

ZAM-41

OPROGRAMOWANIE

JĘZYK EOL

EOL 0-1

JĘZYK EOL

INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA, 1969 r.

S p i s r z e c z y

1. Wstęp	str. 1-1
1.1. Przeznaczenie języka EOL	1-1
1.2. Ogólna struktura języka	1-2
1.3. Formalizm opisu składni	1-3
1.4. Wersje języka EOL	1-3
1.5. Notacja wartości zmiennej	1-4
2. Symbole podstawowe	2-1
2.1. Znaki	2-1
2.1.1. Litery	2-1
2.1.2. Cyfry	2-1
2.1.3. Znaki specjalne	2-1
3. Zmienne i ich wartości	3-1
3.1. Wejście	3-1
3.2. Wyjście	3-1
3.3. Wyrażenie	3-1
3.3.1. Słowo	3-2
3.3.2. Liczba	3-2
3.3.3. Adres zapisu	3-2
3.4. Pliki	3-3
3.5. Zmienna logiczna	3-3
3.6. Stos adresów rozkazów	3-3
3.7. Konwencja wartości początkowych	3-3
4. Postać ogólna rozkazów	4-1
4.1. Struktura rozkazu	4-1
4.2. Argumenty	4-1
4.2.1. Litera z indeksem	4-1
4.2.1.1. Sposób czytania	4-2
4.2.1.2. Sposób dopisywania	4-2
4.2.1.3. Sposób wykorzystania	4-3
4.2.1.4. Inne litery	4-4
4.2.2. Etykieta	4-4
4.2.3. Symbol klasy znaków	4-4
4.2.4. Tekst	4-5
4.2.5. Całkowita	4-5
4.2.6. Brak argumentu	4-5
4.3. Oznaczoność wyniku	4-5

5. Rozkazy, deklaracje, komentarze	str.	5-1
 5.1. Przesłania		5-1
5.1.1. Prześlij		5-1
5.1.2. Czytaj		5-2
5.1.3. Pisz		5-3
5.1.4. Kopiuj		5-4
5.1.5. Wstaw		5-5
5.1.6. Pobierz		5-5
5.1.7. Schowaj		5-5
5.1.8. Ustaw		5-5
5.1.9. Zamień		5-5
 5.2. Umieszczanie		5-6
 5.3. Usunięcie		5-6
 5.4. Relacja		5-7
5.4.1. Relacja arytmetyczna		5-7
5.4.2. Relacja tekstowa		5-8
5.4.3. Relacja zapisu		5-9
 5.5. Rozkaz sterujący		5-9
5.5.1. Skok		5-9
5.5.2. Skok warunkowy		5-10
5.5.3. Skok zwrotnicowy		5-10
5.5.4. Skok powrotny		5-11
 5.6. Rozkaz arytmetyczny		5-12
 5.7. Przekształcenie		5-12
5.7.1. Przekształcenie na słowo		5-13
5.7.2. Przekształcenie na liczbę		5-13
5.7.3. Zbijanie		5-14
5.7.4. Rozbijanie		5-14
 5.8. Szukanie		5-15
5.8.1. Szukaj		5-15
5.8.2. Omiń		5-15
 5.9. Zliczanie		5-16
5.9.1. Znайдź		5-16
5.9.2. Szukaj i licz		5-17
5.9.3. Omiń i licz		5-17
 5.10. Różne		5-17
5.10.1. Cofnij		5-18
5.10.2. Pakuj		5-18
5.10.3. Stop		5-18
5.10.4. Idź		5-18

5.11. Deklaracja	str. 5-18
5.11.1. Zwrotnica	5-18
5.11.2. Robocze	5-19
5.11.3. Start	5-19
5.12. Komentarz	5-19
6. Programy i sekcje	6-1
6.1. Program	6-1
7. Makrodefinicje	7-1
7.1. Odwołanie do makrodefinicji	7-2
Zbiór znaków dla danych	Dodatek A
Użycie w programie sekcji w SAS-ie	Dodatek B
Przypisywanie urządzeń zmiennym wejściowym /wyjściowym/	Dodatek C
Sygnalizacja błędów i przyczyny zakończenia programu	Dodatek D
Lista słów kluczowych	Dodatek E

1. Wstęp

=====

1.1. Przeznaczenie języka EOL

EOL jest prostym językiem do manipulacji symbolami.

EOL ma na celu ułatwienie pisania programów, w zakresie przykładowo następującej problematyki:

- /a/ Tłumaczenie z jednego formalnego języka na drugi, w szczególności w zakresie języków programowania.
- /b/ Przekształcanie wyrażeń arytmetycznych lub logicznych, na przykład symboliczne różniczkowanie funkcji.
- /c/ Ułatwienie wymiany informacji pomiędzy operatorem a maszyną.

Przy projektowaniu języka EOL starano się uzyskać język o stosunkowo prostej budowie, łatwy do nauczenia się oraz realizacji.

W języku EOL zastosowano wiele idei użytych wcześniej w innych językach do manipulacji symbolami, a w szczególności w językach IPL-V oraz COMIT.

1.2 Ogólna struktura języka

W języku EOL wszystkie programy oraz dane wejściowe i wyjściowe mają postać ciągów, złożonych ze znaków pisarskich.

W szczególności każdy program w języku EOL może stanowić dane wejściowe lub być wynikiem działania innego programu w tym języku.

Programy w języku EOL składają się z rozkazów i deklaracji. Wykonanie programu polega na kolejnym wykonywaniu rozkazów zawartych w tym programie. Pierwszy rozkaz wykonywany w programie określony jest przez sam program. Po zakończeniu wykonania każdego rozkazu, o ile treść jego nie mówi wyraźnie inaczej, wykonany zostaje rozkaz występujący w najbliższej kolejności jego wypisania w programie, czyli tak zwanej sekwiencji normalnej. Wykonanie każdego rozkazu powoduje z zasady wykonanie prostej czynności jak zmiana wartości zmiennej lub zmiana normalnej sekwencji wykonania rozkazów.

Program w języku EOL składa się z ciągu sekcji. Wewnątrz sekcji rozkazy i deklaracje mogą być grupowane w zależności od siebie lub niezależne procedury. Struktura procedur języka EOL wzorowana jest na PL/I.

Ponadto w języku EOL przewidziana jest możliwość deklarowania makrodefinicji, pozwalających na automatyczną modyfikację programu przed jego wykonaniem.

Cechą ta ułatwia znacznie układanie programów, gdyż umożliwia stosowanie indeksów symbolicznych.

1.3. Formalizm opisu składni

W niniejszej publikacji opis składni programu i danych sporządzono według notacji zaproponowanej przez Backusa i zastosowanej w opisie języka ALGOL. Notację tą uzupełniono następującymi konwencjami:

$\{a b\}$	oznacza "dokładnie jeden z symboli a,b"
$[a b]$	" " żaden lub jeden z symboli a,b"

Powyższe reguły uogólnia się w sposób naturalny na dowolną ilość symboli.

$\{a\}..$	oznacza "dowolny ciąg niepusty symboli, z których każdy jest określony przez a"
$\{a\}..$	" " dowolny, ewentualnie pusty, ciąg symboli, z których każdy jest określony przez a".

Przykłady

$\{A B\}$	może oznaczać A
$\{A B\}..$	" " ABA
$[A B]..$	" " ciąg pusty

W niniejszej publikacji dla zaznaczenia dosłowności stosować będziemy symbol cudzysłowu.

1.4 Wersje języka EOL

Istnieją dwie wersje językowe EOL-u - polska i angielska.

Program może się składać z ciągu sekcji napisanych zarówno w wersji polskiej jak i angielskiej. W obrębie jednej sekcji należy stosować wyłącznie słowa kluczowe polskie bądź też angielskie.

W niniejszym opracowaniu podano postać syntaktyczną poszczególnych jednostek w obu wersjach.

1.5. Notacja wartości zmiennej

W przykładach podanych w niniejszej publikacji celem ilustracji wartości zmiennych stosować będziemy następującą notację pomocniczą:

<nazwa zmiennej>:<wartość zmiennej>

Do opisu wartości zmiennej oprócz symboli podstawowych używane są znaczniki.

<znacznik>::= $\bar{ } \circ \wedge \vee \uparrow$

gdzie:

- - znacznik słowa
- \circ - " - liczby
- \wedge - " - adresu
- \vee - " - zapisu
- \uparrow - " - wskazówki

Przykłady:

```
I1 :ABCD
I3 :
E4 : $\bar{A1} =^{\circ}3$ 
P4 : $^{\vee} - BC - D^{\circ}91^{\wedge} \uparrow^{\vee} - R$ 
```

Należy podkreślić, że zapisy powyższego typu mają znaczenie wyłącznie pomocnicze i nie mają żadnego wpływu na przebieg wykonania programu lub strukturę danych.

2. Symbole podstawowe

Język EOL zbudowany jest z następujących symboli podstawowych:

2.1. Znaki

$\langle \text{znak} \rangle ::= \langle \text{litera} \rangle | \langle \text{cyfra} \rangle | \langle \text{znak specjalny} \rangle$

2.1.1. Litery

$\langle \text{litera} \rangle ::= \text{A} | \text{B} | \dots | \text{Z}$

2.1.2. Cyfry

$\langle \text{cyfra} \rangle ::= \text{0} | \text{1} | \text{2} | \dots | \text{9}$

2.1.3. Znaki specjalne

$\langle \text{znak specjalny} \rangle ::= \sqcup | + | ; | - | , | : | ' | \lambda$

Znaczenie symboli:

a/ λ - oznacza przejście do nowej linii

b/ \sqcup - " spacja

Symbole spacji / \sqcup / i przecinka /,/ mogą być zastąpione przez dowolną nie-pustą kombinację spacji i przecinków.

Każdy znak ";" , ":" i " λ " może być otoczony z obu stron przez dowolną kombinację spacji i przecinków.

Należy zauważyć, że zbiór znaków języka EOL jest węższy niż zbiór znaków dla danych.

Pełny zbiór znaków dla danych zawarty jest w Dodatku A.

3. Zmienne i ich wartości

W programach operujemy na ograniczonej ilości zmiennych, wybranych z następującego zbioru:

- 16 zmiennych wejścia I₁, I₂, ..., I₁₆
- 16 " wyjścia Q₁, Q₂, ..., Q₁₆
- 32 " wyrażeń E₁, E₂, ..., E₃₂
- 32 " plików P₁, P₂, ..., P₃₂
- 1 zmienna logiczna H
- 1 " stosu adresów rozkazów /SAR/

W trakcie wykonywania programu każdej z powyższych zmiennych przypisywane są różne kolejne wartości, których struktura opisana jest poniżej. Sposób przypisania zmiennym wejścia lub wyjścia urządzeń wejścia lub wyjścia podano w Dodatku C.

3.1. Wejście

<wartość wejścia> ::= <ciąg znaków>
 <ciąg znaków> ::= [<znak>] . .

Przykłady

I₁ :JAN-KOWALSKI
 I₃ :A = B*(C+D) - λ

3.2. Wyjście

<wartość wyjścia> ::= <ciąg znaków>

Przykłady

Q₂ :X - 3.1415; END

3.3. Wyrażenie

<wartość wyrażenia> ::= [<składnik>] . .
 <składnik> ::= <słowo> | <liczba> | <adres zapisu>

3.3.1. Słowo

$\langle \text{słowo} \rangle ::= \{ \langle \text{znak} \rangle \} . .$

Przykłady

"ABC

"X=A+B \sqcup λ ;

"38

Słowa służą na ogólny do zapisu tekstów.

