

1. SYMBOLE. LICZBY. ZMIENNE. FUNKCJE

1.1. Symbole używane w GIER-ALGOLu

W GIER-ALGOLu używamy cyfr dziesiętnych 0,1,...,9, następnie zarówno małych jak i dużych liter alfabetu, nie zawierającego polskich liter ą, ć, ę, ł, ń, ó, ś, ź, ż, natomiast zawierającego duńskie litery æ i ø. Ich dużymi odpowiednikami są Æ i Ø. (Oryginalny ALGOL-60 nie dopuszcza tych duńskich liter).

Ponadto używamy następujących symboli, których znaczenie będziemy później stopniowo poznawać:

$\wedge \vee \times / - + _ \uparrow = _ : ; , . [] () < > _ \dagger \ddagger \leq \geq \dagger \equiv \Rightarrow \neg, :=$

W programach pisanych ręcznie wygodnie jest zamiast symbolu \times używać $*$ (aby nie mylić z literą x), zamiast symbolu \uparrow używać \dagger , zamiast $_$ symbol \div , zamiast \leq symbol \leq , zamiast \geq symbol \geq , a zamiast \neg , symbol $_$.

Znak₁₀, tzw. d z i e s i ą t k a a l g o l o w s k a stanowi nierozdzielny symbol, pisany i drukowany nieco niżej od innych symboli, podobnie jak znak podkreślający $_$.

W oryginalnym ALGOLu-60 nie używa się znaku \times ani $*$ a tylko \times , nie używa się \Rightarrow a tylko \supset , nie używa się \neg , a tylko $_$, zamiast $_$ używa się $_$, zamiast \leq i \geq symboli \leq i \geq .

Na flexowriterze odstęp (SPACE; czytaj: spejs) uzyskuje się przyciskając sztabkę u dołu klawiatury, analogicznie jak na maszynie do pisania.

Odskok karetki na flexowriterze z równoczesnym przejściem do następnego wiersza uzyskuje się przez przyciśnięcie klawisza CAR RET (po angielsku CARRIage RETurn znaczy powrót karetki; czytaj: keridż riteru).

Należy zapamiętać, że translator ALGOLu - wyjawszy pewne sytuacje, o których będzie mowa później - ignoruje przejścia do nowego wiersza i odstępy. Wynika stąd, że pisząc program w języku ALGOL możemy dowolnie umieszczać odstępy lub je pomijać, jak również w dowolnym miejscu programu przechodzić do nowego wiersza, niezależnie od tego, czy program piszemy ręcznie, czy na flexowriterze lub maszynie do pisania. Nie dotyczy to jednak wyrazów podkreślonych takich jak begin, real itp., gdyż są one w ALGOLu traktowane jak pojedyncze symbole i wobec tego nie można ich dzielić. Nie dotyczy to również symboli złożonych, takich jak $:=$ czy \Rightarrow .

Na przykład program podany przykładowo w paragrafie 0.3 może być napisany również następująco:

```

begin real a
,
b ,c x
;a := 1;b:=2x+3;c:=(a+6.1
) / (b+5)-0.1
5; a:=1.2x
a;b:=x:=c:=a+
b+c;end

```

a translator potraktuje go tak samo, jak w poprzednim przypadku.

1.2. Liczby całkowite

W ALGOLu zapisujemy liczby całkowite w sposób tradycyjny w systemie dziesiętnym, pamiętając, że:

- znak + przed liczbą może być opuszczony,
- cyfry można grupować w grupy oddzielając je odstępami,
- liczbę można poprzedzić zerami, co nie ma znaczenia.
- liczba zero może być ze znakiem minus.

Przykłady liczb całkowitych:

85	197 576 312	-0	005
-2	1 000 000	0000	-007 312 639
0	+5	-00000	+00001

1.3. Liczby dziesiętne

W ALGOLu zapisujemy liczby dziesiętne również w sposób tradycyjny, z tym, że część całkowitą oddzielamy od ułamkowej za pomocą kropki (nie wolno tu użyć przecinka). Liczby całkowite mogą być traktowane jako liczby dziesiętne.

Należy jednak pamiętać o następujących regułach:

- znak + przed liczbą może być opuszczony,
- można opuścić pojedyncze zero przed kropką,
- nie można użyć kropki bez napisania po niej co najmniej jednej cyfry,
- liczbę można poprzedzać zerami, co nie ma znaczenia,
- cyfry można dzielić na grupy za pomocą odstępów.

