ZAM-41

Język programowania

sako

Jan Szmelter, Krystyna Balińska-Deloff

Język programowania

SAKO

dla ZAM-41

Opis

Warszawa 1971
Instytut Maszyn Matematycznych
Branżowy Ośrodek INTE

Projekt okładki: Bożena Bratkowska

Komitet Redakcyjny

Jan BOROWIEC /red.nacz./, Wojciech KOSSAKOWSKI, Antoni MAZURKIEWICZ, Jan WIERZBOWSKI, Andrzej WIŚNIEWSKI, Witold WUDEL /sekr. red./

Opracowanie redakcyjne:
Hanna DROZDOWSKA

Adres Redakcji
Warszawa, ul.Krzywickiego 34,
tel.28-37-29

SPIS TREŚCI

W S T Ę P

Celem niniejszego opracowania jest podanie podstawowych zasad korzystania z elektronicznej maszyny liczącej ZAM-41 za pomocą języka SAKO. W stosunku do pełnego opisu języka SAKO - ograniczono się tylko do najprostszych reguł, które jednak pozwalają rozwiązywać nawet skomplikowane zadania matematyczne. Pełny opis programowania w języku SAKO zawiera książka Leona Łukaszewicza i Antoniego Mazurkiewicza pod tytułem „System Automatycznego Kodowania SAKO". Zaawansowany programista dalsze wiadomości może czerpać z podręcznika: ZAM 41 Oprogramowanie, tom III - Język SAKO, wydanego przez IMM w 1971 r.

§ 1. Ogólne uwagi o obliczeniach matematycznych

Obliczenia matematyczne polegają na podstawianiu wartości liczbowych do wzorów matematycznych w celu wykonania działań określonych we wzorach i otrzymania wyników.

Np. na przebieg obliczenia pierwiastków równania kwadratowego

$$ax^2 + bx + c = 0$$

składają się następujące etapy:

1/ ułożenie programu, to znaczy kolejnych wzorów, według których ma przebiegać obliczenie

$$\Delta = b^2 - 4ac$$

$$x_1 = (-b-\sqrt{\Delta}) \quad /2a$$

$$x_2 = (-b+\sqrt{\Delta}) \quad /2a$$

2/ podstawienie danych liczbowych na miejsce zmiennych a,b,c, np.

$$a = 1$$
$$b = -3$$
$$c = 2$$

3/ wykonanie rachunków według programu i podanie wyników:

$$x_1 = 1$$

$$x_2 = 2$$

Gdy podamy rachmistrzowi program i dane liczbowe, to stosując się ściśle do nich otrzyma on poprawną odpowiedź, chociażby nie rozumiał zadania. Dlatego można go zastąpić automatem, jakim jest maszyna matematyczna. Do wykonania tego zadania maszyna musi posiadać następujące urządzenia:

1/ wejście, to znaczy urządzenie, za pomocą którego przekazujemy program i dane,

2/ wyjście, za pomocą którego otrzymujemy wyniki,

3/ urządzenie pozwalające maszynie na wykonanie zadania. Znajomość tego urządzenia nie jest konieczna do korzystania z maszyny.

§ 2. Urządzenia wejścia i wyjścia

Treść zadania, a więc program i dane dla maszyny, przygotowujemy na dalekopisie. Jest to przyrząd podobny do zwykłej maszyny do pisania, na którym piszemy przez naciskanie odpowiednich klawiszy. Najczęściej spotykanymi typami dalekopisów są dalekopisy z kodem międzynarodowym M2.

Znaki pisarskie dalekopisu z kodem międzynarodowym M2 są następujące:

1/ litery

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

2/ cyfry

0 1 2 3 4 5 6 7 8 9

3/ inne

+ - ◊ / * =] . , : ( ) spacja linia powrót

(Spacja oznacza odstęp między znakami; linia - przesunięcie wałka maszyny o jedną linię; powrót - ustawienie karetki z wałkiem do pisania od początku wiersza). Na dalekopisie pisze się wiersz za wierszem, a więc np. kolumny liczb

321	430
425	-18
-31	-22

trzeba wypisywać w kolejności: 321, 430, 425, -18, -31, -22, a nie: 321, 425, -31, 430, -18, -22.

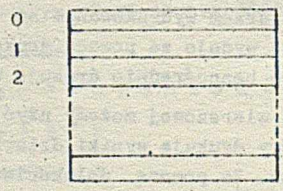
Przy pisaniu dowolnego tekstu, jednocześnie wychodzi z dalekopisu papierowa taśma perforowana, która jest kopią tego tekstu, zapisaną układem dziurek. Jeżeli taśmę wsuniemy do dalekopisu, to zapisaną na niej treść wydrukuje on automatycznie, tak samo jak zwykła maszyna do pisania. Zatem dalekopis może służyć do otrzymania taśmy, jak również do przetłumaczenia jej na normalne pismo.

Teksty wprowadzane do maszyny liczącej są przygotowane na taśmie dalekopisowej. Taśmę tę wsuwamy do czytnika. Taśma przebiegając przez czytnik wzbudza w maszynie impulsy elektryczne odpowiadające dziurkom wydrukowanym na niej. Odpowiedź maszyna wydaje za pomocą drukarki wierszowej, która bezpośrednio drukuje wyniki.

Zamiast drukarki wierszowej możemy użyć również perforatora, który drukuje wyniki dziurkami na taśmie dalekopisu. Za pomocą dalekopisu odczytujemy treść tej odpowiedzi .

§ 3. Pamięć

Naśladując pamięć rachmistrza, maszyna musi mieć możność przechowywania danych, wzorów, rozkazów, wyników działań pośrednich i końcowych. Organem wykonującym to zadanie jest pamięć operacyjna w maszynie.
Wyobrażamy ją sobie jako pewną liczbę /do kilkunastu tysięcy/ komórek ponumerowanych



Do każdej komórki możemy wpisać dowolną informację. Przy wpisywaniu poprzednia zawartość komórki ulega zniszczeniu. Przy pobieraniu informacji z komórki, zawartość jej pozostaje niezmieniona.

Oprócz pamięci operacyjnej maszyna posiada pamięć bębnową zawierającą większą liczbę (rzędu kilkudziesięciu tysięcy) komórek i stanowiącą magazyn informacji. (Nazwa pochodzi stąd, że zbudowana jest ona w postaci bębna zawierającego 128 ścieżek magnetofonowych).

Liczby, na których wykonuje się aktualnie działania arytmetyczne muszą znajdować się w pamięci operacyjnej. Gdy chwilowo są niepotrzebne, wówczas całymi partiami przenosi się je do pamięci bębnowej w celu zmagazynowania.

§ 4. Liczby

W obliczeniach posługujemy się liczbami **całkowitymi** i **ułamkowymi**. W zapisie różnią się one tym, że liczba ułamkowa posiada **kropkę pozycyjną**, a liczba całkowita nie.

Liczba ułamkowa zapisana jest w pamięci maszyny w dwóch kolejnych komórkach.

Liczba całkowita mieści się w jednej komórce pamięci. Nie może zawierać więcej niż siedem cyfr.

A oto przykłady poprawnego zapisu liczb:

1962	liczba całkowita dodatnia
+1962	ta sama liczba całkowita dodatnia
01962	ta sama liczba całkowita dodatnia
1962.	ta sama liczba ułamkowa dodatnia
1962.00	ta sama liczba ułamkowa dodatnia
-27	liczba całkowita ujemna
3.14	liczba ułamkowa dodatnia
+3.14	ta sama liczba ułamkowa dodatnia
3.1400	ta sama liczba ułamkowa dodatnia
03.1400	ta sama liczba ułamkowa dodatnia
-0.3140	liczba ułamkowa ujemna
-.314	ta sama liczba ułamkowa ujemna
+2.18E 4	liczba ułamkowa zmiennoprzecinkowa o wartości $+2.18 \times 10^4$

-2.18E-2	liczba ułamkowa zmiennoprzecinkowa o wartości -2.18×10^{-2}
+.218E+2	liczba ułamkowa zmiennoprzecinkowa o wartości 0.218×10^{2}

Przykłady błędnego zapisu liczb:

98427381	za duża liczba całkowita
42 738	niedozwolona spacja między cyframi
- 42381	niedozwolona spacja między znakiem a liczbą
- 3.183	niedozwolona spacja między znakiem a liczbą
4,57	niedozwolony przecinek zamiast kropki

§.5. Obliczenie wartości wyrażenia arytmetycznego

Zadanie. Obliczyć wartość wyrażenia

$$y = \frac{[(3 \cdot 0{,}5^2 - 4{,}27)\ 0{,}127]\ (7 + 3{,}21)}{457^{-0{,}35}}$$

Rozwiązanie

Drukujemy na dalekopisie następujący program:

```
WYJSCIE:O=PDW(DW,DDW1) ;
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

Y= ((3◊0.5*2-4.27)◊0.127)◊(7+3.21)/(457*-0.35))
DRUKUJ (5.2):Y
STOP NASTEPNY
KONIEC:O
```

Taśmę dalekopisową z wydziurkowanym na niej tym programem wprowadzamy do czytnika maszyny. Po uruchomieniu czytnika, taśma przesuwa się przez czytnik, a wydziurkowane na niej znaki zostają przekazane do pamięci w postaci impulsów elektrycznych. Czynność ta zwana czytaniem przerywa się w chwili, gdy maszyna przeczyta słowo KONIEC:O. Słowo to musi być umieszczone na końcu każdego programu, aby spowodować zatrzymanie wczytywania programu.

Po skończeniu czytania programu maszyna zaczyna wykonywać rozkazy. Każdy rozkaz ma postać krótkiego zdania napisanego w jednym wierszu. Rozkazy wykonywane są przez maszynę w takiej kolejności, w jakiej były napisane, to znaczy, że po skończeniu wykonywania jakiegoś rozkazu maszyna przechodzi automatycznie do wykonywania następnego. Zatrzyma się dopiero wtedy, gdy natrafi na rozkaz STOP NASTEPNY.

Omówimy teraz czynności maszyny w kolejności zapisanych rozkazów. Pierwsze dwa rozkazy

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.
```

tworzą tak zwaną czołówkę napisaną w języku operacyjnym maszyny. (Szczegółowy opis tego języka Czytelnik znajdzie w publikacji: ZAM 41, Oprogramowanie, tom I - Jezyk Operacyjny Maszyny JOM). Pierwsze zdanie czołówki każe maszynie sporządzać wydruki na drukarce wierszowej, co powinno być regułą we wszystkich programach. Gdybyśmy to zdanie opuścili w czołówce, wydruki otrzymalibyśmy w postaci taśmy perforowanej, której treść moglibyśmy odczytać dopiero za pomocą dalekopisu, co niepotrzebnie zwiększa liczbę manipulacji. Drugie zdanie czołówki wymienia po dwukropku czynności, które maszyna ma kolejno wykonać. Najpierw ma wczytać program w języku SAKO, a potem według tego programu ma wykonać PROGRAM. Po tej czołówce taśma perforowana powinna być pusta na odcinku około pół metra.

Następne dwa rozkazy