3.3.2. Liczba

$\langle \text{liczba} \rangle ::= \langle \text{całkowita} \rangle$

$\langle \text{całkowita} \rangle ::= [+|-] \{ \langle \text{cyfra} \rangle \} . .$

Liczby przedstawiają wielkości wyrażone w układzie dziesiętnym. Bezwzględna wartość liczby nie może być większa niż 8388607.

Przykłady

"36 "-48 "0006

3.3.3. Adres zapisu

$\langle \text{adres zapisu} \rangle ::= \{ \langle \text{znak} \rangle \} . .$

Adresem zapisu jest pewien symbol, określający jednoznacznie położenie tego zapisu w pliku.

Przykłady

Wyrażenia mogą przyjmować przykładowo następujące wartości:

E1 : "X1" = "A" * ("B" + "GAMMA")

E7 : "Z" + "Y" = "X"

E16 : "13" POB - λ

3.4. Pliki

$\langle \text{wartość pliku} \rangle ::= [\text{zapis}] \dots \uparrow [\text{zapis}] \dots$
 $\langle \text{zapis} \rangle ::= \langle \text{wartość wyrażenia} \rangle$

Przykłady

P1 : \uparrow
P3 : $\wedge A \wedge B \wedge C \wedge 38 \uparrow$
P6 : $\wedge a \wedge b \wedge c \uparrow \wedge d \wedge e$

gdzie a,b,... oznaczają zapisy

3.5. Zmienna logiczna

$\langle \text{wartość logiczna} \rangle ::= + | -$

Przykłady

H : +
H : -

3.6. Stos adresów rozkazów

$\langle \text{wartość stosu adresów} \rangle ::= [\text{adres rozkazu}] \dots$
 $\langle \text{adres rozkazu} \rangle ::= {}^{\wedge} \{ \text{znak} \} \dots$

Stos adresów rozkazów służy do zapamiętywania adresów rozkazów typu WYKONAJ /CALL/, które jednoznacznie wyznaczają pozycję tych rozkazów w programie. Adresy te mogą być następnie wykorzystane przez rozkaz WRÓĆ /RETURN/.

Przykład

SAR : ${}^{\wedge} 38 {}^{\wedge} 63 {}^{\wedge} 18$

3.7. Konwencja wartości początkowych

Przyjmuje się, że w momencie rozpoczęcia wykonywania programu zmienna H ma wartość "+", natomiast wszystkie pozostałe zmienne są wyzerowane, to znaczy odpowiadające im wartości są ciągami pustymi.

4. Postać ogólna rozkazów

4.1. Struktura rozkazu

Struktura każdego rozkazu jest następująca:

<operator> [<pierwszy argument> [, <drugi argument> [, <trzeci argument>]]]
Operator jest słowem kluczowym w postaci ciągu dużych liter np. PRZEŚLIJ, CZYTAJ,
WYKONAJ.

Postać i znaczenie różnego typu argumentów opisane są poniżej.

4.2. Argumenty

Argumenty w rozkazach mają jedną z następujących postaci:

- Litera indeksowana
- Etykieta
- Symbol klasy znaków
- Tekst
- Całkowita

W niektórych rozkazach pominięcie ostatniego argumentu może mieć sprecyzowane znaczenie.

Z każdą z podanych powyżej postaci argumentu związane jest jego znaczenie opisane poniżej.

4.2.1. Litera z indeksem

Argument tego typu ma postać:

```
<litera indeksowana> ::= <litera argumentu> <indeks>
<litera argumentu> ::= A|B|C|D|E|I|N|P|Q|T|Y|Z|K
<indeks> ::= 1|2...|32
```

Argument w postaci <litera indeksowana> służy do określenia zmiennej oraz wskazuje na sposób operowania wartością tej zmiennej.

Wielkość <litera argumentu> może określać:

- a/ sposób czytania wartości zmiennej;
- b/ sposób dopisywania do wartości zmiennej;
- c/ sposób wykorzystania wartości zmiennej.

Ponadto w niektórych rozkazach występują litery posiadające jeszcze inne znaczenie.

Wielkość <indeks> jest rzeczywistym indeksem zmiennej. Dla przejrzystości opisu rozkazów wprowadzamy następujące symbole dla oznaczenia indeksów:

<n> ::= <indeks>
<m> ::= <indeks>
<k> ::= <indeks>

4.2.3.1. Sposób czytania

Sposób czytania określony jest przez poszczególne litery następująco:

- A czytanie kolejnych składników początkowych z wyrażenia E z jednoczesnym usuwaniem tych składników.
- B czytanie kolejnych składników początkowych z wyrażenia E bez ich usuwania.
- I czytanie kolejnych znaków z wejścia I, z jednoczesnym ich usuwaniem.
- C czytanie kolejnych zapisów znajdujących się bezpośrednio za wskazówką w pliku P z jednoczesnym usuwaniem każdego przeczytanego zapisu.
- D czytanie kolejnych zapisów znajdujących się bezpośrednio za wskazówką w pliku P połączone z jednoczesnym przeskokiem wskazówki o każdy przeczytany zapis. Przeczytane zapisy nie są usuwane z pliku P.

4.2.1.2. Sposób dopisywania

Sposób dopisywania określony jest przez poszczególne litery następująco:

- A dopisywanie na początek E ciągu składników z odwróceniem ich pierwotnej kolejności.

- B dopisywanie na początek E ciągu składników z zachowaniem ich pierwotnej kolejności.
- Y dopisywanie na koniec E ciągu składników z odwróceniem ich pierwotnej kolejności.
- Z dopisywanie na koniec E ciągu składników z zachowaniem ich pierwotnej kolejności.
- Q dopisywanie na końcu Q kolejnych składników wyrażenia E.
- C dopisywanie ciągu zapisów bezpośrednio za wskazówką \uparrow w pliku P wraz z jednoczesnym odwróceniem pierwotnego porządku tych zapisów.
- D dopisanie ciągu zapisów bezpośrednio za wskazówką \uparrow w pliku P bez zmiany jednoczesnym odwróceniem pierwotnego porządku tych zapisów.
- D dopisanie ciągu zapisów bezpośrednio za wskazówką w pliku P bez zmiany porządku tych zapisów. Jednocześnie wskazówka umieszczona zostaje bezpośrednio za ostatnim elementem dopisanego ciągu.

4.2.1.3. Sposób wykorzystania

Przez wykorzystanie składnika rozumiemy jego użycie jako elementu porównawczego, argumentu w działaniach arytmetycznych, elementu określającego etykietę itp.

- A wykorzystanie pierwszego składnika E, a następnie usunięcie tego składnika z wyrażenia.
- B wykorzystanie pierwszego składnika E bez jego usunięcia z wyrażenia.
- C porównanie z dowolnym składnikiem E bez jego usunięcia. Porównanie uznaje się za spełnione, gdy jest spełnione w stosunku do chociaż jednego składnika E.
- E porównanie z całym wyrażeniem E bez zmiany wartości tego wyrażenia lub użycie całego wyrażenia.
- I porównanie z pierwszym znakiem wejścia I bez jego usunięcia.

4.2.1.4. Inne litery

Podane niżej litery mają następujące znaczenie:

- N argument określa wyrażenie, którego pierwszy składnik należy interpretować jako liczbę.
- K argument określa wyrażenie, do którego jako pierwszy składnik dopisuje się liczbę wskazującą położenie elementu.
- P argument odnosi się do pliku.

4.2.2. Etykieta

```
<etykieta> ::= <identyfikator>
<identyfikator> ::= <litera> [<litera> | <cyfra>] ..
```

Długość identyfikatora nie może przekraczać 60 znaków.

Etykiety stosowane są w programach do oznaczania rozkazów, zwrotnic i procedur.

4.2.3. Symbol klasy znaków

W wielu rozkazach zachodzi potrzeba określenia klasy znaków.

$$<\text{klasa}> ::= \{L|D|B|R\}..$$

Znaczenie powyższych symboli jest następujące:

- L oznacza duże litery
- D " cyfry
- B " znak " " "
- R " pozostałe znaki

Klasy określone przez powyższe symbole są rozłączne, a jednocześnie wyczerpują wszystkie stosowane znaki.

4.2.4. Tekst

<tekst> ::= '<ciąg znaków>'

Ze względu na budowę tekstów niektóre znaki mają reprezentację dwuznakową a mianowicie:

- * L oznacza symbol nowej linii
- * ! " " /
- * * " " *

Przyjęto również dwuznakową reprezentację dla nietypowych symboli kodu wewnętrznego. Szczególny na ten temat zawiera Dodatek A. Długość tekstu nie może przekraczać 60 znaków. Sekcja programu może zawierać co najwyżej 128 różnych tekstów więcej niż jednoznakowych.

4.2.5. Całkowita

Liczby całkowite są używane w testach, jako argumenty rozkazów arytmetycznych, rozkazów porównania i w rozkazie UMIEŚĆ. Na równi z liczbą całkowitą w podanych rozkazach może wystąpić wielkość POLE /SPACE/.

Wielkość ta określa ilość bloków pamięci bębnowej zajętych przez pliki. Długość bloku wynosi 128 słów.

Sekcja programu nie może zawierać więcej niż 128 różnych liczb, których wartość bezwzględna przekracza 15.

4.2.6. Brak argumentu

W niektórych rozkazach brak ostatniego argumentu posiada ścisłe sprecyzowane znaczenie.

4.3. Oznaczoność wyniku

W opisie działania wielu rozkazów przyjmuje się, że argumenty spełniają pewne założenia.

Przykładowe:

W rozkazach arytmetycznych zakłada się, że wielkości określone przez oba argumenty rozkazu są liczbami.

W rozkazach zawierających porównania zakłada się, że obie porównywane wielkości są tego samego typu, to znaczy są to dwa słowa, dwie liczby lub dwa adresy zapisów. W przypadku, gdy jakikolwiek warunek założony w opisie rozkazu nie jest spełniony, wynik działania tego rozkazu jest nieokreślony.

5. Rozkazy, deklaracje, komentarze

```
=====
```

`<rozkaz> ::= <przesłanie> | <umieszczenie> | <usunięcie> | <relacja> |
 <rozkaz sterujący> | <rozkaz arytmetyczny> | <przekształcenie> |
 <szukanie> | <zliczanie> | <różne>`

5.1. Przesłania

`<przesłanie> ::= <prześlij> | <czytaj> | <pisz> | <kopiuj> | <wstaw> | <pobierz>
 <schowaj> | <ustaw> | <zamień>`

5.1.1. Prześlij .

`<prześlij> ::= {PRZEŚLIJ | MOVE} [A|B] <n>, {A|B|Y|Z} <m> [, <test przesłania>]
 <test przesłania> ::= <ilość> | <klasa> | <tekst> | {B|T} <k>
 <ilość> ::= <całkowita> | N <k>`

Przenieś kolejne składniki z E <n> do E <m>.

Pierwszy argument określa sposób pobierania składników z E <n> .

Drugi argument określa sposób dopisywania tych składników do E <m>.