Przykłady liczb dziesiętnych:

0	+0.7800	5	-.00087	1238 576.312
279 283	-268.974	5.0	-0.00087	+0
0006	.5894	5.000	-000.00087	.0

1.4. Liczby zmiennoprzecinkowe

Gdy liczba jest bardzo duża albo bardzo mała, zwykły zapis liczby dziesiętnej jest trudny. W tradycyjnym języku matematycznym stosujemy w takich przypadkach zapis liczby w postaci

$$m \cdot 10^c,$$

gdzie c jest liczbą całkowitą, a m dowolną liczbą dziesiętną, np.

$$3,16 \cdot 10^7, \quad 0,23 \cdot 10^{-5},$$

Liczby tak zapisane nazywamy **z m i e n n o p r z e c i n k o w y m i *** (po angielsku używa się tu nazwy floating-point numbers, tzn. liczby z pływającą kropką). Nazwa "liczby zmiennoprzecinkowe" przyjęła się w Polsce, chociaż w ALGOLu bardziej stosowne jest określenie angielskie, ponieważ kropka, a nie przecinek oddziela część całkowitą od ułamkowej.

W podanym wyżej zapisie liczby zmiennoprzecinkowej liczbę m nazywamy **m a n t y s ą**, a liczbę c **c e c h ą** tej liczby zmiennoprzecinkowej.

W ALGOLu można używać liczb zmiennoprzecinkowych. Zapis ich jest jednak nieco różny od tradycyjnego, a mianowicie:

- zamiast liczby 10 pisanej tradycyjnie na jednym poziomie z mantysą, używa się w ALGOLu tzw. dziesiątki algolowskiej 10 , o której była już mowa w paragrafie 1.1.,

- cechę pisze się na jednym poziomie z mantysą, co jest możliwe ze względu na obniżone położenie dziesiątki algolowskiej (w ten sposób uzyskuje się jednopoziomowy zapis algolowski dla liczb zmiennoprzecinkowych, zgodnie z ogólnymi założeniami dla autokodów, o czym była mowa w paragrafie 0.1.),

- między mantysą a dziesiątką algolowską nie pisze się żadnego znaku,

- gdy mantysa jest równa 1 lub -1, można tej cyfry i w ogóle nie pisać,

- gdy cecha jest równa 0, można jej w ogóle nie pisać, opuszczając wtedy również dziesiątkę algolowską,

- nie można pisać dziesiątki algolowskiej, gdy po niej nie pisze się cechy,

- można opuszczać znak + zarówno przed mantysą, jak i przed cechą,

*Istnieje tutaj pewna rozbieżność między matematycznym pojęciem liczby a algolowskim, ponieważ w tym ostatnim uwzględnia się jeszcze sposób zapisu liczby.

- nie używa się nigdy zapisu zmiennoprzecinkowego dla zmiennych, tzn. zarówno mantysa jak i cecha są w ALGOLu zawsze liczbami, a nigdy nie są zmiennymi,

- ani cechy ani mantysy nie ujmują się w nawiasy,

- do mantysy odnoszą się wszystkie reguły podane w paragrafie poprzednim dla liczb dziesiętnych, a do cechy wszystkie reguły podane w paragrafie 1.2 dla liczb całkowitych.

A oto przykłady liczb zmiennoprzecinkowych:

$+ .78_{10}6$, $- .065_{10}-2$, $+_{10}+3$, $324\ 576_{10}-1$,
 $18.37_{10}+13$, $-_{10}9$, $1_{10}0$, $3.24\ 57_{10}4$,
 $3_{10}-3$, $_{10}-5$, $-_{10}-1$, $.0_{10}0$,

Niektóre z powyższych przykładów nie mają praktycznego znaczenia, ale zostały podane dla ilustracji możliwości zapisu algolowskiego. Najbardziej dziwaczne - a przy tym w pełni zgodne z zasadami pisowni algolowskiej - postaci liczb można uzyskać wykorzystując fakt, że odstęp i przejścia do nowego wiersza są przez translator ALGOLu ignorowane. Przykłady takie nie mają jednak praktycznego znaczenia.