```
NIWELUJ NADMIARY:TAK
TRANSLACJA
```

stanowią zdania czołówki SAKO. W trakcie wyko-

nywania działań arytmetycznych przez maszynę może się zdarzyć, że jakaś liczba będzie tak duża lub tak małe, że nie da się zapisać w komórce pamięci maszyny. Jeśli opuścimy zdanie NIWELUJ NADMIARY:TAK, to w tych przypadkach maszyna zatrzyma się i nie będzie dalej wykonywać programu. Jeśli w czołówce SAKO występuje zdanie NIWELUJ NADMIARY:TAK, to maszyna nie przerywa pracy, ale za duże liczby zastępuje największą liczbą, jaka mieści się w maszynie (z odpowiednim znakiem), a bardzo małe (np. 10^{-100}) zastępuje zerem. Zdanie TRANSLACJA jest sygnałem końca czołówki SAKO.

Od tego miejsca zaczyna się właściwy program SAKO.

Pierwszy rozkaz

Y= ((30◊0.5*2-4.27)◊0.127)◊(7+3.21)/(457*(-0.35))

wyraża działania arytmetyczne, które należy wykonać w celu obliczenia wartości Y. Rozkaz ten po lewej stronie musi zawierać zmienną, to znaczy symbol wartości, którą chcemy obliczyć. Drugim znakiem musi być znak równości, a następnie wzór napisany w jednym wierszu, za pomocą następujących znaków:

+	dodawanie
-	odejmowanie
◊	mnożenie
/	dzielenie
*	potęgowanie

Kolejność wykonywania działań jest zgodna z ogólnie przyjętymi zasadami, a tam gdzie trzeba, zapewniona jest przez wprowadzenie nawiasów (tylko okrągłych, stosowanych tyle razy, ile tego wymaga zadanie).

Proste reguły zapisu wyjaśnią następujące przykłady:

Źle	Dobrze
5 + -7	5 + (-7)
5 ◊ -7	5 ◊ (-7)
(2 - 7) 5	(2 - 7) ◊ 5
2 * -3	2 * (-3)

Drugi rozkaz

DRUKUJ (5.2) : Y

spowoduje wydrukowanie wartości Y ze znakiem, pięcioma cyframi przed przecinkiem i dwiema po przecinku. Ilość tych cyfr wskazały liczby w nawiasach, rozdzielone kropką. Ewentualne zera na początku liczby zostają przy drukowaniu zastąpione spacjami (wolnymi miejscami). Rozkaz rezerwuje na napisanie liczby 5 + 2 miejsc na cyfry, + 1 miejsce na znak, + 1 miejsce na kropkę, co razem stanowi 9 miejsc wydruku. Ewentualne dalsze wydruki zaczynałyby się od następnego miejsca i w tej samej linii, w której zakończył się ten wydruk .

§ 6. Obliczenie wartości wyrażenia algebraicznego

Wyrażenie algebraiczne różni się tym od arytmetycznego, że niektóre liczby mogą być zastąpione symbolami zmiennych, które je wyrażają, np:

Zadanie. Obliczyć wartość wyrażenia:

$$Z = \frac{X^2 + Y^2}{2} - 3\,(X - Y)^2$$

gdy $X = 5u^2 - v$

$Y = 2,7$

$u = 0,3$

$v = -4$

Rozwiązanie

Pierwsze rozkazy dotyczą działań algebraicznych. Obowiązuje tu zasada, że jednym rozkazem nadajemy wartość liczbową jednej zmiennej, której symbol napisany jest na początku wiersza. Po nim następuje znak równości, a po nim wzór na obliczenie tej zmiennej. Wszystkie zmienne po prawej stronie równania muszą mieć wartości nadane przez rozkazy, które były wykonane wcześniej. Zatem poprawny przebieg

rachunków odbędzie się dalej według następującego programu:

```
WYJSCIE:O=PDW(DW,DLW1) ;
PROBLEM:SAKO,PROGRAM.
NIWELUJ NADMIARY:TAK
TRANSLACJA
    U = 0.3
    V = -4
    Y = 2.7
    X = 5◊U*2 - V
    Z = (X *2 + Y * 2) / 2 - 1◊(X - Y)*2
```

Program, w którym odwrócilibyśmy porządek rozkazów byłby błędny, bo najpierw trzeba znać U i V aby móc obliczyć X, a potem dopiero znając X i Y można obliczyć Z.

Zakończenie programu może mieć postać następujących rozkazów:

```
DRUKUJ (2.8) : Z
STOP NASTEPNY
KONIEC:O
```

których znaczenie omówione było w poprzednim paragrafie.

§ 7. Funkcje standardowe

W maszynie są zapisane pewne funkcje standardowe, z którymi spotykamy się bardzo często w obliczeniach matematycznych. Najważniejsze z tych funkcji są następujące:

Skrót	Nazwa	
SIN(X)	Sinus	X mierzone w radianach
COS(X)	Cosinus	
TG(X)	Tangens	
ASN(X)	Arcus sinus	
ACS(X)	Arcus cosinus	
ATG(X)	Arcus tangens	
PWK(X)	Pierwiastek kwadratowy	
PWS(X)	Pierwiastek sześcienny	
LN(X)	Logarytm naturalny	
EXP(X)	Eksponent = e^X	
ABS(X)	Wartość absolutna = $\|x\|$	
ENT(X)	Część całkowita	
DIV(N,M)	Część całkowita ilorazu dwu liczb całkowitych	
MOD(N,M)	Reszta dzielenia dwu liczb całkowitych	

X w tych wyrażeniach może być liczbą całkowitą, ułamkową, zmienną lub całym wyrażeniem algebraicznym. Funkcje te piszemy w programie tak samo jak w zwykłych wyrażeniach algebraicznych.

Przykład. Ułożyć program rozwiązania równania kwadratowego z § 1.

Rozwiązanie. Program ma postać:

```
WYJSCIE:O=PDW(DW,DDW1) ;
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

A = 1
B = -3
C = 2
DELTA = B*2 - 4 ◊ A ◊ C
X1 = (-B - PWK (DELTA)) / (2 ◊ A)
X2 = (-B + PWK (DELTA)) / (2 ◊ A)
DRUKUJ (2.3) :  X1, X2
STOP  NASTEPNY
KONIEC:O
```

Widzimy, że zmienne możemy oznaczać nie tylko pojedynczą literą, ale również całymi słowami zaczynającymi się od litery i dalej złożonymi z liter lub cyfr, jak np.: DELTA, X1, X2. Przy takim oznaczaniu znaczenie mają tylko pierwsze cztery znaki, a reszta jest pomijana przez maszynę. Dlatego, np. zmienne oznaczone WARIACJA i WARIANCJA będą traktowane jako jedna zmienna. Nie można także zmiennych nazywać tak samo jak funkcji standardowych. A oto przykłady błędnych oznaczeń zmiennych:

COS	myli się z oznaczeniem funkcji cosinus
X)1	zawiera niedozwolony znak
X,,	zawiera niedozwolone znaki ,,
2X	zaczyna się od cyfry

Widzimy również, że rozkaz DRUKUJ (2.3) : X1 X2 zawiera dwie zmienne, oddzielone przecinkiem Jest to równoważne dwu kolejnym rozkazom

```
DRUKUJ (2.3) : X1
DRUKUJ (2.3) : X2
```

Wartości liczb X1 i X2 będą wydrukowane jedna p drugiej, w tym samym wierszu i w kolejności, w jakiej występują w rozkazie. Możliwe jest umieszczenie po rozkazie DRUKUJ większej ilości zmiennych oddzielonych od siebie przecinkiem.

§ 8. Funkcje definiowane

Przykład.

Ułożyć program na obliczenie wartości wyrażenia:

$$y = \sin(x - \varphi)\ sh\ (x - \varphi) + \cos(2x)\ sh^2(2\varphi)$$

dla $x = 0.2$

$\varphi = 1.5$

Rozwiązanie byłoby proste, gdybyśmy dysponowali funkcją sinus hiperboliczny. Jak wiadomo funkcja ta ma postać

$$sh\ x = \frac{e^x - e^{-x}}{2}$$

Możemy ją zatem zdefiniować jako podprogram za pomocą rozkazów. Wyjaśni to program rozwiązujący nasze zadanie:

```
WYJSCIE:O=PDW(DW,DDW1) ;
PROBLEM:SAKO,PROGRAM.
NIWELUJ NADMIARY:TAK
TRANSLACJA
X = 0.2
FI = 1.5
Y = SIN (X - FI) ◊ SH(X - FI)+COS(2 ◊ X) ◊ (SH(2xFI))*2
DRUKUJ (3.7) : Y
STOP NASTEPNY
```

```
PODPROGRAM : SH (X)
A = EXP (X)
B = EXP (-X)
SH () = (A - B) / 2
WROC
KONIEC:O
```

Pierwsze pięć rozkazów po czołówkach zawiera program główny. Jest on tak napisany, jakby funkcja sinus hiperboliczny, oznaczona przez SH(X), uzupełniała funkcje standardowe z poprzedniego paragrafu. Ponieważ nie jest ona standardowa, to znaczy nie jest trwale zapisana w pamięci maszyny, zdefiniowana została za pośrednictwem kolejnych rozkazów, zaczynających się od rozkazu PODPROGRAM i kończących rozkazem WROC.

Podprogram definiujący funkcję pisze się po programie głównym.

Pierwszy rozkaz

PODPROGRAM : SH(X)

zawiera po dwukropku nazwę funkcji, a po niej w nawiasach argument, to znaczy zmienną niezależną.

Nazwą funkcji jest słowo złożone z liter i cyfr, zaczynające się od litery z tym, że maszyna wczytuje tylko pierwsze trzy znaki nazwy. Aby nie było dwuznaczności, te trzy pierwsze znaki muszą różnić się od początku nazw wszystkich funkcji maszyny.*) W naszym

*) Zastrzeżonymi nazwami są: SIN,COS,TG,ASN ACS,ATG,PWK,PWS,LN,EXP,ABS,ENT,DIV,MOD, SUM,ILN,DOL,ODL,SEL,MAX

przypadku sinus hiperboliczny nazwaliśmy SH, a nie mogliśmy nazwać przez SINH, bo wtedy ta nazwa dla maszyny nie różniłaby się od SIN oznaczającego sinus zwykły.

Argument funkcji może być oznaczony tak, jak dowolna zmienna, przy czym oznaczenia w podprogramie mają własne znaczenie, niezależne od oznaczeń używanych w programie głównym. Np. w naszym zadaniu X w podprogramie oznacza inną zmienną niż X w programie głównym.

Podprogram buduje się według tych samych zasad co program główny, przy uwzględnieniu następujących zastrzeżeń:

1) rozkazy programu głównego nie przechodzą do podprogramu i jeśli zachodzi konieczność, to trzeba je w podprogramie powtórzyć,

2) argument występujący w tytule podprogramu (w naszym przypadku X) traktuje się jak wielkość o znanej wartości liczbowej,

3) rozkaz wykonywany jako przedostatni w podprogramie ma postać: nazwa funkcji, po niej dwa nawiasy (), po nich znak równości, po nim wzór nadający ostateczną wartość funkcji. W naszym zadaniu jest to rozkaz

SH() = (A - B) /2

4) rozkaz wykonywany jako ostatni w podprogramie ma postać

WROC

§ 9. Kilka funkcji definiowanych

Może zdarzyć się, że w jednym programie występuje kilka funkcji definiowanych. Określamy każdą z nich oddzielnym podprogramem, napisanym według zasad § 8. Wszystkie podprogramy piszemy w dowolnej kolejności po zakończeniu programu głównego.

Przykład.

Obliczyć wartość wyrażenia:

$y = \sin x \cdot sh\ 2x + tg\ x \cdot th\ 3x + \cos x \cdot ch\ 4x$

dla $x = 0{,}5$

Rozwiązanie. Program zbudujemy wzbogacając funkcje standardowe funkcjami hiperbolicznymi, a mianowicie

$$sh\ x = \frac{e^x - e^{-x}}{2}$$ sinus hiperboliczny

$$ch\ x = \frac{e^x + e^{-x}}{2}$$ cosinus hiperboliczny

$$th\ x = \frac{sh\ x}{ch\ x}$$ tangens hiperboliczny

Program wygląda następująco:

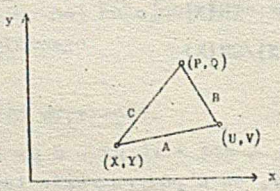
```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.
```

```
NIWELUJ NADMIARY:TAK
TRANSLACJA

X = 0.5

Y=SIN(X)◊SH(2◊X)+TG(X)◊TH(3◊X)+COS(X)◊CH(4◊X)

STOP NASTEPNY

PODPROGRAM : SH(X)

SH() = (EXP(X) - EXP(-X))/2

WROC

PODPROGRAM : CH(X)

CH() = (EXP(X) + EXP(-X))/2

WROC

PODPROGRAM : TH(X)

TH() = SH(X)/CH(X)

