT20052
TO 70 BY D.                                                      FLT20053
VAL B = 1 + 0.                                                   FLT20054
READ NEXT B.                                                     FLT20055
FLG C = 2.                                                       FLT20056
FLG D = C.                                                       FLT20057
TO 01 IF FLG C NE 2.                                             FLT20058
TO 01 IF FLG D NE 2.                                             FLT20059
TO 01 IF FLG D NE C.                                             FLT20060
TO 02.                                                           FLT20061
LOC 01.                                                          FLT20062
VAL W = 4 + 0.                                                   FLT20063
WRITE NEXT W.                                                    FLT20064
LOC 02.                                                          FLT20065
VAL B = 1 + 0.                                                   FLT20066
READ NEXT B.                                                     FLT20068
FLG X = 0.                                                       FLT20068
VAL X = PTR 3.                                                   FLT20069
PTR X = VAL 0.                                                   FLT20070
VAL X = X + 4.                                                   FLT20071
TO 03 IF VAL X NE 7.                                             FLT20072
TO 03 IF PTR X NE 0.                                             FLT20073
TO 03 IF FLG X NE 0.                                             FLT20074
TO 04.                                                           FLT20075
LOC 03.                                                          FLT20076
VAL W = 4 + 0.                                                   FLT20077
WRITE NEXT W.                                                    FLT20078
LOC 04.                                                          FLT20079
VAL B = 1 + 0.                                                   FLT20080
READ NEXT B.                                                     FLT20081
FLG X = 0.                                                       FLT20082
VAL X = PTR 3.                                                   FLT20083
PTR X = VAL 0.                                                   FLT20084
VAL X = 2 + X.                                                   FLT20085
TO 05 IF VAL X NE 5.                                             FLT20086
TO 05 IF PTR X NE 0.                                             FLT20087
TO 05 IF FLG X NE 0.                                             FLT20088
TO 06.                                                           FLT20089
LOC 05.                                                          FLT20090
VAL W = 4 + 0.                                                   FLT20091
WRITE NEXT W.                                                    FLT20092
LOC 06.                                                          FLT20093
VAL B = 1 + 0.                                                   FLT20094
READ NEXT B.                                                     FLT20095
FLG X = 0.                                                       FLT20096
PTR X = VAL 0.                                                   FLT20097
VAL X = PTR 3.                                                   FLT20098
VAL X = X - 2.                                                   FLT20099
TO 07 IF VAL X NE 1.                                             FLT20100
TO 07 IF PTR X NE 0.                                             FLT20101
TO 07 IF FLG X NE 0.                                             FLT20102
TO 08.                                                           FLT20103
LOC 07.                                                          FLT20104
VAL W = 4 + 0.                                                   FLT20105
```

```
WRITE NEXT W.                                                    FLT20106
LOC 08.                                                          FLT20107
VAL B = 1 + 0.                                                   FLT20108
READ NEXT B.                                                     FLT20109
FLG X = 0.                                                       FLT20110
PTR X = VAL 0.                                                   FLT20111
VAL X = PTR 3.                                                   FLT20112
VAL X = 7 - X.                                                   FLT20113
TO 09 IF VAL X NE 4.                                             FLT20114
TO 09 IF FLG X NE 0.                                             FLT20115
TO 09 IF PTR X NE 0.                                             FLT20116
TO 10.                                                           FLT20117
LOC 09.                                                          FLT20118
VAL W = 4 + 0.                                                   FLT20119
WRITE NEXT W.                                                    FLT20120
LOC 10.                                                          FLT20121
VAL B = 1 + 0.                                                   FLT20122
READ NEXT B.                                                     FLT20123
FLG X = 0.                                                       FLT20124
VAL X = PTR 0.                                                   FLT20125
PTR Y = VAL 6.                                                   FLT20126
PTR X = VAL 4.                                                   FLT20127
PTR X = X + 2.                                                   FLT20128
TO 11 IF PTR X NE Y.                                             FLT20129
TO 11 IF VAL X NE 0.                                             FLT20130
TO 11 IF FLG X NE 0.                                             FLT20131
TO 12.                                                           FLT20132
LOC 11.                                                          FLT20133
VAL W = 4 + 0.                                                   FLT20134
WRITE NEXT W.                                                    FLT20135
LOC 12.                                                          FLT20136
VAL B = 1 + 0.                                                   FLT20137
READ NEXT B.                                                     FLT20138
FLG X = 0.                                                       FLT20139
VAL X = PTR 0.                                                   FLT20140
PTR X = VAL 3.                                                   FLT20141
PTR X = 3 + X.                                                   FLT20142
TO 13 IF PTR X NE Y.                                             FLT20143
TO 13 IF VAL X NE 0.                                             FLT20144
TO 13 IF FLG X NE 0.                                             FLT20145
TO 14.                                                           FLT20146
LOC 13.                                                          FLT20147
VAL W = 4 + 0.                                                   FLT20148
WRITE NEXT W.                                                    FLT20149
LOC 14.                                                          FLT20150
VAL B = 1 + 0.                                                   FLT20151
READ NEXT B.                                                     FLT20152
FLG X = 0.                                                       FLT20153
VAL X = PTR 0.                                                   FLT20154
PTR X = VAL 5.                                                   FLT20155
PTR X = X - 3.                                                   FLT20156
TO 15 IF PTR X NE 2.                                             FLT20157
TO 15 IF VAL X NE 0.                                             FLT20158
TO 15 IF FLG X NE 0.                                             FLT20159
TO 16.                                                           FLT20160
```

```
LOC 15.                                                    FLT20161
VAL W = 4 + 0.                                             FLT20162
WRITE NEXT W.                                              FLT20163
LOC 16.                                                    FLT20164
VAL B = 1 + 0.                                             FLT20165
READ NEXT B.                                               FLT20166
FLG X = 0.                                                 FLT20167
VAL X = PTR 0.                                             FLT20168
PTR X = VAL 2.                                             FLT20169
PTR X = 3 - X.                                             FLT20170
TO 17 IF PTR X NE 1.                                       FLT20171
TO 17 IF VAL X NE 0.                                       FLT20172
TO 17 IF FLG X NE 0.                                       FLT20173
TO 18.                                                     FLT20174
LOC 17.                                                    FLT20175
VAL W = 4 + 0.                                             FLT20176
WRITE NEXT W.                                              FLT20177
LOC 18.                                                    FLT20178
VAL B = 1 + 0.                                             FLT20179
READ NEXT B.                                               FLT20180
FLG X = 0.                                                 FLT20181
VAL X = PTR 0.                                             FLT20182
PTR X = VAL 2.                                             FLT20183
PTR X = X * 3.                                             FLT20184
TO 19 IF PTR X NE Y.                                       FLT20185
TO 19 IF VAL X NE 0.                                       FLT20186
TO 19 IF FLG X NE 0.                                       FLT20187
TO 20.                                                     FLT20188
LOC 19.                                                    FLT20189
VAL W = 4 + 0.                                             FLT20190
WRITE NEXT W.                                              FLT20191
LOC 20.                                                    FLT20192
VAL B = 1 + 0.                                             FLT20193
READ NEXT B.                                               FLT20194
FLG X = 0.                                                 FLT20195
VAL X = PTR 0.                                             FLT20196
PTR X = VAL 3.                                             FLT20197
PTR X = 2 * X.                                             FLT20198
TO 21 IF PTR X NE Y.                                       FLT20199
TO 21 IF VAL X NE 0.                                       FLT20200
TO 21 IF FLG X NE 0.                                       FLT20201
TO 22.                                                     FLT20202
LOC 21.                                                    FLT20203
VAL W = 4 + 0.                                             FLT20204
WRITE NEXT W.                                              FLT20205
LOC 22.                                                    FLT20206
VAL B = 1 + 0.                                             FLT20207
READ NEXT B.                                               FLT20208
FLG X = 0.                                                 FLT20209
VAL X = PTR 0.                                             FLT20210
PTR X = 0 + Y.                                             FLT20211
PTR X = X / 2.                                             FLT20212
TO 23 IF PTR X NE 3.                                       FLT20213
TO 23 IF VAL X NE 0.                                       FLT20214
TO 23 IF FLG X NE 0.                                       FLT20215
```

```
TO 24.                                                          FLT20216
LOC 23.                                                         FLT20217
VAL W = 4 + 0.                                                  FLT20218
WRITE NEXT W.                                                   FLT20219
LOC 24.                                                         FLT20220
VAL B = 1 + 0.                                                  FLT20221
READ NEXT B.                                                    FLT20222
FLG X = 0.                                                      FLT20223
VAL X = PTR 0.                                                  FLT20224
PTR X = VAL 2.                                                  FLT20225
PTR X = Y / X.                                                  FLT20226
TO 25 IF PTR X NE 3.                                            FLT20227
TO 25 IF VAL X NE 0.                                            FLT20228
TO 25 IF FLG X NE 0.                                            FLT20229
TO 26.                                                          FLT20230
LOC 25.                                                         FLT20231
VAL W = 4 + 0.                                                  FLT20232
WRITE NEXT W.                                                   FLT20233
LOC 26.                                                         FLT20234
VAL B = 1 + 0.                                                  FLT20235
READ NEXT B.                                                    FLT20236
FLG X = 0.                                                      FLT20237
VAL X = PTR 2.                                                  FLT20238
PTR X = VAL 0.                                                  FLT20239
VAL X = X + X.                                                  FLT20240
TO 27 IF FLG X NE 0.                                            FLT20241
TO 27 IF PTR X NE 0.                                            FLT20242
TO 27 IF VAL X NE 4.                                            FLT20243
TO 28.                                                          FLT20244
LOC 27.                                                         FLT20245
VAL W = 4 + 0.                                                  FLT20246
WRITE NEXT W.                                                   FLT20247
LOC 28.                                                         FLT20248
VAL B = 1 + 0.                                                  FLT20249
READ NEXT B.                                                    FLT20250
FLG X = 0.                                                      FLT20251
VAL X = PTR 2.                                                  FLT20252
PTR X = VAL 0.                                                  FLT20253
VAL X = X - X.                                                  FLT20254
TO 29 IF FLG X NE 0.                                            FLT20255
TO 29 IF PTR X NE 0.                                            FLT20256
TO 29 IF VAL X NE 0.                                            FLT20257
TO 30.                                                          FLT20258
LOC 29.                                                         FLT20259
VAL W = 4 + 0.                                                  FLT20260
WRITE NEXT W.                                                   FLT20261
LOC 30.                                                         FLT20262
VAL B = 1 + 0.                                                  FLT20263
READ NEXT B.                                                    FLT20264
FLG X = 0.                                                      FLT20265
PTR X = VAL 1.                                                  FLT20266
VAL X = PTR 0.                                                  FLT20267
PTR X = X + X.                                                  FLT20268
TO 31 IF FLG X NE 0.                                            FLT20269
TO 31 IF VAL X NE 0.                                            FLT20270
```

```
TO 31 IF PTR X NE 2.                                    FLT20271
TO 32.                                                  FLT20272
LOC 31.                                                 FLT20273
VAL W = 4 + 0.                                          FLT20274
WRITE NEXT W.                                           FLT20275
LOC 32.                                                 FLT20276
VAL B = 1 + 0.                                          FLT20277
READ NEXT B.                                            FLT20278
FLG X = 0.                                              FLT20279
VAL X = PTR 0.                                          FLT20280
PTR X = VAL 4.                                          FLT20281
PTR X = X - X.                                          FLT20282
TO 33 IF FLG X NE 0.                                    FLT20283
TO 33 IF VAL X NE 0.                                    FLT20284
TO 33 IF PTR X NE 0.                                    FLT20285
TO 34.                                                  FLT20286
LOC 33.                                                 FLT20287
VAL W = 4 + 0.                                          FLT20288
WRITE NEXT W.                                           FLT20289
LOC 34.                                                 FLT20290
VAL B = 1 + 0.                                          FLT20291
READ NEXT B.                                            FLT20292
FLG X = 0.                                              FLT20293
VAL X = PTR 0.                                          FLT20294
PTR Y = VAL 4.                                          FLT20295
PTR X = VAL 2.                                          FLT20296
PTR X = X * X.                                          FLT20297
TO 35 IF FLG X NE 0.                                    FLT20298
TO 35 IF VAL X NE 0.                                    FLT20299
TO 35 IF PTR X NE Y.                                    FLT20300
TO 36.                                                  FLT20301
LOC 35.                                                 FLT20302
VAL W = 4 + 0.                                          FLT20303
WRITE NEXT W.                                           FLT20304
LOC 36.                                                 FLT20305
VAL B = 1 + 0.                                          FLT20306
READ NEXT B.                                            FLT20307
FLG X = 0.                                              FLT20308
VAL X = PTR 0.                                          FLT20309
PTR X = VAL 3.                                          FLT20310
PTR X = X / X.                                          FLT20311
TO 37 IF FLG X NE 0.                                    FLT20312
TO 37 IF VAL X NE 0.                                    FLT20313
TO 37 IF PTR X NE 1.                                    FLT20314
TO 38.                                                  FLT20315
LOC 37.                                                 FLT20316
VAL W = 4 + 0.                                          FLT20317
WRITE NEXT W.                                           FLT20318
LOC 38.                                                 FLT20319
VAL B = 1 + 0.                                          FLT20320
READ NEXT B.                                            FLT20321
VAL W = 4 + 0.                                          FLT20322
WRITE NEXT W.                                           FLT20323
VAL B = 1 + 0.                                          FLT20324
READ NEXT B.                                            FLT20325
```

```
FLG X = 0.                                                    FLT20326
PTR X = 0 + 0.                                                FLT20327
VAL X = CHAR.                                                 FLT20328
TO 58 IF FLG X NE 0.                                          FLT20329
TO 58 IF PTR X NE 0.                                          FLT20330
TO 59.                                                        FLT20331
LOC 58.                                                       FLT20332
VAL W = 4 + 0.                                                FLT20333
WRITE NEXT W.                                                 FLT20334
LOC 59.                                                       FLT20335
VAL B = 1 + 0.                                                FLT20336
READ NEXT B.                                                  FLT20337
VAL W = 4 + 0.                                                FLT20338
WRITE NEXT W.                                                 FLT20339
VAL A = 0 - 1.                                                FLT20340
CHAR = VAL A.                                                 FLT20341
VAL W = 4 + 0.                                                FLT20342
WRITE NEXT W.                                                 FLT20343
VAL B = 1 + 0.                                                FLT20344
READ NEXT B.                                                  FLT20345
FLG X = 3.                                                    FLT20346
VAL X = 4 + 0.                                                FLT20347
PTR X = 2 + 0.                                                FLT20348
WRITE NEXT X.                                                 FLT20349
VAL B = 1 + 0.                                                FLT20350
READ NEXT B.                                                  FLT20351
TO 60 IF VAL X = 4.                                           FLT20352
VAL W = 4 + 0.                                                FLT20353
WRITE NEXT W.                                                 FLT20354
LOC 60.                                                       FLT20355
TO 70 BY D.                                                   FLT20356
FLG X = 3.                                                    FLT20357
VAL X = 1 + 0.                                                FLT20358
PTR X = 2 + 0.                                                FLT20359
READ NEXT X.                                                  FLT20360
TO 61 IF VAL X = 1.                                           FLT20361
VAL W = 4 + 0.                                                FLT20362
WRITE NEXT W.                                                 FLT20363
LOC 61.                                                       FLT20364
TO 70 BY D.                                                   FLT20365
FLG X = 3.                                                    FLT20366
VAL X = 2 + 0.                                                FLT20367
PTR X = 2 + 0.                                                FLT20368
REWIND X.                                                     FLT20369
VAL B = 1 + 0.                                                FLT20370
READ NEXT B.                                                  FLT20371
TO 62 IF VAL X = 2.                                           FLT20372
VAL W = 4 + 0.                                                FLT20373
WRITE NEXT W.                                                 FLT20374
LOC 62.                                                       FLT20375
TO 70 BY D.                                                   FLT20376
VAL B = 1 + 0.                                                FLT20377
READ NEXT B.                                                  FLT20378
VAL X = CHAR.                                                 FLT20379
VAL X = CHAR.                                                 FLT20380
```

```
VAL Z = 2 + 0.                                                  FLT20381
WRITE NEXT Z.                                                   FLT20382
VAL B = 1 + 0.                                                  FLT20383
READ NEXT B.                                                    FLT20384
VAL Z = 2 + 0.                                                  FLT20385
WRITE NEXT Z.                                                   FLT20386
VAL B = 1 + 0.                                                  FLT20387
READ NEXT B.                                                    FLT20388
VAL Z = 2 + 0.                                                  FLT20389
REWIND Z.                                                       FLT20390
VAL Z = 2 + 0.                                                  FLT20391
READ NEXT Z.                                                    FLT20392
VAL Z = CHAR.                                                   FLT20393
VAL Z = CHAR.                                                   FLT20394
VAL W = 4 + 0.                                                  FLT20395
WRITE NEXT W.                                                   FLT20396
VAL B = 1 + 0.                                                  FLT20397
READ NEXT B.                                                    FLT20398
TO 63 IF VAL X = Z.                                             FLT20399
VAL W = 4 + 0.                                                  FLT20400
WRITE NEXT W.                                                   FLT20401
LOC 63.                                                         FLT20402
VAL Z = 2 + 0.                                                  FLT20403
READ NEXT Z.                                                    FLT20404
VAL W = 4 + 0.                                                  FLT20405
WRITE NEXT W.                                                   FLT20406
FLG Z = 0.                                                      FLT20407
VAL Z = 2 + 0.                                                  FLT20408
READ NEXT Z.                                                    FLT20409
VAL B = 1 + 0.                                                  FLT20410
READ NEXT B.                                                    FLT20411
TO 64 IF FLG Z = 1.                                             FLT20412
VAL W = 4 + 0.                                                  FLT20413
WRITE NEXT W.                                                   FLT20414
LOC 64.                                                         FLT20415
VAL B = 1 + 0.                                                  FLT20416
READ NEXT B.                                                    FLT20417
VAL W = 3 + 0.                                                  FLT20418
WRITE NEXT W.                                                   FLT20419
VAL B = 1 + 0.                                                  FLT20420
READ NEXT B.                                                    FLT20421
VAL W = 3 + 0.                                                  FLT20422
WRITE NEXT W.                                                   FLT20423
VAL B = 1 + 0.                                                  FLT20424
READ NEXT B.                                                    FLT20425
VAL W = 4 + 0.                                                  FLT20426
WRITE NEXT W.                                                   FLT20427
VAL B = 1 + 0.                                                  FLT20428
READ NEXT B.                                                    FLT20429
TO 39 IF PTR 8 GE 9.                                            FLT20430
TO 40.                                                          FLT20431
LOC 39.                                                         FLT20432
VAL W = 4 + 0.                                                  FLT20433
WRITE NEXT W.                                                   FLT20434
STOP.                                                           FLT20435
```

```
LOC 40.                                                    FLT20436
VAL B = 1 + 0.                                             FLT20437
READ NEXT B.                                               FLT20438
PTR C = 0 + 8.                                             FLT20439
FLG D = 0.                                                 FLT20440
VAL D = PTR 0.                                             FLT20441
PTR D = VAL 0.                                             FLT20442
LOC 41.                                                    FLT20443
STO C = D.                                                 FLT20444
PTR C = C + 7.                                             FLT20445
TO 41 IF PTR 9 GE C.                                       FLT20446
FLG D = 3.                                                 FLT20447
PTR C = 0 + 8.                                             FLT20448
LOC 42.                                                    FLT20449
GET E = C.                                                 FLT20450
TO 43 IF FLG E NE 0.                                       FLT20451
STO C = D.                                                 FLT20452
GET E = C.                                                 FLT20453
TO 43 IF FLG E NE 3.                                       FLT20454
PTR C = C + 7.                                             FLT20455
TO 42 IF PTR 9 GE C.                                       FLT20456
TO 44.                                                     FLT20457
LOC 43.                                                    FLT20458
VAL W = 4 + 0.                                             FLT20459
WRITE NEXT W.                                              FLT20460
LOC 44.                                                    FLT20461
VAL B = 1 + 0.                                             FLT20462
READ NEXT B.                                               FLT20463
VAL D = 0 - 1.                                             FLT20464
VAL F = 0 - 1.                                             FLT20465
PTR C = 0 + 8.                                             FLT20466
LOC 45.                                                    FLT20467
GET E = C.                                                 FLT20468
TO 46 IF VAL E NE 0.                                       FLT20469
STO C = D.                                                 FLT20470
GET E = C.                                                 FLT20471
TO 46 IF VAL E NE F.                                       FLT20472
PTR C = C + 7.                                             FLT20473
TO 45 IF PTR 9 GE C.                                       FLT20474
TO 47.                                                     FLT20475
LOC 46.                                                    FLT20476
VAL W = 4 + 0.                                             FLT20477
WRITE NEXT W.                                              FLT20478
LOC 47.                                                    FLT20479
VAL B = 1 + 0.                                             FLT20480
READ NEXT B.                                               FLT20481
PTR D = 0 - 1.                                             FLT20482
PTR F = 0 - 1.                                             FLT20483
PTR C = 0 + 8.                                             FLT20484
LOC 48.                                                    FLT20485
GET E = C.                                                 FLT20486
TO 49 IF PTR E NE 0.                                       FLT20487
STO C = D.                                                 FLT20488
GET E = C.                                                 FLT20489
TO 49 IF PTR E NE F.                                       FLT20490
```

```
PTR C = C + 7.                                                  FLT20491
TO 48 IF PTR 9 GE C.                                            FLT20492
TO 50.                                                          FLT20493
LOC 49.                                                         FLT20494
VAL W = 4 + 0.                                                  FLT20495
WRITE NEXT W.                                                   FLT20496
LOC 50.                                                         FLT20497
VAL B = 1 + 0.                                                  FLT20498
READ NEXT B.                                                    FLT20499
VAL W = 4 + 0.                                                  FLT20500
WRITE NEXT W.                                                   FLT20501
VAL B = 1 + 0.                                                  FLT20502
READ NEXT B.                                                    FLT20503
VAL W = 4 + 0.                                                  FLT20504
WRITE NEXT W.                                                   FLT20505
LOC 51.                                                         FLT20506
VAL X = CHAR.                                                   FLT20507
PTR D = D + 1.                                                  FLT20508
TO 51 IF VAL X NE F.                                            FLT20509
VAL B = 1 + 0.                                                  FLT20510
READ NEXT B.                                                    FLT20511
VAL A = CHAR.                                                   FLT20512
VAL B = CHAR.                                                   FLT20513
VAL C = CHAR.                                                   FLT20514
VAL D = CHAR.                                                   FLT20515
VAL E = CHAR.                                                   FLT20516
VAL H = CHAR.                                                   FLT20517
PTR X = VAL 5.                                                  FLT20518
PTR X = X + X.                                                  FLT20519
PTR Y = X * X.                                                  FLT20520
PTR A = D / Y.                                                  FLT20521
PTR B = D / X.                                                  FLT20522
PTR C = B * X.                                                  FLT20523
PTR C = D - C.                                                  FLT20524
PTR E = A * X.                                                  FLT20525
PTR B = B - E.                                                  FLT20526
VAL E = PTR A.                                                  FLT20527
VAL E = E + H.                                                  FLT20528
VAL F = PTR B.                                                  FLT20529
VAL F = F + H.                                                  FLT20530
VAL G = PTR C.                                                  FLT20531
VAL G = G + H.                                                  FLT20532
CHAR = VAL A.                                                   FLT20533
CHAR = VAL B.                                                   FLT20534
CHAR = VAL C.                                                   FLT20535
CHAR = VAL D.                                                   FLT20536
CHAR = VAL E.                                                   FLT20537
CHAR = VAL F.                                                   FLT20538
CHAR = VAL G.                                                   FLT20539
VAL W = 4 + 0.                                                  FLT20540
WRITE NEXT W.                                                   FLT20541
VAL B = 1 + 0.                                                  FLT20542
READ NEXT B.                                                    FLT20543
VAL W = 4 + 0.                                                  FLT20544
WRITE NEXT W.                                                   FLT20545
```

```
FLG C = 2.                                                    FLT20546
CHAR = VAL C.                                                 FLT20547
VAL B = 1 + 0.                                                FLT20548
READ NEXT B.                                                  FLT20549
TO 52 IF FLG C = 0.                                           FLT20550
VAL W = 4 + 0.                                                FLT20551
WRITE NEXT W.                                                 FLT20552
LOC 52.                                                       FLT20553
VAL B = 1 + 0.                                                FLT20554
READ NEXT B.                                                  FLT20555
FLG A = 0.                                                    FLT20556
VAL A = 0 - 1.                                                FLT20557
CHAR = VAL A.                                                 FLT20558
TO 53 IF FLG A = 1.                                           FLT20559
VAL W = 4 + 0..                                               FLT20560
WRITE NEXT W.                                                 FLT20561
STOP.                                                         FLT20562
LOC 53.                                                       FLT20563
CHAR = VAL C.                                                 FLT20564
TO 53 IF FLG C = 0.                                           FLT20565
VAL W = 4 + 0.                                                FLT20566
WRITE NEXT W.                                                 FLT20567
VAL B = 1 + 0.                                                FLT20568
READ NEXT B.                                                  FLT20569
VAL W = 4 + 0.                                                FLT20570
WRITE NEXT W.                                                 FLT20571
STOP.                                                         FLT20572
LOC 70.                                                       FLT20573
VAL B = 1 + 0.                                                FLT20574
READ NEXT B.                                                  FLT20575
TO 71 IF PTR X = 2.                                           FLT20576
VAL W = 4 + 0.                                                FLT20577
WRITE NEXT W.                                                 FLT20578
LOC 71.                                                       FLT20579
VAL X = 4 + 0.                                                FLT20580
TO 73 IF FLG X = 0.                                           FLT20581
VAL B = 1 + 0.                                                FLT20582
READ NEXT B.                                                  FLT20583
VAL X = X - 1.                                                FLT20584
TO 72 IF FLG X = 1.                                           FLT20585
VAL B = 1 + 0.                                                FLT20586
READ NEXT B.                                                  FLT20587
VAL X = X - 1.                                                FLT20588
TO 72 IF FLG X = 2.                                           FLT20589
VAL B = 1 + 0.                                                FLT20590
READ NEXT B.                                                  FLT20591
VAL X = X - 1.                                                FLT20592
TO 72 IF FLG X = 3.                                           FLT20593
VAL B = 1 + 0.                                                FLT20594
READ NEXT B.                                                  FLT20595
VAL X = X - 1.                                                FLT20596
LOC 72.                                                       FLT20597
VAL W = 4 + 0.                                                FLT20598
WRITE NEXT W.                                                 FLT20599
LOC 73.                                                       FLT20600
```

```
TO 74 IF VAL X = 0.                                              FLT20601
VAL B = 1 + 0.                                                   FLT20602
READ NEXT B.                                                     FLT20603
VAL X = X - 1.                                                   FLT20604
TO 73.                                                           FLT20605
LOC 74.                                                          FLT20606
RETURN BY D.                                                     FLT20607
END PROGRAM.                                                     FLT20608
```

FLD2