Liczby dziesiętne i liczby zmiennoprzecinkowe obejmujemy jedną nazwą l i c z b r z e c z y w i s t y c h (real), z wyjątkiem liczb całkowitych, które zaliczamy do klasy liczb rzeczywistych lub nie, w zależności od tego, czy piszemy je z użyciem kropki lub dziesiętnej algolowskiej, czy nie. Na przykład liczby 5.0 czy $50_{10}-1$ zaliczymy do rzeczywistych, a liczby 5 czy 50 do całkowitych.

1.5. Zadanie

Które z podanych niżej zapisów są zapisami liczb dozwolonymi w ALGOLu, a które nie:

- | | | |
|-----------------------|----------------------|-----------------------|
| 1) 10 | 9) $+ .0$ | 17) $-1_{10}0$ |
| 2) $- .0$ 01 | 10) $- .0$ | 18) $-1_{10}-0$ |
| 3) 25. | 11) $5.6_{10}(-2)$ | 19) $-1_{10}-000$ |
| 4) $-1.2_{10}k$ | 12) $_{10}0$ | 20) $+111_{10}11$ |
| 5) $+1382.567_{10}+4$ | 13) $2_{10}1.2$ | 21) $3.(6)$ |
| 6) $+3.162$ | 14) $987\ 654\ .321$ | 22) $2 \times_{10}-1$ |
| 7) $11,11_{10}-2$ | 15) -10^2 | 23) $123.456.789$ |
| 8) $599._{10}-4$ | 16) $-_{10}1$ | 24) $_{10}$ |

1.6. Zadanie

1) Następujące liczby napisać jako zmiennoprzecinkowe o mantysie nie mniejszej niż 1 i mniejszej niż 10:

238 571 896	0.01	1
15833	-0.00 0053	.1
-3.21	+0.000 000 000 675	-.111 111

2) następujące liczby napisać jako dziesiętne:

$2.74_{10}2$	$+_{10}+5$	$+_{10}.0_{10}5$	$1_{10}1$
$-389_{10}5$	$-88_{10}-7$	$18_{10}-0$	$-1_{10}-1$
10^{-3}	$.000765_{10}-2$	$0_{10}0$	$-.1_{10}-1$

1.7. Ograniczenia zakresu liczb

W każdej maszynie cyfrowej zakres wielkości liczb jest ograniczony względami technicznymi.

W maszynach typu GIER liczby całkowite (integer) nie mogą przekraczać co do modułu (tzn. co do wartości bezwzględnej) liczby $2^{29} = 536\,870\,912$, natomiast liczby rzeczywiste (real) różne od zera muszą co do modułu zawierać się między $2^{-512} = 7.458_{10} \cdot 10^{-155}$ a $2^{512} = 1.341_{10} \cdot 10^{154}$. Widzimy, że zakres dla liczb rzeczywistych jest, na szczęście, bardzo szeroki.

Należy tu wyjaśnić, że podział liczb rzeczywistych na liczby dziesiętne i liczby zmiennoprzecinkowe istnieje tylko w zapisie programu w języku ALGOL, w zapisie danych wprowadzanych do maszyny i zapisie wyników wyprowadzanych z maszyny, natomiast po wprowadzeniu do maszyny, wszystkie liczby rzeczywiste mają jednolitą postać i są jednakoowo traktowane, niezależnie od tego, czy były do maszyny wprowadzane jako dziesiętne czy zmiennoprzecinkowe.

Liczby całkowite, o ile nie wykraczają poza dozwolony zakres, nie są w maszynie obciążone żadnymi błędami, i rachunki na nich są wykonywane dokładnie. Natomiast liczby rzeczywiste są w maszynie obciążone błędami zaokrągleń i ich dokładność względna jest w maszynach GIER rzędu $3_{10}-9$. Wynika stąd, że w zapisie liczb rzeczywistych, zarówno dziesiętnych jak i zmiennoprzecinkowych, w GIER-ALGOLu nie ma sensu używać więcej jak 10 czy najwyżej 11 cyfr znaczących (można użyć nawet 11, ale tylko ze względu na prawidłowe zaokrąglenie, natomiast dokładność obliczeń nie sięga tak daleko). Nie dotyczy to programów specjalnych na tzw. podwójną czy jeszcze wyższą precyzję. O takich programach nie będzie w niniejszym opracowaniu mowy.