Trzeci argument określa moment zakończenia przeniesień. W szczególności składowa <ilość> określa ilość przeniesionych składników.

Pozostałe składowe trzeciego argumentu określają najbliższy składnik, który nie zostaje przeniesiony i na którym kroki przeniesienia zostają zakończone. Składnik ten jest określony w myśl reguł podanych w rozdziale 4. Brak argumentu oznacza, że należy przenieść wszystkie składniki.

Jeżeli wszystkie składniki z E <n> zostaną przeniesione do E <m> zanim natapi moment zakończenia przeniesień określony przez trzeci argument, to wykonywanie rozkazu zostaje zakończone, a zmiennej H nadana wartość " - ".

Przykłady

Załóżmy, że

E1 : $\lceil XA \rceil = 18^{\circ}3'$;
 E3 : $\lceil ALFA \rceil$
 E8 : $\lceil ; \rceil =$
 H : +

Wykonanie wypisanych poniżej rozkazów PRZEŚLIJ powoduje w stosunku do powyższych wartości następujące przekształcenia:

PRZEŚLIJ	A1,A3,2	{ }	E1 : $\lceil = 18^{\circ}3' ;$
PRZEŚLIJ	A1,A3, ' = '		E3 : $\lceil 1 \rceil \lceil XA \rceil \lceil ALFA \rceil$
PRZEŚLIJ	A1,A3, R		
PRZEŚLIJ	A1,A3,T8		
PRZEŚLIJ	B1,B3, 2		E3 : $\lceil XA \rceil 1 \lceil ALFA \rceil$
PRZEŚLIJ	B3,B3		E3 : $\lceil ALFA \rceil \lceil ALFA \rceil$
			H : +
PRZEŚLIJ	A3,Y3, R		E3 : $\lceil \rceil \lceil ALFA$
			H : -

5.1.2. Czytaj

$\langle \text{czytaj} \rangle ::= \{ \text{CZYTAJ} \mid \text{READ} \} \lceil I \langle n \rangle , \{ A \mid B \mid Y \mid Z \} \langle m \rangle [, \langle \text{test przesłania} \rangle]$

Pobieraj kolejne znaki początkowe z I⟨n⟩, utwórz z nich jedno słowo i dopisz do E⟨m⟩.

Znaki pobierane z I⟨n⟩ są jednocześnie z tego ciągu usuwane. Porządek znaków w utworzonym słowie jest zgodny z porządkiem ich pobierania.

Drugi argument określa sposób dopisania utworzonego słowa do E⟨m⟩.

Trzeci argument, lub jego brak, jest określony podobnie jak trzeci argument w rozkazie <prześlij> z tą różnicą, że odnosi się nie do kolejnych składników wyrażenia, a do kolejnych znaków w ciągu I<n>.

W przypadku gdy trzeci argument rozkazu określa ciąg znaków, to uwzględniony jest jedynie pierwszy znak.

P r z y k l a d y

Niech I1 : X1=28; λ ⊢
 E3 : ~PQ
 H: +

Wypisane poniżej rozkazy powodują:

CZYTEJ	I1,A3,5	I1 : ; λ ⊢ E3 : ~X1=28~PQ
CZYTEJ	I1,Y3,D	I1 : 1=28; λ ⊢ E3 : ~PQ~X
CZYTEJ	I1,A3,.'	I1 : E3 : ~X1=28; λ ⊢~PQ H : -

5.1.3. Pisz

<pisz> ::= {PISZ|WRITE} ⊢ {A|B} <n>, Q <m> [_[<test przesłania>]]

Pobieraj kolejne słowa z E<n> i dopisz je na końcu Q <m> jako następujące po sobie ciągi znaków.

Sposób pobierania składników określony jest przez pierwszy argument. Znaczenie trzeciego argumentu lub jego brak jest takie samo jak w rozkazie <prześlij>.

P r z y k l a d y

Niech E7 : ~X1=13 ⊢ ~λ
 Q2 : WYNIK ⊢

Wykonanie poniższych rozkazów powoduje:

PISZ A7 ,Q2	E7 : Q2 : WYNIK ⊢ X1=13 ⊢ λ
PISZ B7,Q2,1	E7 : ~X1=13 ⊢ ~λ Q2 : WYNIK ⊢ X1

5.1.4. Kopuj

$\langle \text{kopiuj} \rangle ::= \{ \text{KOPIUJ} | \text{COPY} \} \sqcup \{ C | D \} \langle n \rangle, \{ C | D \} \langle m \rangle [, \langle \text{test kopiowania} \rangle]$

$\langle \text{test kopiowania} \rangle ::= \langle \text{ilość} \rangle | \langle \text{tekst} \rangle | \{ B | T | E \} \langle k \rangle$

Przenieś kolejne zapisy z $P \langle n \rangle$ do $P \langle m \rangle$.

Pierwszy argument określa sposób pobierania zapisów z $P \langle n \rangle$. Drugi argument określa sposób dopisywania tych zapisów do $P \langle m \rangle$. Trzeci argument określa moment zakończenia przeniesień. W szczególności składowa $\langle \text{ilość} \rangle$ oznacza ilość przeniesionych zapisów.

Pozostałe składowe trzeciego argumentu określają najbliższy zapis, który nie zostaje przeniesiony i na którym wykonywanie rozkazu zostaje zakończone. Zapis ten jest określony w myśl reguł podanych w rozdziale 4. Jeśli trzeci argument jest wymieniony i wszystkie zapisy z $P \langle n \rangle$ zostaną przeniesione do $P \langle m \rangle$ zanim nastąpi moment zakończenia przeniesień określony przez trzeci argument, to wykonanie rozkazu zostaje zakończone, a zmiennej H nadana wartość " - ".

Brak trzeciego argumentu oznacza przeniesienie do $P \langle m \rangle$ wszystkich zapisów występujących w $P \langle n \rangle$ z pozostawieniem wartości H bez zmiany.

Przykłady

Przyjmijmy, że

$P_1 : ^v_a \uparrow ^v_b ^v_c ^v_d$

$P_3 : ^v_r \uparrow ^v_s ^v_t$

$H : +$

gdzie a,b,c,... przedstawiają pewne zapisy. Podane poniżej rozkazy powodują następujące przekształcenia:

KOPIUJ D1,D3,2

$P_1 : ^v_a ^v_b ^v_c \uparrow ^v_d$

$P_3 : ^v_r ^v_b ^v_c \uparrow ^v_s ^v_t$

$H : +$

KOPIUJ C1,C3,4

$P_1 : ^v_a \uparrow$

$P_3 : ^v_r \uparrow ^v_d ^v_c ^v_b ^v_s ^v_t$

$H : -$

5.1.5. Wstaw

$\langle \text{wstaw} \rangle ::= \{\text{WSTAW} | \text{PUT}\} \sqcup \{A | B\} \langle n \rangle, \{C | D\} \langle m \rangle$

Pobieraj kolejne składniki z E⟨n⟩, utwórz z nich jeden zapis i umieść w P⟨m⟩. Pierwszy argument określa sposób pobierania składników z E⟨n⟩. Drugi argument określa sposób dopisania zapisu do P⟨m⟩.

W przypadku, gdy wyrażenie E⟨n⟩ jest puste, wynikiem działania rozkazu jest jedynie przypisanie H wartości " - ".

5.1.6. Pobierz

$\langle \text{pobierz} \rangle ::= \{\text{POBIERZ} | \text{GET}\} \sqcup \{C | D\} \langle n \rangle, \{A | B | Y | Z\} \langle m \rangle$

Pobierz z P⟨n⟩ zapis i jego składniki dopisz do E⟨m⟩, w sposób określony przez drugi argument.

Pierwszy argument określa sposób pobrania zapisu z P⟨n⟩. W przypadku, gdy wskazówka znajduje się na końcu pliku wynikiem działania rozkazu jest jedynie przypisanie H wartości " - ".

5.1.7. Schowaj

$\langle \text{schowaj} \rangle ::= \{\text{SCHOWAJ} | \text{SAVE}\} \sqcup P \langle n \rangle, \{A | B | Y | Z\} \langle m \rangle$

Adres zapisu w pliku P⟨n⟩, który występuje bezpośrednio przed wskazówką dopisz do wyrażenia E⟨m⟩ w sposób określony przez drugi argument rozkazu.

5.1.8. Ustaw

$\langle \text{ustaw} \rangle ::= \{\text{USTAW} | \text{RESTORE}\} \sqcup \{A | B\} \langle n \rangle, P \langle m \rangle$

Pobierz z E⟨n⟩ adres zapisu w sposób określony przez pierwszy argument, a następnie wskazówkę w P⟨m⟩ umieść bezpośrednio po zapisie wskazanym przez ten adres. Jeśli zapis ten został uprzednio usunięty z pliku, to wynik tego rozkazu jest nieokreślony.

5.1.9. Zamień

$\langle \text{zamień} \rangle ::= \{\text{ZAMIEŃ} | \text{EXCHANGE}\} \sqcup \{E \langle n \rangle, E \langle m \rangle | P \langle n \rangle, P \langle m \rangle\}$

Zamień wzajemnie pomiędzy sobą wartości wyrażeń lub plików, wskazanych przez dwa argumenty rozkazu.

Przykład

Niech E1 : $\neg X_1 = Q$
 E2 : $\neg X_2 = 3$

Po wykonaniu poniższego rozkazu mamy

ZAMIEŃ E1, E2 E1 : $\neg X_2 = 3$
 E2 : $\neg X_1 = Q$

5.2. Umieszczenie

$\langle \text{umieszczenie} \rangle ::= \langle \text{umieść w wyrażeniu} \rangle | \langle \text{umieść w pliku} \rangle | \langle \text{umieść na wyjściu} \rangle$
 $\langle \text{umieść w wyrażeniu} \rangle ::= \{\text{UMIESZCZENIE} | \text{SET}\} \sqcup \{\langle \text{całkowita} \rangle | \langle \text{tekst} \rangle\}, \{A|B|Y|Z\} \langle n \rangle$
 $\langle \text{umieść w pliku} \rangle ::= \{\text{UMIESZCZENIE} | \text{SET}\} \sqcup \{\langle \text{całkowita} \rangle | \langle \text{tekst} \rangle\}, \{C|D\} \langle n \rangle$
 $\langle \text{umieść na wyjściu} \rangle ::= \{\text{UMIESZCZENIE} | \text{SET}\} \sqcup \langle \text{tekst} \rangle, Q \langle n \rangle$

Wielkość określona przez pierwszy argument dopisz do zmiennej określonej przez drugi argument. Argument ten określa jednocześnie sposób dopisania podanej wielkości. Wielkość $\langle \text{całkowita} \rangle$ określa składnik wyrażenia, który jest liczbą o podanej wartości. Jeżeli wskazaną zmienną jest plik, to podana wielkość dopisana jest jako odrębny zapis, składający się z jednego składnika.