```
1.  THIS PROGRAM PERFORMS FURTHER TESTS ON THE FLUB MACROS. IT SHOULD    FLD20001
2.  BE RUN ONLY AFTER FLT1 HAS BEEN COMPLETED SUCCESSFULLY. OUTPUT       FLD20002
3.  FROM THIS PROGRAM CONSISTS OF A SERIES OF NUMBERED LINES AND ERROR   FLD20003
4.  MESSAGES. BOTH REGISTER AND MEMORY OPERATIONS ARE TESTED HERE.       FLD20004
1.  THIS LINE SHOULD APPEAR ONLY ON CHANNEL 3 (MACHINE READABLE OUTPUT)  FLD20005
5.  CONV ERROR MESSAGE SHOULD FOLLOW                                     FLD20006
6.  EXPR ERROR MESSAGE SHOULD FOLLOW                                     FLD20007
7.  FULL ERROR MESSAGE SHOULD FOLLOW                                     FLD20008
8.  IOCH ERROR MESSAGE SHOULD FOLLOW                                     FLD20009
     MESSAGE ALTERS VAL FIELD OF CHANNEL REGISTER.                       FLD20010
     MESSAGE ALTERS PTR FIELD OF CHANNEL REGISTER.                       FLD20011
     MESSAGE SET ENDFILE ON CHANNEL 4.                                   FLD20012
     MESSAGE SET ILLEGAL OPERATION ON CHANNEL 4.                         FLD20013
     MESSAGE DOES NOT ALTER THE FLAG OF THE CHANNEL REGISTER.            FLD20014
     MESSAGE DESTROYS THE FLAG OF THE CHANNEL REGISTER.                  FLD20015
     FLG * = * FAILS FOR BOTH DESCRIPTORS ALPHABETIC.                    FLD20016
     VAL X = X + * FAILS                                                 FLD20017
     VAL X = * + X FAILS                                                 FLD20018
     VAL X = X - * FAILS                                                 FLD20019
     VAL X = * - X FAILS                                                 FLD20020
     PTR X = X + * FAILS                                                 FLD20021
     PTR X = * + X FAILS                                                 FLD20022
     PTR X = X - * FAILS                                                 FLD20023
     PTR X = * - X FAILS                                                 FLD20024
     PTR X = X * * FAILS                                                 FLD20025
     PTR X = * * X FAILS                                                 FLD20026
     PTR X = X / * FAILS                                                 FLD20027
     PTR X = * / X FAILS                                                 FLD20028
     VAL X = X + X FAILS                                                 FLD20029
     VAL X = X - X FAILS                                                 FLD20030
     PTR X = X + X FAILS                                                 FLD20031
     PTR X = X - X FAILS                                                 FLD20032
     PTR X = X * X FAILS                                                 FLD20033
     PTR X = X / X FAILS                                                 FLD20034
9.  REGISTER TESTS COMPLETE.  I/O TESTS FOLLOW.                          FLD20035
    VAL * = CHAR ALTERS FLG OR PTR.                                      FLD20036
10. IF THIS REPEATS, -1 DIDNT CLEAR LINE BUFFER. A BLANK SHOULD FOLLOW.  FLD20037
11. THIS LINE SHOULD BE SEPARATED FROM 10 BY A BLANK LINE.               FLD20038
     WRITE NEXT ALTERS VAL FIELD OF SPECIFYING REGISTER.                 FLD20039
     WRITE NEXT ALTERS PTR FIELD OF SPECIFYING REGISTER.                 FLD20040
     WRITE NEXT SETS ENDFILE ON CHANNEL 4.                               FLD20041
     WRITE NEXT SETS ILLEGAL OPERATION ON CHANNEL 4.                     FLD20042
     WRITE NEXT DOES NOT ALTER FLAG OF SPECIFYING REGISTER.              FLD20043
     WRITE NEXT DESTROYS FLAG OF SPECIFYING REGISTER.                    FLD20044
     READ NEXT ALTERS VAL FIELD OF SPECIFYING REGISTER.                  FLD20045
     READ NEXT ALTERS PTR FIELD OF SPECIFYING REGISTER.                  FLD20046
     READ NEXT SETS ENDFILE ON CHANNEL 1.                                FLD20047
     READ NEXT SETS ILLEGAL OPERATION ON CHANNEL 1.                      FLD20048
     READ NEXT DOES NOT ALTER FLAG OF SPECIFYING REGISTER.               FLD20049
     READ NEXT DESTROYS FLAG OF SPECIFYING REGISTER.                     FLD20050
```

```
      REWIND ALTERS VAL FIELD OF SPECIFYING REGISTER.                    FLD20051
      REWIND ALTERS PTR FIELD OF SPECIFYING REGISTER.                    FLD20052
      REWIND SET ENDFILE ON CHANNEL 2.                                   FLD20053
      REWIND SET ILLEGAL OPERATION ON CHANNEL 2.                         FLD20054
      REWIND DOES NOT ALTER FLAG OF SPECIFYING REGISTER.                 FLD20055
      REWIND DESTROYS FLAG OF SPECIFYING REGISTER.                       FLD20056
12.   THIS LINE WAS WRITTEN TO CHANNEL 2 AND READ BACK.                  FLD20057
13.   THIS LINE WAS ALSO READ BACK FROM CHANNEL 2.  IT WAS THE LAST.     FLD20058
      THE LINE BUFFER WAS UNCHANGED BY A READ FROM CHANNEL 2.            FLD20059
      REWIND DOES NOT REPOSITION CHANNEL 2 PROPERLY.                     FLD20060
      NO ENDFILE FOUND ON CHANNEL 2.                                     FLD20061
2.    THIS LINE SHOULD ONLY APPEAR ON CHANNEL 3                          FLD20062
3.    THIS LINE SHOULD ONLY APPEAR ON CHANNEL 3 (IT IS THE LAST)         FLD20063
14.   I/O TESTS COMPLETE.  MEMORY TESTS FOLLOW.                          FLD20064
      MEMORY SIZE IS ZERO OR NEGATIVE. FURTHER TESTS ABORTED             FLD20065
      ERROR IN STO-GET IN FLG FIELD.                                     FLD20066
      ERROR IN STO-GET IN VAL FIELD.                                     FLD20067
      ERROR IN STO-GET IN PTR FIELD.                                     FLD20068
15.   MEMORY TESTS COMPLETE.  FURTHER TESTS OF CHARACTER I/O FOLLOW.     FLD20069
16.   IF LINE 17 IS WRONG, -1 WAS NOT FOUND AT THE END OF THE LINE BUFFER.FLD20070
17.   000 CHARACTERS WERE EXTRACTED FROM THIS LINE BEFORE FINDING -1.    FLD20071
18.   THE NEXT LINE TESTS THE SETTING OF FLG=1 ON FILLING THE LINE BUFFER.FLD20072
      FLG IS NOT SET TO 0 WHEN A SPACE IS INSERTED INTO THE LINE BUFFER  FLD20073
      FLG IS NOT SET TO 1 WHEN -1 IS INSERTED INTO THE LINE BUFFER       FLD20074
19.   A FULL LINE OF DOTS PROVES THE FLG IS SET PROPERLY BY CHAR = VAL *. FLD20075
```

ST62

```
FLG I = 0.                                                    STG20001
VAL I = 1 + 0.                                                STG20002
PTR I = 0 + 0.                                                STG20003
READ NEXT I.                                                  STG20004
TO 98 IF FLG I NE 0.                                          STG20005
VAL A = CHAR.                                                 STG20006
PTR A = 8 + 0.                                                STG20007
STO A = I.                                                    STG20008
FLG B = 2.                                                    STG20009
VAL B = CHAR.                                                 STG20010
VAL C = CHAR.                                                 STG20011
PTR C = 9 + 0.                                                STG20012
VAL D = CHAR.                                                 STG20013
VAL E = CHAR.                                                 STG20014
PTR E = VAL E.                                                STG20015
VAL F = CHAR.                                                 STG20016
PTR F = A + 7.                                                STG20017
STO F = 0.                                                    STG20018
VAL G = 0 + 0.                                                STG20019
PTR H = 5 * 7.                                                STG20020
FLG J = 1.                                                    STG20021
PTR J = 0 + 0.                                                STG20022
FLG L = 1.                                                    STG20023
VAL L = 0 - 1.                                                STG20024
PTR L = 0 + 0.                                                STG20025
VAL M = CHAR.                                                 STG20026
PTR M = 0 + 0.                                                STG20027
FLG N = 0.                                                    STG20028
VAL N = CHAR.                                                 STG20029
FLG O = 0.                                                    STG20030
VAL O = CHAR.                                                 STG20031
VAL P = CHAR.                                                 STG20032
VAL Q = CHAR.                                                 STG20033
VAL R = CHAR.                                                 STG20034
PTR R = 0 + 0.                                                STG20035
PTR 4 = 7 + 7.                                                STG20036
PTR 8 = F + 7.                                                STG20037
TO 01 BY D.                                                   STG20038
LOC 01.                                                       STG20039
GET I = A.                                                    STG20040
READ NEXT I.                                                  STG20041
TO 98 IF FLG I NE 0.                                          STG20042
PTR I = C + 0.                                                STG20043
VAL Y = 0 + 0.                                                STG20044
PTR Y = C + 0.                                                STG20045
TO 02 IF PTR M = 0.                                           STG20046
PTR M = M - 1.                                                STG20047
TO 01.                                                        STG20048
LOC 02.                                                       STG20049
PTR 9 = I + 0.                                                STG20050
```

```
VAL I = CHAR.                                                    STG20051
PTR I = 9 - 7.                                                   STG20052
TO 97 IF PTR 8 GE I.                                             STG20053
STO 9 = I.                                                       STG20054
TO 04 IF VAL I = L.                                              STG20055
TO 03 IF VAL I = A.                                              STG20056
VAL Y = Y + 1.                                                   STG20057
TO 02 IF VAL I NE B.                                             STG20058
PTR B = I + 0.                                                   STG20059
STO 9 = B.                                                       STG20060
TO 02.                                                           STG20061
LOC 03.                                                          STG20062
PTR 9 = I + 0.                                                   STG20063
VAL I = CHAR.                                                    STG20064
PTR I = 9 - 7.                                                   STG20065
STO 9 = I.                                                       STG20066
TO 97 IF PTR 8 GE I.                                             STG20067
TO 03 IF VAL I NE L.                                             STG20068
LOC 04.                                                          STG20069
PTR U = 9 - 7.                                                   STG20070
STO U = 3.                                                       STG20071
PTR U = U - 7.                                                   STG20072
STO U = 3.                                                       STG20073
PTR U = U - 7.                                                   STG20074
STO U = 3.                                                       STG20075
PTR U = U - 7.                                                   STG20076
STO U = 3.                                                       STG20077
PTR U = U - 7.                                                   STG20078
STO U = 3.                                                       STG20079
PTR U = U - 7.                                                   STG20080
STO U = 3.                                                       STG20081
PTR U = U - 7.                                                   STG20082
STO U = 3.                                                       STG20083
PTR U = U - 7.                                                   STG20084
STO U = 3.                                                       STG20085
PTR V = U - 7.                                                   STG20086
STO V = 3.                                                       STG20087
PTR U = V - 7.                                                   STG20088
PTR 9 = U + 0.                                                   STG20089
TO 97 IF PTR 8 GE 9.                                             STG20090
GET W = A.                                                       STG20091
GET X = Y.                                                       STG20092
FLG Y = 0.                                                       STG20093
PTR Z = A + 0.                                                   STG20094
TO 58 BY B.                                                      STG20095
TO 50 IF FLG B = 2.                                              STG20096
TO 56 IF FLG Y = 0.                                              STG20097
STO 9 = 1.                                                       STG20098
PTR 9 = 9 - H.                                                   STG20099
STO 9 = J.                                                       STG20100
PTR J = 9 + H.                                                   STG20101
PTR 9 = 9 - 7.                                                   STG20102
STO 9 = C.                                                       STG20103
PTR 9 = 9 - 7.                                                   STG20104
STO 9 = D.                                                       STG20105
```

```
PTR 9 = 9 - 7.                                                  STG20106
STO 9 = K.                                                      STG20107
PTR K = U + 0.                                                  STG20108
PTR 9 = 9 - 7.                                                  STG20109
STO 9 = R.                                                      STG20110
PTR R = 0 + 0.                                                  STG20111
PTR C = 9 - 7.                                                  STG20112
TO 97 IF PTR 8 GE C.                                            STG20113
TO 05 BY D.                                                     STG20114
LOC 05.                                                         STG20115
PTR 9 = C + 0.                                                  STG20116
PTR Y = 0 + 0.                                                  STG20117
LOC 06.                                                         STG20118
TO 07 IF PTR M = 0.                                             STG20119
PTR Z = K + 7.                                                  STG20120
GET K = K.                                                      STG20121
GET I = K.                                                      STG20122
TO 08 IF VAL I = 1.                                             STG20123
PTR M = M - 1.                                                  STG20124
GET Z = Z.                                                      STG20125
TO 06 IF FLG Z NE 3.                                            STG20126
PTR Y = Y + 1.                                                  STG20127
TO 06 IF VAL Z = 7.                                             STG20128
PTR Y = Y - 1.                                                  STG20129
TO 06 IF VAL Z NE 8.                                            STG20130
PTR Y = Y - 1.                                                  STG20131
TO 06 IF PTR Y GE 0.                                            STG20132
TO 06 IF PTR R = 0.                                             STG20133
PTR U = R - 7.                                                  STG20134
GET Y = U.                                                      STG20135
TO 49 IF FLG Y NE 1.                                            STG20136
PTR C = R + 0.                                                  STG20137
GET R = R.                                                      STG20138
TO 05.                                                          STG20139
LOC 07.                                                         STG20140
PTR K = K + 7.                                                  STG20141
GET I = K.                                                      STG20142
TO 09 IF FLG I = 2.                                             STG20143
TO 22 IF FLG I = 3.                                             STG20144
PTR I = 9 - 7.                                                  STG20145
STO 9 = I.                                                      STG20146
PTR 9 = I + 0.                                                  STG20147
TO 97 IF PTR 8 GE 9.                                            STG20148
TO 07 IF FLG I = 0.                                             STG20149
PTR Y = C - 9.                                                  STG20150
PTR Y = Y / 7.                                                  STG20151
PTR Y = Y - 1.                                                  STG20152
VAL Y = PTR Y.                                                  STG20153
PTR Y = C + 0.                                                  STG20154
TO 04 IF VAL I NE 1.                                            STG20155
LOC 08.                                                         STG20156
PTR 9 = J - H.                                                  STG20157
GET J = 9.                                                      STG20158
PTR 9 = 9 - 7.                                                  STG20159
GET C = 9.                                                      STG20160
```

```
PTR 9 = 9 - 7.                                                  STG20161
GET D = 9.                                                      STG20162
PTR 9 = 9 - 7.                                                  STG20163
GET K = 9.                                                      STG20164
PTR 9 = 9 - 7.                                                  STG20165
GET R = 9.                                                      STG20166
RETURN BY D.                                                    STG20167
LOC 09.                                                         STG20168
PTR V = J + I.                                                  STG20169
TO 21 IF VAL I = 6.                                             STG20170
GET Y = V.                                                      STG20171
TO 45 IF VAL I = 7.                                             STG20172
TO 23 IF FLG Y = 3.                                             STG20173
GET X = Y.                                                      STG20174
TO 11 IF VAL I = 0.                                             STG20175
TO 10 IF VAL I = 1.                                             STG20176
TO 12 IF VAL I = 2.                                             STG20177
TO 15 IF VAL I = 4.                                             STG20178
PTR X = Y + 0.                                                  STG20179
TO 20 IF VAL I = 3.                                             STG20180
PTR N = VAL Y.                                                  STG20181
TO 18 IF VAL I = 5.                                             STG20182
TO 23 IF VAL Y NE 1.                                            STG20183
PTR N = VAL X.                                                  STG20184
TO 18 IF VAL I = 8.                                             STG20185
MESSAGE CONV TO 4.                                              STG20186
TO 94 BY B.                                                     STG20187
TO 07.                                                          STG20188
LOC 10.                                                         STG20189
PTR V = 9 + 7.                                                  STG20190
GET W = F.                                                      STG20191
PTR Z = F + 0.                                                  STG20192
TO 58 BY B.                                                     STG20193
TO 07 IF FLG Y NE 1.                                            STG20194
FLG I = 0.                                                      STG20195
GET X = Y.                                                      STG20196
LOC 11.                                                         STG20197
TO 07 IF VAL Y = 0.                                             STG20198
GET I = X.                                                      STG20199
PTR X = 9 - 7.                                                  STG20200
STO 9 = X.                                                      STG20201
PTR 9 = X + 0.                                                  STG20202
VAL Y = Y - 1.                                                  STG20203
TO 07 IF VAL Y = 0.                                             STG20204
GET X = I.                                                      STG20205
PTR I = 9 - 7.                                                  STG20206
STO 9 = I.                                                      STG20207
PTR 9 = I + 0.                                                  STG20208
TO 97 IF PTR 8 GE 9.                                            STG20209
VAL Y = Y - 1.                                                  STG20210
TO 11.                                                          STG20211
LOC 12.                                                         STG20212
FLG B = 2.                                                      STG20213
GET W = F.                                                      STG20214
PTR Z = F + 0.                                                  STG20215
```

```
TO 58 BY B.                                                    STG20216
FLG B = 0.                                                     STG20217
GET X = Y.                                                     STG20218
TO 11 IF FLG Y = 1.                                            STG20219
PTR Y = 8 + 0.                                                 STG20220
FLG Y = 1.                                                     STG20221
PTR L = L + 1.                                                 STG20222
PTR X = L + 0.                                                 STG20223
PTR W = 9 + 7.                                                 STG20224
VAL Y = 0 + 0.                                                 STG20225
LOC 13.                                                        STG20226
PTR V = X / 5.                                                 STG20227
PTR Z = V * 5.                                                 STG20228
PTR X = X - Z.                                                 STG20229
VAL X = PTR X.                                                 STG20230
PTR X = V + 0.                                                 STG20231
PTR W = W - 7.                                                 STG20232
STO W = X.                                                     STG20233
VAL Y = Y + 1.                                                 STG20234
TO 97 IF PTR 8 GE W.                                           STG20235
TO 13 IF PTR X NE 0.                                           STG20236
LOC 14.                                                        STG20237
GET X = W.                                                     STG20238
PTR W = W + 7.                                                 STG20239
VAL X = X + E.                                                 STG20240
PTR X = 8 + 7.                                                 STG20241
STO 8 = X.                                                     STG20242
PTR 8 = X + 0.                                                 STG20243
TO 14 IF PTR 9 GE W.                                           STG20244
STO 8 = 0.                                                     STG20245
PTR 8 = 8 + 7.                                                 STG20246
TO 97 IF PTR 8 GE 9.                                           STG20247
STO U = Y.                                                     STG20248
GET X = Y.                                                     STG20249
FLG I = 0.                                                     STG20250
TO 11.                                                         STG20251
LOC 15.                                                        STG20252
TO 74 BY P.                                                    STG20253
TO 18 IF PTR N GE 0.                                           STG20254
PTR O = 9 - 7.                                                 STG20255
TO 97 IF PTR 8 GE O.                                           STG20256
STO 9 = O.                                                     STG20257
PTR 9 = O + 0.                                                 STG20258
PTR N = 0 - N.                                                 STG20259
TO 18.                                                         STG20260
LOC 16.                                                        STG20261
GET Y = V.                                                     STG20262
TO 17 IF FLG Y = 1.                                            STG20263
PTR V = V - 7.                                                 STG20264
TO 16 IF VAL Y NE 1.                                           STG20265
PTR N = Y + 0.                                                 STG20266
TO 18.                                                         STG20267
LOC 17.                                                        STG20268
PTR Y = V + H.                                                 STG20269
TO 23 IF PTR Y = J.                                            STG20270
```

```
PTR L = L + 1.                                                  STG20271
PTR I = L + 0.                                                  STG20272
STO V = I.                                                      STG20273
PTR V = V - 7.                                                  STG20274
GET Y = V.                                                      STG20275
FLG Y = 1.                                                      STG20276
STO V = Y.                                                      STG20277
PTR N = L + 0.                                                  STG20278
LOC 18.                                                         STG20279
PTR Y = N / 5.                                                  STG20280
PTR Z = Y * 5.                                                  STG20281
PTR X = N - Z.                                                  STG20282
FLG X = 0.                                                      STG20283
VAL X = PTR X.                                                  STG20284
PTR N = Y + 0.                                                  STG20285
VAL G = G + 1.                                                  STG20286
PTR 8 = 8 + 7.                                                  STG20287
STO 8 = X.                                                      STG20288
TO 18 IF PTR N NE 0.                                            STG20289
LOC 19.                                                         STG20290
GET X = 8.                                                      STG20291
PTR 8 = 8 - 7.                                                  STG20292
VAL G = G - 1.                                                  STG20293
VAL X = X + E.                                                  STG20294
PTR X = 9 - 7.                                                  STG20295
STO 9 = X.                                                      STG20296
PTR 9 = X + 0.                                                  STG20297
TO 19 IF VAL G NE 0.                                            STG20298
TO 07.                                                          STG20299
LOC 20.                                                         STG20300
GET X = X.                                                      STG20301
VAL Y = Y - 1.                                                  STG20302
TO 20 IF VAL Y NE L.                                            STG20303
TO 07 IF FLG X = 1.                                             STG20304
PTR X = 9 - 7.                                                  STG20305
TO 97 IF PTR 8 GE X.                                            STG20306
STO 9 = X.                                                      STG20307
PTR 9 = X + 0.                                                  STG20308
TO 07.                                                          STG20309
LOC 21.                                                         STG20310
STO 9 = L.                                                      STG20311
PTR K = K + 7.                                                  STG20312
PTR Y = C - 9.                                                  STG20313
PTR Y = Y / 7.                                                  STG20314
FLG Y = 0.                                                      STG20315
VAL Y = PTR Y.                                                  STG20316
PTR Y = C + 0.                                                  STG20317
STO V = Y.                                                      STG20318
PTR C = 9 - 7.                                                  STG20319
TO 05.                                                          STG20320
LOC 22.                                                         STG20321
PTR V = J + 0.                                                  STG20322
TO 16 IF PTR I = 0.                                             STG20323
TO 08 IF VAL I = 9.                                             STG20324
PTR V = V + 7.                                                  STG20325
```

```
PTR K = K + 7.                                                  STG20326
TO 32 IF VAL I = 1.                                             STG20327
TO 32 IF VAL I = 2.                                             STG20328
TO 33 IF VAL I = 3.                                             STG20329
TO 42 IF VAL I = 4.                                             STG20330
TO 36 IF VAL I = 5.                                             STG20331
TO 39 IF VAL I = 6.                                             STG20332
TO 43 IF VAL I = 7.                                             STG20333
TO 47 IF VAL I = 8.                                             STG20334
TO 23 IF VAL I NE 0.                                            STG20335
STOP.                                                           STG20336
LOC 23.                                                         STG20337
MESSAGE CONV TO 4.                                              STG20338
TO 94 BY B.                                                     STG20339
TO 07.                                                          STG20340
LOC 32.                                                         STG20341
GET X = K.                                                      STG20342
VAL W = 3 + 0.                                                  STG20343
TO 24 IF FLG X = 1.                                             STG20344
PTR K = K + 7.                                                  STG20345
VAL W = X - E.                                                  STG20346
GET X = K.                                                      STG20347
TO 24 IF FLG X = 1.                                             STG20348
REWIND W.                                                       STG20349
PTR K = K + 7.                                                  STG20350
LOC 24.                                                         STG20351
TO 31 IF VAL I = 2.                                             STG20352
STO 9 = L.                                                      STG20353
PTR X = C + 0.                                                  STG20354
TO 57 IF PTR C NE 9.                                            STG20355
PTR K = K + 7.                                                  STG20356
GET I = K.                                                      STG20357
TO 25 IF FLG I NE 1.                                            STG20358
PTR K = K - 7.                                                  STG20359
TO 23.                                                          STG20360
LOC 25.                                                         STG20361
PTR Z = VAL I.                                                  STG20362
PTR Z = Z - E.                                                  STG20363
TO 28 IF PTR Z GE 5.                                            STG20364
TO 28 IF PTR 0 GE Z.                                            STG20365
VAL X = I + 0.                                                  STG20366
PTR Z = Z * 7.                                                  STG20367
PTR Y = J + Z.                                                  STG20368
GET Y = Y.                                                      STG20369
TO 27 IF FLG Y = 3.                                             STG20370
GET Z = Y.                                                      STG20371
LOC 26.                                                         STG20372
TO 27 IF VAL Y = 0.                                             STG20373
CHAR = VAL Z.                                                   STG20374
GET Z = Z.                                                      STG20375
VAL Y = Y - 1.                                                  STG20376
PTR K = K + 7.                                                  STG20377
GET I = K.                                                      STG20378
TO 26 IF VAL I = X.                                             STG20379
TO 25.                                                          STG20380
```

```
LOC 27.                                                    STG20381
CHAR = VAL F.                                              STG20382
PTR K = K + 7.                                             STG20383
GET I = K.                                                 STG20384
TO 27 IF VAL I = X.                                        STG20385
TO 25.                                                     STG20386
LOC 28.                                                    STG20387
TO 57 IF FLG I = 1.                                        STG20388
CHAR = VAL I.                                              STG20389
PTR K = K + 7.                                             STG20390
GET I = K.                                                 STG20391
TO 25.                                                     STG20392
LOC 31.                                                    STG20393
GET I = A.                                                 STG20394
TO 29 IF PTR C = 9.                                        STG20395
GET X = C.                                                 STG20396
VAL I = X - E.                                             STG20397
STO A = I.                                                 STG20398
TO 29 IF PTR X = 9.                                        STG20399
REWIND I.                                                  STG20400
TO 98 IF FLG I NE 0.                                       STG20401
LOC 29.                                                    STG20402
GET X = V.                                                 STG20403
TO 05 IF VAL X = 0.                                        STG20404
TO 05 IF FLG X = 3.                                        STG20405
PTR Y = X + 0.                                             STG20406
READ NEXT I.                                               STG20407
TO 98 IF FLG I NE 0.                                       STG20408
LOC 30.                                                    STG20409
TO 05 IF VAL X = 0.                                        STG20410
VAL X = X - 1.                                             STG20411
GET Y = Y.                                                 STG20412
VAL Z = CHAR.                                              STG20413
TO 30 IF VAL Y = Z.                                        STG20414
WRITE NEXT W.                                              STG20415
TO 29 IF FLG W = 0.                                        STG20416
STOP.                                                      STG20417
LOC 33.                                                    STG20418
GET Y = V.                                                 STG20419
TO 23 IF FLG Y = 3.                                        STG20420
TO 05 IF VAL Y = 0.                                        STG20421
GET X = Y.                                                 STG20422
FLG B = 2.                                                 STG20423
GET W = F.                                                 STG20424
PTR Z = F + 0.                                             STG20425
TO 58 BY B.                                                STG20426
FLG B = 0.                                                 STG20427
FLG W = Y.                                                 STG20428
PTR W = U + 0.                                             STG20429
PTR Z = Y + 0.                                             STG20430
PTR V = V + 7.                                             STG20431
GET Y = V.                                                 STG20432
TO 23 IF FLG Y = 3.                                        STG20433
PTR X = Y + 0.                                             STG20434
FLG Z = 1.                                                 STG20435
```

```
VAL Z = Y + 0.                                                  STG20436
TO 35 IF FLG W NE 1.                                            STG20437
STO W = Z.                                                      STG20438
TO 05 IF VAL Y = 0.                                             STG20439
LOC 34.                                                         STG20440
GET X = X.                                                      STG20441
PTR W = Z + 0.                                                  STG20442
GET Z = W.                                                      STG20443
VAL Z = X + 0.                                                  STG20444
STO W = Z.                                                      STG20445
VAL Y = Y - 1.                                                  STG20446
TO 35 IF PTR Z = 0.                                             STG20447
TO 34 IF VAL Y NE 0.                                            STG20448
TO 05.                                                          STG20449
LOC 35.                                                         STG20450
PTR Z = 8 + 0.                                                  STG20451
STO W = Z.                                                      STG20452
PTR 8 = 8 + 7.                                                  STG20453
TO 97 IF PTR 8 GE 9.                                            STG20454
PTR W = Z + 0.                                                  STG20455
GET Z = X.                                                      STG20456
PTR X = Z + 0.                                                  STG20457
VAL Y = Y - 1.                                                  STG20458
TO 35 IF VAL Y NE L.                                            STG20459
STO W = 0.                                                      STG20460
TO 05.                                                          STG20461
LOC 36.                                                         STG20462
GET I = K.                                                      STG20463
PTR K = K + 7.                                                  STG20464
GET Y = V.                                                      STG20465
PTR V = V + 7.                                                  STG20466
GET Z = V.                                                      STG20467
TO 23 IF FLG Y = 3.                                             STG20468
TO 23 IF FLG Z = 3.                                             STG20469
PTR V = V + 7.                                                  STG20470
TO 41 IF VAL Y NE Z.                                            STG20471
TO 38 IF VAL Y = 0.                                             STG20472
PTR X = Z + 0.                                                  STG20473
LOC 37.                                                         STG20474
GET X = X.                                                      STG20475
GET Y = Y.                                                      STG20476
TO 41 IF VAL X NE Y.                                            STG20477
VAL Z = Z - 1.                                                  STG20478
TO 37 IF VAL Z NE 0.                                            STG20479
LOC 38.                                                         STG20480
TO 05 IF VAL I NE E.                                            STG20481
TO 42.                                                          STG20482
LOC 39.                                                         STG20483
GET I = K.                                                      STG20484
PTR K = K + 7.                                                  STG20485
GET Y = V.                                                      STG20486
TO 23 IF FLG Y = 3.                                             STG20487
TO 74 BY P.                                                     STG20488
PTR I = N + 0.                                                  STG20489
PTR V = J + 4.                                                  STG20490
```