Wymienione ograniczenia odnoszą się również do wartości zmiennych i wszelkich innych wartości liczbowych otrzymywanych w trakcie obliczeń.

1.8. Zmienne

W tradycyjnym zapisie matematycznym zmienne są oznaczane jednoliterowymi symbolami z ewentualnym użyciem indeksów. Natomiast dwie litery lub więcej pisane bezpośrednio po sobie oznaczają w zapisie tradycyjnym iloczyn zmiennych. Na przykład symbole:

$p, x, u_1, a_{11}, t_{wew}$

oznaczają pojedyncze zmienne, natomiast:

ab, kit, xyz

oznaczają iloczyny zmiennych odpowiednio a i b ; k , i oraz t ; x , y i z . System powyższy niezupełnie jest konsekwentny, gdyż w zapisie

$\sin x$

\sin wcale nie oznacza iloczynu zmiennych s , i oraz n .

Że w tradycyjnym zapisie ograniczenie symbolu zmiennej do jednej tylko litery było krępujące, możemy zauważyć w niektórych podręcznikach fizyki, gdzie często dla ułatwienia czytelnikowi orientacji pisze się np.

$$\text{szybkość} = \frac{\text{droga}}{\text{czas}},$$

nie bardzo troszcząc się o to, że według obowiązujących w tradycyjnym zapisie reguł np. czas powinien być interpretowany jako iloczyn zmiennych c , z , a oraz s .

Jak zobaczymy w dalszych paragrafach, w ALGOLu zapisujemy iloczyny zawsze za pomocą operatora mnożenia. Wobec tego nic nie stoi na przeszkodzie, aby zmienne oznaczać niekoniecznie tylko jedną literą.

W ALGOLu nazwy zmiennych formuje się z liter alfabetu i cyfr dziesiętnych z dodatkową umową, że pierwszym symbolem musi być litera. Poza tym ograniczeniem istnieje jeszcze drugie, mianowicie nie dopuszcza się nazw zmiennych identycznych z nazwami tzw. procedur standardowych, do których należą m.in. tzw. funkcje standardowe, jak np. \sin (patrz paragraf 1.15).

Oto lista nazw zastrzeżonych w GIER-ALGOLu dla procedur standardowych:

abs	inone	setchar
arctan	input	sign
char	kbon	sin
cos	ln	split
drumplace	lyn	sqrt
entier	outchar	to buf
exp	outclear	to car
from buf	outcopy	to drum
from car	outer	typechar
from drum	output	typein
gier	outsp	write
gierdrum	outsum	writchar
gierproc	outtext	writcopy
inchar	pack	writcr
		writetext

Przy zachowaniu powyższych dwu ograniczeń wszelkie kombinacje liter i cyfr (a także odstępów, które są przez translator z reguły ignorowane) mogą być użyte jako nazwy zmiennych. Oto przykłady nazw zmiennych:

q,	koszt,	Ania z kwiatkiem,	temperatura koncowa,
droga,	lambda,	x1,	a34kTMNs,
int 3,	alfa 2,	x1a2,	kon by sie usmial,
V17a,	volume 13,	xxxx,	ojojjoj,
tu i tam,	koniec,	xyz,-	kropka;

ALGOL dopuszcza używanie jako nazw zmiennych tych wyrazów, które - gdy podkreślone - są zarezerwowane do innych celów. Wynika to z faktu, że w nazwach zmiennych nie dopuszcza się podkreśleń, a translator na podkreślenia reaguje. Na przykład wolno jako nazwy zmiennej użyć wyrazu begin, pomimo że - jak widzieliśmy w paragrafie 0.3 - wyraz begin jest zastrzeżony do innych celów. Samo podkreślenie wyrazu daje tu wystarczające rozróżnienie.