Przykłady

Niech E8 : $\neg ALFA = \sqcup$
 Q3 : $X_1 =$

Wypisane niżej rozkazy powodują

UMIESZCZENIE 3846, B8

E8 : $\neg 3846 = ALFA = \sqcup$

UMIESZCZENIE 'BETA', Y8

E8 : $\neg ALFA = \sqcup \neg BETA$

UMIESZCZENIE 'L', Q3

Q3 : $X_1 = \lambda$

5.3. Usunięcie

$\langle \text{usunięcie} \rangle ::= \langle \text{usuń z wyrażenia} \rangle | \langle \text{usuń z wejścia} \rangle | \langle \text{usuń z pliku} \rangle$
 $\langle \text{usuń z wyrażenia} \rangle ::= \{\text{USUŃ} | \text{CLEAR}\} \sqcup A \langle n \rangle [, \langle \text{test przeszukania} \rangle]$
 $\langle \text{usuń z wejścia} \rangle ::= \{\text{USUŃ} | \text{CLEAR}\} \sqcup I \langle n \rangle [, \langle \text{test przeszukania} \rangle]$
 $\langle \text{usuń z pliku} \rangle ::= \{\text{SKREŚL} | \text{DELETE}\} \sqcup C \langle n \rangle [, \langle \text{test kopiowania} \rangle]$

Usunięcie z wejścia, wyrażenia lub pliku, określonego przez pierwszy argument, elementy w ilości określonej przez drugi argument.

Znaczenie drugiego argumentu jest takie samo jak odpowiednio w rozkazach `<prześlij>`, `<czytaj>` lub `<kopiuj>`.

Przykłady

Niech E1 : `"XA" = "18`
`I3 : λuu SKIP`
`P7 : ^a↑^b^c`

Wypisane poniżej rozkazy powodują

USUŃ A1, 1	E1 : <code>" = "18</code>
USUŃ I3, 1	I3 : <code>u u SKIP</code>
SKREŚL C7	P7 : <code>^ a ↑</code>

5.4. Relacja

`<relacja> ::= <relacja arytmetyczna> | <relacja tekstowa> | <relacja zapisu>`

5.4.1. Relacja arytmetyczna

`<relacja arytmetyczna> ::= <relacja arytmetyczna równości> | <relacja arytmetyczna uporządkowania>`

`<relacja arytmetyczna równości> ::= { RW | EQ } \sqcup { A | B | T } <n>, { całkowita } | N <m> }`

`<relacja arytmetyczna uporządkowania> ::= { { WK | MN } | { GT | LT } } \sqcup B <n>, { całkowita } | B <n>`

Jeśli pomiędzy liczbami wskazanymi przez pierwszy i drugi argument zachodzi relacja wskazana przez operator rozkazu, to wartość zmiennej H pozostaw bez zmiany. W przeciwnym przypadku zmiennej H nadaj wartość `"_"`. W przypadku użycia pierwszego argumentu w postaci A<n> usunięcie argumentu ma miejsce tylko wtedy, gdy relacja jest spełniona.

Znaczenie operatorów relacji arytmetycznych jest następujące:

RW /EQ/- równy
 WK /GT/- większy
 MN /LT/- mniejszy

Przykłady

Niech E8 : "8 3
 E9 :

RW A8,8	E8: 3
	H : bez zmian
RW A8,10	E8:bez zmian
	H : -
WK B9,1	wynik nieokreślony

5.4.2. Relacja tekstowa

$\langle \text{relacja tekstowa} \rangle ::= \langle \text{relacja tekstowa równości} \rangle \mid \langle \text{relacja tekstowa uporządkowania} \rangle$
 $\langle \text{relacja tekstowa równości} \rangle ::= \{ \text{RW} | \text{EQ} \} \cup \{ \text{A} | \text{B} | \text{T} | \text{I} \} \langle n \rangle, \{ \langle \text{tekst} \rangle | \langle \text{klasa} \rangle \} \{ \text{B} | \text{T} \} \langle m \rangle$
 $\langle \text{relacja tekstowa uporządkowania} \rangle ::= \{ \{ \text{PP} | \text{NS} \} \cup \{ \text{PR} | \text{FL} \} \} \cup \text{B} \langle n \rangle, \{ \langle \text{tekst} \rangle | \text{B} \langle n \rangle \}$

Jeśli pomiędzy słowami wskazanymi przez pierwszy i drugi argument rozkazu zachodzi relacja wskazana przez operator rozkazu, to wartość zmiennej H pozostaw bez zmiany. W przeciwnym przypadku zmiennej H nadaj wartość "-". W przypadku użycia pierwszego argumentu w postaci A⟨n⟩ usunięcie argumentu ma miejsce tylko wtedy, gdy relacja jest spełniona.

Znaczenie operatorów relacji tekstowych jest następujące:

RW /EQ/ = równy

PP /PR/ = poprzedza

NS /FL/ = następuje

W przypadku, gdy pierwszy argument ma postać I⟨n⟩ w porównaniach uwzględniany jest tylko pierwszy znak drugiego argumentu. W tekście relacji równości jako argumenty mogą występować adresy zapisów.

Mówimy, że dwa słowa są równe, jeśli są one identyczne.

Mówimy, że słowo X poprzedza słowo Y, jeśli przy porównaniu tych słów kolejno znaki po znaku przy pierwszej parze różnych znaków znak słowa X jest wcześniejszy od odpowiadającego mu znaku słowa Y.

Przyjmujemy przy tym, że brak znaku jest wcześniejszy od jakiegokolwiek znaku.

Mówimy, że słowo X następuje po słowie Y, jeśli słowo Y poprzedza słowo X. Uporządkowanie znaków : patrz Dodatek A.

Przykłady

Niech E1 : - A^-BETA^5
 E2 : - Q^-A^-3
 I7 : - ALFA

Wykonanie następujących rozkazów powoduje

RW - B1, A	H : bez zmian
RW - I7, B1	E1 : - A^-BETA^5
	H : bez zmian

5.4.3. Relacja zapisu

$\langle \text{relacja-zapisu} \rangle ::= \{ \text{RZ} | \text{ER} \} \sqcup P < n >, \{ \langle \text{całkowity} \rangle \langle \text{tekst} \rangle | \{ B | E | N | T \} < m > \}$
 Rozkaz powoduje porównanie pierwszego składnika lub całego zapisu z wartością określona przez drugi argument.
 Znaczenie liter 2-go argumentu patrz 4.2.1.3./Sposób wykorzystania/.
 W przypadku równości wartość H pozostaw bez zmian.
 W przeciwnym przypadku nadaj H wartość "-".

Przykłady

Niech P1: $\forall a \forall b \exists AB \neg \text{TOM}$
 E3: - ROM - AB

Wykonanie następującego rozkazu spowoduje:

RZ P1, T3
 H : bez zmian

5.5. Rozkaz sterujący

$\langle \text{rozkaz-sterujący} \rangle ::= \langle \text{skok} \rangle | \langle \text{skok warunkowy} \rangle | \langle \text{skok zwrotnicowy} \rangle | \langle \text{skok powrotny} \rangle$

Rozkaz

5.5.1. Skok

$\langle \text{skok} \rangle ::= \{ \{ \text{SKOCZ} | \text{WYKONAJ} \} \} | \{ \text{GOTO} | \text{CALL} \} \sqcup \langle \text{etykieta} \rangle$
 Przerwij normalną sekwencję rozkazów i przejdź do wykonania rozkazu oznaczonego etykietą. W przypadku rozkazu z operatorem WYKONAJ /CALL/ adres tego rozkazu umieść na wierzchołku stosu SAR.

5.5.2. Skok warunkowy

$\langle \text{skok warunkowy} \rangle ::= \{\text{SKOMI} | \text{SKOPL} | \text{WYMI} | \text{WYPL}\} | \{\text{GOMI} | \text{GOPL} | \text{CAMI} | \text{CAPL}\} \sqcup \langle \text{etykieta} \rangle$
 Jeśli spełnione są warunki wskazane przez operator rozkazu, to przejdź do rozkazu oznaczonego etykieta.

W przypadku rozkazów z operatorem WYMI / CAMI / i WYPL / CAPL / adres tego rozkazu umieść na wierzchołku stosu SAR.

W każdym przypadku zmiennej H nadaj wartość "+" . Poszczególne operatory rozkazu wskazują następujące warunki:

SKOPL / GOPL /; WYPL / CAPL /
 SKOMI / GOMI /, WYMI / CAMI /

wartość H jest równa "+"
 wartość H jest równa "-"

Przykład

Jeśli H jest równe "+" , to rozkaz WYMI ALFA powoduje umieszczenie adresu tego rozkazu na wierzchołku stosu SAR i przejście do miejsca programu oznaczonego etykietą ALFA oraz zmianę wartości H na "+" .

5.5.3. Skok zwrotnicowy

$\langle \text{skok zwrotnicowy} \rangle ::= \{\text{SKONA} | \text{SKOIN} | \text{WYNA} | \text{WYIN}\} | \{\text{GONA} | \text{GOIN} | \text{CANA} | \text{CAIN}\} \sqcup \{A | B\} \langle n \rangle$,
 nazwa zwrotnicy

Przerwij normalną sekwencję rozkazów i przejdź do rozkazu, którego etykieta określona jest przez pierwszy argument rozkazu oraz znajduje się na liście etykiet zwrotnicy o nazwie podanej jako trzeci argument.

W przypadku operatorów WYNA / CANA / oraz WYIN / CAIN / adres rozkazu umieść na wierzchołku stosu SAR . Typ zwrotnicy oraz związany z tym sposób wyboru etykiety określone są przez operator rozkazu w sposób następujący:

SKONA / GONA /, WYNA / CANA / oznaczają, że rozkaz odnosi się do zwrotnicy typu NAZWA i wskazany składnik wyrażenia E<n> zawiera słowa równe wybranej etykiecie bądź tekstowi przypisanemu etykiecie na liście etykiet zwrotnicy.

SKOIN / GOIN /, WYIN / CAIN / oznaczają, że rozkaz odnosi się do zwrotnicy typu INDEKS i wskazany składnik wyrażenia E<n> jest indeksem wybranej etykiety na liście zwrotnicy. Pierwsza etykieta na liście ma indeks równy jeden.

W przypadku, gdy w zwrotnicy typu NAZWA brak jest wskazanej etykiety bądź tekstu przypisanego etykiecie, wówczas sekwencja rozkazów nie zostaje przerwana, a wartość zmiennej SAR pozostaje bez zmian. Ponadto w przypadku operatora WYNA /CANA/ zmiennej H zostaje przypisana wartość " - ".

W przypadku, gdy pierwszy argument ma postać A<n> jego usunięcie ma miejsce tylko wtedy, gdy szukanie zostaje uwieńczone sukcesem.

W przypadku, gdy w zwrotnicy typu INDEKS brak jest wskazanej etykiety, wówczas wynik działania rozkazu jest nieokreślony.

P r z y k ł a d y

Niech będą zadeklarowane zwrotnice:

```
LX : INDEKS LA, LB, LC
KEY: NAZWA FOR, IF, BEGIN, END
```

Wartości E1 oraz E3 wynoszą:

```
E1 : "2"
E3 : "BEGIN"REAL"X";
```

Wówczas rozkaz

```
SKOIN B1, LX
```

jest równoważny rozkazowi

```
SKOCZ LB
```

Podobnie rozkaz

```
SKONA B3, KEY
```

jest równoważny rozkazowi

```
SKOCZ BEGIN
```

5.5.4. Skok powrotny

<skok powrotny> ::= {WRÓĆ | RETURN}

Przerwij normalną sekwencję rozkazów i przejdź do rozkazu następnego w stosunku do rozkazu, którego adres umieszczony jest na wierzchołku stosu SAR. Jednocześnie adres ten usuń ze stosu.