```
GET Y = V.                                                    STG20491
TO 23 IF FLG Y = 3.                                           STG20492
TO 74 BY P.                                                   STG20493
PTR V = J + 4.                                                STG20494
PTR V = V + 7.                                                STG20495
PTR N = N - 1.                                                STG20496
TO 38 IF PTR N = 0.                                           STG20497
TO 40 IF PTR N GE 0.                                          STG20498
TO 05 IF VAL I = 0.                                           STG20499
TO 41.                                                        STG20500
LOC 40.                                                       STG20501
TO 05 IF VAL I = N.                                           STG20502
LOC 41.                                                       STG20503
TO 05 IF VAL I = E.                                           STG20504
LOC 42.                                                       STG20505
GET Y = V.                                                    STG20506
TO 23 IF FLG Y = 3.                                           STG20507
TO 05 IF VAL Y = 0.                                           STG20508
TO 74 BY P.                                                   STG20509
PTR M = N + 0.                                                STG20510
TO 05.                                                        STG20511
LOC 43.                                                       STG20512
PTR Y = C - 9.                                                STG20513
PTR Y = Y / 7.                                                STG20514
VAL Y = PTR Y.                                                STG20515
TO 07 IF VAL Y = 0.                                           STG20516
PTR Y = C + 0.                                                STG20517
TO 74 BY P.                                                   STG20518
FLG Y = 1.                                                    STG20519
VAL Y = 0 + 0.                                                STG20520
PTR Y = N + 1.                                                STG20521
STO C = R.                                                    STG20522
PTR Z = R + 0.                                                STG20523
PTR R = C + 0.                                                STG20524
PTR C = C - 4.                                                STG20525
STO C = K.                                                    STG20526
LOC 44.                                                       STG20527
PTR C = R + 0.                                                STG20528
PTR R = Z + 0.                                                STG20529
PTR Y = Y - 1.                                                STG20530
TO 05 IF PTR 0 GE Y.                                          STG20531
PTR R = C + 0.                                                STG20532
PTR C = C - 7.                                                STG20533
STO C = Y.                                                    STG20534
PTR C = C - 7.                                                STG20535
GET K = C.                                                    STG20536
PTR C = C - 7.                                                STG20537
TO 05.                                                        STG20538
LOC 45.                                                       STG20539
STO 9 = L.                                                    STG20540
PTR W = C - 9.                                                STG20541
PTR W = W / 7.                                                STG20542
FLG W = 0.                                                    STG20543
VAL W = PTR W.                                                STG20544
PTR W = C + 0.                                                STG20545
```

```
PTR 9 = 9 - 7.                                                  STG20546
FLG B = 2.                                                      STG20547
PTR B = 0 + 0.                                                  STG20548
FLG U = 0.                                                      STG20549
VAL U = R + 0.                                                  STG20550
PTR U = 7 + 0.                                                  STG20551
FLG Z = 1.                                                      STG20552
VAL Z = 0 + 0.                                                  STG20553
PTR Z = 0 + 0.                                                  STG20554
PTR X = 9 - 7.                                                  STG20555
LOC 46.                                                         STG20556
VAL Z = Z + 1.                                                  STG20557
STO 9 = Z.                                                      STG20558
PTR 9 = 9 - 7.                                                  STG20559
STO 9 = U.                                                      STG20560
PTR 9 = 9 - 7.                                                  STG20561
STO 9 = B.                                                      STG20562
PTR 9 = 9 - 7.                                                  STG20563
PTR K = K + 7.                                                  STG20564
GET I = K.                                                      STG20565
PTR I = X - 9.                                                  STG20566
STO 9 = I.                                                      STG20567
PTR X = 9 + 0.                                                  STG20568
PTR 9 = 9 - 7.                                                  STG20569
TO 97 IF PTR 8 GE 9.                                            STG20570
TO 46 IF FLG I NE 1.                                            STG20571
STO 9 = B.                                                      STG20572
FLG B = 0.                                                      STG20573
PTR Z = 9 + 0.                                                  STG20574
PTR 9 = 9 - 7.                                                  STG20575
VAL U = M + 0.                                                  STG20576
STO 9 = U.                                                      STG20577
PTR 9 = 9 - 7.                                                  STG20578
STO 9 = R.                                                      STG20579
PTR 9 = 9 - 7.                                                  STG20580
STO 9 = C.                                                      STG20581
PTR 9 = 9 - 7.                                                  STG20582
STO 9 = V.                                                      STG20583
PTR 9 = 9 - 7.                                                  STG20584
STO 9 = Y.                                                      STG20585
PTR R = 9 - 7.                                                  STG20586
STO R = Z.                                                      STG20587
PTR 9 = R - 7.                                                  STG20588
STO 9 = W.                                                      STG20589
PTR 9 = 9 - 4.                                                  STG20590
STO 9 = K.                                                      STG20591
PTR Z = Z - 7.                                                  STG20592
TO 48.                                                          STG20593
LOC 47.                                                         STG20594
TO 05 IF PTR R = 0.                                             STG20595
GET Z = R.                                                      STG20596
LOC 48.                                                         STG20597
PTR U = R - 7.                                                  STG20598
GET Y = U.                                                      STG20599
TO 44 IF FLG Y = 1.                                             STG20600
```

```
TO 49 IF VAL Y = 0.                                          STG20601
STO U = 0.                                                   STG20602
PTR U = U - 4.                                               STG20603
GET K = U.                                                   STG20604
PTR V = U + 0.                                               STG20605
PTR 9 = U - 7.                                               STG20606
PTR C = 9 + 0.                                               STG20607
GET X = Y.                                                   STG20608
TO 99 BY B.                                                  STG20609
PTR Y = R + 4.                                               STG20610
GET W = Y.                                                   STG20611
PTR Y = R - 4.                                               STG20612
TO 97 IF PTR 8 GE Y.                                         STG20613
GET Y = Y.                                                   STG20614
STO W = Y.                                                   STG20615
TO 05.                                                       STG20616
LOC 99.                                                      STG20617
TO 60 IF VAL Z NE 1.                                         STG20618
FLG X = 0.                                                   STG20619
VAL X = Y - 1.                                               STG20620
VAL Y = 1 + 0.                                               STG20621
PTR U = U + 7.                                               STG20622
STO U = Y.                                                   STG20623
PTR U = U + 7.                                               STG20624
STO U = X.                                                   STG20625
RETURN BY B.                                                 STG20626
LOC 49.                                                      STG20627
TO 44 IF FLG Y = 1.                                          STG20628
PTR R = R + 7.                                               STG20629
GET Y = R.                                                   STG20630
PTR R = R + 7.                                               STG20631
GET W = R.                                                   STG20632
STO W = Y.                                                   STG20633
PTR R = R + 7.                                               STG20634
GET C = R.                                                   STG20635
PTR R = R + 7.                                               STG20636
GET R = R.                                                   STG20637
TO 05.                                                       STG20638
LOC 50.                                                      STG20639
FLG Y = 1.                                                   STG20640
VAL Y = L + 0.                                               STG20641
PTR 8 = 8 - 7.                                               STG20642
TO 54.                                                       STG20643
LOC 51.                                                      STG20644
VAL I = CHAR.                                                STG20645
STO 8 = I.                                                   STG20646
TO 52 IF VAL I = C.                                          STG20647
TO 52 IF VAL I = D.                                          STG20648
VAL I = I - E.                                               STG20649
FLG Z = 3.                                                   STG20650
VAL Z = CHAR.                                                STG20651
VAL Z = Z - E.                                               STG20652
PTR Z = VAL I.                                               STG20653
STO 8 = Z.                                                   STG20654
TO 52 IF PTR 0 GE Z.                                         STG20655
```

```
TO 52 IF PTR Z GE 5.                                    STG20656
FLG Z = 2.                                              STG20657
PTR Z = Z * 7.                                          STG20658
STO 8 = Z.                                              STG20659
LOC 52.                                                 STG20660
PTR 8 = 8 + 7.                                          STG20661
TO 97 IF PTR 8 GE 9.                                    STG20662
VAL I = CHAR.                                           STG20663
STO 8 = I.                                              STG20664
TO 51 IF VAL I = D.                                     STG20665
TO 53 IF VAL I = L.                                     STG20666
TO 52 IF VAL I NE C.                                    STG20667
LOC 53.                                                 STG20668
PTR Y = 8 + 0.                                          STG20669
STO U = Y.                                              STG20670
PTR U = 8 + 0.                                          STG20671
LOC 54.                                                 STG20672
GET I = A.                                              STG20673
READ NEXT I.                                            STG20674
TO 98 IF FLG I NE 0.                                    STG20675
VAL I = CHAR.                                           STG20676
PTR I = 0 + 0.                                          STG20677
PTR 8 = 8 + 7.                                          STG20678
STO 8 = I.                                              STG20679
TO 51 IF VAL I = D.                                     STG20680
TO 52 IF VAL I NE C.                                    STG20681
PTR Y = 8 + 0.                                          STG20682
STO U = Y.                                              STG20683
STO 8 = 1.                                              STG20684
PTR 8 = 8 + 7.                                          STG20685
TO 97 IF PTR 8 GE 9.                                    STG20686
VAL I = CHAR.                                           STG20687
TO 55 IF VAL I NE C.                                    STG20688
FLG B = 0.                                              STG20689
LOC 55.                                                 STG20690
RETURN BY D.                                            STG20691
LOC 56.                                                 STG20692
VAL W = 3 + 0.                                          STG20693
PTR X = C + 0.                                          STG20694
LOC 57.                                                 STG20695
GET X = X.                                              STG20696
CHAR = VAL X.                                           STG20697
TO 57 IF FLG X NE 1.                                    STG20698
WRITE NEXT W.                                           STG20699
TO 98 IF FLG W NE 0.                                    STG20700
TO 55 IF VAL X = L.                                     STG20701
CHAR = VAL X.                                           STG20702
TO 57.                                                  STG20703
LOC 58.                                                 STG20704
PTR Z = W + Z.                                          STG20705
TO 60 IF PTR W NE 0.                                    STG20706
TO 71 IF FLG B = 2.                                     STG20707
LOC 59.                                                 STG20708
TO 70 IF PTR V GE 9.                                    STG20709
GET Z = V.                                              STG20710
```

```
GET Y = @.                                                    STG20711
GET X = Y.                                                    STG20712
TO 63 IF FLG Z = 2.                                           STG20713
TO 64 IF FLG Z = 3.                                           STG20714
PTR V = @ + 7.                                                STG20715
PTR @ = V + 7.                                                STG20716
LOC 60.                                                       STG20717
GET W = Z.                                                    STG20718
TO 69 IF FLG W = 1.                                           STG20719
TO 62 IF FLG W = 2.                                           STG20720
TO 58 IF VAL Y = 0.                                           STG20721
TO 58 IF VAL X NE W.                                          STG20722
TO 61 IF PTR W = 0.                                           STG20723
TO 61 IF FLG X = 3.                                           STG20724
TO 61 IF FLG B = 2.                                           STG20725
PTR @ = V - 7.                                                STG20726
PTR V = @ - 7.                                                STG20727
TO 97 IF PTR 8 GE V.                                          STG20728
STO @ = Y.                                                    STG20729
PTR W = W + Z.                                                STG20730
STO V = W.                                                    STG20731
LOC 61.                                                       STG20732
VAL Y = Y - 1.                                                STG20733
PTR Y = X + 0.                                                STG20734
GET X = X.                                                    STG20735
PTR Z = Z + 7.                                                STG20736
TO 60.                                                        STG20737
LOC 62.                                                       STG20738
TO 61 IF FLG X = 2.                                           STG20739
TO 58 IF FLG B = 2.                                           STG20740
PTR @ = V - 7.                                                STG20741
PTR V = @ - 7.                                                STG20742
TO 97 IF PTR 8 GE V.                                          STG20743
STO @ = Y.                                                    STG20744
FLG Z = 2.                                                    STG20745
STO V = Z.                                                    STG20746
FLG X = 3.                                                    STG20747
TO 58.                                                        STG20748
LOC 63.                                                       STG20749
FLG Z = 3.                                                    STG20750
PTR Z = Z + 7.                                                STG20751
STO V = Z.                                                    STG20752
PTR U = U + 7.                                                STG20753
FLG W = 0.                                                    STG20754
VAL W = 0 + 0.                                                STG20755
PTR W = Y + 0.                                                STG20756
STO U = W.                                                    STG20757
TO 60.                                                        STG20758
LOC 64.                                                       STG20759
TO 68 IF VAL Y = 0.                                           STG20760
TO 68 IF VAL X = R.                                           STG20761
GET W = U.                                                    STG20762
VAL W = W + 1.                                                STG20763
VAL Y = Y - 1.                                                STG20764
PTR Y = X + 0.                                                STG20765
```

```
TO 67 IF VAL X NE M.                                    STG20766
VAL Z = 0 + 0.                                          STG20767
LOC 65.                                                 STG20768
VAL Z = Z + 1.                                          STG20769
LOC 66.                                                 STG20770
TO 68 IF VAL Y = 0.                                     STG20771
GET X = X.                                              STG20772
VAL Y = Y - 1.                                          STG20773
PTR Y = X + 0.                                          STG20774
VAL W = W + 1.                                          STG20775
TO 65 IF VAL X = M.                                     STG20776
TO 66 IF VAL X NE R.                                    STG20777
VAL Z = Z - 1.                                          STG20778
TO 66 IF VAL Z NE 0.                                    STG20779
LOC 67.                                                 STG20780
GET X = X.                                              STG20781
STO @ = Y.                                              STG20782
STO U = W.                                              STG20783
TO 60.                                                  STG20784
LOC 68.                                                 STG20785
STO U = 3.                                              STG20786
PTR U = U - 7.                                          STG20787
PTR V = @ + 7.                                          STG20788
PTR @ = V + 7.                                          STG20789
TO 59.                                                  STG20790
LOC 69.                                                 STG20791
TO 58 IF VAL Y NE 0.                                    STG20792
PTR U = Z + 7.                                          STG20793
GET Y = U.                                              STG20794
LOC 70.                                                 STG20795
RETURN BY B.                                            STG20796
LOC 71.                                                 STG20797
PTR W = 8 - Z.                                          STG20798
STO Z = W.                                              STG20799
TO 73 IF VAL Y = 0.                                     STG20800
LOC 72.                                                 STG20801
VAL Y = Y - 1.                                          STG20802
PTR Y = X + 0.                                          STG20803
PTR X = 0 + 0.                                          STG20804
STO 8 = X.                                              STG20805
PTR 8 = 8 + 7.                                          STG20806
TO 97 IF PTR 8 GE 9.                                    STG20807
GET X = Y.                                              STG20808
TO 72 IF VAL Y NE 0.                                    STG20809
LOC 73.                                                 STG20810
FLG X = 1.                                              STG20811
PTR X = 0 + 0.                                          STG20812
STO 8 = X.                                              STG20813
PTR U = 8 + 7.                                          STG20814
FLG Y = 0.                                              STG20815
PTR Y = U + 0.                                          STG20816
STO U = Y.                                              STG20817
PTR 8 = U + 7.                                          STG20818
TO 97 IF PTR 8 GE 9.                                    STG20819
RETURN BY B.                                            STG20820
```

```
LOC 74.                                                    STG20821
PTR O = 9 + 0.                                             STG20822
VAL S = Y + 0.                                             STG20823
PTR S = Y + 0.                                             STG20824
PTR T = 0 + 0.                                             STG20825
TO 75 IF VAL Y NE 0.                                       STG20826
PTR N = 0 + 0.                                             STG20827
RETURN BY P.                                               STG20828
LOC 75.                                                    STG20829
VAL T = M + 0.                                             STG20830
LOC 76.                                                    STG20831
TO 93 IF VAL S = 0.                                        STG20832
GET X = S.                                                 STG20833
PTR Y = S + 0.                                             STG20834
VAL Y = 0 + 0.                                             STG20835
TO 77 IF VAL X NE M.                                       STG20836
STO 9 = T.                                                 STG20837
PTR 9 = 9 - 7.                                             STG20838
TO 97 IF PTR 8 GE 9.                                       STG20839
VAL S = S - 1.                                             STG20840
PTR S = X + 0.                                             STG20841
TO 75.                                                     STG20842
LOC 77.                                                    STG20843
TO 78 IF VAL X = N.                                        STG20844
TO 78 IF VAL X = O.                                        STG20845
TO 78 IF VAL X = P.                                        STG20846
TO 78 IF VAL X = @.                                        STG20847
TO 78 IF VAL X = R.                                        STG20848
VAL Y = Y + 1.                                             STG20849
GET X = X.                                                 STG20850
TO 77 IF VAL S NE Y.                                       STG20851
VAL X = R + 0.                                             STG20852
VAL S = S + 1.                                             STG20853
LOC 78.                                                    STG20854
VAL J = X + 0.                                             STG20855
PTR N = 0 + 0.                                             STG20856
VAL S = S - Y.                                             STG20857
VAL S = S - 1.                                             STG20858
PTR S = X + 0.                                             STG20859
TO 83 IF VAL Y = 0.                                        STG20860
GET X = Y.                                                 STG20861
PTR U = VAL X.                                             STG20862
PTR U = U - E.                                             STG20863
TO 79 IF PTR U GE 5.                                       STG20864
TO 81 IF PTR U GE 0.                                       STG20865
LOC 79.                                                    STG20866
PTR V = 9 + 7.                                             STG20867
GET W = F.                                                 STG20868
FLG Y = 0.                                                 STG20869
PTR Z = F + 0.                                             STG20870
TO 58 BY B.                                                STG20871
TO 83 IF FLG Y NE 1.                                       STG20872
TO 83 IF VAL Y = 0.                                        STG20873
GET X = Y.                                                 STG20874
FLG N = 1.                                                 STG20875
```

```
TO 82 IF VAL X = 0.                                          STG20876
FLG N = 0.                                                   STG20877
PTR X = Y + 0.                                               STG20878
LOC 80.                                                      STG20879
GET X = X.                                                   STG20880
PTR U = VAL X.                                               STG20881
PTR U = U - E.                                               STG20882
TO 81 IF PTR U = 0.                                          STG20883
TO 93 IF PTR U GE 5.                                         STG20884
TO 93 IF PTR 0 GE U.                                         STG20885
LOC 81.                                                      STG20886
PTR N = N * 5.                                               STG20887
PTR N = N + U.                                               STG20888
LOC 82.                                                      STG20889
VAL Y = Y - 1.                                               STG20890
TO 80 IF VAL Y NE 0.                                         STG20891
TO 83 IF FLG N = 0.                                          STG20892
FLG N = 0.                                                   STG20893
PTR N = 0 - N.                                               STG20894
LOC 83.                                                      STG20895
TO 92 IF VAL J = R.                                          STG20896
TO 90 IF VAL T = M.                                          STG20897
TO 89 IF VAL J = P.                                          STG20898
TO 89 IF VAL J = @.                                          STG20899
LOC 84.                                                      STG20900
TO 87 IF VAL T = @.                                          STG20901
TO 86 IF VAL T = P.                                          STG20902
TO 85 IF VAL T = 0.                                          STG20903
PTR T = T + N.                                               STG20904
TO 88.                                                       STG20905
LOC 85.                                                      STG20906
PTR T = T - N.                                               STG20907
TO 88.                                                       STG20908
LOC 86.                                                      STG20909
PTR T = T * N.                                               STG20910
TO 88.                                                       STG20911
LOC 87.                                                      STG20912
PTR T = T / N.                                               STG20913
LOC 88.                                                      STG20914
VAL T = J + 0.                                               STG20915
TO 76 IF VAL J NE R.                                         STG20916
PTR N = T + 0.                                               STG20917
PTR 9 = 9 + 7.                                               STG20918
GET T = 9.                                                   STG20919
TO 92.                                                       STG20920
LOC 89.                                                      STG20921
TO 86 IF VAL T = P.                                          STG20922
TO 87 IF VAL T = @.                                          STG20923
LOC 90.                                                      STG20924
STO 9 = T.                                                   STG20925
PTR 9 = 9 - 7.                                               STG20926
TO 97 IF PTR 8 GE 9.                                         STG20927
VAL T = J + 0.                                               STG20928
PTR T = N + 0.                                               STG20929
TO 76.                                                       STG20930
```

```
LOC 91.                                                    STG20931
TO 93 IF VAL S NE 0.                                       STG20932
RETURN BY P.                                               STG20933
LOC 92.                                                    STG20934
TO 84 IF VAL T NE M.                                       STG20935
TO 91 IF PTR 9 = 0.                                        STG20936
PTR 9 = 9 + 7.                                             STG20937
GET T = 9.                                                 STG20938
TO 92 IF VAL S = 0.                                        STG20939
GET X = S.                                                 STG20940
VAL S = S - 1.                                             STG20941
PTR S = X + 0.                                             STG20942
VAL J = X + 0.                                             STG20943
TO 92 IF VAL J = R.                                        STG20944
TO 83 IF VAL J = N.                                        STG20945
TO 83 IF VAL J = O.                                        STG20946
TO 83 IF VAL J = P.                                        STG20947
TO 83 IF VAL J = Q.                                        STG20948
LOC 93.                                                    STG20949
MESSAGE EXPR TO 4.                                         STG20950
PTR N = 0 + 0.                                             STG20951
PTR 9 = 0 + 0.                                             STG20952
TO 94 BY B.                                                STG20953
RETURN BY P.                                               STG20954
LOC 94.                                                    STG20955
PTR X = C + 0.                                             STG20956
PTR Y = J + 0.                                             STG20957
TO 96 IF PTR 9 GE C.                                       STG20958
STO 9 = L.                                                 STG20959
LOC 95.                                                    STG20960
GET X = X.                                                 STG20961
CHAR = VAL X.                                              STG20962
TO 95 IF FLG X = 0.                                        STG20963
WRITE NEXT 4.                                              STG20964
TO 98 IF FLG 4 NE 0.                                       STG20965
TO 96 IF VAL X = L.                                        STG20966
CHAR = VAL X.                                              STG20967
TO 95.                                                     STG20968
LOC 96.                                                    STG20969
TO 70 IF PTR Y = 0.                                        STG20970
PTR Y = Y - H.                                             STG20971
PTR X = Y - 7.                                             STG20972
GET Y = Y.                                                 STG20973
GET X = X.                                                 STG20974
TO 95.                                                     STG20975
LOC 97.                                                    STG20976
MESSAGE FULL TO 4.                                         STG20977
TO 94 BY B.                                                STG20978
STOP.                                                      STG20979
LOC 98.                                                    STG20980
MESSAGE IOCH TO 4.                                         STG20981
TO 94 BY B.                                                STG20982
STOP.                                                      STG20983
END PROGRAM.                                               STG20984
```

ST2T

```
.'@'0 (+-*/)                                                    ST2T0001
END PROGRAM.                                                    ST2T0002
4'F7@                                                           ST2T0003
THIS LINE REPEATS 4 TIMES UNDER COUNT CONTROL'F1@               ST2T0004
                                                                ST2T0005
'F8@                                                            ST2T0006
END OF STAGE2 TEST.  THIS LINE CONTAINS TYPE 0 ELEMENTS ONLY@   ST2T0007
'F0@                                                            ST2T0008
@                                                               ST2T0009
TRY USING AN UNDEFINED PARAMETER.                               ST2T0010
THE UNDEFINED PARAMETER IS '10'F1@                              ST2T0011
'F1@                                                            ST2T0012
UNDEFINED PARAMETER IN FORMATTED OUTPUT 111111111@              ST2T0013
@                                                               ST2T0014
TYPE 2 '.                                                       ST2T0015
TEST CHARACTER IS '10@                                          ST2T0016
TRANSFORMATION 1 YIELDS '11@                                    ST2T0017
TRANSFORMATION 2 YIELDS '12@                                    ST2T0018
TRANSFORMATION 3 YIELDS '13@                                    ST2T0019
TRANSFORMATION 4 YIELDS '14@                                    ST2T0020
TRANSFORMATION 5 YIELDS '15@                                    ST2T0021
PARAMETER 2 WAS SET TO '10'26 '20@                              ST2T0022
'10'37.@                                                        ST2T0023
ITERATION ON '10. NEXT MEMBER '30@                              ST2T0024
'F8@                                                            ST2T0025
TRANSFORMATION 8 YIELDS '18@                                    ST2T0026
@                                                               ST2T0027
TEST PARAMETER 0.                                               ST2T0028
'00 IS GENERATED SYMBOL 1@                                      ST2T0029
'01 IS GENERATED SYMBOL 2@                                      ST2T0030
'02 IS GENERATED SYMBOL 3@                                      ST2T0031
'03 IS GENERATED SYMBOL 4@                                      ST2T0032
'04 IS GENERATED SYMBOL 5@                                      ST2T0033
'05 IS GENERATED SYMBOL 6@                                      ST2T0034
'06 IS GENERATED SYMBOL 7@                                      ST2T0035
'07 IS GENERATED SYMBOL 8@                                      ST2T0036
'08 IS GENERATED SYMBOL 9@                                      ST2T0037
'09 IS GENERATED SYMBOL 10@                                     ST2T0038
'00 IS GENERATED SYMBOL 1@                                      ST2T0039
'01 IS GENERATED SYMBOL 2@                                      ST2T0040
'02 IS GENERATED SYMBOL 3@                                      ST2T0041
'03 IS GENERATED SYMBOL 4@                                      ST2T0042
'04 IS GENERATED SYMBOL 5@                                      ST2T0043
'05 IS GENERATED SYMBOL 6@                                      ST2T0044
'06 IS GENERATED SYMBOL 7@                                      ST2T0045
'07 IS GENERATED SYMBOL 8@                                      ST2T0046
'08 IS GENERATED SYMBOL 9@                                      ST2T0047
'09 IS GENERATED SYMBOL 10@                                     ST2T0048
@                                                               ST2T0049
NULL MACRO.                                                     ST2T0050
@
```

```
NINE PARAMETERS ' ' ' ' ' ' ' ' ' '.                                    ST2T0051
NINE PARAMETERS ARE '10 '20 '30 '40 '50 '60 '70 '80 '90.'F1@           ST2T0052
@                                                                       ST2T0053
CODE BODY LINE WITH NO TERMINATOR.                                      ST2T0054
THIS CODE BODY LINE HAS NO TERMINATOR.  IT SHOULD HAVE 80 CHARACTERS.   ST2T0055
@@                                                                      ST2T0056
C  THIS DATA EXERCISES MOST OF THE FEATURES OF STAGE 2.  THE FIRST TWO  ST2T0057
C  LINES WILL NOT BE MATCHED, AND SHOULD BE PRINTED OUT UNCHANGED.      ST2T0058
TEST PARAMETER 0.                                                       ST2T0059
NULL MACRO.                                                             ST2T0060
NINE PARAMETERS 1 2 3 4 5 6 7 8 9.                                      ST2T0061
TYPE 2 A.                                                               ST2T0062
CODE BODY LINE WITH NO TERMINATOR.                                      ST2T0063
TYPE 2 0.                                                               ST2T0064
TYPE 2 13.                                                              ST2T0065
TYPE 2 A.                                                               ST2T0066
TYPE 2 A+13.                                                            ST2T0067
TYPE 2 15/2.                                                            ST2T0068
TYPE 2 5*(A+13).                                                        ST2T0069
TEST PARAMETER 0.                                                       ST2T0070
TYPE 2 THIS IS A LONG STRING.                                           ST2T0071
TYPE 2 L1,L2,(L31,L32,L33),L4,L5.                                       ST2T0072
TYPE 2 (PL1,PL2,PL3,(PL41,PL42,PL43),PL5,PL6,PL7).                      ST2T0073
TRY USING AN UNDEFINED PARAMETER.                                       ST2T0074
END PROGRAM.                                                            ST2T0075
```

BASM