WROC

KONIEC:O
```

Zauważmy, że w podprogramie tangensu hiperbolicznego korzystaliśmy z funkcji SH(X) i CH(X) określonych innymi podprogramami. Takie postępowanie jest dozwolone!

§ 10. Funkcje definiowane wielu zmiennych

Przykład. Obliczyć długość obwodu trójkąta, jeżeli dane są współrzędne wierzchołków: (X,Y), (U,V), (P,Q) w układzie osi prostokątnych X,Y:



Wartość długości każdego z boków trójkąta zależy od wartości czterech współrzędnych, np. dla boku A mamy:

$$A = \sqrt{(X-U)^2 + (Y-V)^2}$$

Układamy następujący podprogram:

```
PODPROGRAM : DLUGOSC (X,Y,U,V)
DLUGOSC () = PWK ((X-U)*2 + (Y-X)*2)
WROC
```

Cały program ma postać następującą:

```
WYJSCIE:O=PDW(DW,DDW1) ;
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

X=O
Y=1
U=3
V=2
P=5
Q=-1
A=DLUGOSC(X,Y,U,V)
B=DLUGOSC(U,V,P,Q)
C=DLUGOSC(P,Q,X,Y)
S=A+B+C
DRUKUJ (3.7) :S
STOP NASTEPNY

PODPROGRAM : DLUGOSC (X,Y,U,V)
DLUGOSC () = PWK ((X-U)*2 + (Y-V) * 2
WROC
KONIEC:O
```

§ 11. Zmienne ułamkowe i całkowite

Każda zmienna użyta w programie ma w zależności od typu zarezerwowaną w pamięci jedną lub dwie komórki, w które maszyna wpisuje obliczoną wartość liczbową zmiennej. Np. w § 7 występują zmienne ułamkowe A,B,C, DELTA, X1, X2, a więc dla nich zarezerwowano 12 komórek pamięci.

Z § 4 wiemy, że w dwóch komórkach pamięci mieści się jedna liczba ułamkowa lub 2 liczby całkowite. Jeśli więc w programie występują takie zmienne, które w toku rachunków przyjmują wartości wyłącznie całkowite, to możemy rozkazać maszynie, aby rezerwowała dla nich tylko po jednej komórce pamięci. Dokonujemy tego za pomocą rozkazu CAŁKOWITE: pisząc po dwukropku te zmienne, które przyjmują wartości tylko całkowite. Np. zadanie z § 7 możemy napisać także w postaci:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA
```

```
CALKOWITE : A, B, C
A=1
B=-3
C=2
DELTA=B ◊ B-4 ◊ A ◊ C
X1= (-B-PWK(DELTA)) / (2 ◊ A)
X2= (-B+PWK(DELTA)) / (2 ◊ A)
DRUKUJ(2.3) :X1, X2
STOP NASTEPNY
KONIEC:O
```

Uwaga: rozkaz „całkowite" powinien być napisany wcześniej, nim napisane zostaną jakiekolwiek rozkazy zawierające tę zmienną. Tak więc błędne byłoby takie zaczęcie programu:

```
A = 1
B = -3
C = 2
CALKOWITE : A, B, C
    itd.
```

§ 12. Czytanie danych wejściowych

W przykładach podanych dotychczas, wszystkie dane liczbowe wprowadzane były do rachunków równocześnie z programem. Można je wprowadzać także w inny sposób, a mianowicie po wprowadzeniu programu. Wtedy całość zadania napisana jest na trzech arkuszach:

1) arkusz zawierający program,
2) arkusz zawierający dane,
3) arkusz wyników.

W dotychczasowych przykładach nie było arkusza drugiego.

Arkusz danych zawiera komentarze słowne i liczby. Komentarze słowne służą do objaśnienia treści arkusza. Komentarz zaczyna się od litery. Komentarz może zajmować jeden lub kilka wierszy. Komentarz może zawierać wszystkie znaki dalekopisu, z wyjątkiem znaku równości „=" i dwukropka „:". Komentarz kończy się jednym z tych dwu znaków.

Dane liczbowe pisze się w postaci liczb całkowitych lub ułamkowych. W jednym wierszu może znajdować się tylko jedna liczba. Liczba ta może być napisana także w tym wierszu,

w którym skończył się komentarz.

Ponieważ maszyna pomija komentarze, arkusz danych może ograniczyć się wyłącznie do liczb.

Sposób wprowadzania danych do maszyny podaje przykład, w którym zadanie z § 1 rozwiążemy zakładając, że parametry a, b, c wprowadzone są z arkusza danych.

Arkusz danych niech ma postać następującą:

```
DANE
A = 1.
B = -3.
C = 2.
```

Pierwszym komentarzem jest DANE A=, drugim B=, trzecim C=. Ponieważ te komentarze maszyna pomija, arkusz danych mógłby mieć postać

```
 1.
-3.
 2.
```

Ponieważ na arkuszu danych wprowadziliśmy liczby ułamkowe (o czym świadczy kropka pozycyjna), więc A,B,C w programie musimy traktować jako zmienne ułamkowe. Program ten wygląda następująco:

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

CZYTAJ : A, B, C

DELTA = B◊B - 4◊A◊C

X1 = (-B - PWK (DELTA)) /(2◊A)

X2 = (-B + PWK (DELTA)) /(2◊A)
```

```
DRUKUJ (2.3) : X1, X2
STOP NASTEPNY
KONIEC:O

DANE
A=1.
B=-3.
C=2.
```

Do czytnika wprowadza się taśmę z programem, który kończy się słowem KONIEC:O. Dalej na taśmie napisane są dane. Postępując zgodnie z czołówką, najpierw maszyna wczyta program. Po natrafieniu na słowo KONIEC:O przejdzie do następnej czynności wymienionej w czołówce, to znaczy do wykonywania kolejnych rozkazów programu. Gdy natrafi na rozkaz

```
CZYTAJ : A, B, C
```

uruchomiony zostanie czytnik, a liczby (bez komentarzy) napisane na taśmie zostaną przeczytane przez maszynę i podstawione na miejsce zmiennych A,B,C wymienionych po rozkazie CZYTAJ:. Wczytanych zostanie tyle liczb z arkusza danych, ile zmiennych występuje po rozkazie CZYTAJ:, przy czym na miejsce pierwszej napisanej zmiennej podstawiona zostanie pierwsza przeczytana liczba, na miejsce drugiej - druga itd.

Ponieważ komentarz nie odgrywa roli a jedynie porządek napisania liczb na arkuszu danych, to nawet gdybyśmy w naszym przykładzie dane przygotowali w następujący sposób:

```
DANE:
C=1.
B=-3.
A=2.
```

to jednak rozkaz CZYTAJ: A,B,C wprowadziłby do maszyny A=1., B=-3., C=2.

Gdy dane na arkuszu danych przedstawione są jako liczby całkowite (tzn. bez kropki pozycyjnej, § 4) , to odpowiadające im zmienne w rozkazie CZYTAJ muszą być też całkowite. To znaczy, że wcześniej od niego musi w programie występować rozkaz CALKOWITE. Gdybyśmy np. dane do naszego zadania napisali w postaci

```
DANE:
A=1
B=-3
C=2.
```

to A i B byłyby całkowite, a C ułamkowe. Odpowiedni program zaczynałby się następująco:

```
CALKOWITE : A,B
CZYTAJ : A,B,C
```

i dalej jak poprzednio.

Główną zaletą osobnego wprowadzania programu i danych jest to, że jednym programem możemy wielokrotnie rozwiązywać te same zadania, dla rozmaitych danych liczbowych.

§ 13. Drukowanie wyników

Drukowanie wartości liczbowej zmiennej ułamkowej Y możemy dokonać za pomocą rozkazu

DRUKUJ (5.2) : Y

opisanego w § 4. Przypominamy, że liczby (5.2) oznaczają, że będzie wydrukowane

znak, 5 cyfr przed kropką, kropka, 2 cyfry po kropce,

co razem zajmuje miejsce dziewięciu znaków wydruku.

Drukowanie zacznie się od następnego miejsca, na którym skończył się poprzedni wydruk. (Oczywiście, w zależności od potrzeby liczby 5.2 i symbol zmiennej Y można zastąpić innymi).

Gdyby zmienna Y była określona jako całkowita przez rozkaz CALKOWITE : Y, to rozkaz DRUKUJ musiałby mieć postać następującą

DRUKUJ (3) : Y

i powodowałby wydrukowanie

znaku, 3 cyfr

a więc zajęłby 4 miejsca. (Oczywiście liczba 3 i zmienna Y w tym przykładzie mogą być zastą-

pione innymi, w zależności od potrzeby). Jak widać teraz w nawiasie występuje tylko jedna liczba zamiast dwu. Gdybyśmy np. rozkaz DRUKUJ (3.0): Y zastosowali do zmiennej całkowitej Y zamiast podanego wyżej, byłby do błąd.

Zmienną ułamkową Y można wydrukować jeszcze inaczej, a mianowicie za pomocą rozkazu

DRUKUJ (3.4,5):Y

Teraz do wydruku liczbowej wartości zmiennej Y zarezerwowano 3 miejsca dziesiętne przed kropką pozycyjną i 4 miejsca po kropce. Trzeci parametr rozkazu, to znaczy 5 oznacza, że wynik ma zawierać co najmniej 5 cyfr znaczących. Jeśli wartość liczby przekroczy zarezerwowaną dla niej ilość znaków, to wydruk będzie miał postać:

znak liczby, ułamek dziesiętny, litera E, znak, liczba całkowita, dwie spacje.

Literę E należy odczytać jako mnożnik 10 w potędze wskazanej przez wydrukowaną po niej liczbę całkowitą. Wydruk liczby w takiej postaci jest wygodny wtedy, gdy nie znamy rzędu jej wielkości i nie możemy z góry określić ile miejsc należy zarezerwować przed przecinkiem i po przecinku. Wydruk liczby w tej postaci zajmuje na tabulogramie tyle miejsc, ile wynosi suma dwu pierwszych parametrów +8. W naszym przykładzie będzie to 3+4+8=15 miejsc

Jeżeli maszyna w swej pracy natrafia kilka razy na rozkazy DRUKUJ, to odpowiednie liczby będą drukowane jedna za drugą, w tym samym

wierszu. Gdybyśmy chcieli aby między nimi powstało np. 5 spacji (to znaczy 5 wolnych miejsc), to maszyna powinna wykonać między rozkazami DRUKUJ dodatkowy rozkaz

SPACJA 5

(Oczywiście liczbę 5 możemy zastąpić dowolną inną, w zależności od potrzeby).

Po kilku rozkazach DRUKUJ i SPACJA może okazać się, że nie ma już więcej miejsca na drukowanie w wierszu. Przejście do nowego wiersza spowoduje rozkaz

LINIA 3

Rozkaz ten każe dalsze wydruki prowadzić od początku trzeciej następnej linijki. (Oczywiście liczbę 3 możemy zastąpić w miarę potrzeby, dowolną całkowitą i dodatnią).

Jeżeli zależy nam na tym, aby oprócz liczb wydrukowany został jakiś tekst, możemy tego dokonać za pomocą rozkazu

TEKST :

po którym w następnym wierszu zamiast nowego rozkazu piszemy to, co maszyna ma napisać. Wydruk tekstu nastąpi począwszy od tego miejsca, na którym zakończył się poprzedni wydruk.

Aby objaśnić działanie podanych rozkazów, prześledźmy następujące przykłady programu zawierającego tylko czytanie i pisanie.

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

TRANSLACJA

CALKOWITE : N,M
CZYTAJ : N,M,A,DELTA
LINIA 1
```

```
TEKST :
DRUKOWANIE WYNIKOW
LINIA 2
TEKST :
N =
DRUKUJ (3) : N
SPACJA 2
TEKST :
M =
DRUKUJ (3) : M
LINIA 1
TEKST :
A =
DRUKUJ (3.3) : A
LINIA 1
TEKST :
DELTA =
DRUKUJ (2.5) : DELTA
STOP NASTEPNY
KONIEC:0

DANE:
127
-5
6.
0.321
```

Pierwszy rozkaz programu nie wymaga objaśnienia. Drugi spowoduje wczytanie kolejnych liczb z taśmy danych i podstawienie ich na miejsce N,M,A,DELTA. Charakter danych, to znaczy tego, czy liczba napisana jest z kropką pozycyjną czy nie, zgadza się z charakterem zmiennych w rozkazie CZYTAJ, z których pierwsze dwie N,M są całkowite, a pozostałe dwie A,DELTA ułamkowe.