W przypadku, gdy stos SAR jest pusty, znaczenie rozkazu WRÓĆ /RETURN/ jest nieokreślone.

5.6. Rozkaz arytmetyczny

$\langle \text{rozkaz arytmetyczny} \rangle ::= \{ \{\text{DODAJ} | \text{ODEJMIJ} | \text{MNÓŻ} | \text{DZIEL}\} | \{\text{ADD} | \text{SUB} | \text{MULT} | \text{DIV}\} \} \sqcup A\langle n \rangle, \langle \text{całkowita} \rangle | B\langle m \rangle$

Wykonaj działanie arytmetyczne określone przez operator na liczbach określonych przez argumenty rozkazu i otrzymany wynik umieść na początku wyrażenia $E\langle n \rangle$ z równoczesnym usunięciem 1-go argumentu.

Znaczenie poszczególnych operatorów jest następujące:

DODAJ /ADD/ Dodaj do siebie dwie liczby, określone przez dwa argumenty rozkazu, a następnie otrzymaną sumę umieść na początku wyrażenia $E\langle n \rangle$.

MNÓŻ /MULT/ Pomnóż przez siebie dwie liczby, określone przez dwa argumenty rozkazu, a następnie otrzymany iloczyn umieść na początku wyrażenia $E\langle n \rangle$.

ODEJMIJ /SUB/ Od liczby określonej przez pierwszy argument rozkazu odejmij liczbę określoną przez drugi jego argument, a następnie otrzymaną różnicę umieść na początku wyrażenia $E\langle n \rangle$.

DZIEL /DIV/ Liczbę, określoną przez pierwszy argument rozkazu podziel przez liczbę określoną przez drugi jego argument. Resztę z tego dzielenia umieść jako początkowy składnik $E\langle n \rangle$, a całkowitą część ilorazu jako drugi składnik tego wyrażenia. Znak reszty jest zawsze równy znakowi dzielnej. W przypadku, gdy liczba określona przez drugi argument jest równa zeru wynik dzielenia jest nieokreślony.

P r z y k l a d y

Niech $E1 : {}^{\circ}9^-A$ $E3 : {}^{\circ}-11^-B$

Wykonanie poniższych rozkazów powoduje

DODAJ A1; B3

$E1 : {}^{\circ}-2^-A$

DZIEL A1, B3

$E1 : {}^{\circ}9 {}^{\circ}0^-A$

5.7. Przekształcenie

$\langle \text{przekształcenie} \rangle ::= \langle \text{przekształcenie na słowo} \rangle | \langle \text{przekształcenie na liczbę} \rangle | \langle \text{zbijanie} \rangle | \langle \text{rozbijanie} \rangle$

5.7.1. Przekształcanie na słowo

<przekształcenie na słowo> ::= {SŁOWO|WORD} {A|B} <n>

Liczbe stanowiąca początkowy składnik E<n> zamień na słwo stanowiące zapis tej liczby w układzie dziesiętnym i otrzymany rezultat umieść na początku E<n>.

Pierwsza cyfra słowa jest różna od zera z wyjątkiem przypadku, gdy liczba równa się 0, a w przypadku liczby dodatniej pominięty jest znak "+". Znaczenie liter pierwszego argumentu patrz 4.2.1.3./Sposób wykorzystania/.

Przykłady

Niech	E3 : $^{\circ}6^{\circ}7$	E5 : $^{-}11^{\circ}A$
	E6 : ^{-}BA	E10:

Wypisane poniżej rozkazy powodują:

SŁOWO A3	E3 : $^{-}6^{\circ}7$
----------	-----------------------

SŁOWO B5	E5 : $^{-}11^{\circ}-11^{\circ}A$
----------	-----------------------------------

SŁOWO A6	wynik nieokreślony
----------	--------------------

SŁOWO B10	wynik nieokreślony
-----------	--------------------

5.7.2. Przekształcanie na liczbę

<przekształcenie na liczbę> ::= {LICZBA|NUMBER} {A|B} <n>

Słowo stanowiące początkowy składnik E<n> zamień na liczbę stanowiącą wartość tego słowa w układzie dziesiętnym i otrzymany rezultat umieść na początku E<n>.

Znaczenie liter pierwszego argumentu patrz 4.2.1.3 /Sposób wykorzystania/.

Przykłady

Niech	E3 : $^{-}11^{\circ}A$	E5 : $^{-}07^{\circ}B$
-------	------------------------	------------------------

Wykonanie poniższych rozkazów powoduje

LICZBA A3	E3 : $^{^{\circ}}11^{\circ}A$
-----------	-------------------------------

LICZBA B5	E5 : $^{-}7^{\circ}-07^{\circ}B$
-----------	----------------------------------

5.7.3. Zbijanie

$\langle \text{zbijanie} \rangle ::= \{\text{ZBIJ} | \text{COMPRESS}\} \sqcup \{A | B\} \langle n \rangle [\langle \text{test przesłania} \rangle]$

Początkowe składniki $E \langle n \rangle$ ułóż kolejno jeden za drugim i powstały w ten sposób jeden składnik umieść na początku $E \langle n \rangle$.

Pierwszy argument rozkazu określa sposób pobierania składników z $E \langle n \rangle$. Drugi argument lub jego brak określa ostatni z pobranych składników, analogicznie jak w rozkazie $\langle \text{prześlij} \rangle$.

Jeśli wszystkie składniki $E \langle n \rangle$ zostaną pobrane zanim warunek określony przez drugi argument zostanie spełniony, to wykonywanie rozkazu zostaje zakończone, a zmiennej H nadana zostaje wartość "-".

Przykłady

Niech $E2 : \text{"ALFA"} = \text{"X"} + \text{"3"} \sqcup H : +$

Wykonanie poniższych rozkazów powoduje

ZBIJ A2; B $E2 : \text{"ALFA"} = \text{"X+3"} \sqcup$

ZBIJ A2, ';' $E2 : \text{"ALFA"} = \text{"X+3"} \sqcup H : -$

5.7.4. Rozbijanie

$\langle \text{rozbijanie} \rangle ::= \{\text{ROZBIJ} | \text{SPLIT}\} \sqcup \{A | B\} \langle n \rangle [\langle \text{test przesłania} \rangle]$

Słowo stanowiące pierwszy składnik $E \langle n \rangle$ zamień na ciąg złożony z kolejnych znaków początkowych tego słowa i umieść go na początku $E \langle n \rangle$.

Drugi argument rozkazu lub jego brak ma takie samo znaczenie w stosunku do kolejnych znaków początkowego słowa w $E \langle n \rangle$ jak wyrażenie $\langle \text{test przesłania} \rangle$ lub jego brak w rozkazie $\langle \text{czytaj} \rangle$.

W przypadku gdy całe pierwsze słowo w $E \langle n \rangle$ zostanie rozbite przed spełnieniem warunku określonego przez drugi argument rozkazu, wykonywanie rozkazu zostaje zakończone, a zmiennej H nadana wartość " - ".

Przykłady

Niech $E1 : \text{"AB"} = \text{"3"} \sqcup$

Wykonanie poniższych rozkazów powoduje

ROZBIJ A1; ; $E1 : \text{"A"} \sqcup$

ROZBIJ B1; ; $E1 : \text{"A"} \text{"B"} = \text{"3"} \sqcup$

ROZBIJ A1, B $E1 : \text{"A"} \text{"B"} = \text{"3"} \sqcup H : -$

5.8. Szukanie

$\langle \text{szukanie} \rangle ::= \langle \text{szukaj} \rangle \mid \langle \text{omiń} \rangle$

5.8.1. Szukaj

$\langle \text{szukaj} \rangle ::= \{\text{SZUKAJ} \mid \text{SEARCH}\} \{A \mid B\} \langle n \rangle, \langle \text{nazwa zwrotnic} \rangle$

Jeśli na liście zwrotnic typu NAZWA, o nazwie określonej przez ostatni argument rozkazu, znajduje się etykieta /bądź tekst przypisany etykiecie/ określona przez pierwszy argument tego rozkazu, to wartość H pozostaw bez zmiany.

W przeciwnym przypadku zmiennej H nadaj wartość "-".

W przypadku, gdy pierwszy argument ma postać A⟨n⟩, usunięcie następuje tylko wtedy, gdy szukanie zostaje uwieńczone sukcesem.

Przykład

Niech dana będzie zwrotnica:

KLUCZ :NAZWA IF, PLUS:'+', MINUS:'-', BEGIN

oraz wartość zmiennych E

E3 :`+`A

E4 :`MARMUR`BAR

Wypisane poniżej rozkazy powodują:

SZUKAJ A3, KLUCZ

E3 :`A

H : bez zmian

SZUKAJ A4, KLUCZ

E4 :bez zmian

H : -

5.8.2. Omiń

$\langle \text{omiń} \rangle ::= \{\text{OMIŃ} \mid \text{SHIFT}\} \sqcup P \langle n \rangle [, \langle \text{test kopiowania} \rangle]$

Przesuwaj wskazówkę w pliku P⟨n⟩ do przodu aż do momentu określonego przez argument ⟨test kopiowania⟩.

Jeżeli argument ten nie występuje w rozkazie, przesuwaj wskazówkę aż do końca pliku. Argument ⟨test kopiowania⟩ określa moment zatrzymania się wskazówki w ten sam sposób jak to ma miejsce przy czytaniu pliku bez usuwania zapisów w rozkazie ⟨kopiuj⟩.

Również w tych samych okolicznościach zmiennej H zostaje nadana wartość "-".

Przykłady

Przyjmijmy, że

E3 : $\neg \text{ALFA} \neg \text{BETA}$

E23 : $\neg S$

P1 : $\neg a \uparrow \neg b \neg c \neg d \neg e$

gdzie d : $\neg \text{ALFA} ^o 3$

natomiast początkowe składniki w zapisach b oraz c są różne od ALFA.

Wówczas wykonanie następujących rozkazów powoduje:

OMIŃ P1, B3

P1 : $\neg a \neg b \neg c \uparrow \neg d \neg e$

OMIŃ P1, 10

P1 : $\neg a \neg b \neg d \neg e \uparrow$

H : -

5.9. Zliczanie

$\langle \text{zliczanie} \rangle ::= \langle \text{znajdź} \rangle \mid \langle \text{szukaj i licz} \rangle \mid \langle \text{omiń i licz} \rangle$

5.9.1. Znajdź

$\langle \text{znajdź} \rangle ::= [\text{ZNAJDŹ} \mid \text{FIND}] \sqcup [A \mid B] \langle n \rangle, K \langle m \rangle \mid , \langle \text{test przesłania} \rangle$

Jeśli w wyrażeniu określonym przez pierwszy argument rozkazu, znajduje się składnik określony przez trzeci argument rozkazu, to wartość H pozostaw bez zmiany, a na początku wyrażenia E⟨m⟩ umieść liczbę określającą ilość składników E⟨n⟩ poprzedzających ten składnik.

Pierwszemu składnikowi tego wyrażenia odpowiada liczba jeden.

Jeśli takiego składnika brak, to jedynie zmiennej H nadaj wartość "-".

W przypadku braku trzeciego argumentu na początku E⟨m⟩ umieszczona zostaje liczba wskazująca pełną ilość składników w E⟨n⟩.