```
.'@'0 (+-*/)                                                    ST2M0001
' ' = ' + 0.                                                    ST2M0002
'10 '20 = '30@                                                  ST2M0003
@                                                               ST2M0004
' ' = '.                                                        ST2M0005
ZFET '10 '30@                                                   ST2M0006
ZSTO '10 '20@                                                   ST2M0007
@                                                               ST2M0008
' ' = ' '.                                                      ST2M1008
ZFET '30 '40@                                                   ST2M2008
ZSTO '10 '20@                                                   ST2M3008
@                                                               ST2M4008
' ' = ' ' '.                                                    ST2M0009
ZFET '10 '30@                                                   ST2M0010
Z'40 '10 '50@                                                   ST2M0011
ZSTO '10 '20@                                                   ST2M0012
@                                                               ST2M0013
Z' ' '.                                                         ST2M0014
'10DR J'20'30@                                                  ST2M0015
@                                                               ST2M0016
'DR '.                                                          ST2M0017
'10'20'26 .CODE'16 '11'20'26 'F3@                               ST2M0018
@                                                               ST2M0019
FET' '.                                                         ST2M0020
.CODE'16 'F3@                                                   ST2M0021
@                                                               ST2M0022
STODR '.                                                        ST2M0023
.CODE'96@                                                       ST2M0024
      '10='91'F1@                                               ST2M0025
@                                                               ST2M0026
TO ' IF ' ' ' '.                                                ST2M0027
ZFET '20 '30@                                                   ST2M0028
'20'50'56@                                                      ST2M0029
ZJ'40,'12,J'50@                                                 ST2M0030
@                                                               ST2M0031
TO ' IF ' ' ' ='.                                               ST2M0032
ZFET '20 '30@                                                   ST2M0033
'58'56@                                                         ST2M0034
ZJ'40,'12,'50@                                                  ST2M0035
@                                                               ST2M0036
TO ' IF ' ' ' = .                                               ST2M1036
ZFET '20 '30@                                                   ST2M2036
ZJ'40,'12,0@                                                    ST2M3036
@                                                               ST2M4036
ZJ',',',.                                                       ST2M0037
.CODE'96@                                                       ST2M0038
IF ('10) NE (=) SKIP 1@                                         ST2M0039
E@'16@                                                          ST2M0040
     IF ('91,'10,'30) GO TO '20'F1@                             ST2M0041
@                                                               ST2M0042
```

```
IF ' NE ' SKIP '.                                                    ST2M0043
'F51@                                                                ST2M0044
@                                                                    ST2M0045
LOC '.                                                               ST2M0046
'12'16@                                                              ST2M0047
'F1@                                                                 ST2M0048
 1111 CONTINUE@                                                      ST2M0049
@                                                                    ST2M0050
TO '.                                                                ST2M0051
      GO TO '12'F1@                                                  ST2M0052
@                                                                    ST2M0053
TO ' BY '.                                                           ST2M0054
.CON'86@                                                             ST2M0055
      JPTR'20='84-199'F1@                                            ST2M0056
      GO TO '12'F1@                                                  ST2M0057
'84'86@                                                              ST2M0058
'F1@                                                                 ST2M0059
 8888 CONTINUE@                                                      ST2M0060
SET .CON TO .CON+1@                                                  ST2M0061
@                                                                    ST2M1061
RETURN BY '.                                                         ST2M2061
      GO TO (200,201,202,203,204,205,206,207,208,209,210,211,212,'F1@ ST2M3061
     1213,214,215,216,217,218), JPTR'10'F1@                          ST2M4061
@                                                                    ST2M5061
VAL ' = CHAR.                                                        ST2M0062
      JVAL'10=LB(LBR)'F1@                                            ST2M0063
      LBR=LBR+1'F1@                                                  ST2M0064
@                                                                    ST2M0065
CHAR = VAL '.                                                        ST2M0066
      CALL IWRCH(JVAL'10,LB,LBW,JFLG'10,LBL)'F1@                     ST2M0067
@                                                                    ST2M0068
READ NEXT '.                                                         ST2M0069
      JFLG'10=IOOP(-1,JVAL'10,LB,1,LBL)'F1@                          ST2M0070
      LB(LBL)=-1'F1@                                                 ST2M0071
      LBR=1'F1@                                                      ST2M0072
@                                                                    ST2M0073
WRITE NEXT '.                                                        ST2M0074
      JFLG'10=IOOP(1,JVAL'10,LB,1,LBL)'F1@                           ST2M0075
      LBW=1'F1@                                                      ST2M0076
@                                                                    ST2M0077
REWIND '.                                                            ST2M0078
      JFLG'10=IOOP(0,JVAL'10,LB,1,1)'F1@                             ST2M0079
      JFLG'10=0'F1@                                                  ST2M0080
@                                                                    ST2M0081
GET ' = '.                                                           ST2M0082
      JFLG'10=L(JPTR'20)'F1@                                         ST2M0083
      JVAL'10=L(JPTR'20+1)'F1@                                       ST2M0084
      JPTR'10=L(JPTR'20+2)'F1@                                       ST2M0085
@                                                                    ST2M0086
STO ' = '.                                                           ST2M0087
      L(JPTR'10)=JFLG'20'F1@                                         ST2M0088
      L(JPTR'10+1)=JVAL'20'F1@                                       ST2M0089
      L(JPTR'10+2)=JPTR'20'F1@                                       ST2M0090
@                                                                    ST2M0091
' EQU '.                                                             ST2M1091
```

```
.'10'16 'F3@                                                    ST2M2091
@                                                               ST2M3091
SET ' TO '.                                                     ST2M4091
'24'26 'F3@                                                     ST2M5091
@                                                               ST2M6091
MESSAGE ' TO '.                                                 ST2M7091
MES E@U 11@                                                     ST2M8091
.MES'56@                                                        ST2M9091
'10'17@                                                         ST2MA091
      MB('54)='18'F1@                                           ST2MB091
SET .MES TO .MES+1@                                             ST2MC091
'F8@                                                            ST2MD091
      JFLG'20=IOOP(1,JVAL'20,MB,1,21)'F1@                       ST2ME091
@                                                               ST2MF091
STOP.                                                           ST2M0092
      RETURN'F1@                                                ST2M0093
@                                                               ST2M0094
END PROGRAM.                                                    ST2M0095
      END'F1@                                                   ST2M0096
'F0@                                                            ST2M0097
@@                                                              ST2M0098
      SUBROUTINE PROGR                                          ST2M0099
      DIMENSION LB(81),MB(20),L(30000)                          ST2M0100
      COMMON MAXCH,JCHAN(36)                                    ST2M0101
      JPTR9=30000                                               ST2M0102
      JFLG0=0                                                   ST2M0103
      JFLG1=1                                                   ST2M0104
      JFLG2=2                                                   ST2M0105
      JFLG3=3                                                   ST2M0106
      JVAL0=0                                                   ST2M0107
      JVAL1=1                                                   ST2M0108
      JVAL2=2                                                   ST2M0109
      JVAL3=3                                                   ST2M0110
      JVAL4=4                                                   ST2M0111
      JVAL5=5                                                   ST2M0112
      JVAL6=6                                                   ST2M0113
      JVAL7=7                                                   ST2M0114
      JVAL8=8                                                   ST2M0115
      JVAL9=9                                                   ST2M0116
      JPTR0=0                                                   ST2M0117
      JPTR1=1                                                   ST2M0118
      JPTR2=2                                                   ST2M0119
      JPTR3=3                                                   ST2M0120
      JPTR5=10                                                  ST2M0121
      JPTR7=3                                                   ST2M0122
      JPTR8=1                                                   ST2M0123
      LBL=1                                                     ST2M0124
      LBR=1                                                     ST2M0125
      LBW=1                                                     ST2M0126
      JPTR9=JPTR8+(JPTR9/JPTR7-1)*JPTR7                         ST2M0127
      DO 1000 J=1,9                                             ST2M0128
 1000 MB(J)=46                                                  ST2M0129
      MB(10)=0                                                  ST2M0130
      MB(15)=0                                                  ST2M0131
      MB(16)=5                                                  ST2M0132
```

```
      MB(17)=18                                                  ST2M0133
      MB(18)=18                                                  ST2M0134
      MB(19)=15                                                  ST2M0135
      MB(20)=18                                                  ST2M0136
CON EQU 200.                                                     ST2M0137
```

WSED

```
204STG20087    ***EDIT***                                    WSED0001
205WSED0003    ***EDIT***                                    WSED0002
LOC SETSCAN.                                                 WSED0003
204STG20689    ***EDIT***                                    WSED0004
205WSED0105    ***EDIT***                                    WSED0005
TO WISPLIN BY D.                                             WSED0006
LOC WISPLIN.                                                 WSED0007
GET I = A.                                                   WSED0008
READ NEXT I.                                                 WSED0009
TO 98 IF FLG I NE 0.                                         WSED0010
WRITE NEXT 4.                                                WSED0011
PTR X = 5 - 1.                                               WSED0012
PTR X = X * 7.                                               WSED0013
PTR X = X + 8.                                               WSED0014
FLG Y = 0.                                                   WSED0015
VAL Y = 0.                                                   WSED0016
PTR Y = C.                                                   WSED0017
VAL Z = 0.                                                   WSED0018
PTR Z = 8 + 7.                                               WSED0019
PTR 9 = C.                                                   WSED0020
VAL I = CHAR.                                                WSED0021
TO COMMA IF VAL I = = .                                      WSED0022
LOC STOCH.                                                   WSED0023
PTR I = 9 - 7.                                               WSED0024
STO 9 = I.                                                   WSED0025
TO SKIPSP IF VAL I = = .                                     WSED0026
TO COMMA IF VAL I = =,.                                      WSED0027
TO PERIOD IF VAL I = A.                                      WSED0028
TO LINEND IF VAL I = L.                                      WSED0029
LOC ADVPT.                                                   WSED0030
PTR 9 = I.                                                   WSED0031
VAL Y = Y + 1.                                               WSED0032
TO GETCH IF VAL I NE ='.                                     WSED0033
VAL I = CHAR.                                                WSED0034
TO PERIOD IF VAL I = L.                                      WSED0035
PTR I = VAL I.                                               WSED0036
PTR W = Z.                                                   WSED0037
PTR U = I / 5.                                               WSED0038
PTR V = U * 5.                                               WSED0039
PTR V = I - V.                                               WSED0040
PTR I = U.                                                   WSED0041
STO W = V.                                                   WSED0042
PTR W = W + 7.                                               WSED0043
VAL Y = Y + 1.                                               WSED0044
LOC CVTCH.                                                   WSED0045
PTR U = I / 5.                                               WSED0046
PTR V = U * 5.                                               WSED0047
PTR V = I - V.                                               WSED0048
PTR I = U.                                                   WSED0049
STO W = V.                                                   WSED0050
PTR W = W + 7.                                               WSED0051
```

```
VAL Y = Y + 1.                                                    WSED0052
TO CVTCH IF PTR I NE 0.                                           WSED0053
LOC MOVECH.                                                       WSED0054
PTR W = W - 7.                                                    WSED0055
GET U = W.                                                        WSED0056
VAL I = PTR U.                                                    WSED0057
VAL I = I + E.                                                    WSED0058
PTR I = 9 - 7.                                                    WSED0059
STO 9 = I.                                                        WSED0060
PTR 9 = I.                                                        WSED0061
TO MOVECH IF PTR W NE Z.                                          WSED0062
LOC GETCH.                                                        WSED0063
VAL I = CHAR.                                                     WSED0064
TO STOCH.                                                         WSED0065
LOC SKIPSP.                                                       WSED0066
VAL I = CHAR.                                                     WSED0067
TO SKIPSP IF VAL I = = .                                          WSED0068
TO SCOMMA IF VAL I = =,.                                          WSED0069
TO PERIOD IF VAL I = A.                                           WSED0070
TO PERIOD IF VAL I = L.                                           WSED0071
PTR 9 = I.                                                        WSED0072
VAL Y = Y + 1.                                                    WSED0073
PTR I = 9 - 7.                                                    WSED0074
STO 9 = I.                                                        WSED0075
TO ADVPT.                                                         WSED0076
LOC SCOMMA.                                                       WSED0077
STO 9 = I.                                                        WSED0078
LOC COMMA.                                                        WSED0079
STO Z = Y.                                                        WSED0080
PTR Z = Z + 7.                                                    WSED0081
VAL Y = 0.                                                        WSED0082
PTR Y = 9.                                                        WSED0083
LOC ISPACE.                                                       WSED0084
VAL I = CHAR.                                                     WSED0085
TO ISPACE IF VAL I = = .                                          WSED0086
TO STOCH.                                                         WSED0087
LOC PERIOD.                                                       WSED0088
STO 9 = L.                                                        WSED0089
LOC LINEND.                                                       WSED0090
STO Z = Y.                                                        WSED0091
VAL Y = 0.                                                        WSED0092
TO MOVE IF PTR Z GE X.                                            WSED0093
LOC STNULL.                                                       WSED0094
PTR Z = Z + 7.                                                    WSED0095
STO Z = Y.                                                        WSED0096
TO STNULL IF PTR Z NE X.                                          WSED0097
LOC MOVE.                                                         WSED0098
GET X = Z.                                                        WSED0099
PTR 9 = 9 - 7.                                                    WSED0100
STO 9 = X.                                                        WSED0101
PTR Z = Z - 7.                                                    WSED0102
TO MOVE IF PTR Z NE 8.                                            WSED0103
PTR V = 9.                                                        WSED0104
TO SETSCAN.                                                       WSED0105
204STG20983    ***EDIT***                                         WSED0106
```

WSPM

```
.@@'0 (+-*/)                                                                    WSPM0001
- @ @ @.                                                                        WSPM0002
ERR '20 '30 ILLEGAL SYMBOL@                                                     WSPM0003
@                                                                               WSPM0004
+ @ @.                                                                          WSPM0005
IF '25 GT 1 SKIP 2@                                                             WSPM0006
- '10DR '20@                                                                    WSPM0007
'F9@                                                                            WSPM0008
IF '10 NE STO SKIP 2@                                                           WSPM0009
ERR '20 IS IMMEDIATE AND CANNOT BE SET@                                         WSPM0010
'F9@                                                                            WSPM0011
IF '10 NE INC SKIP 2@                                                           WSPM0012
ERR '20 IS IMMEDIATE AND CANNOT BE INCREMENTED@                                 WSPM0013
'F9@                                                                            WSPM0014
- '10IM '20@                                                                    WSPM0015
@                                                                               WSPM0016
+ @ @ @.                                                                        WSPM0017
ERR '20 '30 ILLEGAL OPERAND@                                                    WSPM0018
@                                                                               WSPM0019
+ STO '@.                                                                       WSPM0020
ERR ''''10 IS IMMEDIATE AND CANNOT BE SET@                                      WSPM0021
@                                                                               WSPM0022
+ INC '@.                                                                       WSPM0023
ERR ''''10 IS IMMEDIATE AND CANNOT BE INCREMENTED@                              WSPM0024
@                                                                               WSPM0025
+ D@ (@).                                                                       WSPM0026
ERR ('20) ILLEGAL IN ELEMENT DECLARATION BECAUSE IT IS NOT IMMEDIATE@           WSPM0027
@                                                                               WSPM0028
+ @ (@).                                                                        WSPM0029
- '10DR '20@                                                                    WSPM0030
@                                                                               WSPM0031
+ D@ AF @.                                                                      WSPM0032
ERR AF '20 ILLEGAL IN ELEMENT DECLARATION BECAUSE IT IS NOT IMMEDIATE@          WSPM0033
@                                                                               WSPM0034
+ IOP AF @.                                                                     WSPM0035
ERR AF '20 ILLEGAL AS THE SECOND OPERAND OF AN I/O STATEMENT@                   WSPM0036
@                                                                               WSPM0037
+ J@ AF @.                                                                      WSPM0038
ERR AF '20 ILLEGAL JUMP TARGET@                                                 WSPM0039
@                                                                               WSPM0040
+ @ AF @.                                                                       WSPM0041
- '10AF '20@                                                                    WSPM0042
@                                                                               WSPM0043
+ D@ BPF @.                                                                     WSPM0044
ERR BPF '20 ILLEGAL IN ELEMENT DECLARATIONS BECAUSE IT IS NOT IMMEDIATE@        WSPM0045
@                                                                               WSPM0046
+ IOP BPF @.                                                                    WSPM0047
ERR BPF '20 ILLEGAL AS THE SECOND OPERAND OF AN I/O STATEMENT@                  WSPM0048
@                                                                               WSPM0049
+ J@ BPF @.                                                                     WSPM0050
```

```
ERR BPF '20 ILLEGAL JUMP TARGET@                                                WSPM0051
@                                                                               WSPM0052
+ @ BPF @.                                                                      WSPM0053
- '10BP '20@                                                                    WSPM0054
@                                                                               WSPM0055
+ D@ CAR @.                                                                     WSPM0056
ERR CAR '20 ILLEGAL IN ELEMENT DECLARATIONS BECAUSE IT IS NOT IMMEDIATE@        WSPM0051
@                                                                               WSPM0058
+ @ CAR @.                                                                      WSPM0059
- '10CA '20@                                                                    WSPM0060
@                                                                               WSPM0061
+ D@ CDR @.                                                                     WSPM0062
ERR CDR '20 ILLEGAL IN ELEMENT DECLARATIONS BECAUSE IT IS NOT IMMEDIATE@        WSPM0063
@                                                                               WSPM0064
+ @ CDR @.                                                                      WSPM0065
- '10CD '20@                                                                    WSPM0066
@                                                                               WSPM0067
+ L@ @ @.                                                                       WSPM0068
ERR '20 '30 ILLEGAL LOCATION SYMBOL@                                            WSPM0069
@                                                                               WSPM0070
+ L@ @.                                                                         WSPM0071
IF '25 = 1 SKIP 2@                                                              WSPM0072
'20'3701234567890@                                                              WSPM0073
IF '30 NE  SKIP 2@                                                              WSPM0074
ERR '10 ILLEGAL LOCATION SYMBOL@                                                WSPM0075
'F9@                                                                            WSPM0076
L'10IM Z'22@                                                                    WSPM0077
                                          *** Z'22 EQU '20'F14@                 WSPM1077
    SBOUND 1'F1@                                                                WSPM2077
@                                                                               WSPM0078
.                                                                               WSPM0079
IF ('10) = () SKIP 1@                                                           WSPM0080
+ LOC '10@                                                                      WSPM0081
IF ('20) = () SKIP 1@                                                           WSPM0082
'20@                                                                            WSPM0083
IF ('30) = () SKIP 1@                                                           WSPM0084
'30@                                                                            WSPM0085
IF ('40) = () SKIP 1@                                                           WSPM0086
'40@                                                                            WSPM0087
IF ('50) = () SKIP 1@                                                           WSPM0088
'50@                                                                            WSPM0089
IF ('60) = () SKIP 1@                                                           WSPM0090
'60@                                                                            WSPM0091
IF ('70) = () SKIP 1@                                                           WSPM0092
'70@                                                                            WSPM0093
IF ('80) = () SKIP 1@                                                           WSPM0094
'80@                                                                            WSPM0095
IF ('90) = () SKIP 1@                                                           WSPM0096
'90@                                                                            WSPM0097
@                                                                               WSPM0098
TO @.                                                                           WSPM0099
+ JMP '10@                                                                      WSPM0100
@                                                                               WSPM0101
INCR @.                                                                         WSPM0102
+ INC '10@                                                                      WSPM0103
```

```
@                                                                  WSPM0104
@ = @.                                                             WSPM0105
+ FET '20@                                                         WSPM0106
+ STO '10@                                                         WSPM0107
@                                                                  WSPM0108
IO @ ON @.                                                         WSPM0109
+ FET '10@                                                         WSPM0110
+ IOP '20@                                                         WSPM0111
@                                                                  WSPM0112
TO @ IF @ = @.                                                     WSPM0113
+ FET '20@                                                         WSPM0114
+ CMP '30@                                                         WSPM0115
+ JE@ '10@                                                         WSPM0116
@                                                                  WSPM0117
TO @ IF @ NE @.                                                    WSPM0118
+ FET '20@                                                         WSPM0119
+ CMP '30@                                                         WSPM0120
+ JNE '10@                                                         WSPM0121
@                                                                  WSPM0122
TO @ IF @ LT @.                                                    WSPM0123
+ FET '20@                                                         WSPM0124
+ CMP '30@                                                         WSPM0125
+ JLT '10@                                                         WSPM0126
@                                                                  WSPM0127
ELEMENT @ @ @ @.                                                   WSPM0128
+ DAF '10@                                                         WSPM0130
+ DBP '20@                                                         WSPM0131
+ DCA '30@                                                         WSPM0132
+ DCD '40@                                                         WSPM0129
@                                                                  WSPM0133
@ = LINE BUFFER.                                                   WSPM0134
IO 01 ON '10@                                                      WSPM0135
@                                                                  WSPM0136
LINE BUFFER = @.                                                   WSPM0137
IO 02 ON '10@                                                      WSPM0138
@                                                                  WSPM0139
READ @.                                                            WSPM0140
IO 03 ON '10@                                                      WSPM0141
@                                                                  WSPM0142
WRITE @.                                                           WSPM0143
IO 04 ON '10@                                                      WSPM0144
@                                                                  WSPM0145
ENDFILE @.                                                         WSPM0146
IO 05 ON '10@                                                      WSPM0147
@                                                                  WSPM0148
REWIND @.                                                          WSPM0149
IO 06 ON '10@                                                      WSPM0150
@                                                                  WSPM0151
ENTRY @.                                                           WSPM0152
+ LEN '10@                                                         WSPM0153
@                                                                  WSPM0154
USE @.                                                             WSPM0155
+ USE '10@                                                         WSPM0156
@                                                                  WSPM0157
EXIT @.                                                            WSPM0158
```

```
+ XIT '10@                                                    WSPM0159
@                                                            WSPM0160
@ = NEW ELEMENT.                                             WSPM0161
USE GETNEW@                                                  WSPM0162
'10 = (P1)@                                                  WSPM0163
@                                                            WSPM0164
PUSH DOWN AF @.                                              WSPM0165
ERR AF '10 ILLEGAL IN PUSH DOWN@                             WSPM0166
@                                                            WSPM0167
PUSH DOWN BPF @.                                             WSPM0168
ERR BPF '10 ILLEGAL IN PUSH DOWN@                            WSPM0169
@                                                            WSPM0170
POP UP AF @.                                                 WSPM0171
ERR AF '10 ILLEGAL IN POP UP@                                WSPM0172
@                                                            WSPM0173
POP UP BPF @.                                                WSPM0174
ERR BPF '10 ILLEGAL IN POP UP@                               WSPM0175
@                                                            WSPM0176
PUSH DOWN @.                                                 WSPM0177
USE GETNEW@                                                  WSPM0178
(P2) = '10@                                                  WSPM0179
CAR P1 = CAR P2@                                             WSPM0180
CDR P1 = CDR P2@                                             WSPM0181
CDR P2 = (P1)@                                               WSPM0182
@                                                            WSPM0183
POP UP @.                                                    WSPM0184
(P1) = '10@                                                  WSPM0185
(P2) = CDR P1@                                               WSPM0186
CAR P1 = CAR P2@                                             WSPM0187
CDR P1 = CDR P2@                                             WSPM0188
CDR P2 = (FREE)@                                             WSPM0189
(FREE) = (P2)@                                               WSPM0190
@                                                            WSPM0191
COPY TO @.                                                   WSPM0192
'F2@                                                         WSPM0193
@                                                            WSPM0194
@ EQU @.                                                     WSPM0195
'F3@                                                         WSPM0196
@                                                            WSPM0197
@ SET @.                                                     WSPM0198
'24'26@                                                      WSPM0199
'F3@                                                         WSPM0200
@                                                            WSPM0201
SKIP @.                                                      WSPM0202
'F4@                                                         WSPM0203
@                                                            WSPM0204
IF @ = @ SKIP @.                                             WSPM0205
'F50@                                                        WSPM0206
@                                                            WSPM0207
IF @ NE @ SKIP @.                                            WSPM0208
'F51@                                                        WSPM0209
@                                                            WSPM0210
IF @ GT @ SKIP @.                                            WSPM0211
'F6+@                                                        WSPM0212
@                                                            WSPM0213
```

```
ERR @.                                                          WSPM0214
***** '10'F14@                                                  WSPM0215
 'F14@                                                          WSPM0216
@                                                               WSPM0217
@.                                                              WSPM0218
ERR '10 UNRECOGNIZED@                                           WSPM0219
@                                                               WSPM0220
BASIC WISP COMPILER@.                                           WSPM0221
NIL EQU NIL@                                                    WSPM0222
P2 EQU P2@                                                      WSPM0223
 'F14@                                                          WSPM0224
ENV END'16 'F2@                                                 WSPM0225
'16@                                                            WSPM0226
1'F2@                                                           WSPM0227
@                                                               WSPM0228
- @ @.                                                          WSPM0229
IF '25 GT 1 SKIP 3@                                             WSPM0230
'28*2'96@                                                       WSPM0231
'10 BASE+'94@                                                   WSPM0232
'F9@                                                            WSPM0233
'20'3701234567890@                                              WSPM0234
IF '30 =  SKIP 1@                                               WSPM0235
Z'22'26@                                                        WSPM0236
'10 '20@                                                        WSPM0237
@                                                               WSPM0238
+ @ '@.                                                         WSPM0239
'20*2'96@                                                       WSPM1239
'10IM BASE+'94@                                                 WSPM2239
@                                                               WSPM3239
JMPIM @.                                                        WSPM0240
.LAB'26@                                                        WSPM0241
'10'3701234567890@                                              WSPM0242
IF '30 NE  SKIP 3@                                              WSPM0243
 '21  B   '10+'10'F1@                                          WSPM0244
  '26 .LAB'16 'F3@                                              WSPM0245
'F9@                                                            WSPM0246
 '21  B   '10'F1@                                              WSPM0247
  '26 .LAB'16 'F3@                                              WSPM0248
@                                                               WSPM0249
JMPDR @.                                                        WSPM0250
.LAB'26@                                                        WSPM0251
 '21  LD  '10,,6'F1@                                           WSPM0252
    B   JMPDRCD'F1@                                             WSPM0253
  '26 .LAB'16 'F3@                                              WSPM0254
@                                                               WSPM0255
JMPCA @.                                                        WSPM0256
.LAB'26@                                                        WSPM0257
 '21  LD  ('10),,6'F1@                                         WSPM0258
    B   JMPCA'F1@                                               WSPM0259
  '26 .LAB'16 'F3@                                              WSPM0260
@                                                               WSPM0261
JMPCD @.                                                        WSPM0262
.LAB'26@                                                        WSPM0263
 '21  LD  ('10),,6'F1@                                         WSPM0264
    B   JMPDRCD'F1@                                             WSPM0265
```

```
  '26 .LAB'16 'F3@                                                    WSPM0266
@                                                                     WSPM0267
INCDR @.                                                              WSPM0268
.LAB'26@                                                              WSPM0269
 '21  LD   '10,,6'F1@                                                 WSPM0270
      AL   INCR,,6'F1@                                                WSPM0271
      STD  '10,,6'F1@                                                 WSPM0272
  '26 .LAB'16 'F3@                                                    WSPM0273
@                                                                     WSPM0274
INCCA @.                                                              WSPM0275
.LAB'26@                                                              WSPM0276
 '21  LD   ('10),,6'F1@                                               WSPM0277
      AL   INCR,,7'F1@                                                WSPM0278
      STD  ('10),,6'F1@                                               WSPM0279
  '26 .LAB'16 'F3@                                                    WSPM0280
@                                                                     WSPM0281
INCCD @.                                                              WSPM0282
.LAB'26@                                                              WSPM0283
 '21  LD   ('10),,6'F1@                                               WSPM0284
      AL   INCR,,6'F1@                                                WSPM0285
      STD  ('10),,6'F1@                                               WSPM0286
  '26 .LAB'16 'F3@                                                    WSPM0287
@                                                                     WSPM0288
INCAF @.                                                              WSPM0289
.LAB'26@                                                              WSPM0290
 '21  LD   ('10),,6'F1@                                               WSPM0291
      B    INCAF'F1@                                                  WSPM0292
      STD  ('10),,6'F1@                                               WSPM0293
  '26 .LAB'16 'F3@                                                    WSPM0294
@                                                                     WSPM0295
INCBP @.                                                              WSPM0296
.LAB'26@                                                              WSPM0297
 '21  LD   ('10),,6'F1@                                               WSPM0298
      B    INCBP'F1@                                                  WSPM0299
      STD  ('10),,6'F1@                                               WSPM0300
  '26 .LAB'16 'F3@                                                    WSPM0301
@                                                                     WSPM0302
FETIM @.                                                              WSPM0303
.LAB'26@                                                              WSPM0304
'10'3701234567890@                                                    WSPM0305
IF '30 NE  SKIP 3@                                                    WSPM0306
 '21  LI   '10,,3'F1@                                                 WSPM0307
  '26 .LAB'16 'F3@                                                    WSPM0308
'F9@                                                                  WSPM0309
 '21  LI   '10,,3'F1@                                                 WSPM0310
      HR   1,,3'F1@                                                   WSPM0311
  '26 .LAB'16 'F3@                                                    WSPM0312
@                                                                     WSPM0313
FETDR @.                                                              WSPM0314
.LAB'26@                                                              WSPM0315
 '21  LD   '10,,6'F1@                                                 WSPM0316
      B    FETDRCD'F1@                                                WSPM0317
  '26 .LAB'16 'F3@                                                    WSPM0318
@                                                                     WSPM0319
FETAF @.                                                              WSPM0320
```