W rezultacie wykonanie tego rozkazu w komórkach pamięci maszyny będą zapisane następujące liczby:

N=127, M=-5, A=6. , DELTA=0.321

Trzeci rozkaz programu (LINIA 1) ustawia urządzenie drukujące w położeniu początkowym na wypadek, gdyby po poprzednim pisaniu nie znajdowało się ono w takim położeniu. Rozkaz następny

```
TEKST:
DRUKOWANIE WYNIKOW
```

spowoduje wypisanie przez urządzenie piszące tytułu:

DRUKOWANIE WYNIKOW

Następny rozkaz (LINIA 2) ustawi urządzenie piszące w nowym drugim wierszu (to znaczy, że jeden wiersz zostanie opuszczony). Następny rozkaz

```
TEKST :
N=
```

spowoduje wypisanie na początku tego wiersza treści

N=

Rozkaz DRUKUJ (3) :N spowoduje w dalszym ciągu tego wiersza wydrukowanie (całkowitej!) liczby stanowiącej wartość, to znaczy +127. Zatem wiersz teraz będzie miał postać:

N = +127

Rozkaz SPACJA 2 każe dokonać odstępu o długości dwu znaków dalekopisowych, a rozkaz

```
TEKST :
M=
```

wydrukuje potem dalszą treść tak, że wiersz

przybierze postać:

N= +127 M=

Rozkaz DRUKUJ(3) :M każe wydrukować w dalszym ciągu wiersza wartość liczbową M, tzn. -5. Ponieważ (3) w tym rozkazie każe zarezerwować 4 miejsca, natomiast liczba -5 zawiera tylko dwa, więc pierwsze 2 pozycje przy drukowaniu będą opuszczone. W ten sposób wiersz przybierze postać

N = +127 M= -5

Dalsze drukowanie przebiega w podobny sposób. Ostatecznie odpowiedź otrzymana po wykonaniu programu będzie miała postać:

DRUKOWANIE WYNIKOW

N=+127 M= -5

A= +6.000

DELTA= +0.32100

§ 14. Skok warunkowy

Rozwiążmy jeszcze raz przykład z § 1 z tym, że przewidywać będziemy wprowadzanie danych a,b,c z arkusza danych. Nie będziemy zakładali nic o tych liczbach. W związku z tym może się zdarzyć, że wyróżnik Δ będzie ujemny i wtedy nie można obliczyć pierwiastków x_1 i x_2. Chcielibyśmy, aby maszyna zasygnalizowała ten fakt. Możemy to zrobić za pomocą następującego programu:

```
WYJSCIE:O=PDW(DW,DDW1) ;
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

CZYTAJ : A,B,C
DELTA = B◊B - 4◊A◊C
GDY DELTA ] 0 : 1, INACZEJ 2
1) X1 = (-B - PWK (DELTA)) /(2◊A)
X2 = (-B + PWK (DELTA)) /(2◊A)
LINIA 1
TEKST :
X1 =
DRUKUJ (5.5) : X1
SPACJA 5
TEKST :
```

```
X2 =
DRUKUJ (5.5) :X2
STOP NASTEPNY
2) GDY DELTA=0:1, INACZEJ 3
3) LINIA 1
TEKST :
WYROZNIK UJEMNY
STOP NASTEPNY
KONIEC:0
DANE:
A=1.
B=-3.
C=2.
```

Pierwsze 2 rozkazy tego programu były omówione poprzednio. Rozkaz trzeci

GDY DELTA] 0 : 1, INACZEJ 2

rozumie maszyna w ten sposób, że gdy spełniona jest nierówność DELTA > 0 dalsze obliczenia mają przebiegać począwszy od rozkazu o numerze 1) (a więc następnego), a w przeciwnym przypadku o numerze 2). Jak widać obliczenie przebiegać może dwiema różnymi drogami, w zależności od znaku wyróżnika. Jeśli Czytelnik prześledzi program to stwierdzi, że dla nieujemnego wyróżnika w odpowiedzi otrzymamy wydrukowane obydwa pierwiastki, a dla ujemnego maszyna wydrukuje zamiast tego treść: WYROZNIK UJEMNY.

Forma rozkazu GDY jest następująca:

GDY warunek : numer rozkazu, INACZEJ numer rozkazu

Warunek może być nierównością, przy czym dopuszczalny jest tylko znak] (a nie <). Warunek może być także równością (znak =).

Warunku równości nie należy stosować do liczb ułamkowych gdy nie jesteśmy pewni, że na skutek zaokrągleń rachunków równość może być spełniona tylko w przybliżeniu a nie ściśle. (Takie zaokrąglenia powstają na skutek tego, że liczby wprowadzamy w układzie dziesiętnym, a maszyna liczy w układzie dwójkowym. Np. okazuje się, że równość

$$0.1 + 0.1 + 0.1 = 0.3$$

nie jest spełniona ściśle w układzie dwójkowym przy ograniczonej ilości cyfr).

Jeden z numerów rozkazu można zastąpić słowem NASTEPNY. Wtedy odpowiedni tok obliczeń będzie przebiegał od rozkazu następnego. Wyjaśni to przykład, w którym szukamy największej z trzech liczb A,B,C

```
        WYJSCIE:0=PDW(DW,DDW1) ;
        PROBLEM:SAKO,PROGRAM.

        TRANSLACJA

        CZYTAJ : A,B,C
        S=A
        GDY B ]S :  NASTEPNY, INACZEJ 1
        S=B
    1)  GDY C ]S  :  NASTEPNY,INACZEJ 2
        S=C
    2)  DRUKUJ (5.5) : S
        STOP NASTEPNY
        KONIEC:0

        DANE:
        A=120.
        B=230.
        C=3400.
```

Tutaj szukaną największą liczbę oznaczamy przez S. Drugi rozkaz każe podstawić na S

liczbę A. Następnie badamy czy B>S. Jeśli tak, to na nową wartość S przyjmiemy B, (S=B) jeśli nie, to omijamy ten rozkaz. Dalej badamy czy C>S. Jeśli tak, to na nową wartość S przyjmujemy C (S=C), a jeśli nie, to omijamy ten rozkaz. W rezultacie S ma wartość największej z liczb A,B,C i tę wartość wydrukuje maszyna.

§ 15. Skok bezwarunkowy

Przypuśćmy, że mamy obliczyć S = |A|, gdzie A jest dane. Pokażemy, jak można dokonać tego bez użycia funkcji standardowej ABS(X) omówionej w § 7. Odpowiedni fragment programu ma postać

```
          ----------
          ----------
          GDY A ]0  : NASTEPNY, INACZEJ 5
          S=A
          SKOCZ DO 6
       5) S = -A
       6) ----------
          ----------
```

Kreskami zastąpiliśmy początkowe i końcowe rozkazy programu, których tutaj nie będziemy omawiać.

Gdy A jest dodatnie, S = A, gdy ujemne S = -A. Wybór jednej z tych ewentualności dokonuje się za pomocą rozkazu skoku warunkowego

```
GDY A ]0  :  NASTEPNY, INACZEJ 5
```

Gdy zajdzie pierwsza ewentualność należy ominąć rozkaz S=-A.
Dokonuje tego rozkaz

SKOCZ DO 6

który każe maszynie prowadzić dalsze obliczenia począwszy od rozkazu o numerze 6).

§ 16. Powtórzenia

Przykład. Obliczyć i wydrukować wartość:

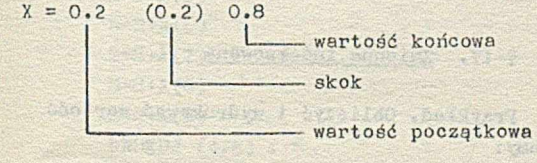
$$y = \sqrt{1 - x^2}$$

dla x = 0.2, 0.4, 0.6, 0.8

Rozwiązanie. Program mógłby wyglądać następująco:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

X=0.2
Y = PWK(1- X◊X)
DRUKUJ (5.5) : Y
X=0.4
Y = PWK (1 - X◊X)
DRUKUJ (5.5) : Y
X=0.6
Y = PWK (1 - X◊X)
DRUKUJ (5.5) : Y
X=0.8
Y = PWK (1 - X◊X)
DRUKUJ (5.5) : Y
STOP NASTEPNY
KONIEC:O
```

Od razu spostrzegamy, że program zawiera wielokrotne powtórzenie tych samych rozkazów z tym, że X przyjmuje najpierw najmniejszą wartość 0.2, a potem powiększa ją po kolei o skok 0.2, aby zakończyć obliczenie po osiągnięciu wartości 0.8. Ten rodzaj zmienności zmiennej X zapisujemy symbolicznie



Program poprzedni znacznie się uprości, gdy napiszemy go w postaci

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

*)Y = PWK(1 - X◊X)
DRUKUJ (5.5):Y
POWTORZ: X = 0.2 (0.2) 0.8
STOP NASTEPNY
KONIEC:0
```

Rozkaz POWTORZ każe wykonywać działania począwszy od miejsca zaznaczonego w programie gwiazdką. Wartości X przy każdym powtórzeniu nadawane są przed wykonaniem rozkazu zaznaczonego gwiazdką zgodnie z tym, co napisano po dwukropku w rozkazie POWTORZ.

Uwaga: gdy X jest całkowite, to liczby określające jego zmienność muszą być całkowite; gdy X jest ułamkowe, to liczby określające jego zmienność muszą być ułamkowe.

§ 17. Zmienna indeksowana

Przykład. Obliczyć i wydrukować wartość sumy:

$$s = a_0 + a_1 + a_2 + \ldots + a_9,$$

gdzie a_0, a_1, ... , a_9 są liczbami danymi.

Zespół liczb a_0, a_1, ..., a_9 wprowadzimy do maszyny jako tzw. blok liczbowy.
Blok liczbowy oznaczamy tak jak zmienną, tylko nazwę jej poprzedzamy gwiazdką. W naszym przykładzie oznaczamy go symbolem: * A. Dowolny element tego bloku, czyli tzw. zmienną indeksowaną oznaczamy: A(I), gdzie A jest nazwą zmiennej, a litera w nawiasach indeksem zmiennej. Indeks może przybierać tylko wartości dodatnie i całkowite.