Przykłady

Przyjmijmy, że

E3 : $\neg \text{AX} \neg \text{AY} \neg \text{BX} \neg \text{CZ}$

E4 : $\neg Q$

Wypisane poniżej rozkazy powodują:

ZNAJDŹ A3, K4, BX

E3 : $\neg \text{BX} \neg \text{CZ}$

-

E4 : $^o 2 \neg Q$

ZNAJDŹ B3, K3

E3 : $^o 4 \neg \text{AX} \neg \text{AY} \neg \text{BX} \neg \text{CZ}$

5.9.2. Szukaj i licz

$\langle \text{szukaj i licz} \rangle ::= \{\text{SZL} | \text{SEC}\} \sqcup \{A | B\} \langle n \rangle, K \langle m \rangle, \langle \text{nazwa zwrotnicy} \rangle$

Wykonaj operację opisaną w 5.8.1. /Szukaj/.

W przypadku znalezienia określonej wartości na liście zwrotnicy umieść liczbę określającą jej położenie na liście na początku wyrażenia określonego przez drugi argument. Pierwszej wartości na liście zwrotnicy odpowiada liczba 1.

P r z y k l a d

Niech dana będzie zwrotnica

KLUCZ : NAZWA IF, FOR, BEGIN, END

oraz wartość zmiennej

E3 : "FOR" A

Poniższy rozkaz powoduje

SZL B3, K3, KLUCZ

E3 : "2" FOR "A"

H : bez zmian

5.9.3. Omiń i licz

$\langle \text{omiń i licz} \rangle ::= \{\text{OML} | \text{SHIC}\} \sqcup P \langle n \rangle, K \langle m \rangle [\langle \text{test kopiowania} \rangle]$

Wykonaj operację opisaną w 5.8.2. /Omiń/.

Liczbe określającą ilość omijanych zapisów umieść na początku wyrażenia określonego przez drugi argument.

P r z y k l a d

Przyjmijmy, że

E3 : "ALFA" BETA

E23 : "S

P1 : "a" "b" "c" "d" "e"

gdzie

d : "ALFA" 3

i pierwsze składniki zapisów b i c są różne od ALFA.

Poniższy rozkaz powoduje

OML P1, K23, B3

E3 : "ALFA" BETA

E23 : "2" S

P1 : "a" "b" "c" "d" "e"

5.10. Różne

$\langle \text{różne} \rangle ::= \langle \text{cofnij} \rangle | \langle \text{pakuj} \rangle | \langle \text{stop} \rangle | \langle \text{idz} \rangle$

5.10.1. Cofnij

$\langle \text{cofnij} \rangle ::= \{\text{COFNIJ} | \text{RESET}\} \sqcup \text{P } \langle n \rangle$

Wskazówkę w pliku P $\langle n \rangle$ ustaw przed początkowym zapisem w pliku.

5.10.2. Pakuj

$\langle \text{pakuj} \rangle ::= \{\text{PAKUJ} | \text{PACK}\}$

Rozkaz pozwala na odzyskanie obszarów pamięci zajętych przez zapisy usunięte wcześniej przez program.

Rozkaz powoduje ustawienie wskazówek we wszystkich plikach przed ich początkowymi zapisy.

Wszystkie adresy wskazówek zapamiętane wcześniej stają się nieaktualne.

5.10.3. Stop

$\langle \text{stop} \rangle ::= \{\text{STOP} | \text{STOP}\}$

Zatrzymaj wykonywanie programu.

5.10.4. Idź

$\langle \text{idź} \rangle ::= \{\text{IDZ} | \text{GOSEC}\} \sqcup \langle \text{nazwa sekcji} \rangle$

$\langle \text{nazwa sekcji} \rangle ::= \langle \text{identyfikator} \rangle$

Przejdz do wykonania sekcji o podanej nazwie. Sekcja ta może być napisana w Języku EOL lub M-SAS. W przypadku przejścia do sekcji napisanej w języku EOL, wyrażenia i SAR zostają wyzerowane, wartość pozostałych zmiennych jest zachowywana.

Informacje niezbędne do pisania programów mieszanych M-SAS-EOL zawiera Dodatek B.

5.11. Deklaracja

$\langle \text{deklaracja} \rangle ::= \langle \text{zwrotnica} \rangle | \langle \text{robocze} \rangle | \langle \text{start} \rangle$

5.11.1. Zwrotnica

$\langle \text{zwrotnica} \rangle ::= \langle \text{zwrotnica indeksowa} \rangle | \langle \text{zwrotnica nazwowa} \rangle$

$\langle \text{zwrotnica indeksowa} \rangle ::= \{ \langle \text{nazwa zwrotnicy} \rangle : \} .. \{ \text{INDEKS} | \text{INDEX} \} \sqcup$

$\langle \text{lista zwrotnicy indeksowej} \rangle$

$\langle \text{lista zwrotnicy indeksowej} \rangle ::= \langle \text{etykieta} \rangle [, [-\lambda] \langle \text{etykieta} \rangle] ..$

$\langle \text{nazwa zwrotnicy} \rangle ::= \langle \text{etykieta} \rangle$

<zwrotnica nazwowa> ::= {<nazwa zwrotnicy>} .. {NAZWA | NAME} ↴
 <lista zwrotnicy nazwowej>
 <lista zwrotnicy nazwowej> ::= <etykieta>[:<tekst>] [, [-λ]<etykieta>[:<tekst>]]]

Ze zwrotnicy indeksowej korzysta za pomocą rozkazów SKOIN /GOIN/, WYIN /CAIN/, zaś ze zwrotnicy nazwowej za pomocą rozkazów SKONA /GONA/ i WYNA /CANA/. Rozkazy te zostały opisane w 5.5.3.

5.11.2. Robocze

<robocze> ::= { ROBOCZE | WORKSPACE } ↴<całkowita>

Deklaracja ta określa ilość bloków pamięci zarezerwowanej na wyrażenia i SAR. Wielkość bloku wynosi 128 słów.

Brak deklaracji równoważny jest zadeklarowaniu maksymalnej ilości pamięci.

5.11.3. Start programu

<start> ::= {START | START}

Deklaracja ta poprzedza rozkaz, który w sekcji programu ma być wykonany jako pierwszy.

Brak deklaracji <start> w sekcji równoważny jest umieszczeniu tej deklaracji przed pierwszym rozkazem sekcji.

5.12. Komentarz

<komentarz> ::= /*dowolny ciąg znaków, w którym para znaków "*/" nie występuje*/

Komentarz nie powoduje wykonania żadnej czynności, a służy jedynie do włączenia odpowiednich objaśnień doprogramu.

6. Programy i Sekcje

6.1. Program

$\langle \text{program} \rangle ::= \{ \langle \text{sekcja} \rangle | \langle \text{sekcja M-SAS} \rangle \} . . \text{KONP} \sqcup \langle \text{nazwa sekcji} \rangle$

Nazwa sekcji po słowie KONP wyznacza sekcję od której rozpoczyna się wykonywanie programu.

$\langle \text{sekcja} \rangle ::= \{ \text{SEKCJA} | \text{SECTION} \} \sqcup \langle \text{nazwa sekcji} \rangle \{ \lambda ; \} \{ \langle \text{zdanie zewnętrzne} \rangle \{ \lambda ; \} \} . . \{ \text{KONS} | \text{ENDS} \}$

$\langle \text{nazwa sekcji} \rangle ::= \langle \text{etykieta} \rangle$

$\langle \text{zdanie zewnętrzne} \rangle ::= \langle \text{zdanie} \rangle$

$\langle \text{zdanie} \rangle ::= [\langle \text{etykieta} \rangle :] . . \langle \text{rozkaz} \rangle | \langle \text{deklaracja} \rangle | \langle \text{procedura} \rangle | \langle \text{komentarz} \rangle$

$\langle \text{procedura} \rangle ::= \{ \langle \text{nazwa procedury} \rangle : \} . . \{ \text{PROC} | \text{PROC} \} \{ \lambda ; \} \{ \langle \text{zdanie} \rangle \{ \lambda ; \} \} . . \{ \text{KONIEC} | \text{END} \} \{ \lambda ; \}$

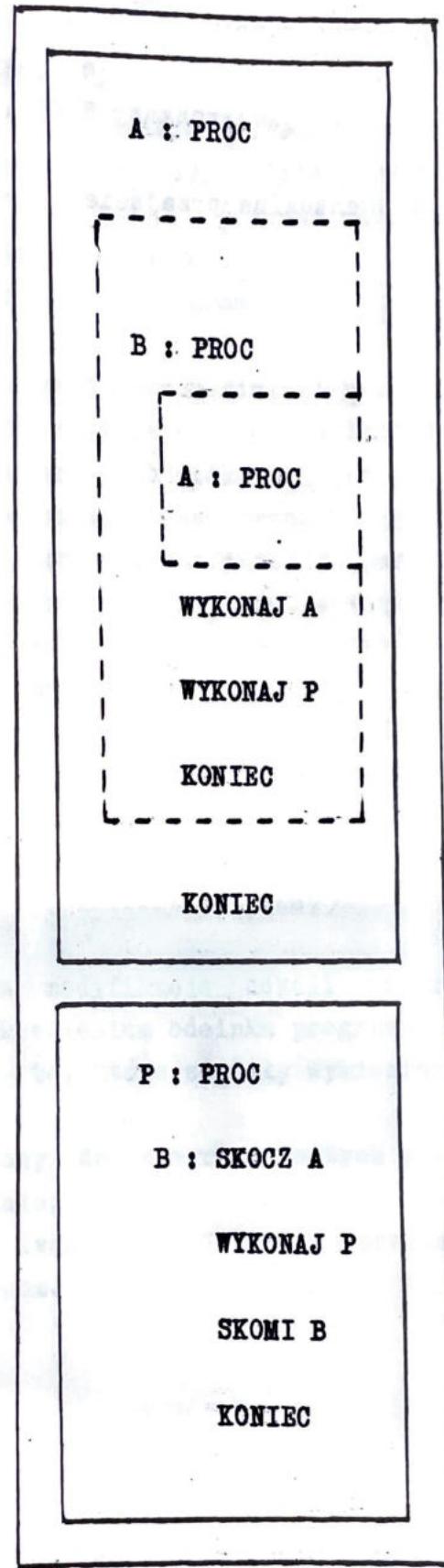
$\langle \text{nazwa procedury} \rangle ::= \langle \text{etykieta} \rangle$

Definicja procedury jest rekursywna, gdyż w skład jednej procedury może wchodzić inna procedura. Zakresem etykiety umieszczonej we wnętrzu procedury P jest cała zawartość procedury P pomniejszona o zawartość tych procedur, w których wnętrzu ta sama etykieta występuje ponownie. Zakresem etykiet zdań zewnętrznych / t.zn. etykiet tych rozkazów oraz nazw tych procedur i zwrotnic, które są zdaniami zewnętrznymi/ jest cała sekcja również pomniejszona o zawartość procedur, w których wnętrzu ta sama etykieta występuje ponownie.

Zakresem nazwy sekcji jest cały program.

P r z y k l a d

Przykład różnych zakresów etykiet przedstawiony jest w poniższym programie, złożonym z dwóch procedur zewnętrznych o nazwie A oraz P.