```
.LAB'26@                                                        WSPM0321
 '21  LD   ('10),,6'F1@                                         WSPM0322
      B    FETAF'F1@                                            WSPM0323
  '26 .LAB'16 'F3@                                              WSPM0324
@                                                               WSPM0325
FETBP @.                                                        WSPM0326
.LAB'26@                                                        WSPM0327
 '21  LD   ('10),,6'F1@                                         WSPM0328
      B    FETBP'F1@                                            WSPM0329
  '26 .LAB'16 'F3@                                              WSPM0330
@                                                               WSPM0331
FETCA @.                                                        WSPM0332
.LAB'26@                                                        WSPM0333
 '21  LD   ('10),,6'F1@                                         WSPM0334
      B    FETCA'F1@                                            WSPM0335
  '26 .LAB'16 'F3@                                              WSPM0336
@                                                               WSPM0337
FETCD @.                                                        WSPM0338
.LAB'26@                                                        WSPM0339
 '21  LD   ('10),,6'F1@                                         WSPM0340
      B    FETDRCD'F1@                                          WSPM0341
  '26 .LAB'16 'F3@                                              WSPM0342
@                                                               WSPM0343
STODR @.                                                        WSPM0344
      LD   '10,,6'F1@                                           WSPM0345
      B    STODRCD'F1@                                          WSPM0346
      STD  '10,,6'F1@                                           WSPM0347
@                                                               WSPM0348
STOAF @.                                                        WSPM0349
      LD   ('10),,6'F1@                                         WSPM0350
      B    STOAF'F1@                                            WSPM0351
      STD  ('10),,6'F1@                                         WSPM0352
@                                                               WSPM0353
STOBP @.                                                        WSPM0354
      LD   ('10),,6'F1@                                         WSPM0355
      B    STOBP'F1@                                            WSPM0356
      STD  ('10),,6'F1@                                         WSPM0357
@                                                               WSPM0358
STOCA @.                                                        WSPM0359
      LD   ('10),,6'F1@                                         WSPM0360
      B    STOCA'F1@                                            WSPM0361
      STD  ('10),,6'F1@                                         WSPM0362
@                                                               WSPM0363
STOCD @.                                                        WSPM0364
      LD   ('10),,6'F1@                                         WSPM0365
      B    STODRCD'F1@                                          WSPM0366
      STD  ('10),,6'F1@                                         WSPM0367
@                                                               WSPM0368
CMPIM @.                                                        WSPM0369
'10'3701234567890@                                              WSPM0370
IF '30 NE  SKIP 3@                                              WSPM0371
      LI   '10,,6'F1@                                           WSPM0372
      ST   COMP,,6'F1@                                          WSPM0373
'F9@                                                            WSPM0374
      LI   '10,,6'F1@                                           WSPM0375
```

```
     HR   1,,6'F1@                                                    WSPM0376
     ST   COMP,,6'F1@                                                 WSPM0377
@                                                                     WSPM0378
CMPDR @.                                                              WSPM0379
     LD   '10,,6'F1@                                                  WSPM0380
     B    CMPDRCD'F1@                                                 WSPM0381
@                                                                     WSPM0382
CMPAF @.                                                              WSPM0383
     LD   ('10),,6'F1@                                                WSPM0384
     B    CMPAF'F1@                                                   WSPM0385
@                                                                     WSPM0386
CMPBP @.                                                              WSPM0387
     LD   ('10),,6'F1@                                                WSPM0388
     B    CMPBP'F1@                                                   WSPM0389
@                                                                     WSPM0390
CMPCA @.                                                              WSPM0391
     LD   ('10),,6'F1@                                                WSPM0392
     B    CMPCA'F1@                                                   WSPM0393
@                                                                     WSPM0394
CMPCD @.                                                              WSPM0395
     LD   ('10),,6'F1@                                                WSPM0396
     B    CMPDRCD'F1@                                                 WSPM0397
@                                                                     WSPM0398
JE@IM @.                                                              WSPM0399
'10'3701234567890@                                                    WSPM0400
IF '30 NE  SKIP 4@                                                    WSPM0401
     LI   '10,,6'F1@                                                  WSPM0402
     HL   1,,6'F1@                                                    WSPM0403
     B    JE@IDCD'F1@                                                 WSPM0404
'F9@                                                                  WSPM0405
     LI   '10,,6'F1@                                                  WSPM0406
     B    JE@IDCD'F1@                                                 WSPM0407
@                                                                     WSPM0408
JE@DR @.                                                              WSPM0409
     LD   '10,,6'F1@                                                  WSPM0410
     B    JE@IDCD'F1@                                                 WSPM0411
@                                                                     WSPM0412
JE@CA @.                                                              WSPM0413
     LD   ('10),,6'F1@                                                WSPM0414
     B    JE@CA'F1@                                                   WSPM0415
@                                                                     WSPM0416
JE@CD @.                                                              WSPM0417
     LD   ('10),,6'F1@                                                WSPM0418
     B    JE@IDCD'F1@                                                 WSPM0419
@                                                                     WSPM0420
JNEIM @.                                                              WSPM0421
'10'3701234567890@                                                    WSPM0422
IF '30 NE  SKIP 4@                                                    WSPM0423
     LI   '10,,6'F1@                                                  WSPM0424
     HL   1,,6'F1@                                                    WSPM0425
     B    JNEIDCD'F1@                                                 WSPM0426
'F9@                                                                  WSPM0427
     LI   '10,,6'F1@                                                  WSPM0428
     B    JNEIDCD'F1@                                                 WSPM0429
@                                                                     WSPM0430
```

```
JNEDR @.                                                        WSPM0431
     LD   '10,,6'F1@                                            WSPM0432
     B    JNEIDCD'F1@                                           WSPM0433
@                                                               WSPM0434
JNECA @.                                                        WSPM0435
     LD   ('10),,6'F1@                                          WSPM0436
     B    JNECA'F1@                                             WSPM0437
@                                                               WSPM0438
JNECD @.                                                        WSPM0439
     LD   ('10),,6'F1@                                          WSPM0440
     B    JNEIDCD'F1@                                           WSPM0441
@                                                               WSPM0442
JLTIM @.                                                        WSPM0443
'10'3701234567890                                               WSPM0444
IF '30 NE  SKIP 4@                                              WSPM0445
     LI   '10,,6'F1@                                            WSPM0446
     HL   1,,6'F1@                                              WSPM0447
     B    JLTIDCD'F1@                                           WSPM0448
'F9@                                                            WSPM0449
     LI   '10,,6'F1@                                            WSPM0450
     B    JLTIDCD'F1@                                           WSPM0451
@                                                               WSPM0452
JLTDR @.                                                        WSPM0453
     LD   '10,,6'F1@                                            WSPM0454
     B    JLTIDCD'F1@                                           WSPM0455
@                                                               WSPM0456
JLTCA @.                                                        WSPM0457
     LD   ('10),,6'F1@                                          WSPM0458
     B    JLTCA'F1@                                             WSPM0459
@                                                               WSPM0460
JLTCD @.                                                        WSPM0461
     LD   ('10),,6'F1@                                          WSPM0462
     B    JLTIDCD'F1@                                           WSPM0463
@                                                               WSPM0464
DAFIM @.                                                        WSPM0465
,LAB'26@                                                        WSPM0466
 '21  DA  2'F1@                                                 WSPM0467
'10'26 .AF'16 'F3@                                              WSPM0468
@                                                               WSPM0469
DBPIM @.                                                        WSPM0470
'10'26 .BP'16 'F3@                                              WSPM0471
@                                                               WSPM0472
DCAIM @.                                                        WSPM0473
'10'9701234567890                                               WSPM0474
IF '90 NE  SKIP 2@                                              WSPM0475
'10+'10'26 .CA'16 'F3@                                          WSPM0476
'F9@                                                            WSPM0477
'10'26 .CA'16 'F3@                                              WSPM0478
@                                                               WSPM0479
DCDIM @.                                                        WSPM0480
'10'9701234567890                                               WSPM0481
IF '90 NE  SKIP 2@                                              WSPM0482
'10+'10'26 .CD'16 'F3@                                          WSPM0483
SKIP 1@                                                         WSPM0484
'10'26 .CD'16 'F3@                                              WSPM0485
```

```
     B   *+6'F12@                                                    WSPM0486
.AF'16 .BP'26 .CA'36 .CD'46 .LAB'56@                                 WSPM0487
 '51F  DC  A'''11'''F12@                                             WSPM0488
 '51B  DC  A'''21'''F12@                                             WSPM0489
 '51A  DC  A'''31'''F12@                                             WSPM0490
 '51D  DC  A'''41'''F12@                                             WSPM0491
     L   '51F,,6'F12@                                                WSPM0492
     C   CONST1,,6'F12@                                              WSPM0493
     BH  FLGERR'F12@                                                 WSPM0494
     AL  '51D,,6'F12@                                                WSPM0495
     L   '51B,,7'F12@                                                WSPM0496
     C   CONST1,,7'F12@                                              WSPM0497
     BH  FLGERR'F12@                                                 WSPM0498
     AL  '51A,,7'F12@                                                WSPM0499
     STD '51,,6'F12@                                                 WSPM1499
  '26 .LAB'16 'F3@                                                   WSPM0500
@                                                                    WSPM0501
LOCIM @.                                                             WSPM0502
'10'26 .LAB'16 'F3@                                                  WSPM0503
@                                                                    WSPM0504
LENIM @.                                                             WSPM0505
 '10  ST  *+4,,0'F1@                                                 WSPM0506
     B   *+3'F1@                                                     WSPM0507
     DA  1'F1@                                                       WSPM0508
@                                                                    WSPM0509
XITIM @.                                                             WSPM0510
.LAB'26@                                                             WSPM0511
 '21  B  ('10+4)'F1@                                                 WSPM0512
  '26 .LAB'16 'F3@                                                   WSPM0513
@                                                                    WSPM0514
USEIM @.                                                             WSPM0515
.LAB'26@                                                             WSPM0516
 '21  B  '10'F1@                                                     WSPM0517
  '26 .LAB'16 'F3@                                                   WSPM0518
@                                                                    WSPM0519
STOP.                                                                WSPM0520
.LAB'16@                                                             WSPM0521
 '11  B  (PROGR+4)'F1@                                               WSPM0522
  '26 'F3@                                                           WSPM0523
@                                                                    WSPM0524
END PROGRAM.                                                         WSPM0525
KRAJ'F12@                                                            WSPM0526
 ELAS  ST  *+4,,0'F1@                                                WSPM0528
       B   *+3'F1@                                                   WSPM0529
       DA  1'F1@                                                     WSPM0530
KRAJ'16@                                                             WSPM0531
2R'F23@                                                              WSPM0532
       B   (ELAS+4)'F1@                                              WSPM0533
          END'F1@                                                    WSPM1533
'F0@                                                                 WSPM0534
@                                                                    WSPM0535
IOPIM @.                                                             WSPM0601
'10'3701234567890@                                                   WSPM0602
IF '30 NE  SKIP 5@                                                   WSPM0603
     LI  '10,,6'F1@                                                  WSPM0604
```

```
     HL   1,,6'F1@                                                    WSPM0605
     L    CONST3,,2'F1@                                               WSPM1605
     B    IOOPOPR'F1@                                                 WSPM0606
'F9@                                                                  WSPM0607
     LI   '10,,6'F1@                                                  WSPM0608
     L    CONST3,,2'F1@                                               WSPM1608
     B    IOOPOPR'F1@                                                 WSPM0609
@                                                                     WSPM0610
IOPDR @.                                                              WSPM0611
     LD   '10,,6'F1@                                                  WSPM0612
     L    CONST0,,2'F1@                                               WSPM0613
     B    IOOPOPR'F1@                                                 WSPM0614
     STD  '10,,6'F1@                                                  WSPM0615
@                                                                     WSPM0616
IOPCA @.                                                              WSPM0617
     LD   ('10),,6'F1@                                                WSPM0618
     L    CONST2,,2'F1@                                               WSPM0619
     B    IOOPOPR'F1@                                                 WSPM0620
     STD  ('10),,6'F1@                                                WSPM0621
@                                                                     WSPM0622
IOPCD @.                                                              WSPM0623
     LD   ('10),,6'F1@                                                WSPM0624
     L    CONST1,,2'F1@                                               WSPM0625
     B    IOOPOPR'F1@                                                 WSPM0626
     STD  ('10),,6'F1@                                                WSPM0627
@@
    BASIC WISP COMPILER - VERSION 01 JUNE 1974.
 PROGR    ST   *+4,,0
          B    PROGRST
          DA   1
 PROGRST  NOP
ENV END
```

WIOCS

```
HELP    FLMNT LIST                                                    WIOS0001
MAIN    SECT                                                          WIOS0002
     B  START                                                         WIOS0003
        SBOUND 1                                                      WIOS0004
FREESPB  DA XXXXX               THE SIZE MUST BE PROPERLY SET
FREESPE  DA 2
BASE     DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
         DC A'BASE+257'
         DC A'*+1'
```

```
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'FREESPB+1'
DC A'*+1'
DC A'FREESPE+1'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
```

```
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
```

```
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
DC A'*+1'
DC A'BASE+257'
```

```
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
        DC A'BASE+257'
        DC A'*+1'
ZNIL    DC A'BASE+257'
        DC A'BASE+1'
TABCI   DC  A'BASE+256'
        DC  A'BASE+256'                                    WIOST001
        DC  A'BASE+256'                                    WIOST002
        DC  A'BASE+256'                                    WIOST003
        DC  A'BASE+256'                                    WIOST004
        DC  A'BASE+256'                                    WIOST005
                                                           WIOST006
```

```
DC  A'BASE+256'                                                    WIOST007
DC  A'BASE+256'                                                    WIOST008
DC  A'BASE+256'                                                    WIOST009
DC  A'BASE+256'                                                    WIOST010
DC  A'BASE+256'                                                    WIOST011
DC  A'BASE+256'                                                    WIOST012
DC  A'BASE+256'                                                    WIOST013
DC  A'BASE+256'                                                    WIOST014
DC  A'BASE+256'                                                    WIOST015
DC  A'BASE+256'                                                    WIOST016
DC  A'BASE+256'                                                    WIOST017
DC  A'BASE+256'                                                    WIOST018
DC  A'BASE+256'                                                    WIOST019
DC  A'BASE+256'                                                    WIOST020
DC  A'BASE+256'                                                    WIOST021
DC  A'BASE+256'                                                    WIOST022
DC  A'BASE+256'                                                    WIOST023
DC  A'BASE+256'                                                    WIOST024
DC  A'BASE+256'                                                    WIOST025
DC  A'BASE+256'                                                    WIOST026
DC  A'BASE+256'                                                    WIOST027
DC  A'BASE+256'                                                    WIOST028
DC  A'BASE+256'                                                    WIOST029
DC  A'BASE+256'                                                    WIOST030
DC  A'BASE+256'                                                    WIOST031
DC  A'BASE+256'                                                    WIOST032
DC  A'BASE+256'                                                    WIOST033
DC  A'BASE+256'                                                    WIOST034
DC  A'BASE+256'                                                    WIOST035
DC  A'BASE+256'                                                    WIOST036
DC  A'BASE+256'                                                    WIOST037
DC  A'BASE+256'                                                    WIOST038
DC  A'BASE+256'                                                    WIOST039
DC  A'BASE+256'                                                    WIOST040
DC  A'BASE+256'                                                    WIOST041
DC  A'BASE+256'                                                    WIOST042
DC  A'BASE+256'                                                    WIOST043
DC  A'BASE+256'                                                    WIOST044
DC  A'BASE+256'                                                    WIOST045
DC  A'BASE+256'                                                    WIOST046
DC  A'BASE+256'                                                    WIOST047
DC  A'BASE+256'                                                    WIOST048
DC  A'BASE+256'                                                    WIOST049
DC  A'BASE+256'                                                    WIOST050
DC  A'BASE+256'                                                    WIOST051
DC  A'BASE+256'                                                    WIOST052
DC  A'BASE+256'                                                    WIOST053
DC  A'BASE+256'                                                    WIOST054
DC  A'BASE+256'                                                    WIOST055
DC  A'BASE+256'                                                    WIOST056
DC  A'BASE+256'                                                    WIOST057
DC  A'BASE+256'                                                    WIOST058
DC  A'BASE+256'                                                    WIOST059
DC  A'BASE+256'                                                    WIOST060
DC  A'BASE+256'                                                    WIOST061
```

```
DC    A'BASE+256'                                                    WIOST062
DC    A'BASE+256'                                                    WIOST063
DC    A'BASE+256'                                                    WIOST064
DC    A'BASE'                                                        WIOST065
DC    A'BASE+2'                                                      WIOST066
DC    A'BASE+4'                                                      WIOST067
DC    A'BASE+6'                                                      WIOST068
DC    A'BASE+8'                                                      WIOST069
DC    A'BASE+10'                                                     WIOST070
DC    A'BASE+12'                                                     WIOST071
DC    A'BASE+14'                                                     WIOST072
DC    A'BASE+16'                                                     WIOST073
DC    A'BASE+18'                                                     WIOST074
DC    A'BASE+74'                                                     WIOST075
DC    A'BASE+76'                                                     WIOST076
DC    A'BASE+78'                                                     WIOST077
DC    A'BASE+80'                                                     WIOST078
DC    A'BASE+82'                                                     WIOST079
DC    A'BASE+84'                                                     WIOST080
DC    A'BASE+86'                                                     WIOST081
DC    A'BASE+20'                                                     WIOST082
DC    A'BASE+22'                                                     WIOST083
DC    A'BASE+24'                                                     WIOST084
DC    A'BASE+26'                                                     WIOST085
DC    A'BASE+28'                                                     WIOST086
DC    A'BASE+30'                                                     WIOST087
DC    A'BASE+32'                                                     WIOST088
DC    A'BASE+34'                                                     WIOST089
DC    A'BASE+36'                                                     WIOST090
DC    A'BASE+88'                                                     WIOST091
DC    A'BASE+90'                                                     WIOST092
DC    A'BASE+92'                                                     WIOST093
DC    A'BASE+94'                                                     WIOST094
DC    A'BASE+96'                                                     WIOST095
DC    A'BASE+98'                                                     WIOST096
DC    A'BASE+100'                                                    WIOST097
DC    A'BASE+102'                                                    WIOST098
DC    A'BASE+38'                                                     WIOST099
DC    A'BASE+40'                                                     WIOST100
DC    A'BASE+42'                                                     WIOST101
DC    A'BASE+44'                                                     WIOST102
DC    A'BASE+46'                                                     WIOST103
DC    A'BASE+48'                                                     WIOST104
DC    A'BASE+50'                                                     WIOST105
DC    A'BASE+52'                                                     WIOST106
DC    A'BASE+104'                                                    WIOST107
DC    A'BASE+106'                                                    WIOST108
DC    A'BASE+108'                                                    WIOST109
DC    A'BASE+110'                                                    WIOST110
DC    A'BASE+112'                                                    WIOST111
DC    A'BASE+114'                                                    WIOST112
DC    A'BASE+54'                                                     WIOST113
DC    A'BASE+56'                                                     WIOST114
DC    A'BASE+58'                                                     WIOST115
DC    A'BASE+60'                                                     WIOST116
```

```
DC  A'BASE+62'                                                  WIOST117
DC  A'BASE+64'                                                  WIOST118
DC  A'BASE+66'                                                  WIOST119
DC  A'BASE+68'                                                  WIOST120
DC  A'BASE+70'                                                  WIOST121
DC  A'BASE+72'                                                  WIOST122
DC  A'BASE+116'                                                 WIOST123
DC  A'BASE+118'                                                 WIOST124
DC  A'BASE+120'                                                 WIOST125
DC  A'BASE+122'                                                 WIOST126
DC  A'BASE+124'                                                 WIOST127
DC  A'BASE+126'                                                 WIOST128
DC  A'BASE+128'                                                 WIOST129
DC  A'BASE+130'                                                 WIOST130
DC  A'BASE+132'                                                 WIOST131
DC  A'BASE+134'                                                 WIOST132
DC  A'BASE+136'                                                 WIOST133
DC  A'BASE+138'                                                 WIOST134
DC  A'BASE+140'                                                 WIOST135
DC  A'BASE+142'                                                 WIOST136
DC  A'BASE+144'                                                 WIOST137
DC  A'BASE+146'                                                 WIOST138
DC  A'BASE+148'                                                 WIOST139
DC  A'BASE+150'                                                 WIOST140
DC  A'BASE+152'                                                 WIOST141
DC  A'BASE+154'                                                 WIOST142
DC  A'BASE+156'                                                 WIOST143
DC  A'BASE+158'                                                 WIOST144
DC  A'BASE+160'                                                 WIOST145
DC  A'BASE+162'                                                 WIOST146
DC  A'BASE+164'                                                 WIOST147
DC  A'BASE+166'                                                 WIOST148
DC  A'BASE+168'                                                 WIOST149
DC  A'BASE+170'                                                 WIOST150
DC  A'BASE+172'                                                 WIOST151
DC  A'BASE+174'                                                 WIOST152
DC  A'BASE+176'                                                 WIOST153
DC  A'BASE+178'                                                 WIOST154
DC  A'BASE+180'                                                 WIOST155
DC  A'BASE+182'                                                 WIOST156
DC  A'BASE+184'                                                 WIOST157
DC  A'BASE+186'                                                 WIOST158
DC  A'BASE+188'                                                 WIOST159
DC  A'BASE+190'                                                 WIOST160
DC  A'BASE+192'                                                 WIOST161
DC  A'BASE+194'                                                 WIOST162
DC  A'BASE+196'                                                 WIOST163
DC  A'BASE+198'                                                 WIOST164
DC  A'BASE+200'                                                 WIOST165
DC  A'BASE+202'                                                 WIOST166
DC  A'BASE+204'                                                 WIOST167
DC  A'BASE+206'                                                 WIOST168
DC  A'BASE+208'                                                 WIOST169
DC  A'BASE+210'                                                 WIOST170
DC  A'BASE+212'                                                 WIOST171
```

```
DC  A'BASE+214'                                                    WIOST172
DC  A'BASE+216'                                                    WIOST173
DC  A'BASE+218'                                                    WIOST174
DC  A'BASE+220'                                                    WIOST175
DC  A'BASE+222'                                                    WIOST176
DC  A'BASE+224'                                                    WIOST177
DC  A'BASE+226'                                                    WIOST178
DC  A'BASE+228'                                                    WIOST179
DC  A'BASE+230'                                                    WIOST180
DC  A'BASE+232'                                                    WIOST181
DC  A'BASE+234'                                                    WIOST182
DC  A'BASE+236'                                                    WIOST183
DC  A'BASE+238'                                                    WIOST184
DC  A'BASE+240'                                                    WIOST185
DC  A'BASE+242'                                                    WIOST186
DC  A'BASE+244'                                                    WIOST187
DC  A'BASE+246'                                                    WIOST188
DC  A'BASE+248'                                                    WIOST189
DC  A'BASE+250'                                                    WIOST190
DC  A'BASE+252'                                                    WIOST191
DC  A'BASE+254'                                                    WIOST192
DC  A'BASE'                                                        WIOST193
DC  A'BASE+2'                                                      WIOST194
DC  A'BASE+4'                                                      WIOST195
DC  A'BASE+6'                                                      WIOST196
DC  A'BASE+8'                                                      WIOST197
DC  A'BASE+10'                                                     WIOST198
DC  A'BASE+12'                                                     WIOST199
DC  A'BASE+14'                                                     WIOST200
DC  A'BASE+16'                                                     WIOST201
DC  A'BASE+18'                                                     WIOST202
DC  A'BASE+74'                                                     WIOST203
DC  A'BASE+76'                                                     WIOST204
DC  A'BASE+78'                                                     WIOST205
DC  A'BASE+80'                                                     WIOST206
DC  A'BASE+82'                                                     WIOST207
DC  A'BASE+84'                                                     WIOST208
DC  A'BASE+86'                                                     WIOST209
DC  A'BASE+20'                                                     WIOST210
DC  A'BASE+22'                                                     WIOST211
DC  A'BASE+24'                                                     WIOST212
DC  A'BASE+26'                                                     WIOST213
DC  A'BASE+28'                                                     WIOST214
DC  A'BASE+30'                                                     WIOST215
DC  A'BASE+32'                                                     WIOST216
DC  A'BASE+34'                                                     WIOST217
DC  A'BASE+36'                                                     WIOST218
DC  A'BASE+88'                                                     WIOST219
DC  A'BASE+90'                                                     WIOST220
DC  A'BASE+92'                                                     WIOST221
DC  A'BASE+94'                                                     WIOST222
DC  A'BASE+96'                                                     WIOST223
DC  A'BASE+98'                                                     WIOST224
DC  A'BASE+100'                                                    WIOST225
DC  A'BASE+102'                                                    WIOST226
```

```
         DC   A'BASE+38'                                                    WIOST227
         DC   A'BASE+40'                                                    WIOST228
         DC   A'BASE+42'                                                    WIOST229
         DC   A'BASE+44'                                                    WIOST230
         DC   A'BASE+46'                                                    WIOST231
         DC   A'BASE+48'                                                    WIOST232
         DC   A'BASE+50'                                                    WIOST233
         DC   A'BASE+52'                                                    WIOST234
         DC   A'BASE+104'                                                   WIOST235
         DC   A'BASE+106'                                                   WIOST236
         DC   A'BASE+108'                                                   WIOST237
         DC   A'BASE+110'                                                   WIOST238
         DC   A'BASE+112'                                                   WIOST239
         DC   A'BASE+114'                                                   WIOST240
         DC   A'BASE+54'                                                    WIOST241
         DC   A'BASE+56'                                                    WIOST242
         DC   A'BASE+58'                                                    WIOST243
         DC   A'BASE+60'                                                    WIOST244
         DC   A'BASE+62'                                                    WIOST245
         DC   A'BASE+64'                                                    WIOST246
         DC   A'BASE+66'                                                    WIOST247
         DC   A'BASE+68'                                                    WIOST248
         DC   A'BASE+70'                                                    WIOST249
         DC   A'BASE+72'                                                    WIOST250
         DC   A'BASE+116'                                                   WIOST251
         DC   A'BASE+118'                                                   WIOST252
         DC   A'BASE+120'                                                   WIOST253
         DC   A'BASE+122'                                                   WIOST254
         DC   A'BASE+124'                                                   WIOST255
         DC   A'BASE+126'                                                   WIOST256
TABIC    DC   C' ABCDEFGHIJKLMNOP@RSTUVWXYZ012345678'                       WIOST301
      DC  X'F94A4B4C4D4E4F505A5B5C5D5E5F606161A6B6C6D6E6F7A7B7C7D7E7F'      WIOST302
         DC   X'80818283848586878889898A8B8C8D8E8F'                         WIOST303
         DC   X'909192939495969798999A9B9C9D9E9F'                           WIOST304
         DC   X'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'                           WIOST305
         DC   X'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'                           WIOST306
CRBUF    DA   520                                                           WIOS0005
LPBUF    DA   69                                                            WIOS0006
LPLBUF   DC   X'0140'                                                       WIOS0007
LBUF     DC   C136' '                                                       WIOS0008
CRFCB FCBPS  ACNAM1=UIN,RCDFM=F,RCDS=80,BLKS=1040,BUFS=1040,              CWIOS0009
             BUFAD1=CRBUF,EOFEX=KRAJ                                        WIOS1009
LPFCB FCBPS  ACNAM1=LIST,RCDFM=FC,RCDS=138,BLKS=138,BUFS=138,             CWIOS0010
             BUFAD1=LPBUF                                                   WIOS1010
CROPEN   OPEN   CRFCB,IN                                                    WIOS0011
LPOPEN   OPEN   LPFCB,OT                                                    WIOS0012
CRCLOSE  CLOSE  CRFCB                                                       WIOS0013
LPCLOSE  CLOSE  LPFCB                                                       WIOS0014
CRREAD   GETS   CRFCB,LBUF                                                  WIOS0015
LPWRITE  WRITE  LPFCB,LPLBUF,S                                              WIOS0016
CRCHECK  CHECK  CRFCB                                                       WIOS0017
LPCHECK  CHECK  LPFCB                                                       WIOS0018
START    CALLF  S,OPEN,CROPEN                                               WIOS0019
         CALLF  S,OPEN,LPOPEN                                               WIOS0020
         B   ELAS                                                           WIOS0022
```