Odpowiedni program może mieć postać następującą:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA
```

```
BLOK(9) : A
CZYTAJ: * A
S=0
S=S+A(0)
S=S+A(1)
S=S+A(2)
S=S+A(3)
S=S+A(4)
S=S+A(5)
S=S+A(6)
S=S+A(7)
S=S+A(8)
S=S+A(9)
DRUKUJ (4.6) : S
STOP NASTEPNY
KONIEC:0

DANE:
1.
-2.
5.
100.
200.
55.5
0.
0.1
0.02
7.
*
```

Rozkaz BLOK(9) : A powoduje zarezerwowanie dziesięciu kolejnych komórek w pamięci maszyny dla liczb: A(0), A(1), A(2), A(3), A(4), A(5), A(6), A(7), A(8), A(9), tworzących blok: *A.

Uwaga: w rozkazie BLOK opuszcza się gwiazdkę, gdyż z samego rozkazu wynika, że dotyczy on bloku, a nie zmiennej.
Przy wykonywaniu rozkazu CZYTAJ : *A maszyna przeczyta tyle liczb z taśmy danych, ile zarezerwowano komórek rozkazem BLOK i umieści je w przeznaczonych dla nich komórkach pamięci.

Liczby (dane) tworzące blok umieszczamy na taśmie danych kolejno, każdą liczbę w nowej linii, tak jak dane dotyczące zmiennej prostej. Możemy także umieścić je po kilka w jednym wierszu oddzielając spacjami, np.:

```
1.  -2.  5.  100.  200.
55.5  0.  0.1  0.02
*
```

Znakiem kończącym blok liczb danych jest gwiazdka wydrukowana w nowej linii.

Sumowanie liczb danych wykonuje się począwszy od rozkazu piątego: S = 0 i kończy się rozkazem: S = S + A(9).
Niech S oznacza komórkę pamięci, w której tworzy się suma wzrastając od zera o kolejno dodawane składniki. Dlatego zaczynamy od rozkazu:

S = 0

Następnie do tej wartości sumy dodajemy pierwszy składnik A(0) i wynik umieszczamy znów w tej komórce S:

S = S + A(0).

Uwaga: znak równości w języku SAKO ma nieco inne znaczenie niż w zwykłej algebrze. Napisane równanie czytamy:

Nowa wartość S = Poprzednia wartość S + A(0)

Nie wolno skrócić S występującego po obydwu stronach znaku równości! (Patrz § 6).

Podobnie postępujemy ze składnikiem A(1) itd. aż do uzyskania sumy wszystkich danych składników. Drukowania wyniku obliczeń dokonuje rozkaz: DRUKUJ (4.6) : S. Po wykonaniu tego rozkazu maszyna zatrzyma się: STOP NASTEPNY.

Program można znacznie uprościć wprowadzając rozkaz POWTORZ : I = 0 (1) 9, zastępując dziesięć podobnych rozkazów jednym o postaci ogólnej:

$$S = S + A(I).$$

Ponieważ indeks I jest liczbą całkowitą należy rozkazy te poprzedzić rozkazem:
CALKOWITE : (I).

Ostatecznie mamy program następujący:

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

CALKOWITE : I
BLOK (9) : A
CZYTAJ : *A
S=0
*) S=S+A(I)
POWTORZ : I=0(1)9
DRUKUJ (4.6) : S
STOP NASTEPNY
KONIEC:0

DANE:
1.
-2.
```

```
5.
100.
200.
55.5
0.
0.1
0.02
7.
*
```

§ 18. Zmienna indeksowana o dwóch wskaźnikach

Przykład. Wyznaczyć sumę elementów zawartych w tablicy:

$$\begin{array}{cccc} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{array}$$

gdzie $a_{00}, a_{01}, \ldots, a_{23}$ są liczbami danymi.

Oznaczmy element tablicy przez A(I,J), gdzie indeks I odpowiada numerowi wiersza, a indeks J - numerowi kolumny tablicy. Wartość indeksu I zmienia się od zera do dwóch, wartość indeksu J - od zera do trzech.

Układamy program:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

BLOK (2,3) : A
CZYTAJ : *A
```

```
S=0
S=S+A(0,0)
S=S+A(0,1)
S=S+A(0,2)
S=S+A(0,3)

S=S+A(1,0)
S=S+A(1,1)
S=S+A(1,2)
S=S+A(1,3)

S=S+A(2,0)
S=S+A(2,1)
S=S+A(2,2)
S=S+A(2,3)
DRUKUJ (4.6) : S
STOP NASTEPNY
KONIEC:0

DANE:
-2.
3.5
20.
131.1
-0.051
10.2
8.777
1.
155.
5600.
-99.9
-40.44
*
```

Rozkaz BLOK (2.3) : A oznacza, że blok liczb danych jest dwuwymiarowy tj. posiada trzy wiersze i cztery kolumny. Rozkaz powoduje zarezerwowanie w pamięci maszyny (2+1)x(3+1)=12

komórek dla zmiennej A.

Część programu zawarta między rozkazami CZYTAJ :*A i DRUKUJ (4.6) : S wykonuje następujące czynności:

1) zeruje komórkę S : S=0

2) sumuje elementy pierwszego wiersza:

```
S = S + A(0,0)
S = S + A(0,1)
S = S + A(0,2)
S = S + A(0,3)
```

3) do otrzymanej sumy dodaje kolejno elementy drugiego wiersza:

```
S = S + A(1,0)
S = S + A(1,1)
S = S + A(1,2)
S = S + A(1,3)
```

4) do tej sumy dodaje elementy trzeciego wiersza:

```
S = S + A(2,0)
S = S + A(2,1)
S = S + A(2,2)
S = S + A(2,3)
```

Sumowanie elementów pierwszego wiersza zawiera rozkazy, które można napisać w postaci ogólnej:

```
S = S + A(0,J) ,
```

wprowadzając powtórzenie dla J = 0(1)3.

Podobnie dla następnych wierszy:

```
S = S + A(1,J),
S = S + A(2,J).
```

Dlatego program można napisać prościej:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA
CALKOWITE : J
BLOK(2,3): A
CZYTAJ : *A
S=O
*) S=S+A(O,J)
POWTORZ : J=O(1)3
*) S=S+A(1,J)
POWTORZ : J=O(1)3
*) S=S+A(2,J)
POWTORZ : J=O(1)3
DRUKUJ(4.6): S
STOP NASTEPNY
KONIEC:O

DANE:
-2.   3.5   20.   131.1   -0.051   10.2
8.777   1.   155.   56000.   -99.9   -40.44
*
```

W tym programie także powtarzają się trzykrotnie odcinki odpowiadające zmianom A(O,J), A(1,J), A(2,J). Zatem można tę część programu zastąpić wyrażeniem ogólnym zawierającym wyraz A(I,J) i nakazać powtórzenie w zakresie I = O(1)2.

Ostatecznie mamy następujący program:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

CALKOWITE : I,J
BLOK (2,3) : A
```

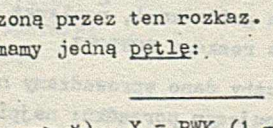
```
CZYTAJ : *A
S=0
* *) S=S+A(I,J)
POWTORZ : J=0(1)3
POWTORZ : I=0(1)2
DRUKUJ (4.6) : S
STOP NASTEPNY
KONIEC:0

DANE
-2.   3.5   20.   131.1   -0.051   10.2
8.777   1.   155.   5600.   -99.9   -40.44
*
```

Uwaga: rozkaz **) S = S + A (I,J) jest opatrzony dwiema gwiazdkami, ponieważ do niego odnoszą się dwa rozkazy POWTORZ.

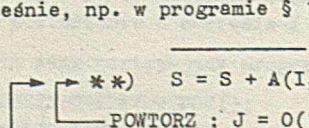
Uwaga: liczby dane wprowadzamy na taśmę danych w następującym porządku: najpierw wszystkie elementy pierwszego wiersza A(0,0), A(0,1), A(0,2), A(0,3), następnie kolejno elementy drugiego i trzeciego wiersza.

§ 19. Uwagi o powtórzeniach

Część programu, która podlega powtarzaniu przy wykonywaniu obliczeń przez maszynę dzięki użyciu rozkazu POWTORZ, nazywa się pętlą utworzoną przez ten rozkaz. Np. w programie § 16 mamy jedną pętlę:

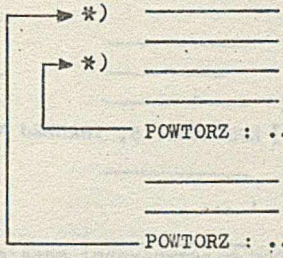


Istnieje możliwość tworzenia kilku pętli jednocześnie, np. w programie § 18 mamy dwie pętle:

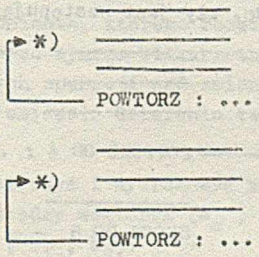


Jeśli w programie jest kilka rozkazów POWTORZ, to pętle należące do nich nie mogą przecinać

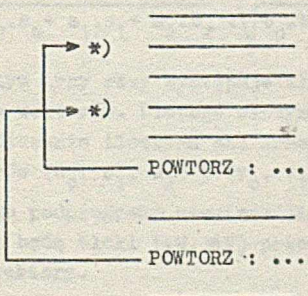
się. Prawidłowe są więc następujące układy pętli:



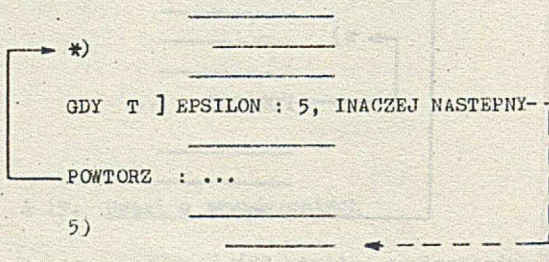
oraz (porównaj str.58)



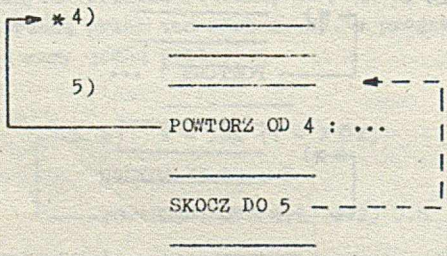
A oto przykład złego programu /pętle przecinają się):



Dozwolone są wyskoki z pętli przed zakończeniem wszystkich obliczeń określonych rozkazem POWTORZ. Np.:



Nie wolno natomiast zarządzać skoku do pętli z zewnątrz. Np. zły jest następujący program:



§ 20. Zmienna indeksowana w podprogramie (użycie rozkazu STRUKTURA)

Jeżeli zagadnienie, którego rozwiązanie mamy zaprogramować wymaga kilkakrotnego wykonania analogicznych operacji nad zmienną indeksowaną, to warto umieścić działania te w podprogramie.

Przykład. Obliczyć i wydrukować wartość cosinusa kąta między wektorem (a_0, a_1, a_2) i wektorem (b_0, b_1, b_2):

$$\cos\alpha = \frac{a_0 \cdot b_0 + a_1 \cdot b_1 + a_2 \cdot b_2}{\sqrt{a_0 \cdot a_0 + a_1 \cdot a_1 + a_2 \cdot b_2} \cdot \sqrt{b_0 \cdot b_0 + b_1 \cdot b_1 + b_2 \cdot b_2}}$$

W zadaniu tym trzy razy występuje iloczyn skalarny dwóch wektorów. Dlatego ułożymy podprogram na obliczenie iloczynu skalarnego dowolnych wektorów (u_0, u_1, u_2) i (v_0, v_1, v_2). Funkcję tego podprogramu nazwiemy ILOCZYN, a jej argumentami będą bloki (*U, *V) przedstawiające obydwa wektory.