Zakres nazwy procedury zewnętrznej A nie przenosi się do wnętrza procedury B, gdyż we wnętrzu tym umieszczona jest inna etykieta A.

Dlatego też rozkaz WYKONAJ A odnosi się do procedury A zawartej w procedurze B.

Zakres nazwy procedury zewnętrznej P rozciąga się na całą zawartość procedur

A oraz P, gdyż nazwa ta nigdzie nie jest zadeklarowana powtórnie.

Dlatego też oba wypisane powyżej rozkazy WYKONAJ P odnoszą się do tej samej procedury P.

Rozkaz SKOMI B powoduje ewentualne przejście do rozkazu SKOCZ A, umieszczonego w procedurze P.

7. Makrodefinicje

```

=====
<sekcja źródłowa> ::= <sekcja> | <sekcja modyfikowana>
<sekcja modyfikowana> ::= { SEKCJA | SECTION } [ <nazwa sekcji> { λ | ; } ]
                           (lista makrodefinicji) { λ | ; } <treść sekcji modyfikowanej>
<lista makrodefinicji> ::= { <makrodefinicja> } . .
<makrodefinicja> ::= <nazwa makrodefinicji> [ <lista parametrów> [ λ ] ]
                           ( <treść makrodefinicji> )
<nazwa makrodefinicji> ::= <identyfikator>
<lista parametrów> ::= [ <parametr> { , | λ } <parametr> ] . .
<parametr> ::= <identyfikator>
<treść makrodefinicji> ::= [ <element makrodefinicji> ] . .
<element makrodefinicji> ::= <identyfikator> | <identyfikator> ' <parametr> | <calkowita>
                           <parametr> | <tekst> | [ , | : | ; | λ | <komentarz> | <odwołanie wewnętrzne> ]
<odwołanie wewnętrzne> ::= * { <nazwa makrodefinicji> <parametr> } [ <argument wewnętrzny> [ <argument wewnętrzny> ] . . ] { λ | ; }
<argument wewnętrzny> ::= <identyfikator> | <identyfikator> ' <parametr> | <calkowita>
                           <parametr> | <tekst>
<treść sekcji modyfikowanej> ::= { <element sekcji modyfikowanej> } . .
<element sekcji modyfikowanej> ::= <identyfikator> | <calkowita> | [ , | : | ; | λ | <komentarz> | <odwołanie> | <tekst> | <odwołanie> ]
<odwołanie> ::= * <nazwa makrodefinicji> [ <argument> [ <argument> ] . . ] { λ | ; }
<argument> ::= <identyfikator> | <calkowita> | <tekst>

```

Makrodefinicje pozwalają na modyfikację sekcji programu przed wykonaniem. Każda makrodefinicja jest określeniem odcinka programu, w którym mogą występować parametry, lecz tylko te, które zostały wymienione na liście parametrów danej makrodefinicji.

Odcinek ten zostaje włączony do programu w tych punktach, gdzie nastąpiło odwołanie do danej makrodefinicji.

W wyniku takich modyfikacji <sekcja modyfikowana> przekształca się w <sekcję>, która jest następnie wykonywana.

Przykłady

a. WYPISZ PERFORATOR

(PISZ A1, Q'PERFORATOR)

Nazwą powyższej makrodefinicji jest WYPISZ.

Makrodefinicja jest jednoparametrowa. Parametrem jest PERFORATOR.

b. ABC LA, TEKST, ARG
 (UMIEŚĆ TEKST, A'ARG
 SKOMI' LA)

Nazwą powyższej makrodefinicji jest ABC.

Makrodefinicja jest trójparametrowa. Parametrami jej są LA, TEKST, ARG.

c. QX TOM, EX, L
 (USUŃ T'L, D
 *ABC EX, 'STOP', 5
 SKOCZ TOM

Nazwą powyższej makrodefinicji jest QX.

Makrodefinicja jest trójparametrowa. Jej parametrami są TOM, EX, L.
 Makrodefinicja ta zawiera odwołanie do makrodefinicji o nazwie ABC.

d. PUSTE()

Powyższa makrodefinicja nosi nazwę PUSTE.

Makrodefinicja jest bezparametrowa. Jej treść jest pusta.

Możliwość stosowania makrodefinicji o pustej treści umożliwia np. odwoływanie do pewnych makrodefinicji na etapie uruchamiania programu, a po uruchomieniu zamianę treści tych makrodefinicji na puste, co pozwala na uniknięcie usuwania zbędnych odwołań z treści sekcji modyfikowanej.

e. POMOCNICZA A, B()

Powyższa makrodefinicja nosi nazwę POMOCNICZA. Jej parametrami są A i B.
 Treść tej makrodefinicji jest pusta.

7.1. Odwołanie do makrodefinicji

Włączenie makrodefinicji do programu ma miejsce tylko wtedy, gdy spełnione są następujące warunki:

- Makrodefinicja o danej nazwie została zdefiniowana na liście makrodefinicji danej sekcji modyfikowanej.
- Ilość argumentów odwołania jest identyczna z ilością parametrów makrodefinicji.

W momencie odwołania jest ustanawiana odpowiedniość między parametrami i argumentami w ten sposób, że każdemu parametrowi z listy parametrów makrodefinicji odpowiada argument zajmujący tą samą pozycję na liście argumentów. Po ustaleniu odpowiedniości następuje włączenie makrodefinicji. Włączenie makrodefinicji polega na zastąpieniu odwołania do makrodefinicji, treścią tejże makrodefinicji z równoczesnym podstawieniem argumentów w miejscach odpowiadających im parametrów.

Jeśli odwołanie kończy się średnikiem, to po włączeniu treści makrodefinicji średnik jest usuwany.

W przypadku, gdy element makrodefinicji ma postać <identyfikator>'<parametr>, przy podstawianiu usuwany jest apostrof.

Jeśli odwołanie do makrodefinicji powoduje włączenie takiej treści makrodefinicji, w której znajduje się ponowne odwołanie do makrodefinicji, to wykonywane są następujące czynności:

- a. W przypadku, gdy nazwa makrodefinicji jest parametrem pierwotnej makrodefinicji, to zostaje ona zastąpiona odpowiednim argumentem.
- b. Parametry pierwotnej makrodefinicji, które występują w odwołaniu zostają zamienione na odpowiadające im argumenty nadrzednego odwołania.
- c. Zmodyfikowane zgodnie z p. a i b odwołanie zostaje zastąpione treścią odpowiadającą mu makrodefinicji zgodnie z omówionymi wcześniej regułami.

Z powyższego wynika, że włączanie makrodefinicji powoduje przechodzenie argumentów z zewnętrznych odwołań do wewnętrznych.

Makrodefinicje muszą być tak zdefiniowane, żeby po ich włączeniu<sekcja modyfikowana> została przekształcona w poprawną syntaktycznie <sekcję>.

Przykłady

a/ Przytoczymy przykład sekcji modyfikowanej o nazwie STARTOWA i uproszczony schemat jej przekształcenia w <sekcję>. Przykładem argumentu przenoszonego z zewnętrznego odwołania do wewnętrznego jest tutaj GAMMA.

SEKCJA STARTOWA

(ABC LA, TEKST, ARG

(UMIEŚĆ TEKST, A'ARG

SKOMI LA)

QX TOM, EX, L

(USUÑ I'L, D

ABC EX, STOP ,5

SKOCZ TOM))

ALFA : PROC

* QX BETA, GAMMA, 1

GAMMA : STOP

BETA : CZYTAJ I1, B3, B

SKOCZ ALFA

KONIEC

KONS

Pierwszy etap modyfikacji daje w wyniku:

SEKCJA STARTOWA

ALFA : PROC

USUŃ I1, D

*ABC GAMMA, 'STOP', 5

SKOCZ BETA

GAMMA : STOP

BETA : CZYTAJ I1, B3, B

SKOCZ ALFA

KONIEC

KONS

Drugi etap modyfikacji daje w wyniku <sekcję> :

SEKCJA STARTOWA

ALFA : PROC

USUŃ I1, D

UMIEŚĆ 'STOP', A5

SKOMI GAMMA

SKOCZ BETA

GAMMA : STOP

BETA : CZYTAJ I1, B3, B

SKOCZ ALFA

KONIEC

KONS

- b/ Poniżej przytoczymy przykład sekcji modyfikowanej o nazwie PIERWSZA, w której makrodefinicja P1 zawiera odwołanie do makrodefinicji, której nazwa jest parametrem formalnym ARA.

Schemat przekształcenia tej <sekcji modyfikowanej> na <sekcję> podano niżej.

SEKCJA PIERWSZA

(P1 B1, ARA, C

(PRZEŚLIJ A'B1, B'C

*ARA, 3)

P2 M

(PISZ A'M, Q1)

P3 T

(CZYTAJ I1, B'T, L))

START : PROC
* P1 4, P3, 7
UMIEŚĆ 'ALFABET', Q1
* P1 2, P2, 8
KONIEC
KONS

Pierwszy etap modyfikacji daje w wyniku:

SEKCJA PIERWSZA

START : PROC
PRZEŚLIJ A4, B7
* P3, 3
UMIEŚĆ 'ALFABET', Q1
PRZEŚLIJ A2, B8
* P2, 3
KONIEC
KONS

Następny etap modyfikacji daje w wyniku sekcje :

SEKCJA PIERWSZA

START : PROC
PRZEŚLIJ A4, B7
CZYTAJ I1, B3, L
UMIEŚĆ 'ALFABET', Q1
PRZEŚLIJ A2, B8
PISZ A3, Q1
KONIEC
KONS

Dodatek A

Zbiór znaków dla danych

Uporządkowanie znaków dla danych programu w języku EOL i tekstów zawartych w programie jest zgodne z Normą Zakładową Kodu Wewnętrznego /KW-ZAW/.

Poniższa tabela zawiera pełny zbiór znaków dla danych.

!	Spacja	!	L	!
!	.	!	M	!
!	(!	N	!
!	+	!	O	!
!	*	!	P	!
!)	!	Q	!
!	:	!	R	!
!	-	!	S	!
!	/	!	T	!
!	,	!	U	!
!	:	!	V	!
!	' /apostrof/	!	W	!
!	=	!	X	!
!	[!	Y	!
!	!	Z	!	
!	<	!	SI /duże litery/	!
!	>	!	0	!
!	?	!	1	!
!	ESC	!	2	!
!	so /małe litery/	!	3	!
!	A	!	4	!
!	B	!	5	!
!	C	!	6	!
!	D	!	7	!
!	E	!	8	!
!	F	!	9	!
!	G	!	LF /nowa linia/	!
!	H	!	◊	!
!	I	!	FF	!
!	J	!	CS	!
!	K	!	CR	!
		!	DEL	!

Sposób przyporządkowania znakom kodu wewnętrznego znaków kodów zewnętrznych zawarty jest w opracowaniu "Maszyna ZAM-4.1, oprogramowanie, część I, Maszyna Użytkowa".
 Znaki nietypowe kodu wewnętrznego mają w tekstach następującą reprezentację:

Znak kodu wewnętrznego

	Reprezentacja w tekście
[* U
]	* V
<	* W
>	* X
?	* Y
ESC	* Z
SO	* M
SJ	* D
LF	* L
FF	* E
CS	* F
CR	* G
DEL	* S

W przypadku, gdy kod zewnętrzny posiada jednoznakową reprezentację nietypowego znaku kodu wewnętrznego, można jej użyć zamiast wymienionej wyżej reprezentacji dwuznakowej.