```
         B   PROGR                                                    WIOS0024
KRAJ     CALLF S.CLOSE,CRCLOSE                                        WIOS0025
         CALLF S.CLOSE,LPCLOSE                                        WIOS0026
      RETURN                                                          WIOS0027
IOERR    TR  4,2
         L   CONST0,,2
IOERRC   L   IOERMESS,2,5
         ST  MESSAREA,2,5
         A   CONST1,,2
         C   CONST10,,2
         BNL ERPRINT
         B   IOERRC
FLERMESS DC C20'ELAS OF AF OR BPF   '
FLGERR   L   CONST0,,2
FLGERRC  L   FLERMESS,2,5
         ST  MESSAREA,2,5
         A   CONST1,,2
         C   CONST10,,2
         BNL ERPRINT
         B   FLGERRC
IOERMESS DC C20'I/O DNUM OR OPCODE  '
RETADR   DA  1
IOOP     DA  1
JUMP     DA  1
COMP     DA  1
INCR     DC  X'0002'
INCROR   DC  X'0001'
INCRAND  DC  X'FFFE'
CONST0   DC  F'0'                                                    WIOS0032
CONST1   DC  F'1'                                                    WIOS0033
CONST2   DC  F'2'                                                    WIOS0034
CONST3   DC  F'3'                                                    WIOS0035
CONST4   DC  F'4'                                                    WIOS0036
CONST6   DC  F'6'                                                    WIOS0037
CONST80  DC  F'80'                                                   WIOS0038
CONST136 DC  F'136'                                                  WIOS0039
CONST81  DC F'81'
LIMIT    DC F'136'
CONST10  DC F'10'
CONSTNIL DC A'ZNIL'                                                  WIOS0125
CONSTBEG DC A'BASE'                                                  WIOS0126
CONSTEND DC A'ZNIL'                                                  WIOS0127
CONSTSP  DC C' '                                                     WIOS0128
CONOTCHR DC C'@@'                                                    WIOS0129
ERPRINT  CALLF F,WRITE,ERWRITE                                       WIOS0065
         CALLF F,CHECK,LPCHECK                                       WIOS0066
         B   (RETADR)                                                WIOS0067
ERWRITE  WRITE LPFCB,ERAREA,S                                        WIOS0068
ERAREA   DC X'0140'                                                  WIOS0069
         DC C26'*************** ERROR IN  '                         WIOS1069
MESSAREA DC C110'  '                                                 WIOS2069
JMPDRCD  HR  1,,6                                                    WSPSR001
         HL  1,,6                                                    WSPSR002
         ST  JUMP,,6                                                 WSPSR003
         B  (JUMP)                                                   WSPSR004
```

```
JMPCA    HR    1,,7                                          WSPSR005
         HL    1,,7                                          WSPSR006
         ST    JUMP,,7                                       WSPSR007
         B     (JUMP)                                        WSPSR008
INCAF    ST    RETADR,,0                                     WSPSR009
         TR    4,6                                           WSPSR010
         BOB   *+6,,15                                       WSPSR011
         OR    INCROR,,6                                     WSPSR012
         B     *+4                                           WSPSR013
         AND   INCRAND,,6                                    WSPSR014
         B     (RETADR)                                      WSPSR015
INCBP    ST    RETADR,,0                                     WSPSR016
         TR    4,7                                           WSPSR017
         BOB   *+6,,15                                       WSPSR018
         OR    INCROR,,7                                     WSPSR019
         B     *+4                                           WSPSR020
         AND   INCRAND,,7                                    WSPSR021
         B     (RETADR)                                      WSPSR022
FETDRCD  ST    RETADR,,0                                     WSPSR023
         TR    3,6                                           WSPSR024
         HR    1,,3                                          WSPSR025
         B     (RETADR)                                      WSPSR026
FETAF    ST    RETADR,,0                                     WSPSR027
         TR    3,6                                           WSPSR028
         HL    15,,3                                         WSPSR029
         HR    15,,3                                         WSPSR030
         B     (RETADR)                                      WSPSR031
FETBP    ST    RETADR,,0                                     WSPSR032
         TR    3,7                                           WSPSR033
         HL    15,,3                                         WSPSR034
         HR    15,,3                                         WSPSR035
         B     (RETADR)                                      WSPSR036
FETCA    ST    RETADR,,0                                     WSPSR037
         TR    3,7                                           WSPSR038
         HR    1,,3                                          WSPSR039
         B     (RETADR)                                      WSPSR040
STODRCD  ST    RETADR,,0                                     WSPSR041
         TR    4,6                                           WSPSR042
         HRD   1,,4                                          WSPSR043
         TR    4,3                                           WSPSR044
         HLD   1,,4                                          WSPSR045
         TR    6,4                                           WSPSR046
         B     (RETADR)                                      WSPSR047
STOAF    ST    RETADR,,0                                     WSPSR048
         TR    4,6                                           WSPSR049
         HRD   1,,4                                          WSPSR050
         HL    15,,3                                         WSPSR051
         TR    5,3                                           WSPSR052
         HLD   1,,4                                          WSPSR053
         TR    6,4                                           WSPSR054
         B     (RETADR)                                      WSPSR055
STOBP    ST    RETADR,,0                                     WSPSR056
         TR    4,7                                           WSPSR057
         HRD   1,,4                                          WSPSR058
         HL    15,,3                                         WSPSR059
```

```
         TR   5,3                                                  WSPSR060
         HLD  1,,4                                                 WSPSR061
         TR   7,4                                                  WSPSR062
         B    (RETADR)                                             WSPSR063
STOCA    ST   RETADR,,0                                            WSPSR064
         TR   4,7                                                  WSPSR065
         HRD  1,,4                                                 WSPSR066
         TR   4,3                                                  WSPSR067
         HLD  1,,4                                                 WSPSR068
         TR   7,4                                                  WSPSR069
         B    (RETADR)                                             WSPSR070
CMPDRCD  ST   RETADR,,0                                            WSPSR071
         HR   1,,6                                                 WSPSR072
         ST   COMP,,6                                              WSPSR073
         B    (RETADR)                                             WSPSR074
CMPAF    ST   RETADR,,0                                            WSPSR075
         HL   15,,6                                                WSPSR076
         HR   15,,6                                                WSPSR077
         ST   COMP,,6                                              WSPSR078
         B    (RETADR)                                             WSPSR079
CMPBP    ST   RETADR,,0                                            WSPSR080
         HL   15,,7                                                WSPSR081
         HR   15,,7                                                WSPSR082
         ST   COMP,,7                                              WSPSR083
         B    (RETADR)                                             WSPSR084
CMPCA    ST   RETADR,,0                                            WSPSR085
         HR   1,,7                                                 WSPSR086
         ST   COMP,,7                                              WSPSR087
         B    (RETADR)                                             WSPSR088
JE@IDCD  ST   RETADR,,0                                            WSPSR089
         HR   1,,6                                                 WSPSR090
         HL   1,,6                                                 WSPSR091
         ST   JUMP,,6                                              WSPSR092
         C    COMP,,3                                              WSPSR093
         BE   (JUMP)                                               WSPSR094
         B    (RETADR)                                             WSPSR095
JE@CA    ST   RETADR,,0                                            WSPSR096
         HR   1,,7                                                 WSPSR097
         HL   1,,7                                                 WSPSR098
         ST   JUMP,,7                                              WSPSR099
         C    COMP,,3                                              WSPSR100
         BE   (JUMP)                                               WSPSR101
         B    (RETADR)                                             WSPSR102
JNEIDCD  ST   RETADR,,0                                            WSPSR103
         HR   1,,6                                                 WSPSR104
         HL   1,,6                                                 WSPSR105
         ST   JUMP,,6                                              WSPSR106
         C    COMP,,3                                              WSPSR107
         BNE  (JUMP)                                               WSPSR108
         B    (RETADR)                                             WSPSR109
JNECA    ST   RETADR,,0                                            WSPSR110
         HR   1,,7                                                 WSPSR111
         HL   1,,7                                                 WSPSR112
         ST   JUMP,,7                                              WSPSR113
         C    COMP,,3                                              WSPSR114
```

```
         BNE  (JUMP)                                                WSPSR115
         B    (RETADR)                                              WSPSR116
JLTIDCD  ST   RETADR,,0                                             WSPSR117
         HR   1,,6                                                  WSPSR118
         HL   1,,6                                                  WSPSR119
         ST   JUMP,,6                                               WSPSR120
         C    COMP,,3                                               WSPSR121
         BL   (JUMP)                                                WSPSR122
         B    (RETADR)                                              WSPSR123
JLTCA    ST   RETADR,,0                                             WSPSR124
         HR   1,,7                                                  WSPSR125
         HL   1,,7                                                  WSPSR126
         ST   JUMP,,7                                               WSPSR127
         C    COMP,,3                                               WSPSR128
         BL   (JUMP)                                                WSPSR129
         B    (RETADR)                                              WSPSR130
IOOPOPR  ST   RETADR,,0                                             WIOSR201
         TR   1,3                                                   WIOSR202
         BM   IOERR                                                 WIOSR203
         S    CONST6,,1                                             WIOSR204
         BP   IOERR                                                 WIOSR205
         HL   1,,3                                                  WIOSR206
         B    *+2,3                                                 WIOSR207
         B    (RETADR)                                              WIOSR208
         B    READCHR                                               WIOSR209
         B    WRITECHR                                              WIOSR210
         B    READL                                                 WIOSR211
         B    WRITEL                                                WIOSR212
         B    EOFL                                                  WIOSR213
         B    REWINDL                                               WIOSR214
READL    C    CONST3,,2                                             WIOSR249
         BNE  IOERR                                                 WIOSR250
         ST   IOOP,,6                                               WIOSR251
         LD   (IOOP),,6                                             WIOSR252
         TR   1,7                                                   WIOSR253
         HR   1,,1                                                  WIOSR254
         C    CONST1,,1                                             WIOSR255
         BNE  IOERR                                                 WIOSR256
         CALLF F,GETS,CRREAD                                        WIOSR257
         L    CONST81,,2                                            WIOSR258
         ST   LIMIT,,2                                              WIOSR258
         B    IOOPRES                                               WIOSR259
WRITEL   C    CONST3,,2                                             WIOSR260
         BNE  IOERR                                                 WIOSR260
         ST   IOOP,,6                                               WIOSR260
         LD   (IOOP),,6                                             WIOSR261
         TR   1,7                                                   WIOSR262
         HR   1,,1                                                  WIOSR263
         C    CONST4,,1                                             WIOSR264
         BNE  IOERR                                                 WIOSR265
SPINS    L    LIMIT,,2
         S    CONST1,,2
         L    CONSTSP,,3
SPINSC   C    CONST136,,2
         BNL  ACTWR
```

```
         STB  LBUF,2,3                                          WIOSR265
         A    CONST1,,2                                         WIOSR266
         B    SPINSC                                            WIOSR267
ACTWR    CALLF F,WRITE,LPWRITE                                  WIOSR268
         CALLF F,CHECK,LPCHECK                                  WIOSR269
         B    IOOPRES                                           WIOSR270
IOOPRES  TR   4,6                                               WIOSR271
         HRD  1,,4                                              WIOSR272
         L    CONST0,,4                                         WIOSR273
         HLD  1,,4                                              WIOSR274
         TR   6,4                                               WIOSR275
         STD  (IOOP),,6                                         WIOSR276
         B    (RETADR)                                          WIOSR277
READCHR  C    CONST3,,2                                         WIOSR299
         BE   RDTOIM                                            WIOSR300
         C    CONST2,,2                                         WIOSR301
         BE   RDTOCA                                            WIOSR302
         LD   ZNIL,,4                                           WIOSR303
         TR   2,4                                               WIOSR304
         HRD  1,,2                                              WIOSR305
         C    LIMIT,,2                                          WIOSR306
         BNL  NIL01D                                            WIOSR307
         C    CONST0,,2                                         WIOSR308
         BE   NIL01D                                            WIOSR309
         S    CONST1,,2                                         WIOSR310
         LB   LBUF,2,1                                          WIOSR311
         L    TABCI,1,0                                         WIOSR313
         C    CONSTNIL,,0                                       WIOSR314
         BE   NIL01D                                            WIOSR315
         TR   1,0                                               WIOSR316
         HR   1,,1                                              WIOSR317
         A    CONST2,,2                                         WIOSR318
         C    LIMIT,,2                                          WIOSR319
         BE   NIL00D                                            WIOSR320
SETNILD  HLD  1,,2                                              WIOSR321
         TR   4,2                                               WIOSR322
         STD  ZNIL,,4                                           WIOSR323
         TR   4,6                                               WIOSR324
         HRD  1,,4                                              WIOSR325
         TR   4,1                                               WIOSR326
         HLD  1,,4                                              WIOSR327
         TR   6,4                                               WIOSR328
         B    (RETADR)                                          WIOSR329
NIL00D   L    CONST0,,2                                         WIOSR330
         B    SETNILD                                           WIOSR331
NIL01D   L    CONSTNIL,,1                                       WIOSR332
         HR   1,,1                                              WIOSR333
         L    CONST1,,2                                         WIOSR334
         B    SETNILD                                           WIOSR335
RDTOCA   LD   ZNIL,,4                                           WIOSR336
         TR   2,4                                               WIOSR337
         HRD  1,,2                                              WIOSR338
         C    LIMIT,,2                                          WIOSR339
         BNL  NIL01A                                            WIOSR340
         C    CONST0,,2                                         WIOSR341
```

```
         BE    NIL01A                                                  WIOSR342
         S     CONST1,,2                                               WIOSR343
         LB    LBUF,2,1                                                WIOSR344
         L     TABCI,1,0                                               WIOSR346
         C     CONSTNIL,,0                                             WIOSR347
         BE    NIL01A                                                  WIOSR348
         TR    1,0                                                     WIOSR349
         HR    1,,1                                                    WIOSR350
         A     CONST2,,2                                               WIOSR351
         C     LIMIT,,2                                                WIOSR352
         BE    NIL00A                                                  WIOSR353
SETNILA  HLD   1,,2                                                    WIOSR354
         TR    4,2                                                     WIOSR355
         STD   ZNIL,,4                                                 WIOSR356
         TR    4,7                                                     WIOSR357
         HRD   1,,4                                                    WIOSR358
         TR    4,1                                                     WIOSR359
         HLD   1,,4                                                    WIOSR360
         TR    7,4                                                     WIOSR361
         B     (RETADR)                                                WIOSR362
NIL00A   L     CONST0,,2                                               WIOSR363
         B     SETNILA                                                 WIOSR364
NIL01A   L     CONSTNIL,,1                                             WIOSR365
         HR    1,,1                                                    WIOSR366
         L     CONST1,,2                                               WIOSR367
         B     SETNILA                                                 WIOSR368
RDTOIM   B     IOERR                                                   WIOSR369
WRITECHR C     CONST3,,2                                               WIOSR399
         BE    WRFROMIM                                                WIOSR400
         C     CONST2,,2                                               WIOSR401
         BE    WRFROMCA                                                WIOSR402
         TR    1,6                                                     WIOSR403
FETNIL   LD    ZNIL,,4                                                 WIOSR404
         TR    2,4                                                     WIOSR405
         HRD   1,,2                                                    WIOSR406
         C     CONST136,,2                                             WIOSR407
         BNL   NIL01                                                   WIOSR408
         C     CONST0,,2                                               WIOSR409
         BE    NIL01                                                   WIOSR410
         HR    1,,1                                                    WIOSR411
         HL    1,,1                                                    WIOSR412
         C     CONSTNIL,,1                                             WIOSR413
         BE    CRINS                                                   WIOSR414
         C     CONSTBEG,,1                                             WIOSR415
         BL    NOTCHAR                                                 WIOSR416
         C     CONSTEND,,1                                             WIOSR417
         BH    NOTCHAR                                                 WIOSR418
         S     CONST1,,2                                               WIOSR419
         S     CONSTBEG,,1                                             WIOSR420
         HR    1,,1                                                    WIOSR421
         LB    TABIC,1,0                                               WIOSR422
         STB   LBUF,2,0                                                WIOSR423
NILINCR  A     CONST2,,2                                               WIOSR424
         C     CONST136,,2                                             WIOSR425
         BE    NILLIM                                                  WIOSR426
```

```
SETNILW  HLD  1,,2                                          WIOSR427
         TR   4,2                                          WIOSR428
         STD  ZNIL,,4                                      WIOSR429
         B    (RETADR)                                     WIOSR430
CRINS    ST   LIMIT,,2
NIL01    L    CONST1,,2                                    WIOSR431
         B    SETNILW                                      WIOSR432
NILLIM   ST   LIMIT,,2
         L    CONST0,,2                                    WIOSR440
         B    SETNILW                                      WIOSR441
NOTCHAR  S    CONST1,,2                                    WIOSR442
         L    CONOTCHR,,0                                  WIOSR443
         STB  LBUF,2,0                                     WIOSR444
         B    NILINCR                                      WIOSR445
WRFROMCA TR   1,7                                          WIOSR446
         B    FETNIL                                       WIOSR447
WRFROMIM TR   1,6                                          WIOSR448
         B    FETNIL                                       WIOSR449
EOFL     B    IOERR
REWINDL  B    IOERR
         END
```

WSPT

```
      IO 03 ON INPUT, IO 04 ON PRINT.                                  WSPT0001
      READ INPUT, WRITE PRINT.                                         WSPT0002
      READ INPUT, TO L1 IF 01 = 01.                                    WSPT0003
      WRITE PRINT.                                                     WSPT0004
L1,   READ INPUT, TO L2 IF 01 = 02, TO L3.                             WSPT0005
L2,   WRITE PRINT.                                                     WSPT0006
L3,   READ INPUT, TO L4 IF 01 NE 02.                                   WSPT0007
      WRITE PRINT.                                                     WSPT0008
L4,   READ INPUT, TO L5 IF 01 NE 01, TO L6.                            WSPT0009
L5,   WRITE PRINT.                                                     WSPT0010
L6,   READ INPUT, TO L7 IF 01 LT 02.                                   WSPT0011
      WRITE PRINT.                                                     WSPT0012
L7,   READ INPUT, TO L8 IF 01 LT 01, TO L9.                            WSPT0013
L8,   WRITE PRINT.                                                     WSPT0014
L9,   READ INPUT, TO L10 IF 02 LT 01, TO L11.                          WSPT0015
L10,  WRITE PRINT.                                                     WSPT0016
L11,  READ INPUT, WRITE PRINT.                                         WSPT0017
      READ INPUT, TO L12 IF ONE = (PTR).                               WSPT0018
      WRITE PRINT.                                                     WSPT0019
L12,  READ INPUT, TO L13 IF PTR NE (PTR).                              WSPT0020
      WRITE PRINT.                                                     WSPT0021
L13,  READ INPUT, TO L14 IF PRINT LT (PTR).                            WSPT0022
      WRITE PRINT.                                                     WSPT0023
L14,  READ INPUT, TO L15 IF 00 = AF PTR.                               WSPT0024
      WRITE PRINT.                                                     WSPT0025
L15,  READ INPUT, TO L16 IF 01 = BPF PTR.                              WSPT0026
      WRITE PRINT.                                                     WSPT0027
L16,  READ INPUT, TO L17 IF 02 = CAR PTR.                              WSPT0028
      WRITE PRINT.                                                     WSPT0029
L17,  READ INPUT, TO L18 IF 03 = CDR PTR.                              WSPT0030
      WRITE PRINT.                                                     WSPT0031
L18,  READ INPUT, WRITE PRINT.                                         WSPT0032
      READ INPUT, TO L19 IF (PTR2) = PTR.                              WSPT0033
      WRITE PRINT.                                                     WSPT0034
L19,  READ INPUT, TO L20 IF AF PTR2 = 01.                              WSPT0035
      WRITE PRINT.                                                     WSPT0036
L20,  READ INPUT, TO L21 IF BPF PTR2 = 00.                             WSPT0037
      WRITE PRINT.                                                     WSPT0038
L21,  READ INPUT, TO L22 IF CAR PTR2 = 04.                             WSPT0039
      WRITE PRINT.                                                     WSPT0040
L22,  READ INPUT, TO L23 IF CDR PTR2 = ONE.                            WSPT0041
      WRITE PRINT.                                                     WSPT0042
L23,  READ INPUT, WRITE PRINT.                                         WSPT0043
      READ INPUT, (PTR2) = ONE.                                        WSPT0044
      TO L24 IF (PTR2) = ONE, WRITE PRINT.                             WSPT0045
L24,  READ INPUT, AF PTR = 01.                                         WSPT0046
      TO L25 IF AF PTR = 01, WRITE PRINT.                              WSPT0047
L25,  READ INPUT, BPF PTR = 00.                                        WSPT0048
      TO L26 IF BPF PTR = 00, WRITE PRINT.                             WSPT0049
L26,  READ INPUT, CAR PTR = 05.                                        WSPT0050
```

```
      TO L27 IF CAR PTR = 05, WRITE PRINT.                          WSPT0051
L27,  READ INPUT, CDR PTR = 06.                                     WSPT0052
      TO L28 IF CDR PTR = 06, WRITE PRINT.                          WSPT0053
L28,  READ INPUT, WRITE PRINT.                                      WSPT0054
      READ INPUT, INCR AF PTR3.                                     WSPT0055
      TO L29 IF AF PTR3 = 00, WRITE PRINT.                          WSPT0056
L29,  READ INPUT, INCR BPF PTR3.                                    WSPT0057
      TO L30 IF BPF PTR3 = 01, WRITE PRINT.                         WSPT0058
L30,  READ INPUT, INCR (PTR3).                                      WSPT0059
      TO L31 IF (PTR3) = PTR2, WRITE PRINT.                         WSPT0060
L31,  READ INPUT, INCR AF PTR4.                                     WSPT0061
      TO L32 IF AF PTR4 = 01, WRITE PRINT.                          WSPT0062
L32,  READ INPUT, INCR BPF PTR4.                                    WSPT0063
      TO L33 IF BPF PTR4 = 00, WRITE PRINT.                         WSPT0064
L33,  READ INPUT, INCR CAR PTR4.                                    WSPT0065
      TO L34 IF CAR PTR4 = PTR3, WRITE PRINT.                       WSPT0066
L34,  READ INPUT, INCR CDR PTR4.                                    WSPT0067
      TO L35 IF CDR PTR4 = INC, WRITE PRINT.                        WSPT0068
L35,  READ INPUT, WRITE PRINT.                                      WSPT0069
      READ INPUT, TO (LB1) IF 01 = 01.                              WSPT0070
      WRITE PRINT.                                                  WSPT0071
L36,  READ INPUT, TO (LB2) IF 01 = 02, TO L38.                      WSPT0072
L37,  WRITE PRINT.                                                  WSPT0073
L38,  READ INPUT, TO CAR LB3 IF 01 = 01.                            WSPT0074
      WRITE PRINT.                                                  WSPT0075
L39,  READ INPUT, TO CAR LB4 IF 01 = 02, TO L41.                    WSPT0076
L40,  WRITE PRINT.                                                  WSPT0077
L41,  READ INPUT, TO CDR LB3 IF 01 = 01.                            WSPT0078
      WRITE PRINT.                                                  WSPT0079
L42,  READ INPUT, TO CDR LB4 IF 01 = 02, TO L44.                    WSPT0080
L43,  WRITE PRINT.                                                  WSPT0081
L44,  READ INPUT, WRITE PRINT.                                      WSPT0082
      READ INPUT, TO (LB5) IF 01 NE 02.                             WSPT0083
      WRITE PRINT.                                                  WSPT0084
L45,  READ INPUT, TO (LB6) IF 01 NE 01, TO L47.                     WSPT0085
L46,  WRITE PRINT.                                                  WSPT0086
L47,  READ INPUT, TO CAR LB7 IF 01 NE 02.                           WSPT0087
      WRITE PRINT.                                                  WSPT0088
L48,  READ INPUT, TO CAR LB8 IF 01 NE 01, TO L50.                   WSPT0089
L49,  WRITE PRINT.                                                  WSPT0090
L50,  READ INPUT, TO CDR LB7 IF 01 NE 02.                           WSPT0091
      WRITE PRINT.                                                  WSPT0092
L51,  READ INPUT, TO CDR LB8 IF 01 NE 01, TO L53.                   WSPT0093
L52,  WRITE PRINT.                                                  WSPT0094
L53,  READ INPUT, WRITE PRINT.                                      WSPT0095
      READ INPUT, TO (LB9) IF 01 LT 02.                             WSPT0096
      WRITE PRINT.                                                  WSPT0097
L54,  READ INPUT, TO (LB10) IF 01 LT 01, TO L56.                    WSPT0098
L55,  WRITE PRINT.                                                  WSPT0099
L56,  READ INPUT, TO (LB11) IF 02 LT 01, TO L58.                    WSPT0100
L57,  WRITE PRINT.                                                  WSPT0101
L58,  READ INPUT, TO CAR LB12 IF 01 LT 02.                          WSPT0102
      WRITE PRINT.                                                  WSPT0103
L59,  READ INPUT, TO CAR LB13 IF 01 LT 01, TO L61.                  WSPT0104
L60,  WRITE PRINT.                                                  WSPT0105
```

```
L61,  READ INPUT, TO CAR LB14 IF 02 LT 01, TO L63.                    WSPT0106
L62,  WRITE PRINT.                                                    WSPT0107
L63,  READ INPUT, TO CDR LB12 IF 01 LT 02,                            WSPT0108
      WRITE PRINT.                                                    WSPT0109
L64,  READ INPUT, TO CDR LB13 IF 01 LT 01, TO L66.                    WSPT0110
L65,  WRITE PRINT.                                                    WSPT0111
L66,  READ INPUT, TO CDR LB14 IF 02 LT 01, TO L68.                    WSPT0112
L67,  WRITE PRINT.                                                    WSPT0113
L68,  READ INPUT, WRITE PRINT.                                        WSPT0114
      (NIL) = 01.                                                     WSPT0115
      READ INPUT, IO 01 ON (ONE).                                     WSPT0116
      TO L69 IF (ONE) = ' , WRITE PRINT.                              WSPT0117
L69,  READ INPUT, IO 01 ON CAR PTR.                                   WSPT0118
      TO L70 IF CAR PTR = ' , WRITE PRINT.                            WSPT0119
L70,  READ INPUT, IO 01 ON CDR PTR.                                   WSPT0120
      TO L71 IF (ONE) = 'I, WRITE PRINT.                              WSPT0121
L71,  READ INPUT, TO L72 IF (NIL) = 04.                               WSPT0122
      WRITE PRINT.                                                    WSPT0123
L72,  READ INPUT, WRITE PRINT.                                        WSPT0124
L73,  IO 01 ON A, TO L73 IF A NE NIL.                                 WSPT0125
      READ INPUT, WRITE PRINT.                                        WSPT0126
      READ INPUT, TO L74 IF (NIL) = 01.                               WSPT0127
      WRITE PRINT.                                                    WSPT0128
L74,  READ INPUT, (NIL) = 01.                                         WSPT0129
      IO 02 ON ' , TO L75 IF (NIL) = 02.                              WSPT0130
      WRITE PRINT, STOP.                                              WSPT0131
L75,  READ INPUT, WRITE PRINT.                                        WSPT0132
      READ INPUT, WRITE PRINT.                                        WSPT0133
      (NIL) = 01.                                                     WSPT0134
L76,  IO 02 ON '+, TO L76 IF (NIL) NE 00.                             WSPT0135
      IO 02 ON '/, WRITE PRINT.                                       WSPT0136
      READ INPUT, WRITE PRINT.                                        WSPT0137
      (NIL) = 02, IO 02 ON '6.                                        WSPT0138
      (NIL) = 04, IO 02 ON NIL, WRITE PRINT.                          WSPT0139
      READ INPUT, TO L77 IF (NIL) = 01.                               WSPT0140
      WRITE PRINT.                                                    WSPT0141
L77,  READ INPUT, WRITE PRINT, STOP.                                  WSPT0142
INPUT,ELEMENT 00 00 01 00.                                            WSPT0143
PRINT,ELEMENT 00 00 04 00.                                            WSPT0144
ONE,  ELEMENT 00 01 02 03.                                            WSPT0145
PTR,  ELEMENT 01 00 04 ONE.                                           WSPT0146
PTR2, ELEMENT 00 00 00 PTR.                                           WSPT0147
PTR3, ELEMENT 00 01 00 PTR.                                           WSPT0148
INC,  ELEMENT 00 01 PTR2 PTR3.                                        WSPT0149
PTR4, ELEMENT 00 00 00 INC.                                           WSPT0150
LB1,  ELEMENT 00 00 00 L36.                                           WSPT0151
LB2,  ELEMENT 00 00 00 L37.                                           WSPT0152
LB3,  ELEMENT 00 00 00 LB31.                                          WSPT0153
LB31, ELEMENT 00 00 L39 L42.                                          WSPT0154
LB4,  ELEMENT 00 00 00 LB41.                                          WSPT0155
LB41, ELEMENT 00 00 L40 L43.                                          WSPT0156
LB5,  ELEMENT 00 00 00 L45.                                           WSPT0157
LB6,  ELEMENT 00 00 00 L47.                                           WSPT0158
LB7,  ELEMENT 00 00 00 LB71.                                          WSPT0159
LB71, ELEMENT 00 00 L48 L51.                                          WSPT0160
```