```
PODPROGRAM : ILOCZYN(*U, *V)
STRUKTURA (2) : U
STRUKTURA (2) : V
S=0
S=S+U(0)◊V(0)
S=S+U(1)◊V(1)
S=S+U(2)◊V(2)
ILOCZYN ( ) = S
WROC
```

W podprogramie rozkaz STRUKTURA pełni tę samą rolę, co rozkaz BLOK w programie głównym. Różnica polega tylko na tym, że rozkaz STRUKTURA nie rezerwuje dla podprogramu nowych miejsc w pamięci, ale każe korzystać z miejsc zarezerwowanych w programie głównym. Gdyby w podprogramie trzeba było zarezerwować nowe miejsca, o których nie było mowy w programie głównym, to zamiast rozkazu STRUKTURA użylibyśmy rozkazu BLOK.

Rozkaz STRUKTURA stosujemy do określenia rozmiarów bloków liczbowych. W naszym przykładzie rozkazy

```
STRUKTURA (2) : U
STRUKTURA (2) : V
```

oznaczają, że bloki U i V zawierają po trzy liczby każdy (gwiazdkę przed nazwą bloku opuszczamy, ponieważ rozkaz STRUKTURA dotyczy tylko bloków liczbowych).

Można te dwa rozkazy zastąpić jednym, ponieważ bloki U i V mają jednakową strukturę:

```
STRUKTURA (2) : U, V.
```

Podobnie jak w przykładach omówionych poprzednio (§ 17, § 18) możemy wprowadzić wyrażenie arytmetyczne o postaci ogólnej:

```
S = S + U(I)◊V(I)
```

i nakazać powtarzanie tych działań dla I zmieniającego się od zera do dwóch:

```
I = 0 (1) 2
```

Otrzymujemy podprogram o następującej postací:

```
PODPROGRAM : ILOCZYN (*U, *V)
CALKOWITE : I
STRUKTURA (2) : U,V
S=0
 *) S = S + U(I)◊V(I)
POWTORZ : I=0(1)2
ILOCZYN ( )=S
WROC
```

Napiszmy teraz cały program, rozwiązujący dane zagadnienie:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA
BLOK (2) : A,B
CZYTAJ: *A, *B
P = ILOCZYN(*A, *B)
Q = ILOCZYN(*A, *A)
R = ILOCZYN(*B, *B)
C = P/PWK (Q◊R)
DRUKUJ (4.6) : C
STOP NASTEPNY
PODPROGRAM : ILOCZYN(*U, *V)
CALKOWITE : I
STRUKTURA (2) : U,V
S=0
 *) S=S+U(I)◊V(I)
```

```
POWTORZ : I=0(1)2
ILOCZYN ( ) = S
WROC
KONIEC

DANE:
-1.     2.5     -0.76
*
10.     8.1     -3.01
*
```

§ 21. Dalsze uwagi o rozkazie STRUKTURA

Rozkaz STRUKTURA możemy zastosować wtedy, gdy operujemy blokami o rozmiarach, które ma określić sam program. Objaśni to następujący prosty przykład: wczytać blok (N+1) liczb A(0),A(1),...,A(N) i wydrukować sumę ich kwadratów. Rozwiązanie przedstawia następujący program:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.
NIWELUJ NADMIARY:TAK
TRANSLACJA
CALKOWITE:N,I
CZYTAJ:N
BLOK(99):A
STRUKTURA(N):A
CZYTAJ:*A
S=0
 *) S=S+A(I)*2
POWTORZ:I=0(1)N
LINIA 1
DRUKUJ(3.3,6):S
STOP NASTEPNY
KONIEC:O
```

```
DANE
N=2
A= 1.2   3.4   5.6
*
```

Pierwszą daną, którą wczytuje rozkaz CZYTAJ:N jest liczba całkowita, określająca indeks ostatniego wyrazu bloku A. Dla bloku A zarezerwowano 100 komórek pamięci rozkazem BLOK(99). Z tego bloku rozkaz STRUKTURA(N) każe do dalszych obliczeń brać tylko pierwsze (N+1) komórek A(0), A(1),...,A(N). Oczywiście program będzie poprawny dla N nie większych od 99. Dla mniejszych wartości N reszta bloku A będzie po prostu niewykorzystana. Rozkaz CZYTAJ: *A każe wczytać blok o strukturze N. Dlatego w przykładzie danych dla N=2 blok ten zawiera N+1=3 liczby. Pozostałe rozkazy nie wymagają objaśnień.

W podobny sposób można zastosować rozkaz STRUKTURA do bloków wielowymiarowych. Jeśli np. każemy wczytać z arkusza danych wartości całkowite N i M określające indeksy ostatniego wyrazu bloku A(0,0),...,A(N,M) to pierwsze rozkazy programu (po czołówkach) będą mogły mieć postać:

```
CALKOWITE:N,M
BLOK(499):A
CZYTAJ:N,M
STRUKTURA(N,M):A
CZYTAJ: *A
      itd.
```

Tutaj dla bloku A zarezerwowano 500 komórek pamięci (długich), z których program wykorzystuje pierwsze (N+1).(M+1) komórek. W dalszych obliczeniach blok A jest traktowany jako macierz dwuindeksowa, zawierająca N+1 wierszy i M+1 kolumn.

§ 22. Podprogram, którego wynik jest liczbą całkowitą

Napiszmy program, który oblicza liczbę kombinacji z n elementów po m elementów w grupie. Jak wiadomo liczba ta wyraża się wzorem n!/(m!.(n-m)!). Z budowy tego wzoru wynika, że wygodnie będzie zbudować podprogram obliczenia silni. Wynikiem tego podprogramu jest liczba całkowita. Dlatego nazwa podprogramu, zarówno w programie głównym, jak i w podprogramie musi występować na liście wielkości wymienianych w rozkazie CALKOWITE. Przedstawia to poniższy przykład.

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

TRANSLACJA

CALKOWITE:N,M,C,SILNIA
CZYTAJ:N,M
C=SILNIA(N)/(SILNIA(M)◊SILNIA(N-M))
DRUKUJ(10):C
STOP NASTEPNY
PODPROGRAM:SILNIA(K)
CALKOWITE:SILNIA,K,I,J
*)GDY I=0:NASTEPNY,INACZEJ !
```

```
    J=1
    SKOCZ DO 2
1)J=J◊I
2)POWTORZ:I=0(1)K
    SILNIA()=J
    WROC
    KONIEC:0

    10
    3
```

§ 23. Podprogram, który oblicza wartości kilku zmiennych

Przypuśćmy, że znamy współrzędne punktu (X,Y) na płaszczyźnie. Jeśli układ osi współrzędnych obrócimy o kąt α, to nowe współrzędne (U,V) wyrażą się następująco:

$$U = X.\cos\alpha + Y.\sin\alpha$$
$$V = -X.\sin\alpha + Y.\cos\alpha$$

Podprogram transformacji układu współrzędnych według tych wzorów może mieć postać:

```
PODPROGRAM:(U,V)=TRANS(X,Y,ALFA)
U=X◊COS(ALFA) + Y◊SIN(ALFA)
V=Y◊COS(ALFA) - X◊SIN(ALFA)
WROC
```

Podprogram ten różni się od poznanych poprzednio tym, że wynikiem jego działania są wartości kilku zmiennych (w przykładzie dwu). Listę tych zmiennych zapisujemy w nawiasach oddzielając je przecinkami. Lista ta występuje w nagłówku podprogramu, po niej następuje znak równości i dalej nazwa podprogramu z ujętą w nawiasy listą argumentów. W treści podprogramu trzeba napisać wszystkie rozkazy, wykonanie których pozwoli obliczyć zmienne

występujące na liście wyników przy założeniu, że zmienne na liście argumentów są dane. Ostatnim wykonywanym rozkazem podprogramu jest WROC.

Jeśli w programie głównym chcemy wywołać opisany podprogram, to możemy tego dokonać przez napisanie rozkazu o przykładowej postaci:

(R,S)=TRANS(P,Q,BETA)

Rozkaz ten każe podstawić do podprogramu TRANS wartości P,Q,BETA na miejsce X,Y,ALFA. Następnie według podprogramu TRANS mają zostać obliczone wartości U,V. Dalej wartości U,V mają zostać podstawione do programu głównego na miejsce R,S. Przy tym wywołaniu charakter zmiennych występujących w wywołaniu musi zgadzać się z charakterem zmiennych określonych w nagłówku podprogramu. (Np. R nie może być ani całkowitą, ani nazwą bloku, bo U jest zmienną ułamkową.)

Tej samej formy używamy, gdy wynikiem podprogramu jest blok. Wyjaśni to przykład, w którym podprogram SU ma obliczyć wypadkową R wektorów X i Y w przestrzeni (N+1) wymiarowej. Każdy z tych wektorów przedstawimy jako blok, którego kolejne komórki zawierają współrzędne wektora. Jak wiadomo, współrzędne wypadkowej są sumami odpowiednich współrzędnych wektorów składowych. Dlatego podprogram ma postać

```
PODPROGRAM:(*R)=SU(*X, *Y,N)
CALKOWITE:N,I
STRUKTURA(N):R,X,Y
*)R(I)=X(I)+Y(I)
POWTORZ:I=0(1)N
WROC
```

Przypuśćmy, że chcemy zastosować ten podprogram w celu dodania trzech wektorów U,V,W w przestrzeni trójwymiarowej i wynik zapisać w postaci wektora Z. W programie głównym trzeba zarezerwować miejsca dla składowych tych wektorów rozkazem

```
BLOK(2):U,V,W,Z,T
```

dalej powinny następować rozkazy, które wprowadzą odpowiednie wartości do komórek bloków U, V, W. Wypadkową wektorów U, V zapiszemy w bloku T, a potem wypadkową wektorów T i W zapiszemy w bloku R. Wykonają to następujące rozkazy:

```
(*T)=SU(*U, *V,2)
(*R)=SU(*T, *W,2)
```

§ 24. Podstawianie argumentów do podprogramu

W poprzednim paragrafie wystąpił rozkaz:

(*T)=SU(*U,*V,2)

Oznaczał on, że w podprogramie SU należy nazwę bloku X zastąpić nazwą bloku U, nazwę Y zastąpić nazwą V, nazwę R zastąpić nazwą T i na miejsce wartości N wprowadzić wartość równą liczbie 2. Widać, że mieliśmy tu do czynienia z dwoma rodzajami podstawień: przez nazwę i przez wartość. Wszystkie bloki, (których nazwy w wywołaniu i w definiowaniu podprogramu należy poprzedzać gwiazdką) przenoszą się do podprogramu przez nazwę. Wszystkie zmienne przenoszą się przez wartość.

Argumenty przenoszone przez wartość mogą w wywołaniu podprogramu być zastąpione wyrażeniami obliczającymi tę wartość. Np. poprzedni rozkaz moglibyśmy zapisać także w postaci:

(*T)=SU(*U,*V,1+1)

Należało przy tym dbać o to, by wprowadzana wartość była całkowita, jeśli tak ją określono w podprogramie (lub ułamkowa, gdy w podprogramie jest ona ułamkowa). W naszym przy-

kładzie nie wolno było np. napisać

```
(*T)=SU(*U,*V,2.0)
```

lub

```
(*T)=SU(*U,*V,6/3),
```

bo 2.0 jest liczbą ułamkową, podobnie jak wynik dzielenia, ale wolno było napisać

```
(*T)=SU(*U,*V,ENT(6/3)),
```

bo funkcja ENT ma wartość całkowitą.

Podstawianie argumentów po prawej stronie podprogramu można wykonać etapami. Służy do tego rozkaz PODSTAW. Przykład takiego podstawiania może być następujący

```
PODSTAW:SU(*U,.,2)
(*T)=SU(.,*V,.)
```

W rozkazie PODSTAW następuje przesłanie odpowiednich argumentów do podprogramu, bez wykonania działań podprogramu. Te argumenty, których nie przesyłamy, zastępujemy kropką. Po takim podstawieniu, przy wywołaniu podprogramu wystarczy wymienić tylko te argumenty, które nie zostały przesłane, pozostałe zastępując kropką. Jeśli kropka występuje na końcu listy argumentów, można ją po prostu opuścić. A więc możemy także napisać

```
PODSTAW:SU(*U,.,2)
(*T)=SU(., *V)
```

Jeślibyśmy podstawili uprzednio wszystkie argumenty, to jednak jedną kropkę należy zostawić. Np.

```
PODSTAW:SU(*U:*V,2)
(*T)=SU(.)
```

Rozkaz PODSTAW stosujemy wtedy, gdy chcemy oszczędzić pracy maszynie. Np. pod koniec poprzedniego paragrafu napisaliśmy dwa rozkazy, w których powtarzał się ten sam argument w wywołaniach podprogramu SU. Zmniejszymy liczbę przesyłań, gdy prześlemy go na początku rozkazem PODSTAW

```
PODSTAW:SU(.,.,2)
(*T)=SU(*U,*V)
(*R)=SU(*T,*W)
```

§ 25. Podprogramy, których argumentem jest funkcja

Napiszmy podprogram obliczenia całki oznaczonej

$$I = \int_A^B F(X)dX$$

Najprościej będzie podzielić przedział $A \leqslant X \leqslant B$ na kilka, np. 8 przedziałów, obliczyć średnią wartość funkcji obliczonej w punktach podziału (których jest 1+8=9) i pomnożyć ją przez długość przedziału. Oto ten podprogram:

```
PODPROGRAM:CALKA(A,B,F())
DX=(B-A)/8
S=0
*)S=S+F(X)
POWTORZ:X=A(DX)B
CALKA()=(B-A)*S/9
WROC
```

Argumentami podprogramu CALKA są: dolna granica całkowania A, górna granica całkowania B oraz nazwa funkcji F(). Nazwę funkcji, będącej argumentem podprogramu, uzupełniamy dwoma nawiasami (). Funkcję tę należy określić jakimś podprogramem.