Np: w kodzie M-2 zamiast '*U' można napisać '[' , tekst '*U' będzie w tym przypadku równoważny tekstowi '[' .

EOL B-1

Dodatek B
Użycie w programie sekcji w SAS-ie

W tym samym programie mogą występować zarówno sekcje w EOL-u jak i w SAS-ie. Komunikacja między sekcjami napisanymi w różnych językach możliwa jest zasadniczo tylko poprzez odpowiednie rozkazy przejścia między sekcjami. Dalej przedstawiona jest pewna inna możliwość. Poniżej przedstawiony jest schemat pamięci wewnętrznej w czasie wykonywania programu.

0 !	<u>ROBOCZE SYSTEMU</u>	!
!	<u>OPERACYJNEGO</u>	!
!	<u>EGZEKUTOR</u>	!
!	/program realizujący	!
!	<u>przejścia międzysekcjne/</u>	!
!	<u>SEKCJA W EOL-U LUB SEKCJA</u>	!
!	<u>WYMIENNA W SAS-ie</u>	!
*/	<u>PAMIĘĆ ROBOCZA</u>	!
!	<u>/tylko dla sekcji EOL-u/</u>	!
*/	!	!
!	<u>SEKCJA STAŁA W SAS-ie</u>	!
!		!
!		!
*/	<u>SEKCJA STAŁA W SAS-ie</u>	!
!	<u>INTERPRETER</u>	!
!	/program realizujący	!
!	<u>operacje EOL-u/</u>	!

Z powyższego schematu wynika, że w przypadku współdziałania sekcji w EOL-u z sekcjami stałymi należy tak określić wielkość pamięci roboczej, aby ta nie nakładała się na sekcje stałe /wielkość pamięci roboczej jest określana za pomocą deklaracji ROBOCZE /WORKSPACE//.

*/adresy zmienne zależne od wielkości poszczególnych obiektów.

Jeśli do sekcji stałej w SAS-ie dokonano przejścia z sekcji w EOL-u za pomocą rozkazu IDZ/GOSEC/, to można powrócić do wykonania następnego rozkazu stosując sekwencję rozkazów:

UMA 7428 *

DOA +5 -

PZA 7428 *

SKO 7429 *

Takie przejście nie powoduje zerowania wyrażeń ani SAR. Interpreter EOL-u wykorzystuje numery przesyłań 28, 29, 30, 31. Interpreter definiuje swoje przesyłania przy każdym przejściu z SAS-u do EOL-u za pomocą makrorozkazu IBZ DO SEKCJI.

Natomiast przy przejściu opisanym powyżej, przesyłania nie są definiowane i wymienionych wyżej numerów nie wolno używać. Wymiana informacji między sekcjami różnojęzycznymi odbywa się zasadniczo poprzez wyrażenia.

Budowa wyrażeń

Wyrażenie zapisane jest w pamięci roboczej jako ciąg tzw. ogniw. Ognivo jest parą kolejnych słów, lecz same ogniva nie muszą być w pamięci rozmieszczone kolejno.

Pierwsze słowo ogniva zawiera adres następnego ogniva, drugie zaś informacje. Ostatnie ognivo zamiast adresu zawiera liczbę -0. Składnik zapisywany jest w całkowitej ilości ogniw. Ostatnie ognivo składnika zawiera 1 w pozycji znaku słowa zawierającego adres.

W słowie informacyjnym mieści się do czterech znaków kodu wewnętrznego lub liczba binarna. Znaki zapisywane są od lewej strony słowa poczynając. Jeśli znaków jest mniej niż cztery, słowo dopełnia się znakami pustymi. Adres pierwszego ogniva i-tego wyrażenia mieści się w 7607+2i-tym słowie pamięci. W następującym słowie mieści się adres ostatniego ogniva tego wyrażenia. Oba adresy mają 1 w pozycji znaku. Jeśli wyrażenie jest puste oba adresy mają wartość -0.

Przykład

Niech E2:-ABCDEF° 5 -OK0-+12

Wyrażenie to może być zapisane w pamięci maszyny następująco:

adres	zawartość	typ zawartości
1200	1350	binarna
1201	ABCD	znakowa

Dodatek C

Przypisywanie urządzeń zmiennym wejściowym /wyjściowym/

Zmienna $I^{(n)}/Q^{(n)}$ przyporządkowana jest n-1-emu symbolicznemu /wyjściu/ systemu operacyjnego. Wejściu Posługując się zmiennymi $I^{(n)}/Q^{(n)}$ dla $n \geq 1$ należy przedtem dołączyć odpowiedni podprogram i dekoder. Dołączenia dokonuje się w SAS-ie. Określając obszar na podprogram i dekoder należy uwzględnić rozmieszczenie programu głównego, pamięci roboczej i interpretera.

D o d a t e k D

Sygnalizacja błędów i przyczyny zakończenia programu

1. W czasie translacji programu w języku EOL sygnalizowane są następujące błędy:

<u>Numer błędu</u>	<u>T r e s ć</u>
!	!
!	! Brak nazwy sekcji
!	! Za dużo argumentów
!	! Niewłaściwy argument
!	! Brak testu
!	! Niewłaściwy test lub niewłaściwy argument
!	! Niepoprawna struktura blokowa
!	! Podwójna etykieta
!	! Podwójne wystąpienie deklaracji START
!	! Brak nazwy procedury lub nazwy zwrotnicy
!	! Łączna długość programu i zadeklarowanej pamięci roboczej przekracza pojemność PAO
!	! Niepoprawny rozkaz
!	! Przekroczone pojemność tablicy liczb lub napisów
!	! Niezadeklarowana etykieta
!	! Niezadeklarowana zwrotnica
!	! Użyto niedozwolonego znaku
!	! Niedozwolony znak w tekście
!	! Liczba, tekst lub identyfikator przekracza 60 znaków
!	! Niepoprawna nazwa lub parametr makrodefinicji
!	! Identyczne nazwy makrodefinicji
!	! Użyto niezadeklarowanego parametru formalnego
!	! Odwołanie do nieistniejącej makrodefinicji
!	! Niezgodna ilość parametrów i argumentów makrodefinicji
!	! Przekroczone listę odwołań do makrodefinicji
!	! Łączna długość przedrostka i argumentu przekracza 60 znaków
!	! Niepoprawne zakończenie listy makrodefinicji
!	! Niepoprawne zakończenie programu
!	! Niepoprawne zakończenie odwołania wewnętrznego

Informacje o błędach występujących w danej sekcji poprzedzone są tekstem SEKCJA, po którym wypisywana jest nazwa sekcji.

Po szczególne rodzaje błędów sygnalizowane są następująco:

a/ Błąd 00 sygnalizowany jest tekstem BLAD 00.

b/ Błędy 12 i 13 sygnalizowane są za pomocą tekstu BLAD, po którym następuje numer błędu, a po nim niezadeklarowana etykieta /12/ lub nazwa niezadeklarowanej zwrócińicy /13/, np. BLAD 12 SORT oznacza, że w sekcji programu nie zadeklarowano etykiety SORT.

c/ Pozostałe błędy są sygnalizowane w następujący sposób: Wypisywane jest słowo BLAD, po którym następuje numer błędu a po nim etykieta i liczony względem tej etykiety numer wiersza, w którym wystąpił błąd.

Jeżeli błąd wystąpił przed pojawiением się pierwszej deklaracji etykiety, to zamiast etykiety wypisywane są trzy gwiazdki, a numer wiersza liczony jest względem pierwszego wiersza sekcji nie wliczając makrodefinicji.

Np: BLAD 10 RELACJA 04

oznacza, że w czwartym wierszu względem etykiety RELACJA wystąpił niepoprawny rozkaz */

Błędy w makrodefinicji sygnalizowane są podobnie, z tym że zamiast etykiety wypisywana jest nazwa makrodefinicji, a numer wiersza jest liczony względem wiersza zawierającego tę nazwę.

W przypadku, gdy sekcja programu została napisana w wersji angielskiej zamiast SEKCJA wypisywane jest słowo SECTION, a zamiast słowa BLAD -słowo ERROR.

*/ Przy lokalizacji błędu należy uwzględnić, że komentarze nie sąbrane pod uwagę, a zamiast odwołań do makrodefinicji, do programu wstawiana jest treść makrodefinicji.

2. Przyczyny zakończenia programu.

EOL D-3

W programach EOL-u są wykorzystywane pewne numery przyczyn zakończenia dozwolone dla programów użytkowych. Znaczenie tych numerów jest następujące:

- 99 - wykonanie operacji STOP;
- 100 - przepełnienie pamięci roboczej /w pamięci tej zapisywane są wyrażenia i SAR/;
- 101 - próba wykonania operacji plikowej , jeśli kiedykolwiek przedtem w danej sekcji zapisano więcej niż n-1 bloków pamięci roboczej, gdzie określa ilość zadeklarowanych bloków;
- 102 - przepełnienie pamięci bęбnowej /w pamięci bębnowej zapisywane są pliki/.

Dodatek E
Lista słów kluczowych

COFNIJ	RESET	SCHOWAJ	SAVE
CZYTAJ	READ	SEKCJA	SECTION
DODAJ	ADD	SKOCZ	GOTO
DZIEL	DIV	SKOIN	GOIN
IDŹ	GOSEC	SKOMI	GOMI
INDEKS	INDEX	SKONA	GONA
KONIEC	END	SKOPL	GOPL
KONP	KONP	SKREŚL	DELETE
KONS	ENDS	SŁOWO	WORD
KOPIUJ	COPY	START	START
LICZBA	NUMBER	STOP	STOP
MN	LT	SZL	SEC
MNÓŻ	MULT	SZUKAJ	SEARCH
NS	FL	UMIESC	SET
NAZWA	NAME	USTAW	RESTORE
ODEJMIIJ	SUB	USUŃ	CLEAR
OML	SHIC	WK	GT
OMIN	SHIFT	WRÓĆ	RETURN
PAKUJ	PACK	WSTAW	PUT
PISZ	WRITE	WYIN	CAIN
POBIERZ	GET	WYKONAJ	CALL
POLE	SPACE	WYMI	CAMI
PP	PR	WYNA	CANA
PROC	PROC	WYPL	CAPL
PRZEŚLIJ	MOVE	ZAMIEŃ	EXCHANGE
ROBOCZE	WORKSPACE	ZBIJ	COMPRESS
ROZBIJ	SPLIT	ZNAJDZ	FIND
RW	EQ		
RZ	ER		

Przy wprowadzeniu do maszyny program w języku EOL dekodowany jest na postać wewnętrzną zgodnie z regułami użytego dekodera.

Tak więc, translator EOL-u przetwarza program przedstawiony w kodzie wewnętrznym. Podczas translacji programu pomijany jest znak CR, o ile nie występuje wewnętrznie do tekstów. Jak wynika z tabeli KW-ZAM podanej w Dodatku A znaki: Ć, Ń, Ԃ, Š, Ž w kodzie wewnętrznym nie występują. W opisie użytych wyłącznie dla przejrzystości zamiast znaków: C, N, O, S, Z, Z.



2 Listopad 1970