```
LB8,  ELEMENT 00 00 00 LB81.                                    WSPT0161
LB81, ELEMENT 00 00 L49 L52.                                    WSPT0162
LB9,  ELEMENT 00 00 00 L54.                                     WSPT0163
LB10, ELEMENT 00 00 00 L55.                                     WSPT0164
LB11, ELEMENT 00 00 00 L57.                                     WSPT0165
LB12, ELEMENT 00 00 00 LB121.                                   WSPT0166
LB121,ELEMENT 00 00 L59 L64.                                    WSPT0167
LB13, ELEMENT 00 00 00 LB131.                                   WSPT0168
LB131,ELEMENT 00 00 L60 L65.                                    WSPT0169
LB14, ELEMENT 00 00 00 LB141.                                   WSPT0170
LB141,ELEMENT 00 00 L62 L67.                                    WSPT0171
      END PROGRAM.                                              WSPT0172
```

WSPD

```
1)  IF THE MACROS ARE ALL CORRECT, THE OUTPUT CONSISTS OF NUMBERED      WSPD0001
2)  LINES ONLY.  THE FIRST TWO LINES DEPEND ONLY UPON RECORD I/O.       WSPD0002
  TO L1 IF 01 = 01 FAILS                                               WSPD0003
  TO L2 IF 01 = 02 FAILS                                               WSPD0004
  TO L4 IF 01 NE 02 FAILS                                              WSPD0005
  TO L5 IF 01 NE 01 FAILS                                              WSPD0006
  TO L7 IF 01 LT 02 FAILS                                              WSPD0007
  TO L8 IF 01 LT 01 FAILS                                              WSPD0008
  TO L10 IF 02 LT 01 FAILS                                             WSPD0009
3)  TESTS OF J--IM COMPLETE.                                           WSPD0010
  TO L12 IF ONE = (PTR) FAILS                                          WSPD0011
  TO L13 IF PTR NE (PTR) FAILS                                         WSPD0012
  TO L14 IF PRINT LT (PTR) FAILS                                       WSPD0013
  TO L15 IF 00 = AF PTR FAILS                                          WSPD0014
  TO L16 IF 01 = BPF PTR FAILS                                         WSPD0015
  TO L17 IF 02 = CAR PTR FAILS                                         WSPD0016
  TO L18 IF 03 = CDR PTR FAILS                                         WSPD0017
4)  TESTS OF CMP COMPLETE.                                             WSPD0018
  TO L19 IF (PTR2) = PTR FAILS                                         WSPD0019
  TO L20 IF AF PTR2 = 01 FAILS                                         WSPD0020
  TO L21 IF BPF PTR2 = 00 FAILS                                        WSPD0021
  TO L22 IF CAR PTR2 = 04 FAILS                                        WSPD0022
  TO L23 IF PTR2 = ONE FAILS                                           WSPD0023
5)  TESTS OF FET COMPLETE.                                             WSPD0024
  (PTR2) = ONE FAILS                                                   WSPD0025
  AF PTR = 01 FAILS                                                    WSPD0026
  BPF PTR = 00 FAILS                                                   WSPD0027
  CAR PTR = 05 FAILS                                                   WSPD0028
  CDR PTR = 06 FAILS                                                   WSPD0029
6)  TESTS OF STO COMPLETE.                                             WSPD0030
  INCR AF PTR3 FAILS                                                   WSPD0031
  INCR BPF PTR3 FAILS                                                  WSPD0032
  INCR (PTR3) FAILS                                                    WSPD0033
  INCR AF PTR4 FAILS                                                   WSPD0034
  INCR BPF PTR4 FAILS                                                  WSPD0035
  INCR CAR PTR4 FAILS                                                  WSPD0036
  INCR CDR PTR4 FAILS                                                  WSPD0037
7)  TESTS OF INC COMPLETE.                                             WSPD0038
  TO (LB1) IF 01 = 01 FAILS                                            WSPD0039
  TO (LB2) IF 01 = 02 FAILS                                            WSPD0040
  TO CAR LB3 IF 01 = 01 FAILS                                          WSPD0041
  TO CAR LB4 IF 01 = 02 FAILS                                          WSPD0042
  TO CDR LB3 IF 01 = 01 FAILS                                          WSPD0043
  TO CDR LB4 IF 01 = 02 FAILS                                          WSPD0044
8)  TESTS OF JE@ COMPLETE.                                             WSPD0045
  TO (LB5) IF 01 NE 02 FAILS                                           WSPD0046
  TO (LB6) IF 01 NE 01 FAILS                                           WSPD0047
  TO CAR LB7 IF 01 NE 02 FAILS                                         WSPD0048
  TO CAR LB8 IF 01 NE 01 FAILS                                         WSPD0049
  TO CDR LB7 IF 01 NE 02 FAILS                                         WSPD0050
```

```
  TO CDR LB8 IF 01 NE 01 FAILS                                          WSPD0051
9)  TESTS OF JNE COMPLETE.                                              WSPD0052
  TO (LB9) IF 01 LT 02 FAILS                                            WSPD0053
  TO (LB10) IF 01 LT 01 FAILS                                           WSPD0054
  TO (LB11) IF 02 LT 01 FAILS                                           WSPD0055
  TO CAR LB12 IF 01 LT 02 FAILS                                         WSPD0056
  TO CAR LB13 IF 01 LT 01 FAILS                                         WSPD0057
  TO CAR LB14 IF 02 LT 01 FAILS                                         WSPD0058
  TO CDR LB12 IF 01 LT 02 FAILS                                         WSPD0059
  TO CDR LB13 IF 01 LT 01 FAILS                                         WSPD0060
  TO CDR LB14 IF 02 LT 01 FAILS                                         WSPD0061
10) TESTS OF JLT COMPLETE.                                              WSPD0062
  IO ON (ONE) FAILS                                                     WSPD0063
  IO ON CAR PTR FAILS                                                   WSPD0064
  IO ON CDR PTR FAILS                                                   WSPD0065
  LINE BUFFER POINTER IS NOT INCREMENTED BY IO 01                       WSPD0066
11) IF THIS IS THE LAST LINE, NO CR/LF WAS FOUND IN THE LINE BUFFER.    WSPD0067
12) A CR/LF WAS FOUND IN THE LINE BUFFER.                               WSPD0068
  THE LINE BUFFER POINTER DID NOT RETURN TO 1 AFTER THE CR/LF WAS FOUND.WSPD0069
  A WRITE INTO THE LINE BUFFER DOES NOT ALTER THE POINTER. TEST ABORTS. WSPD0070
13) IF (14) IS THE LAST LINE, THE LBP DOES NOT WRAP AROUND AT FULL LB.  WSPD0071
14) A FULL LINE OF PLUSES FOLLOWS. THE SLASH IN LB(0) MUST NOT PRINT.   WSPD0072
15) IF LINE 16 IS NOT BLANK, WRITING NIL DOES NOT TERMINATE A LINE.     WSPD0073
  LINE BUFFER POINTER IS NOT RESET TO 1 AFTER NIL IS INSERTED.          WSPD0074
17) WISP MACRO TEST PROGRAM COMPLETED NORMALLY.                         WSPD0075
```

HELP

```
        USE INITL.                                              HELP0001
        A = NIL. F = NIL.                                       HELP0002
        READ CHAR. (NIL) = 01.                                  HELP0003
        C = ' .                                                 HELP0004
        (CHAR) = NIL. USE FINPUT.                               HELP0005
        TO QUIT IF Z NE 01.                                     HELP0006
        (TRUE) = CDR Y.                                         HELP0007
        USE FINPUT.                                             HELP0008
        TO QUIT IF Z NE 11.                                     HELP0009
        USE FINPUT.                                             HELP0010
        TO QUIT IF Z NE 01.                                     HELP0011
        (FALSE) = CDR Y.                                        HELP0012
        USE FINPUT.                                             HELP0013
        TO QUIT IF Z NE 11.                                     HELP0014
        USE FINPUT.                                             HELP0015
        TO QUIT IF Z NE 03.                                     HELP0016
        X = NEW ELEMENT.                                        HELP0017
        CAR X = L11. CDR X = 'F.                                HELP0018
        Y = CDR Y. CAR Y = X.                                   HELP0019
        USE FINPUT.                                             HELP0020
        TO QUIT IF Z NE 03.                                     HELP0021
        X = NEW ELEMENT.                                        HELP0022
        CAR X = L12. CDR X = 'F.                                HELP0023
        Y = CDR Y. CAR Y = X.                                   HELP0024
        USE FINPUT.                                             HELP0025
        TO QUIT IF Z NE 03.                                     HELP0026
        X = NEW ELEMENT.                                        HELP0027
        CAR X = L13. CDR X = 'F.                                HELP0028
        Y = CDR Y. CAR Y = X.                                   HELP0029
        USE FINPUT.                                             HELP0030
        TO QUIT IF Z NE 03.                                     HELP0031
        X = NEW ELEMENT.                                        HELP0032
        CAR X = L14. CDR X = 'F.                                HELP0033
        Y = CDR Y. CAR Y = X.                                   HELP0034
        USE FINPUT.                                             HELP0035
        TO QUIT IF Z NE 03.                                     HELP0036
        X = NEW ELEMENT.                                        HELP0037
        CAR X = L15. CDR X = 'F.                                HELP0038
        Y = CDR Y. CAR Y = X.                                   HELP0039
        USE FINPUT.                                             HELP0040
        TO QUIT IF Z NE 09.                                     HELP0041
        @ = NEW ELEMENT. CDR @ = NIL.                           HELP0042
        T = NEW ELEMENT.                                        HELP0043
NEWR.   R = NEW ELEMENT.                                        HELP0044
NEXTE.  CAR R = NIL. CDR R = NIL.                               HELP0045
        CAR T = EVAL. CDR T = NIL.                              HELP0046
        V = NIL.                                                HELP0047
        (NIL) = 01. LINE BUFFER = NIL.                          HELP0048
        WRITE OUTPUT.                                           HELP0049
        (CHAR) = NIL. USE FINPUT.                               HELP0050
```

```
        TO EX IF Z NE 12.                                          HELP0051
        USE FINPUT.                                                HELP0052
        P = CDR Y.                                                 HELP0053
        TO ERR18 IF Z NE 03.                                       HELP0054
        TO ERR19 IF CAR Y NE NIL.                                  HELP0055
        USE FINPUT.                                                HELP0056
        TO FIXDEF IF Z = 08.                                       HELP0057
BVARS.  TO ERR20 IF Z NE 02.                                       HELP0058
        TO ERR21 IF CDR Y NE NIL.                                  HELP0059
        CAR R = L4, PUSH DOWN R.                                   HELP0060
        CDR Y = CDR R.                                             HELP0061
        USE FINPUT.                                                HELP0062
        TO FIXDEF IF Z = 08.                                       HELP0063
        TO ERR22 IF Z NE 11.                                       HELP0064
        USE FINPUT, TO BVARS.                                      HELP0065
ERR18.  1 = '1, 2 = '8, TO ERROR.                                  HELP0066
ERR19.  1 = '1, 2 = '9, TO ERROR.                                  HELP0067
ERR20.  1 = '2, 2 = '0, TO ERROR.                                  HELP0068
ERR21.  1 = '2, 2 = '1, TO ERROR.                                  HELP0069
ERR22.  1 = '2, 2 = '2, TO ERROR.                                  HELP0070
FIXDEF. USE FINPUT.                                                HELP0071
        CAR T = FIXDF1.                                            HELP0072
        TO REX IF Z = 19.                                          HELP0073
        1 = '2, 2 = '3, TO ERROR.                                  HELP0074
FIXDF1. TO L01 IF Z NE 09.                                         HELP0075
        CAR P = R.                                                 HELP0076
        (NIL) = 01.                                                HELP0077
        Y = P, USE PRATOM.                                         HELP0078
        LINE BUFFER = NIL, WRITE OUTPUT.                           HELP0079
        TO NEWR.                                                   HELP0080
.                                                                  HELP0081
.       ANALYZER FOR EXPRESSION.                                   HELP0082
.                                                                  HELP0083
REX.    USE FINPUT.                                                HELP0084
EX.     TO SE IF Z NE 13.                                          HELP0085
        PUSH DOWN T, CAR T = EX1.                                  HELP0086
        TO RSE.                                                    HELP0087
.                                                                  HELP0088
EX1.    PUSH DOWN R, CAR T = EX2.                                  HELP0089
        TO RSE IF Z = 14.                                          HELP0090
.                                                                  HELP0091
        1 = '1, 2 = '5, TO ERROR.                                  HELP0092
EX2.    PUSH DOWN R, CAR T = EX3.                                  HELP0093
        TO REX IF Z = 15.                                          HELP0094
.                                                                  HELP0095
        1 = '1, 2 = '6, TO ERROR.                                  HELP0096
EX3.    W = CDR R, X = CDR W.                                      HELP0097
        CDR R = CDR X.                                             HELP0098
        CDR W = CAR R, CDR X = W.                                  HELP0099
        PUSH DOWN X, CAR X = L10.                                  HELP0100
RETX.   CAR R = X.                                                 HELP0101
RET.    POP UP T, TO CAR T.                                        HELP0102
.                                                                  HELP0103
.       ANALYZER FOR SIMPLE EXPRESSION                             HELP0104
.                                                                  HELP0105
```

```
RSE.    USE FINPUT.                                                    HELP0106
SE.     PUSH DOWN T. CAR T = SE1.                                      HELP0107
        TO LE.                                                         HELP0108
SE1.    TO SE2 IF Z = 16.                                              HELP0109
        TO RET IF Z NE 17.                                             HELP0110
        PUSH DOWN R. CAR R = L9.                                       HELP0111
        TO SE3.                                                        HELP0112
SE2.    PUSH DOWN R. CAR R = L8.                                       HELP0113
SE3.    PUSH DOWN R.                                                   HELP0114
        CAR T = SE4. TO RSE.                                           HELP0115
.                                                                      HELP0116
SE4.    W = CDR R. X = CDR W.                                          HELP0117
        CDR R = CDR X.                                                 HELP0118
SE5.    CDR X = CAR R.                                                 HELP0119
        CAR R = W. TO RET.                                             HELP0120
.                                                                      HELP0121
.       ANALYZER FOR LOGICAL EXPRESSION                                HELP0122
.                                                                      HELP0123
RLE.    USE FINPUT.                                                    HELP0124
LE.     PUSH DOWN T.                                                   HELP0125
        TO LE4 IF Z = 18.                                              HELP0126
        CAR T = LE1. TO PR.                                            HELP0127
LE1.    TO LE2 IF Z = 06.                                              HELP0128
        TO RET IF Z NE 10.                                             HELP0129
        PUSH DOWN R. CAR R = L7.                                       HELP0130
        TO LE3.                                                        HELP0131
LE2.    PUSH DOWN R. CAR R = L6.                                       HELP0132
LE3.    PUSH DOWN R.                                                   HELP0133
        CAR T = SE4. TO RPR.                                           HELP0134
.                                                                      HELP0135
LE4.    CAR T = LE5. TO RLE.                                           HELP0136
.                                                                      HELP0137
LE5.    X = NEW ELEMENT.                                               HELP0138
        CAR X = L5. CDR X = CAR R.                                     HELP0139
        TO RETX.                                                       HELP0140
.                                                                      HELP0141
.       ANALYZER FOR PRIMARY                                           HELP0142
.                                                                      HELP0143
RPR.    USE FINPUT.                                                    HELP0144
PR.     TO PR4 IF Z = 01.                                              HELP0145
        TO PR5 IF Z = 02.                                              HELP0146
        TO PR6 IF Z = 03.                                              HELP0147
        TO PR15 IF Z = 04.                                             HELP0148
        PUSH DOWN T. CAR T = PR1.                                      HELP0149
        TO REX IF Z = 07.                                              HELP0150
        1 = '1. 2 = '0. TO ERROR.                                      HELP0151
PR1.    TO PR2 IF Z = 08.                                              HELP0152
        1 = '1. 2 = '4. TO ERROR.                                      HELP0153
PR2.    POP UP T.                                                      HELP0154
PR3.    USE FINPUT. TO CAR T.                                          HELP0155
PR4.    X = NEW ELEMENT.                                               HELP0156
        CAR X = L1. CDR X = CDR Y.                                     HELP0157
        CAR R = X. TO PR3.                                             HELP0158
.                                                                      HELP0159
.       BOUND VARIABLE                                                 HELP0160
```

```
.                                                                   HELP0161
PR5,    Y = CDR Y, CAR R = Y.                                       HELP0162
        TO PR3 IF CAR Y = L4.                                       HELP0163
.                                                                   HELP0164
        1 = '1, 2 = '1, TO ERROR.                                   HELP0165
.                                                                   HELP0166
.       FUNCTION CALL                                               HELP0167
.                                                                   HELP0168
PR6,    CAR R = CDR Y, PUSH DOWN R.                                 HELP0169
        USE FINPUT.                                                 HELP0170
        TO PR14 IF Z = 08.                                          HELP0171
        PUSH DOWN T, CAR T = PR11.                                  HELP0172
.                                                                   HELP0173
.       ANALYZE A LIST OF EXPRESSIONS                               HELP0174
.                                                                   HELP0175
PR7,    CAR R = 'F, PUSH DOWN R.                                    HELP0176
        PUSH DOWN T, CAR T = PR8.                                   HELP0177
        TO EX.                                                      HELP0178
PR8,    TO PR9 IF Z NE 11.                                          HELP0179
        PUSH DOWN R, TO REX.                                        HELP0180
PR9,    X = NIL.                                                    HELP0181
PR10,   W = X, X = R, R = CDR R.                                    HELP0182
        CDR X = W.                                                  HELP0183
        TO PR10 IF CAR R NE 'F.                                     HELP0184
        PUSH DOWN X, TO RET.                                        HELP0185
.                                                                   HELP0186
.       COMPLETE A FUNCTION CALL                                    HELP0187
.                                                                   HELP0188
PR11,   POP UP T.                                                   HELP0189
        TO PR12 IF Z = 08.                                          HELP0190
        1 = '1, 2 = '2, TO ERROR.                                   HELP0191
PR14,   X = NEW ELEMENT.                                            HELP0192
        CDR X = NIL.                                                HELP0193
PR12,   W = R, R = CDR R.                                           HELP0194
        CAR X = CAR R.                                              HELP0195
        CAR W = L3, CDR W = X.                                      HELP0196
        CAR R = W, TO PR3.                                          HELP0197
.                                                                   HELP0198
.       ANALYZE A BRACKETED LIST                                    HELP0199
.                                                                   HELP0200
PR15,   USE FINPUT.                                                 HELP0201
        TO PR18 IF Z = 05.                                          HELP0202
        PUSH DOWN T, CAR T = PR16.                                  HELP0203
        TO PR7.                                                     HELP0204
PR16,   POP UP T.                                                   HELP0205
        TO PR17 IF Z = 05.                                          HELP0206
        1 = '1, 2 = '3, TO ERROR.                                   HELP0207
PR18,   X = NEW ELEMENT.                                            HELP0208
        CDR X = NIL.                                                HELP0209
PR17,   CAR X = L2, CAR R = X.                                      HELP0210
        TO PR3.                                                     HELP0211
.                                                                   HELP0212
.       EVALUATE AN EXPRESSION JUST COMPILED                        HELP0213
.                                                                   HELP0214
EVAL,   TO L01 IF Z NE 09.                                          HELP0215
```

```
        (NIL) = 01.                                                    HELP0216
        E = CAR R.                                                     HELP0217
        S = NEW ELEMENT.                                               HELP0218
        CAR S = NIL. CDR S = NIL.                                      HELP0219
        CAR T = LO. TO CAR E.                                          HELP0220
LO.                                                                    HELP0221
        Y = CAR R. USE PRLIST.                                         HELP0222
        LINE BUFFER = NIL.                                             HELP0223
        WRITE OUTPUT.                                                  HELP0224
        TO NEXTE.                                                      HELP0225
LO1.    1 = '1. 2 = '7.                                                HELP0226
ERROR.  TO NONE IF (NIL) = 01.                                         HELP0227
        Y = (NIL). (NIL) = 01.                                         HELP0228
DASH.   LINE BUFFER = '-.                                              HELP0229
        TO DASH IF (NIL) LT Y.                                         HELP0230
NONE.   WRITE OUTPUT.                                                  HELP0231
        TO MESSG IF Z = 09.                                            HELP0232
        TO SKPS1.                                                      HELP0233
SKPS.   USE NEXTCH.                                                    HELP0234
SKPS1.  TO SKPS IF (CHAR) NE ''.                                       HELP0235
        USE NEXTCH.                                                    HELP0236
        TO SKPS IF (CHAR) NE '.                                        HELP0237
MESSG.  (NIL) = 01.                                                    HELP0238
        LINE BUFFER = 'E.                                              HELP0239
        LINE BUFFER = 'R.                                              HELP0240
        LINE BUFFER = 'R.                                              HELP0241
        LINE BUFFER = 'O.                                              HELP0242
        LINE BUFFER = 'R.                                              HELP0243
        LINE BUFFER = ' .                                              HELP0244
        LINE BUFFER = 1.                                               HELP0245
        LINE BUFFER = 2.                                               HELP0246
        LINE BUFFER = NIL.                                             HELP0247
        WRITE OUTPUT.                                                  HELP0248
        TO IFAIL IF 1 = ' .                                            HELP0249
        TO SKIPL.                                                      HELP0250
NXCALL. Y = CAR @. USE PRATOM.                                         HELP0251
        LINE BUFFER = '(.                                              HELP0252
        TO FSTRG IF (NIL) NE 00.                                       HELP0253
        WRITE OUTPUT. (NIL) = 01.                                      HELP0254
FSTRG.  POP UP @. Y = CAR @.                                           HELP0255
        TO PARG IF Y NE 'F.                                            HELP0256
        TO ENDAL.                                                      HELP0257
SEP.    LINE BUFFER = '.                                               HELP0258
        TO PARG IF (NIL) NE 00.                                        HELP0259
        WRITE OUTPUT. (NIL) = 01.                                      HELP0260
PARG.   Y = CDR Y. USE PRLIST.                                         HELP0261
        POP UP @. Y = CAR @.                                           HELP0262
        TO SEP IF Y NE 'F.                                             HELP0263
ENDAL.  POP UP @.                                                      HELP0264
        LINE BUFFER = ').                                              HELP0265
        LINE BUFFER = NIL.                                             HELP0266
        WRITE OUTPUT.                                                  HELP0267
IFAIL.  TO NXCALL IF CDR @ NE NIL.                                     HELP0268
SKIPL.  LINE BUFFER = NIL.                                             HELP0269
        WRITE OUTPUT. WRITE OUTPUT.                                    HELP0270
```

```
        TO NEXTE.                                                    HELP0271
FAIL.   1 = ' , TO MESSG.                                            HELP0272
.                                                                    HELP0273
L1.         EVALUATE  AN ATOM                                        HELP0274
.                                                                    HELP0275
        CAR R = CDR E.                                               HELP0276
        TO CAR T.                                                    HELP0277
.                                                                    HELP0278
L2.         EVALUATE A LIST                                          HELP0279
.                                                                    HELP0280
        TO L21 IF CDR E NE NIL.                                      HELP0281
        CAR R = NIL. TO CAR T.                                       HELP0282
L21.    PUSH DOWN T. CAR T = L22.                                    HELP0283
        TO C23.                                                      HELP0284
L22.    X = NIL.                                                     HELP0285
L23.    W = R. R = CDR R.                                            HELP0286
        CDR W = X. X = W.                                            HELP0287
        TO L23 IF CAR R NE 'F.                                       HELP0288
        TO RETX.                                                     HELP0289
.                                                                    HELP0290
L3.         EVALUATE A FUNCTION CALL                                 HELP0291
.                                                                    HELP0292
        E = CDR E. W = CAR E.                                        HELP0293
        TO L322 IF CAR W = NIL.                                      HELP0294
        PUSH DOWN T.                                                 HELP0295
        TO L331 IF CDR E = NIL.                                      HELP0296
        PUSH DOWN S. CAR S = W.                                      HELP0297
        CAR T = L31.                                                 HELP0298
.                                                                    HELP0299
.         COMMON CODE FOR EVALUATING A LIST  OF EXPRESSIONS          HELP0300
.                                                                    HELP0301
C23.                                                                 HELP0302
        E = CDR E.                                                   HELP0303
        CAR R = 'F.                                                  HELP0304
        PUSH DOWN S.                                                 HELP0305
        PUSH DOWN T. CAR T = C232.                                   HELP0306
C231.   PUSH DOWN R.                                                 HELP0307
        CAR S = CDR E.                                               HELP0308
ECARE.  E = CAR E. TO CAR E.                                         HELP0309
C232.   E = CAR S.                                                   HELP0310
        TO C231 IF E NE NIL.                                         HELP0311
        POP UP S. TO RET.                                            HELP0312
.                                                                    HELP0313
L31.        RETURN FROM EVALUATING ARGUMENTS                         HELP0314
.                                                                    HELP0315
        W = CAR S. POP UP S.                                         HELP0316
        E = CAR W. Y = CDR E.                                        HELP0317
        TO CAR E IF Y = 'F.                                          HELP0318
        PUSH DOWN @. CAR @ = 'F.                                     HELP0319
.                                                                    HELP0320
L32.        SET UP NEW ASSOCIATIONS FOR THE BOUND VARIABLES          HELP0321
.                                                                    HELP0322
        TO L321 IF Y = NIL.                                          HELP0323
        Z = R. R = CDR R.                                            HELP0324
        CDR Z = CAR Z. CAR Z = Y.                                    HELP0325
```