Przypuśćmy, że chcemy obliczyć następujące całki:

$$I1 = \int_0^1 \sqrt{1 - X^3}\, dX$$

$$I2 = \int_{0.5}^1 \sqrt{1 - K.\sin^2 X}\, dX$$

Program ma postać następującą:

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

CZYTAJ:K
I1=CALKA(0.0,1.0,G())
PODSTAW:H(.,K)
I2=CALKA(0.5,1.0,H())
LINIA  1
DRUKUJ(3.5):I1,I2
STOP NASTEPNY

PODPROGRAM:CALKA(A,B,F())
DX=(B-A)/8
S=0
*)S=S+F(X)
POWTORZ:X=A(DX)B
CALKA()=(B-A)◊S/9
WROC

PODPROGRAM:G(X)
G()=PWK(1-X*3)
WROC
PODPROGRAM:H(X, K)
H()=PWK(1-K◊SIN(X)*2)
WROC
KONIEC:0
0.5
```

Rozkaz I1=CALKA(0.0,1.0,G()) każe do podprogramu CALKA podstawić A=0.0 , B=1.0 oraz na nazwę funkcji F() podstawić nazwę G(). Następnie rozkaz ten każe wykonać obliczenia według podprogramu CALKA i wynik przesłać do komórki I1.

Rozkaz I2=CALKA(0.5,1.0,H()) dokonuje podobnych przesłań. Teraz rolę funkcji F() odgrywa funkcja H(). Jak widać, w podprogramie CALKA funkcja F() występuje w rozkazie S=S+F(X). A więc tutaj wywołuje się tylko pierwszy argument funkcji H(). Dlatego przed rozkazem I2=CALKA(0.5,1.0,H()) należało drugi argument funkcji H(X,K) wprowadzić do jej podprogramu rozkazem PODSTAW:H(.,K).

§ 26. Podział na rozdziały

Jeżeli programujemy rozwiązanie obszernego zagadnienia, to dobrze jest podzielić cały program na samodzielne części zwane rozdziałami. Np. podzielimy na rozdziały zadanie z § 12:

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

ROZDZIAL : 1
BLOK(0) : A,B,C
CZYTAJ : *A, *B, *C
IDZ DO ROZDZIALU : 2

ROZDZIAL : 2
BLOK(0) : A,B,C
DELTA = B◊B - 4◊A◊C
X1 = (-B - PWK(DELTA))/(2◊A)
X2 = (-B + PWK(DELTA))/(2◊A)
DRUKUJ (2.3) : X1, X2
STOP NASTEPNY
KONIEC:1
```

```
DANE :
1.
*
-3.
*
2.
*
```

Każdy rozdział rozpoczynamy rozkazem

ROZDZIAL : nazwa rozdziału

Nazwą rozdziału musi być liczba naturalna nie większa niż 32000. Przejście do innego rozdziału dokonuje się za pomocą rozkazu

IDZ DO ROZDZIALU:nazwa innego rozdziału.

W każdym nowym rozdziale wszelkie rozkazy, nazwy zmiennych i nazwy podprogramów używane w starym rozdziale - przestają być aktualne.

Jeżeli chcemy przenieść wyniki obliczeń danego rozdziału do innego rozdziału, to na początku obu tych rozdziałów piszemy rozkaz BLOK, rezerwujący tę samą liczbę komórek pamięci. W naszym przykładzie przeniesienie liczb A,B,C z rozdziału 1 do rozdziału 2 dokonuje się przez zaczęcie obu rozdziałów identycznie brzmiącymi rozkazami:

BLOK (0) : A,B,C .

Numer, który występuje po dwukropku w rozkazie KONIEC:1 oznacza nazwę tego rozdziału, od którego ma zaczynać się liczenie.

§ 27. Pisanie na bęben i czytanie z bębna zmiennych prostych

Dysponujemy jeszcze jedną możliwością przenoszenia wyników liczbowych z jednego rozdziału do innego. Napiszmy program z § 26 w postaci następującej:

```
WYJSCIE:0=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.

NIWELUJ NADMIARY:TAK
TRANSLACJA

ROZDZIAL : 1
CZYTAJ : A,B,C
PISZ NA BEBEN OD 100 : A,B,C
IDZ DO ROZDZIALU : 2

ROZDZIAL : 2
CZYTAJ Z BEBNA OD 100 : A,B,C
DELTA = B◊B - 4◊A◊C
X1 = (-B - PWK (DELTA))/(2◊A)
X2 = (-B + PWK( DELTA))/ (2◊A)
DRUKUJ (2.3) : X1,X2
STOP NASTEPNY
KONIEC:1
```

DANE :
1.
-3.
2.

Wykonując rozkaz

PISZ NA BEBEN OD 100 : A,B,C

maszyna kopiuje w kolejnych komórkach pamięci bębnowej, rozpoczynając od komórki 100, liczby zapisane w komórkach A,B,C pamięci operacyjnej. Poprzednia zawartość wspomnianych komórek pamięci bębnowej ulega zniszczeniu, a zawartość komórek A,B,C pamięci operacyjnej zostaje zachowana.

Przy wykonywaniu rozkazu

CZYTAJ Z BEBNA OD 100 : A, B, C

odbywa się proces odwrotny. Oznacza to, że zawartość kolejnych komórek pamięci bębnowej, zaczynając od komórki 100 zostanie skopiowana w komórkach A, B, C pamięci operacyjnej. Przy tym poprzednia zawartość komórek A, B, C ulegnie zniszczeniu, natomiast zawartość pamięci bębnowej nie zmieni się.

Liczbę ułamkową maszyna zapisuje w dwóch kolejnych komórkach pamięci bębnowej. W naszym programie użyliśmy więc komórek 100, 101, ...,105. Natomiast liczbę całkowitą maszyna zapisuje w jednej komórce.

Uwaga: bęben zawiera pewną liczbę komórek ponumerowanych kolejno 0, 1, 2, ... Liczba ta zależy od zestawu maszyny.

§ 28. Pisanie na bęben i czytanie z bębna bloków liczbowych

Przykład. Obliczyć sumę:

$$\sum_{0}^{1000} a_1 \qquad \text{gdzie } a_1 \text{ są dane.}$$

Ułóżmy następujący program:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM.
NIWELUJ NADMIARY:TAK
TRANSLACJA
ROZDZIAL : 1
CALKOWITE : K
BLOK (199) : A
*) CZYTAJ : *A
PISZ NA BEBEN OD K : *A
POWTORZ : K=0(400)1600
IDZ DO ROZDZIALU : 2
ROZDZIAL : 2
CALKOWITE : K,L
BLOK (199): A
S=0
```

```
*) CZYTAJ Z BEBNA OD K : *A
*) S=S+A(L)
POWTORZ : L=0(1)199
POWTORZ : K=0(400)1600
DRUKUJ(5.5) : S
STOP NASTEPNY
KONIEC:1

DANE:
1.7  41.2  6.02  45.7  0.1  0.65  1.  78.
647.  2.5  30.  9.9  0.004  -2.  -74.  itd.
*
```

W rozdziale 1 przy wykonywaniu obliczeń w pętli zostają wczytane wszystkie liczby dane grupami, po 200 liczb i zapisane na bębnie. Każda taka grupa zajmuje 400 komórek pamięci bębnowej.

Duża pętla w rozdziale 2 kopiuje liczby zapisane na bębnie w pamięci operacyjnej, grupami po 200 liczb. Liczby te są sumowane za pomocą rozkazów w małej pętli.

Na taśmie danych należy liczby umieszczać blokami po 200 liczb, czyli po wypisaniu każdej grupy 200 liczb drukujemy gwiazdkę w nowej linii.

§ 29. Kilkakrotne wykonanie programu

We wszystkich poprzednich przykładach zakładaliśmy, że każdy program był wykonywany jeden raz. Jeśli chcemy po wczytaniu programu wykonać działania wskazane tym programem dwa razy, to zaznaczamy to w pierwszej czołówce przez dwukrotne napisanie słowa PROGRAM. Oto przykład takiej czołówki:

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM,PROGRAM.
```

Oczywiście dla takiego programu należy napisać dwa warianty danych, jeden po drugim, przy czym na taśmie perforowanej między obiema partiami danych musi występować odcinek conajmniej dziesięciu centymetrów pustej taśmy. Gdybyśmy chcieli obliczenia wykonać więcej razy, np. cztery, to czołówka miałaby postać

```
WYJSCIE:O=PDW(DW,DDW1);
PROBLEM:SAKO,PROGRAM,
PROGRAM,PROGRAM, PROGRAM.
```

§ 30. Zestawienie omówionych rozkazów języka SAKO

Uwaga: w każdym rozkazie została podkreślona niezmienna część tego rozkazu.

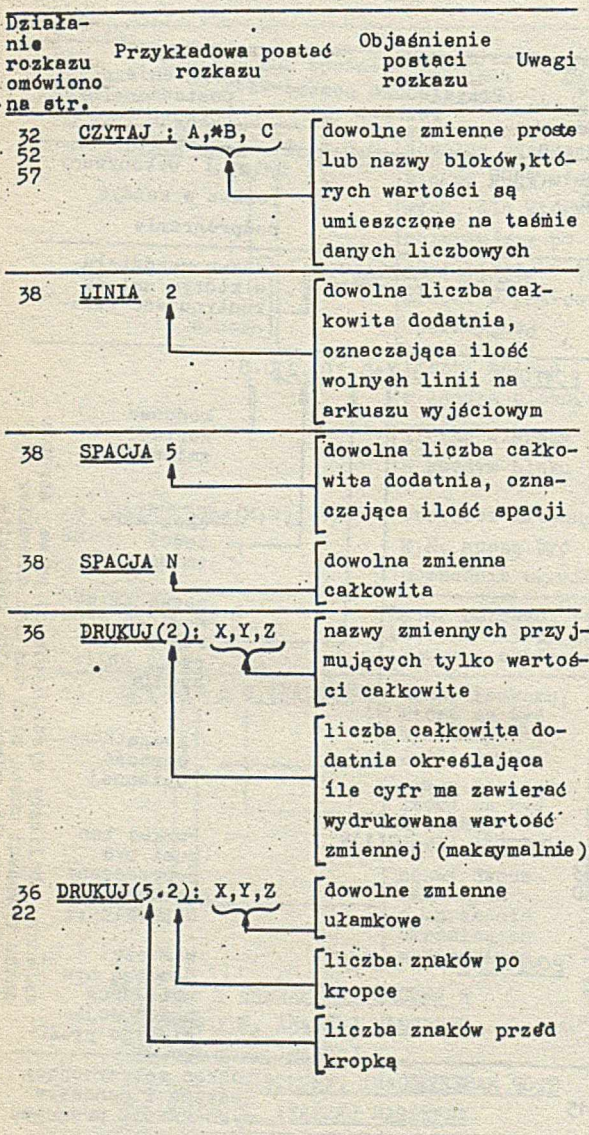
Działanie rozkazu omówiono na str.	Przykładowa postać rozkazu	Objaśnienie postaci rozkazu	Uwagi
15	WYJSCIE:O=PDW(DW,DDW1); PROBLEM:SAKO,PROGRAM. NIWELUJ NADMIARY:TAK TRANSLACJA	Typowa czołówka programu, która każe wczytać program i potem raz go wykonać drukując wyniki na drukarce wierszowej. Po kropce po słowie PROGRAM. należy na taśmie perforowanej wydziurkować znak powrotu karetki, potem około pół metra pustej taśmy.	
86	WYJSCIE:O=PDW(DW,DDW1); PROBLEM:SAKO,PROGRAM, PROGRAM,PROGRAM. NIWELUJ NADMIARY:TAK TRANSLACJA	Jak wyżej z tym, że program zostanie wykonany trzykrotnie	
81	ROZDZIAL : 1	nazwa rozdziału w postaci liczby całkowitej	