1.9. Zadanie

Które z podanych niżej ciągów symboli tworzą nazwy zmiennych, a które nie:

- | | | | |
|-----------|-------------|---------------|----------------------|
| 1) ZZZZZ, | 6) temp., | 11) V7a, | 16) x ₀ , |
| 2) x/3, | 7) -c, | 12) cos, | 17) A1B2C3, |
| 3) end, | 8) Astra-2, | 13) q888q, | 18) 35, |
| 4) char, | 9) zer0, | 14) fiołek, | 19) 135, |
| 5) v, | 10) 2m, | 15) Liczba 2, | 20) b:2; |

1.10. Zmienne liczbowe

Zmienne, których wartościami są liczby, będziemy nazywać liczbowymi. W zależności od tego, czy wartościami zmiennej są liczby całkowite czy rzeczywiste, zmienne liczbowe dzielimy na zmienne całkowite (integer; czytaj: intidża) i zmienne rzeczywiste (real; czytaj: rijal). Nazwy zmiennych całkowitych i nazwy zmiennych rzeczywistych tworzymy na tych samych zasadach. O tym, czy dana nazwa jest nazwą zmiennej całkowitej

tej czy zmiennej rzeczywistej decyduje tzw. deklaracja typu, o czym będzie mowa w paragrafie 3.4. Taką deklarację musimy umieścić w programie, zanim będziemy korzystali z danej zmiennej. Brak deklaracji jest przez maszynę sygnalizowany. Będzie o tym mowa w paragrafie 6.12.

1.11. Wartości logiczne. Zmienne booleowskie

W obliczeniach często występuje sytuacja, że program rozdziela się na pewną liczbę przypadków, w zależności od tego, jakie warunki są potrzebne dla dalszego toku obliczeń. Na przykład w momencie napotkania równania kwadratowego program może rozdzielić się na 3 przypadki:

- I. Równanie ma 2 różne pierwiastki.
- II. Równanie ma tylko 1 pierwiastek.
- III. Równanie nie ma pierwiastków.

W każdym z tych przypadków dalszy tok obliczeń może być zupełnie inny. Wybór przypadku następuje poprzez obliczenie wyróżnika Δ danego równania kwadratowego. Jeżeli $\Delta > 0$, mamy do czynienia z przypadkiem I, jeżeli $\Delta = 0$, z przypadkiem II, a jeżeli $\Delta < 0$, to z przypadkiem III.

Łatwo zauważyć, że dowolne rozgałęzienie programu można sprowadzić do alternatyw, tzn. do rozgałęzień na dwa tylko przypadki. I tak w podanym wyżej przykładzie można by rozpatrywać alternatywę:

Δ mniejsza od zera albo Δ nie mniejsza od zera,

a w przypadku, gdy Δ jest nie mniejsza od zera, następną alternatywę:

Δ równa zero albo Δ nierówna zero.

W ALGOLu istnieją możliwości szerokiego stosowania alternatyw, co stanowi jedną z głównych zalet tego języka. Wybór jednego z dwu przypadków zawartych w alternatywie następuje w zależności od tego, czy warunek podany w alternatywie jest spełniony czy nie. Warunki zapisujemy w ALGOLu za pomocą wyrażeń logicznych, czyli *b o o l e o w s k i c h* (od nazwiska słynnego logika: Boole; czytaj: bul, bulowskie). Będzie im poświęcony paragraf 2.10.

Wyrażeniem booleowskim będzie na przykład

delta < 0

jak również

delta \neq 0

Spełnienie warunku jest równoznaczne z prawdziwością zdania zapisanego danym wyrażeniem booleowskim. Jeśli warunek jest spełniony, tzn. dane wyrażenie booleowskie przedstawia zdanie prawdziwe, mówimy, że to wyrażenie ma wartość true (czytaj: tru; true po angielsku znaczy prawdziwe). Gdy warunek

nie jest spełniony, tzn. dane wyrażenie booleowskie przedstawia zdanie nieprawdziwe, mówimy, że to wyrażenie ma wartość false (czytaj: fałs; false po angielsku znaczy fałszywe). Wyrażenia booleowskie mogą mieć zatem tylko dwie wartości: true albo false. Wartości te nazywamy wartościami logicznymi. Wyrazy true i false oznaczające wartości logiczne zawsze podkreślamy.