```
        PUSH DOWN @, CAR @ = Z.                                          HELP0326
        Y = CDR Y.                                                       HELP0327
        TO L32 IF CAR R NE 'F.                                           HELP0328
        TO L321 IF Y NE NIL.                                             HELP0329
.                                                                        HELP0330
L33.      EVALUATE A USER-DEFINED FUNCTION                               HELP0331
.                                                                        HELP0332
        PUSH DOWN @, CAR @ = W.                                          HELP0333
        CAR T = L332, TO ECARE.                                          HELP0334
L331,   E = CAR W.                                                       HELP0335
        PUSH DOWN @, CAR @ = 'F.                                         HELP0336
        TO L33 IF CDR E = NIL.                                           HELP0337
L321,   PUSH DOWN @, CAR @ = W.                                          HELP0338
        2 = '0, TO FAIL.                                                 HELP0339
L322,   PUSH DOWN @, CAR @ = 'F.                                         HELP0340
        TO L321.                                                         HELP0341
L332,   POP UP @, TO L332 IF CAR @ NE 'F.                                HELP0342
        POP UP @, TO RET.                                                HELP0343
.                                                                        HELP0344
L4.       EVALUATE A BOUND VARIABLE                                      HELP0345
.                                                                        HELP0346
        Z = @.                                                           HELP0347
L41,    Z = CDR Z, Y = CAR Z.                                            HELP0348
        TO L41 IF CAR Y NE E.                                            HELP0349
        CAR R = CDR Y.                                                   HELP0350
        TO CAR T.                                                        HELP0351
.                                                                        HELP0352
L5.       NEGATE A LOGICAL EXPRESSION                                    HELP0353
.                                                                        HELP0354
        PUSH DOWN T, CAR T = L51.                                        HELP0355
ECDRE,  E = CDR E, TO CAR E.                                             HELP0356
L51,    TO L53 IF CAR R = (FALSE).                                       HELP0357
        TO L52 IF CAR R = (TRUE).                                        HELP0358
        2 = '6, TO FAIL.                                                 HELP0359
L52,    CAR R = (FALSE), TO RET.                                         HELP0360
L53,    CAR R = (TRUE), TO RET.                                          HELP0361
.                                                                        HELP0362
L6.       EVALUATE P1 = P2                                               HELP0363
.                                                                        HELP0364
        PUSH DOWN T, CAR T = L61.                                        HELP0365
.                                                                        HELP0366
.         COMMON CODE FOR THE EVALUATION OF TWO PRIMARIES                HELP0367
.                                                                        HELP0368
C67.                                                                     HELP0369
        E = CDR E.                                                       HELP0370
        PUSH DOWN S, CAR S = CDR E.                                      HELP0371
        PUSH DOWN T, CAR T = C671.                                       HELP0372
        TO ECARE.                                                        HELP0373
C671,   E = CAR R.                                                       HELP0374
        TO FAIL4 IF AF E = 00.                                           HELP0375
        PUSH DOWN R.                                                     HELP0376
        E = CAR S, POP UP S.                                             HELP0377
        CAR T = C672, TO CAR E.                                          HELP0378
FAIL4,  2 = '4, TO FAIL.                                                 HELP0379
C672,   E = CAR R, POP UP R.                                             HELP0380
```

```
      TO RET IF AF E = 01.                                          HELP0381
      2 = '5, TO FAIL.                                              HELP0382
.                                                                   HELP0383
.        CONTINUE PROCESSING P1 = P2.                               HELP0384
.                                                                   HELP0385
L61,  TO L53 IF CAR R = E, TO L52.                                  HELP0386
.                                                                   HELP0387
L7.      EVALUATE P1 NE P2                                          HELP0388
.                                                                   HELP0389
      PUSH DOWN T, CAR T = L71.                                     HELP0390
      TO C67.                                                       HELP0391
L71,  TO L53 IF CAR R NE E, TO L52.                                 HELP0392
.                                                                   HELP0393
L8.      EVALUATE LE AND SE                                         HELP0394
.                                                                   HELP0395
      PUSH DOWN T, CAR T = L81.                                     HELP0396
      TO L101.                                                      HELP0397
L81,  E = CAR S, POP UP S.                                          HELP0398
      TO RET IF CAR R = (FALSE).                                    HELP0399
      TO L92 IF CAR R NE (TRUE).                                    HELP0400
.                                                                   HELP0401
L82.     VALUE OF EXPRESSION IS VALUE OF SE                         HELP0402
.                                                                   HELP0403
      CAR T = L83, TO CAR E.                                        HELP0404
L83,  TO RET IF CAR R = (TRUE).                                     HELP0405
      TO RET IF CAR R = (FALSE).                                    HELP0406
      2 = '8, TO FAIL.                                              HELP0407
.                                                                   HELP0408
L9.      EVALUATE LE OR SE                                          HELP0409
.                                                                   HELP0410
      PUSH DOWN T, CAR T = L91.                                     HELP0411
      TO L101.                                                      HELP0412
L91,  E = CAR S, POP UP S.                                          HELP0413
      TO RET IF CAR R = (TRUE).                                     HELP0414
      TO L92 IF CAR R = (FALSE).                                    HELP0415
.                                                                   HELP0416
L92,  2 = '7, TO FAIL.                                              HELP0417
.                                                                   HELP0418
L10.     EVALUATE AN EXPRESSION                                     HELP0419
.                                                                   HELP0420
      PUSH DOWN T, CAR T = L102.                                    HELP0421
L101, E = CDR E.                                                    HELP0422
      PUSH DOWN S, CAR S = CDR E.                                   HELP0423
      TO ECARE.                                                     HELP0424
L102, E = CAR S, POP UP S.                                          HELP0425
      POP UP T.                                                     HELP0426
      TO ECARE IF CAR R = (TRUE).                                   HELP0427
      TO ECDRE IF CAR R = (FALSE).                                  HELP0428
      2 = '9, TO FAIL.                                              HELP0429
.                                                                   HELP0430
L11.     EVALUATE THE BUILT-IN FUNCTION CAR                         HELP0431
.                                                                   HELP0432
      Z = CAR R, POP UP R.                                          HELP0433
      TO L111 IF CAR R NE 'F.                                       HELP0434
      CAR R = CAR Z.                                                HELP0435
```

```
        TO RET IF AF Z = 00.                                        HELP0436
L111.   2 = '1. TO FAIL.                                            HELP0437
.                                                                   HELP0438
L12.        EVALUATE THE BUILT-IN FUNCTION CDR                      HELP0439
.                                                                   HELP0440
        Z = CAR R. POP UP R.                                        HELP0441
        TO L121 IF CAR R NE 'F.                                     HELP0442
        CAR R = CDR Z.                                              HELP0443
        TO RET IF AF Z = 00.                                        HELP0444
L121.   2 = '2. TO FAIL.                                            HELP0445
.                                                                   HELP0446
L13.        EVALUATE THE BUILT-IN FUNCTION CONS                     HELP0447
.                                                                   HELP0448
        Z = CAR R. POP UP R.                                        HELP0449
        TO L131 IF CAR R = 'F.                                      HELP0450
        E = R. R = CDR R.                                           HELP0451
        TO L131 IF CAR R NE 'F.                                     HELP0452
        CDR E = Z.                                                  HELP0453
        CAR R = E.                                                  HELP0454
        TO RET IF Z = NIL.                                          HELP0455
        TO RET IF AF Z = 00.                                        HELP0456
L131.   2 = '3. TO FAIL.                                            HELP0457
.                                                                   HELP0458
L14.        EVALUATE THE BUILT-IN FUNCTION ATOM                     HELP0459
.                                                                   HELP0460
        Z = CAR R. POP UP R.                                        HELP0461
        TO L321 IF CAR R NE 'F.                                     HELP0462
        TO L142 IF AF Z = 00.                                       HELP0463
L141.   CAR R = (TRUE). TO RET.                                     HELP0464
L142.   CAR R = (FALSE). TO RET.                                    HELP0465
.                                                                   HELP0466
L15.        EVALUATE THE BUILT-IN FUNCTION NULL                     HELP0467
.                                                                   HELP0468
        Z = CAR R. POP UP R.                                        HELP0469
        TO L321 IF CAR R NE 'F.                                     HELP0470
        TO L141 IF Z = NIL. TO L142.                                HELP0471
.                                                                   HELP0472
.                                                                   HELP0473
.                                                                   HELP0474
        ENTRY FINPUT.                                               HELP0475
        TO SKIP IF (CHAR) NE NIL.                                   HELP0476
        USE NEXTCH.                                                 HELP0477
SKIP.   TO ATOM IF (CHAR) = '*.                                     HELP0478
        TO CONT1 IF (CHAR) = ' .                                    HELP0479
        TO IDFN IF (CHAR) LT ' .                                    HELPA479
CONT1.                                                              HELPB479
        TO MULT IF (CHAR) = '.                                      HELP0480
      TO SEMIC IF (CHAR) = ';.                                      HELP1480
      TO LBKTC IF (CHAR) = '<.                                      HELP2480
      TO RBKTC IF (CHAR) = '>.                                      HELP3480
        TO RPAR IF (CHAR) NE '(.                                    HELP0481
        Z = 07. TO NOCH.                                            HELP0482
RPAR.   TO COMM IF (CHAR) NE ').                                    HELP0483
        Z = 08. TO NOCH.                                            HELP0484
COMM.   TO EQUL IF (CHAR) NE ',.                                    HELP0485
```

```
        Z = 11, TO NOCH.                                        HELP0486
EQUL,   TO ERRC IF (CHAR) NE '=,                                HELP0487
        Z = 06, TO NOCH.                                        HELP0488
ERRC,   Z = 00.                                                 HELP0489
NOCH,   (CHAR) = NIL, EXIT FINPUT.                              HELP0490
ATOM,   USE NEXTCH, USE READST.                                 HELP0491
        D = 'A, M = 'X.                                         HELP0492
        USE LOOKUP.                                             HELP0493
        TO BAKA IF CDR Y NE NIL.                                HELP0494
        Z = NEW ELEMENT, CDR Y = Z.                             HELP0495
        AF Z = 01.                                              HELP0496
        X = NIL, CAR Z = CAR X.                                 HELP0497
        CAR X = Z.                                              HELP0498
        CDR Z = S.                                              HELP0499
BAKA,   Z = 01, EXIT FINPUT.                                    HELP0500
IDFN,   USE READST.                                             HELP0501
        D = BASIC, M = NIL.                                     HELP0502
        USE LOOKUP.                                             HELP0503
        TO BSYM IF Y NE NIL.                                    HELP0504
        TO FUNC IF (CHAR) = '(.                                 HELP0505
        D = 'V, M = 'X.                                         HELP0506
        USE LOOKUP.                                             HELP0507
        Z = 02, EXIT FINPUT.                                    HELP0508
BSYM,   Z = CDR Y, EXIT FINPUT.                                 HELP0509
FUNC,   D = 'F, M = 'X.                                         HELP0510
        USE LOOKUP.                                             HELP0511
        TO BAKF IF CDR Y NE NIL.                                HELP0512
        Z = NEW ELEMENT, CDR Y = Z.                             HELP0513
        CAR Z = NIL, CDR Z = S.                                 HELP0514
BAKF,   Z = 03, TO NOCH.                                        HELP0515
MULT,   USE NEXTCH.                                             HELP0516
        TO RBKT IF (CHAR) NE '(.                                HELP0517
LBKTC,  Z = 04, TO NOCH.                                        HELP0518
RBKT,   TO SEMI IF (CHAR) NE '),                                HELP0519
RBKTC,  Z = 05, TO NOCH.                                        HELP0520
SEMI,   TO ASGN IF (CHAR) NE ',.                                HELP0521
SEMIC,  Z = 09, TO NOCH.                                        HELP0522
ASGN,   TO ERRC IF (CHAR) NE '=.                                HELP0523
ASGNC,  Z = 19, TO NOCH.                                        HELP0524
.                                                               HELP0525
.                                                               HELP0526
.                                                               HELP0527
        ENTRY LOOKUP.                                           HELP0528
        Z = S.                                                  HELP0529
NEXT,   TO ABSENT IF CDR D = NIL.                               HELP0530
        D = CDR D, Y = CAR D.                                   HELP0531
CHECK,  TO NEXT IF CAR Y NE CAR Z.                              HELP0532
        TO BACK IF CAR Y = NIL.                                 HELP0533
STEP,   Y = CDR Y, Z = CDR Z.                                   HELP0534
        TO CKSUB IF CAR Y NE CAR Z.                             HELP0535
        TO STEP IF CAR Y NE NIL.                                HELP0536
        EXIT LOOKUP.                                            HELP0537
CKSUB,  D = Y, Y = CAR D.                                       HELP0538
        TO CHECK IF AF Y = 00.                                  HELP0539
        TO NONSRT IF M = NIL.                                   HELP0540
```

```
        Y = NEW ELEMENT.                                        HELP0541
        CAR Y = CAR D, CDR Y = CDR D.                           HELP0542
        CAR D = Y.                                              HELP0543
INSERT, CDR D = NEW ELEMENT.                                    HELP0544
        D = CDR D, CDR D = NIL.                                 HELP0545
        CAR D = Z, Y = 'Z.                                      HELP0546
TOEND,  Y = CDR Y.                                              HELP0547
        TO TOEND IF CAR Y NE NIL.                               HELP0548
BACK,   EXIT LOOKUP.                                            HELP0549
ABSENT, TO INSERT IF M NE NIL.                                  HELP0550
NONSRT, Y = NIL, EXIT LOOKUP.                                   HELP0551
.                                                               HELP0552
.                                                               HELP0553
.                                                               HELP0554
        ENTRY READST.                                           HELP0555
        C = CHAR.                                               HELP0556
        Z = 'S, TO LOOP1.                                       HELP0557
STAR,   CAR Z = ' .                                             HELP0558
LOOP,   USE NEXTCH.                                             HELP0559
LOOP1,  CDR Z = NEW ELEMENT.                                    HELP0560
        Z = CDR Z, CAR Z = (CHAR).                              HELP0561
        TO CONT2 IF (CHAR) = ' .                                HELP0562
        TO LOOP IF (CHAR) LT ' .                                HELPA562
CONT2,                                                          HELPB562
        TO STAR IF (CHAR) = '*.                                 HELP0563
        CAR Z = NIL, CDR Z = NIL.                               HELP0564
        C = ' .                                                 HELP0565
        TO GOTCH IF (CHAR) NE ' .                               HELP0566
        USE NEXTCH.                                             HELP0567
GOTCH,  EXIT READST.                                            HELP0568
.                                                               HELP0569
.                                                               HELP0570
.                                                               HELP0571
        ENTRY NEXTCH.                                           HELP0572
GETCH,  (CHAR) = LINE BUFFER.                                   HELP0573
CKCHR,  TO GETCH IF (CHAR) = C.                                 HELP0574
        TO NEWL IF (CHAR) = NIL.                                HELP0575
        EXIT NEXTCH.                                            HELP0576
NEWL,   READ INPUT.                                             HELP0577
        TO QUIT IF (INPUT) NE 00.                               HELP0578
        WRITE OUTPUT.                                           HELP0579
        (CHAR) = ' , TO CKCHR.                                  HELP0580
QUIT,   STOP.                                                   HELP0581
.                                                               HELP0582
.                                                               HELP0583
.                                                               HELP0584
        ENTRY PRLIST.                                           HELP0585
        TO NOATOM IF AF Y = 00.                                 HELP0586
        USE PRATOM.                                             HELP0587
        EXIT PRLIST.                                            HELP0588
NOATOM, X = 'Y.                                                 HELP0589
LIST,   PUSH DOWN S.                                            HELP0590
        CAR S = X.                                              HELP0591
        X = Y, LINE BUFFER = '(.                                HELP0592
        TO NEXOUT IF (NIL) NE 00.                               HELP0593
```

```
        WRITE OUTPUT, (NIL) = 01.                                   HELP0594
NEXOUT, Y = CAR X.                                                  HELP0595
        TO LIST IF AF Y = 00.                                       HELP0596
        USE PRATOM.                                                 HELP0597
ADV,    TO ENDLIS IF CDR X = NIL.                                   HELP0598
        LINE BUFFER = ',.                                           HELP0599
        TO STEP1 IF (NIL) NE 00.                                    HELP0600
        WRITE OUTPUT, (NIL) = 01.                                   HELP0601
STEP1,  X = CDR X, TO NEXOUT.                                       HELP0602
ENDLIS, LINE BUFFER = ').                                           HELP0603
        TO POPUP IF (NIL) NE 00.                                    HELP0604
        WRITE OUTPUT, (NIL) = 01.                                   HELP0605
POPUP,  X = CAR S.                                                  HELP0606
        POP UP S.                                                   HELP0607
        TO ADV IF AF X = 00.                                        HELP0608
        EXIT PRLIST.                                                HELP0609
.                                                                   HELP0610
.                                                                   HELP0611
.                                                                   HELP0612
        ENTRY PRATOM.                                               HELP0613
        TO ADV1 IF Y NE NIL.                                        HELP0614
        Y = NILAT, TO NOTEND.                                       HELP0615
ADV1,   Y = CDR Y.                                                  HELP0616
CHECK1, TO NOTEND IF CAR Y NE NIL.                                  HELP0617
        EXIT PRATOM.                                                HELP0618
NOTEND, Z = CAR Y.                                                  HELP0619
        TO PRINT IF AF Z = 01.                                      HELP0620
        Y = Z, TO CHECK1.                                           HELP0621
PRINT,  LINE BUFFER = Z.                                            HELP0622
        TO ADV1 IF (NIL) NE 00.                                     HELP0623
        WRITE OUTPUT.                                               HELP0624
        (NIL) = 01, TO ADV1.                                        HELP0625
.                                                                   HELP0626
.          DATA ELEMENTS                                            HELP0627
.                                                                   HELP0628
CHAR,   ELEMENT 00 00 01 NIL.                                       HELP0629
INPUT,  ELEMENT 00 00 01 00.                                        HELP0630
OUTPUT, ELEMENT 00 00 04 00.                                        HELP0631
FALSE,  ELEMENT 00 00 00 00.                                        HELP0632
TRUE,   ELEMENT 00 00 00 00.                                        HELP0633
BASIC,  ELEMENT 01 00 00 B1.                                        HELP0634
B1,     ELEMENT 00 00 B11 B2.                                       HELP0635
B2,     ELEMENT 00 00 B21 B3.                                       HELP0636
B3,     ELEMENT 00 00 B31 B4.                                       HELP0637
B4,     ELEMENT 00 00 B41 B5.                                       HELP0638
B5,     ELEMENT 00 00 B51 B6.                                       HELP0639
B6,     ELEMENT 00 00 B61 B7.                                       HELP0640
B7,     ELEMENT 00 00 B71 NIL.                                      HELP0641
B11,    ELEMENT 00 00 'A  B12.                                      HELP0642
B12,    ELEMENT 00 00 'N  B13.                                      HELP0643
B13,    ELEMENT 00 00 'D  B14.                                      HELP0644
B14,    ELEMENT 00 00 NIL 16.                                       HELP0645
B21,    ELEMENT 00 00 'D  B22.                                      HELP0646
B22,    ELEMENT 00 00 'E  B23.                                      HELP0647
B23,    ELEMENT 00 00 'F  B24.                                      HELP0648
```

```
B24.    ELEMENT 00 00 NIL 12.                                   HELP0649
B31.    ELEMENT 00 00 'E  B32.                                  HELP0650
B32.    ELEMENT 00 00 'L B33.                                   HELP0651
B33.    ELEMENT 00 00 'S B34.                                   HELP0652
B34.    ELEMENT 00 00 'E B35.                                   HELP0653
B35.    ELEMENT 00 00 NIL 15.                                   HELP0654
B41.    ELEMENT 00 00 'I B42.                                   HELP0655
B42.    ELEMENT 00 00 'F B43.                                   HELP0656
B43.    ELEMENT 00 00 NIL 13.                                   HELP0657
B51.    ELEMENT 00 00 'N B52.                                   HELP0658
B52.    ELEMENT 00 00 B521 B53.                                 HELP0659
B53.    ELEMENT 00 00 B531 NIL.                                 HELP0660
B521.   ELEMENT 00 00 'E B522.                                  HELP0661
B522.   ELEMENT 00 00 NIL 10.                                   HELP0662
B531.   ELEMENT 00 00 'O B532.                                  HELP0663
B532.   ELEMENT 00 00 'T B533.                                  HELP0664
B533.   ELEMENT 00 00 NIL 18.                                   HELP0665
B61.    ELEMENT 00 00 'O B62.                                   HELP0666
B62.    ELEMENT 00 00 'R B63.                                   HELP0667
B63.    ELEMENT 00 00 NIL 17.                                   HELP0668
B71.    ELEMENT 00 00 'T B72.                                   HELP0669
B72.    ELEMENT 00 00 'H B73.                                   HELP0670
B73.    ELEMENT 00 00 'E B74.                                   HELP0671
B74.    ELEMENT 00 00 'N B75.                                   HELP0672
B75.    ELEMENT 00 00 NIL 14.                                   HELP0673
NILAT.  ELEMENT 00 00 '( N1.                                    HELP0674
N1.     ELEMENT 00 00 ') N2.                                    HELP0675
N2.     ELEMENT 00 00 NIL NIL.                                  HELP0676
.                                                               HELP0677
.                                                               HELP0678
.                                                               HELP0680
        END PROGRAM.                                            HELP0681
```

DYNL

```
        ENTRY INITL.                                                    DYNL0001
        A = 8.                                                          DYNL0002
LINK,   B = A. INCR A.                                                  DYNL0003
        CDR B = A.                                                      DYNL0004
        AF B = 00, BPF B = 00.                                          DYNL0005
        TO LINK IF A LT 9.                                              DYNL0006
        TO BTOP IF A NE 9.                                              DYNL0007
.                                                                       DYNL0008
        AF A = 00, BPF A = 00.                                          DYNL0009
        B = A.                                                          DYNL0010
BTOP,   CDR B = NIL.                                                    DYNL0011
        (FREE) = 8.                                                     DYNL0012
        (BOT) = 8, (TOP) = B.                                           DYNL0013
        EXIT INITL.                                                     DYNL0014
FREE,   ELEMENT 00 00 00 NIL.                                           DYNL0015
BOT,    ELEMENT 00 00 00 00.                                            DYNL0016
TOP,    ELEMENT 00 00 00 00.                                            DYNL0017
        ENTRY TRACER.                                                   DYNL0018
        TO ENDTR IF CDR P1 LT (BOT).                                    DYNL0019
        TO ENDTR IF (TOP) LT CDR P1.                                    DYNL0020
        (P3) = CDR P1.                                                  DYNL0021
        TO ENDTR IF AF P3 = 01.                                         DYNL0022
        (P2) = (P1).                                                    DYNL0023
FORWD,  CDR P1 = (P2).                                                  DYNL0024
MARK,   AF P3 = 01.                                                     DYNL0025
        (P2) = (P1), (P1) = (P3).                                       DYNL0026
        TO REV IF CDR P1 LT (BOT).                                      DYNL0027
        TO REV IF (TOP) LT CDR P1.                                      DYNL0028
        (P3) = CDR P1.                                                  DYNL0029
        TO FORWD IF AF P3 = 00.                                         DYNL0030
REV,    TO CHKBR IF CAR P1 LT (BOT).                                    DYNL0031
        TO CHKBR IF (TOP) LT CAR P1.                                    DYNL0032
        (P3) = CAR P1.                                                  DYNL0033
        TO BRNCH IF AF P3 = 00.                                         DYNL0034
CHKBR,  TO ENDBR IF BPF P2 = 01.                                        DYNL0035
        (P3) = CDR P2, CDR P2 = (P1).                                   DYNL0036
        (P1) = (P2), (P2) = (P3).                                       DYNL0037
        TO REV IF (P1) NE (P2).                                         DYNL0038
ENDTR,  EXIT TRACER.                                                    DYNL0039
BRNCH,  CAR P1 = (P2), BPF P1 = 01.                                     DYNL0040
        TO MARK.                                                        DYNL0041
ENDBR,  BPF P2 = 00.                                                    DYNL0042
        (P3) = CAR P2, CAR P2 = (P1).                                   DYNL0043
        (P1) = (P2), (P2) = (P3).                                       DYNL0044
        TO CHKBR.                                                       DYNL0045
P1,     ELEMENT 00 00 00 00.                                            DYNL0046
P2,     ELEMENT 00 00 00 00.                                            DYNL0047
P3,     ELEMENT 00 00 00 00.                                            DYNL0048
        ENTRY GETNEW.                                                   DYNL0049
        TO ELOK IF (FREE) NE NIL.                                       DYNL0050
```

```
      TO FAIL3 IF (BOT) = (TOP).                                          DYNL0051
      (P1) = NIL.                                                         DYNL0052
SE@LS,BPF P1 = 01, (P1) = CAR P1.                                         DYNL0053
      TO FAIL1 IF BPF P1 = 01.                                            DYNL0054
      USE TRACER.                                                         DYNL0055
      TO SE@LS IF CAR P1 NE NIL.                                          DYNL0056
      (P1) = (BOT), (P2) = FREE.                                          DYNL0057
TSTFL,TO CLRFL IF AF P1 = 01.                                             DYNL0058
      CDR P2 = (P1), (P2) = (P1).                                         DYNL0059
      TO ADVPT.                                                           DYNL0060
CLRFL,AF P1 = 00.                                                         DYNL0061
ADVPT,TO FIXUP IF (P1) = (TOP).                                           DYNL0062
      INCR (P1), TO TSTFL.                                                DYNL0063
FIXUP,TO FAIL2 IF (P2) = FREE.                                            DYNL0064
      CDR P2 = NIL.                                                       DYNL0065
      (P1) = NIL.                                                         DYNL0066
FIXAT,AF P1 = 01, BPF P1 = 00.                                            DYNL0067
      (P1) = CAR P1.                                                      DYNL0068
      TO FIXAT IF (P1) NE NIL.                                            DYNL0069
ELOK, (P1) = (FREE).                                                      DYNL0070
      (FREE) = CDR FREE.                                                  DYNL0071
      EXIT GETNEW.                                                        DYNL0072
FAIL1,(NIL) = 06, LINE BUFFER = '1.  CORRUPTED CAR CHAIN.                 DYNL0073
      TO FAILI.                                                           DYNL0074
FAIL2,(NIL) = 06, LINE BUFFER = '2.  NO GARBAGE COLLECTED.                DYNL0075
      TO FAILI.                                                           DYNL0076
FAIL3,(NIL) = 06, LINE BUFFER = '3.  MEMORY HAS NOT BEEN INITIALIZED.     DYNL0077
FAILI,LINE BUFFER = NIL, (NIL) = 01.                                      DYNL0078
      LINE BUFFER = 'F.                                                   DYNL0079
      LINE BUFFER = 'A.                                                   DYNL0080
      LINE BUFFER = 'I.                                                   DYNL0081
      LINE BUFFER = 'L.                                                   DYNL0082
      LINE BUFFER = ' .                                                   DYNL0083
      WRITE OUTPT,  STOP.                                                 DYNL0084
.                                                                         DYNL0085
.  OUTPUT CHANNEL SPECIFIER.                                              DYNL0086
.                                                                         DYNL0087
OUTPT, ELEMENT 00 00 04 00.                                               DYNL0088
      END DYNALC.                                                         DYNL0089
```

5.7. ПРИЛОЖЕНИЯ*

5.7.1. Приложение А: Две програми на HELP

Приложени са листингите от изпълнението на двете програми, описани в раздел 4.4 и показани на фиг.4.20 и фиг.4.21. Първата програма определя еднаквост на триъгълници по зададени три елемента - страни и/или ъгли. Втората програма извършва формално диференциране на алгебрични изрази.

5.7.2. Приложение Б: Листинги на изходните масиви, използувани при реализацията на процесора HELP за изчислителната машина FACOM 230-45S

5.7.3. Приложение В: Листинги на етапите на реализацията на процесора HELP за изчислителната машина FACOM 230-45S

* Приложенията са оформени в отделен том.

Литература

1. Кетзън,Х., Съвременно програмиране, изд. Техника, София,1971г.
2. Женюи,Ф.,/ред./, Языки программирования, изд. Мир, Москва,1972г.
3. Хопгуд,Ф., Методы компиляции, изд. Мир, Москва,1972г.
4. Фостер,Дж., Обработка списков, изд. Мир, Москва,1974г.
5. Попов,Г., Реализация на сложни програми-процесори чрез вграждане и моделиране, сп. Автоматизация на производството и управлението, кн. 1, 1975г., изд. на КНТПВО, София
6. Попов,Г., Реализация на функционален процесор за обработка на символна списъчна информация, сп. Системи и управление, кн. 1, 1975г., изд. на Висшата следдипломна школа за системоорганизатори при НИИ "К.Маркс", София
7. Кушнерев - Неменман - Цагельский, Программирование для ЭВМ "МИНСК-32", изд. Статистика, Москва,1973г.
8. Колектив с рък. П. Бърнев, ФОР32 - Транслатор от ФОРТРАН за машината "МИНСК-32", изд. на БАН, Институт по математика и механика, София,1973г.
9. Sammet,J., Programming Languages: History and Fundamentals, Prentice-Hall
10. Aho - Ullman, The Theory of Parsing, Translation, and Compiling: Volume I and Volume II, Prentice-Hall
11. Brillinger - Cohen, Introduction to Data Structures and Non-Numeric Computations, Prentice-Hall
12. Waite,W., Implementing Software for Non-Numeric Applications, Prentice-Hall
13. Bobrow,D.,(editor), Symbol Manipulation Languages and Techniques, North-Holland Publishing Company
14. FACOM 230-45 OS II, FORTRAN Specifications
15. FACOM 230-45 OS II, FASP Specifications
16. FACOM 230-45 OS II, Job Control Language
17. FACOM 230-45 OS II, Data Managment

Съдържание