Działanie rozkazu omówiono na str.	Przykładowa postać rozkazu	Objaśnienie postaci rozkazu	Uwagi
24 26	PODPROGRAM: SH(X)	argument funkcji definiowanej nazwa funkcji definiowanej	
28	PODPROGRAM:DLUGOSC(X,Y,U,V)	argument funkcji wielu zmiennych nazwa funkcji wielu zmiennych	
64	PODPROGRAM: F (X, *A)	nazwa bloku, zmienna } argumenty	
77	PODPROGRAM:CALKA(A,B,F())	nazwa funkcji, zmienna, argumenty nazwa funkcji	
71 72	PODPROGRAM: (A, *B)=F(C,*D,E(),F)	argumenty nazwa podprogramu lista wyników	
14	KONIEC:0	rozkaz kończący program, w którym nie podano nazwy rozdziału.	
80	KONIEC:57	Koniec programu wielorozdziałowego. Nazwa rozdziału, od którego należy zacząć wykonywanie programu	

Działanie rozkazu omówiono na str.	Przykładowa postać rozkazu	Objaśnienie postaci rozkazu	Uwagi
30 53 69	CAŁKOWITE: A,B,C,*D,*E	dowolne zmienne, bloki lub nazwy podprogramów	
81	BLOK (0) : A,B,C	dowolne zmienne proste, dla których rezerwujemy po jednym miejscu w pamięci operacyjnej	
51 68	BLOK(10): A,B,C	dowolne oznaczenie bloków, dla których rezerwujemy po 10+1 miejsc w pamięci operacyjnej.	
		liczba miejsc zarezerwowanych dla każdego bloku minus 1.	
66	BLOK (2,3): A,B	dowolne oznaczenia bloków dwuwymiarowych, dla których rezerwujemy po (2+1) (3+1)= 12 miejsc w pamięci operacyjnej	
		liczba kolumn w bloku minus 1	
		liczba wierszy w bloku minus 1	
64	STRUKTURA(2): U,V	nazwy bloków liczbowych	
		ilość liczb pojedynczego bloku liczbowego minus 1	

Działanie rozkazu omówiono na str.	Przykładowa postać rozkazu	Objaśnienie postaci rozkazu	Uwagi
64	STRUKTURA(2,3): A,B,C	dowolne oznaczenia bloków dwuwymiarowych, dla których rezerwujemy po (2+1)x(3+1) = 12 miejsc w pamięci operacyjnej	
		liczba kolumn w każdym bloku	
		liczba wierszy w każdym bloku	
67 68	STRUKTURA(M,N): A,B	wartości zmiennych M,N, muszą być uprzednio określone	
46	SKOCZ DO 6	numer rozkazu, od którego należy kontynuować obliczenia	
43	GDY A]B : 5,INACZEJ 6	numer rozkazu, który ma być wykonany, jeśli A ≤ B,	A i B są to liczby, zmienne lub wyrażenia algebraiczne
		numer rozkazu, który ma być wykonany, jeśli A > B,	
		nazwy dwóch zmiennych o uprzednio określonych wartościach	
	GDY A]B : NASTEPNY, INACZEJ 6		
	GDY A]B : 5, INACZEJ NASTEPNY		
	GDY A=B : 5, INACZEJ 6		
	GDY A=B : NASTEPNY, INACZEJ 6		
	GDY A=B : 5, INACZEJ NASTEPNY		

Działanie rozkazu omówiono na str.	Przykładowa postać rozkazu	Objaśnienie postaci rozkazu	Uwagi
25 72	WROC	ostatni wykonywany rozkaz w każdym podprogramie	
81	IDZ DO ROZDZIAŁU: 2	nazwa rozdziału, w którym należy kontynuować obliczenia	
49	POWTORZ : X=0.2(0.2)0.8	końcowa wartość zmiennej skok wartości zmiennej nazwa zmiennej, dla której powtarzamy obliczenia początkowa wartość zmiennej	zmienne lub liczby występujące po prawej stronie znaku równości muszą mieć ten sam charakter tj. wszystkie są albo ułamkowe albo całkowite
53 60	POWTORZ : I = 0(1)9	rozkaz ten musi być poprzedzony rozkazem CALKOWITE:I	
	POWTORZ : I = A(B)C	wartości A,B, C muszą być obliczone przed wejściem do pętli	
15	STOP NASTEPNY	rozkaz zatrzymujący maszynę i kończący wykonywanie programu	


Działanie rozkazu omówiono na str.
Przykładowa postać rozkazu
Objaśnienie postaci rozkazu
Uwagi
32 52 57
CZYTAJ : A,*B, C
dowolne zmienne proste lub nazwy bloków, których wartości są umieszczone na taśmie danych liczbowych
38
LINIA 2
dowolna liczba całkowita dodatnia, oznaczająca ilość wolnych linii na arkuszu wyjściowym
38
SPACJA 5
dowolna liczba całkowita dodatnia, oznaczająca ilość spacji
38
SPACJA N
dowolna zmienna całkowita
36
DRUKUJ(2): X,Y,Z
nazwy zmiennych przyjmujących tylko wartości całkowite
liczba całkowita dodatnia określająca ile cyfr ma zawierać wydrukowana wartość zmiennej (maksymalnie)
36 22
DRUKUJ(5.2): X,Y,Z
dowolne zmienne ułamkowe
liczba znaków po kropce
liczba znaków przed kropką

Działanie rozkazu omówiono na str.

37 DRUKUJ (2.3.4)

- minimalna ilość cyfr znaczących
- ilość cyfr po kropce
- ilość cyfr przed kropką

38 TEKST:
DRUKOWANIE WYNIKOW

- dowolny tekst, jaki chcemy umieścić na arkuszu wyników obliczeń

82 PISZ NA BEBEN OD 100 : A,B,C

- nazwy zmiennych
- numer komórki pamięci bębnowej, począwszy od której będą zapisane wartości zmiennych

84 PISZ NA BEBEN OD 60 : *A

- dowolny blok liczbowy

82 CZYTAJ Z BEBNA OD 100 : A,B,C
84

- dowolne zmienne proste
- numer komórki pamięci bębnowej, począwszy od której będą skopiowane w komórkach A,B,C pamięci operacyjnej liczby A,B,C

§ 31. Błędy programu

W trakcie wczytywania programu SAKO mogą pojawić się błędy. Niektóre z nich maszyna sama zauważy i zasygnalizuje wypisując listę błędów w następujący sposób: najpierw wypisuje numer rozdziału, w którym wystąpił błąd, a dalej tabelkę zawierającą cztery kolumny.

W pierwszej kolumnie maszyna wydrukowuje numer rozkazu (taki jak występuje w programie), po którym nastąpił błąd. Jeśli takiego numeru nie było, to drukuje się słowo POCZ.

W drugiej kolumnie występuje liczba wskazująca wiersz, w którym jest błąd. Liczba ta jest zerem, gdy błąd wystąpił w wierszu o numerze podanym w pierwszej kolumnie, jedynką gdy w następnym wierszu itd.

W trzeciej kolumnie maszyna wypisuje słowo BLAD oraz numer, który według poniższego zestawienia określa rodzaj błędu.

W czwartej kolumnie może wystąpić słowo CZOL (gdy błąd dotyczy czołówki), słowo GLOW (gdy błąd dotyczy programu głównego) lub słowo będące nazwą podprogramu (gdy błąd wystąpił w tym podprogramie).

BLAD 1. Ciąg znaków pobrany przez translator został potraktowany jako wiersz programu zawierający więcej niż 128 znaków

BLAD 2. Wiersz pobrany przez translator nie został rozpoznany jako zdanie języka SAKO

BLAD 4. Liczba występująca w języku SAKO nie jest poprawna

BLAD 5. Nie rozpoznano końca programu

BLAD 8. Zdanie rozpoznane jako ROZDZIAL posiada niepoprawny parametr

BLAD 10. Translator natrafił na zdanie POWTORZ, dla którego nie wystąpiła gwiazdka

BLAD 20. Błędna składnia zdania

BLAD 21. Zbyt dużo miejsc zajmuje zdanie BLOK

BLAD 22. Zabrakło translatorowi miejsca w pamięci operacyjnej na listy identyfikatorów użytych w zdaniach: BLOK, STRUKTURA, CALKOWITE, POWTORZ

BLAD 23. Argument jest całkowity a powinien być ułamkowy

BLAD 24. Niewłaściwa liczba indeksów zmiennej indeksowanej

BLAD 25. W wyrażeniu arytmetycznym występuje nadmiar lub dzielenie przez zero

BLAD 26. Wystąpiło w programie więcej gwiazdek niż odpowiadających im rozkazów POWTORZ

BLAD 27. Ujemny lub ułamkowy zakres wymiaru w deklaracji blok

BLAD 30. Argument jest ułamkowy a powinien być całkowity

BLAD 31. Parametry POWTORZ są niejednolite

BLAD 32. WROC w programie głównym

BLAD 33. Niezadeklarowany BLOK lub STRUKTURA

BLAD 34. Zdanie typu F ( )= wystąpiło w programie głównym

BLAD 39. Brak nawiasu zamykającego w wyrażeniu arytmetycznym

BLAD 71. Użyto zbyt dużo zmiennych całkowitych

BLAD 72. Brak miejsca w translatorze na listę nazw i listę stałych

BLAD 75. Powtórna deklaracja numeru rozkazu

BLAD 76. Powtórna deklaracja podprogramu

BLAD 78. Powtarza się nazwa bloku

BLAD 92. Brak pamięci dla translatora przy dołączeniu funkcji języka

BLAD 96. Brak miejsca na liście stałych w procesie adresacji

BLAD 97. Brak numeru rozkazu

BLAD 98. Brak deklaracji BLOK lub PODPROGRAM

BLAD 99. Program wynikowy nie mieści się w pamięci operacyjnej

BLAD 89. Program zawiera więcej rozdziałów niż przewiduje translator

BLAD 100. Brak pamięci bębnowej dla dalszej translacji

BLAD 200. Rozpoznany zapis nie jest poprawnym zdaniem języka czołówki

BLAD 201. Obraz wiersza na tabulogramie nie odpowiada informacjom umieszczonym na taśmie (niewłaściwe użycie znaku CR)

BLAD 202. Niezidentyfikowany tekst po końcu zdania czołówki SAKO

BLAD 203. Błędny parametr w zdaniu języka czołówki

BLAD 204. Koniec taśmy (nadmiar wejścia) w taśmie czołówki

BLAD 205. Brak znaku nowej linii po zdaniu końca czołówki

§ 32. Błędy w czasie wykonywania programu

Nazwa błędu	Znaczenie
ZZ	Dzielenie przez zero zmiennoprzecinkowe
ZP	Wystąpił podmiar zmiennoprzecinkowy
ZW	Wystąpił nadmiar zmiennoprzecinkowy
PA	Brak pamięci dla rozdziału, zbyt duży rozdział
RB	Błędny numer rozdziału
RN	Brak rozdziału
PC	Drukuj argument nieznormalizowany
BD	Błędne parametry zdania DRUKUJ
NC	Koniec danych dla CZYTAJ WIERSZ (nadmiar wejścia)
EX	Błędny argument funkcji EXP
LN	Błędny argument funkcji LN
BK	Nadmiar wejścia przy CZYTAJ
BC	Błędne dane dla CZYTAJ
IL	Błędna liczba elementów bloku
NG	Gwiazdka zamiast pojedynczej liczby w CZYTAJ

Nazwa błędu	Znaczenie
DK	Nadmiar potęgi całkowitej
UW	Ujemny wykładnik potęgi całkowitej lub ułamkowej o wykładniku całkowitym
AC	Błąd argumentu funkcji ACS lub ASN
TG	Błąd argumentu funkcji TG

§. 33. Słownik wyrazów użytych w tekście

Liczby oznaczają strony z objaśnieniem hasła

WPM "WEMA" Warszawa 1972. Wyd. I. Format 100x197. Nakład 2000+45 egz
Ark. wyd. 3,2. Ark. druk. 8,67. Zam. 803/71-5-Z/S
WPM "WEMA" Zakład Poligraficzny w Białymstoku Zam. 13 U-95