W ALGOLu używamy również zmiennych, które mogą przyjmować wartości logiczne. Zmienne takie nazywamy z m i e n n y m i l o g i c z n y m i albo z m i e n n y m i b o o l e o w s k i m i. Jeśli na przykład warunek w postaci wyrażenia booleowskiego

$$\text{delta} < 0$$

pojawia się bardzo często, będzie można użyć podstawienia

$$b := \text{delta} < 0$$

i w ten sposób cały warunek uznać za zmienną booleowską b. Wtedy interesuje nas już tylko, czy b ma wartość true czy false. Należy zauważyć, że na ogół jedno i to samo wyrażenie booleowskie przyjmować może zarówno wartość true, jak i false w zależności od aktualnych wartości zmiennych wchodzących w skład tego wyrażenia. Na przykład wyrażenie booleowskie

$$\text{delta} < 0$$

w momencie, gdy $\Delta = -2$, będzie miało wartość true, a w momencie, gdy $\Delta = 3$, wartość false. Natomiast wyrażenie booleowskie

$$xxx + yxy \geq 0,$$

gdzie x i y są zmiennymi liczbowymi, będzie miało zawsze wartość true.

Tak jak na zmienne liczbowe można podstawiać wartości liczbowe, jak na przykład

$$x := 2;$$

tak można w ALGOLu również podstawiać na zmienne booleowskie wartości logiczne, jak na przykład

$$b := \text{true};$$

Nazwy zmiennych liczbowych i zmiennych logicznych czyli booleowskich tworzymy na tych samych zasadach. O tym czy dana nazwa jest nazwą zmiennej całkowitej (integer), zmiennej rzeczywistej (real) czy zmiennej booleowskiej (Boolean; czytają: bulan), decyduje tzw. deklaracja typu, o czym będzie mowa w paragrafie 3.4.

1.12. Zmienne indeksowane. Tablice

Zmiennym - zarówno liczbowym jak i logicznym - możemy w ALGOLu dawać indeksy. Indeksy piszemy w nawiasach kwadratowych, na jednym poziomie z nazwą zmiennej. W ALGOLu używamy

nawiasów kwadratowych tylko dla indeksów. Jeśli zmienna ma więcej niż jeden indeks, to indeksy oddzielamy od siebie przecinkami. Przecinek w ALGOLu służy zawsze do oddzielania jakichś elementów, gdy mamy do czynienia z całą ich listą.

Każdy indeks może być dowolnym wyrażeniem arytmetycznym (patrz paragraf 2.5), a więc m.in. może również zawierać zmienne z indeksami. Ponieważ indeks ma zawsze wartość integer, tzn. całkowitą, a wyrażenie arytmetyczne może mieć wartość real, w ALGOLu obliczenie wartości indeksu danego przez wyrażenie arytmetyczne W o wartości real dokonuje się poprzez funkcję standardową

entier($W+0.5$)

(patrz paragraf 1.15), tzn. przez zaokrąglenie wartości wyrażenia W do najbliższej liczby całkowitej.

Możliwość używania jako indeksów nie tylko liczb ale również dowolnych wyrażeń arytmetycznych jest dla obliczeń bardzo dogodna, gdyż pozwala modyfikować indeksy w czasie liczenia.

A oto przykłady zmiennych z indeksami:

$A[1]$,	$Beta[0,7,1]$,	$Q8[1+\sin(x),x-y]$,
$A[-2]$,	$Beta[1+j,1-1,j]$,	$Q8[0,0]$,
$A[n+1]$,	$Beta[2x1,-j,k+mxn]$,	$Q8[A[3],Beta[n+1,1,-4]-1]$,

W każdej z kolumn mamy tu trzy zmienne o tej samej nazwie, ale o różnych indeksach. Każda ze zmiennych w pierwszej kolumnie ma nazwę A i jeden indeks. Każda ze zmiennych w drugiej kolumnie ma nazwę $Beta$ i trzy indeksy. Każda ze zmiennych w trzeciej kolumnie ma nazwę $Q8$ i dwa indeksy. W ostatnim przykładzie pierwszy indeks jest utworzony przez wyrażenie arytmetyczne redukujące się do jednej tylko zmiennej indeksowanej $A[3]$, a drugi jest różnicą zmiennej indeksowanej $Beta[n+1,1,-4]$ i liczby 1.

Indeksy z tradycyjnego zapisu matematycznego nie zawsze przechodzą do ALGOLu jako indeksy, bo mogą wejść do nazwy zmiennej. Na przykład pierwiastki równania kwadratowego możemy wprowadzać jako zmienne x_1 i x_2 bez indeksów, a zmienną x_n możemy zapisać jako xn , o ile później nie będzie podstawiania wartości na n . Takie podstawianie jest możliwe w indeksach, ponieważ są one wyrażeniami arytmetycznymi, ale nie jest możliwe w samej nazwie zmiennej. Gdyby zachodziła potrzeba podstawiania wartości na zmienną n , zmienną x_n zapisalibyśmy jako $x[n]$.

Staramy się zawsze używać jak najmniej zmiennych z indeksami i jak najmniej indeksów, ponieważ indeksy bardzo wyraźnie przedłużają czas pracy maszyny.

Zmiennej z indeksami można użyć tylko wtedy, gdy jest ona wyrazem czyli komponentą tzw. **t a b l i c y** (array: czytają: arej). Nazwa tablicy jest zawsze nazwą jej wyrazów, z tym, że wyrazy tablicy są indeksowane, a nazwa całej tablicy nie jest nigdy indeksowana. W podanych wyżej przykładach mamy w pierwszej kolumnie trzy wyrazy jakiejś jednowymiarowej tablicy A , w drugiej kolumnie trzy wyrazy jakiejś trójwymiarowej tablicy $Beta$, a w trzeciej kolumnie trzy wyrazy jakiejś

dwuwymiarowej tablicy Q8. Liczba indeksów zmiennej indeksowanej odpowiada liczbie wymiarów tablicy. Na przykład tablica dwuwymiarowa będzie miała wyrazy będące zmiennymi o dwu indeksach. Jeśli wyrazy takiej dwuwymiarowej tablicy T napiszemy tak, by pierwszy indeks był numerem wiersza a drugi numerem kolumny, i jeśli tablica będzie w ten sposób miała m wierszy i n kolumn, to możemy ją sobie wyobrazić jako złożoną z wyrazów:

T[1,1]	T[1,2]	...	T[1,n]
T[2,1]	T[2,2]	...	T[2,n]
.....			
T[m,1]	T[m,2]	...	T[m,n]

Jak zobaczymy później, indeksy w tablicy nie muszą rozpoczynać się od numeru 1.

Tablice wymagają zawsze specjalnej deklaracji. Będzie o tym mowa w paragrafie 3.4.

Nazwy tablic tworzymy na tych samych zasadach, co nazwy zmiennych. O tym, czy dana nazwa jest nazwą tablicy czy zmiennej, decyduje deklaracja, o czym właśnie będzie mowa we wspomnianym paragrafie 3.4.

1.13. Zadanie

Które z podanych niżej ciągów symboli tworzą zmienne indeksowane, a które nie:

- | | | |
|-------------|-----------------------|---------------------------|
| 1) Q[2], | 4) TABLICA[1-2], | 7) MAK[1+A[1+B[1+C[1]]]], |
| 2) A[A[2]], | 5) B[sin(A[sin(C)])], | 8) A[A[1],A[2]], |
| 3) P(3), | 6) PIRI[M[N[Q[4]]], | 9) A[A[1,1],A[1,2]], |

1.14. Zadanie

Napisać w ALGOLu:

- | | | |
|------------------------|----------------------------------|--------------------------------|
| 1) $a_{1,m+1}$, | 4) $v_{k_1 k_2}$, | 7) T_{wewn} , |
| 2) $\beta_{i-1,i+1}$, | 5) A_{pqrs} , | 8) $Z_{\alpha\beta}$, |
| 3) x_{11} , | 6) $\psi_{ok-\varphi_{1,k-1}}$, | 9) $u_{n_{k+1}-1,0,m_{i+2}}$, |

1.15. Funkcje

W wyrażeniach algolowskich - podobnie jak w tradycyjnych wzorach matematycznych - oprócz liczb i zmiennych używamy również funkcji, np.:

$$\sin(a+b)$$

O ile jednak w tradycyjnym języku matematycznym w niektórych przypadkach dopuszczano się pisanie argumentu funkcji bez nawiasów, jak np.:

$$\sin x,$$

o tyle w ALGOLu przestrzega się bardzo rygorystycznie zasady, że argumenty funkcji piszemy zawsze w nawiasach i to zawsze okrągłych, a więc np.:

$$\sin(x)$$

W ALGOLu dzielimy funkcje na dwie kategorie: procedury funkcyjne i funkcje standardowe. Użycie procedury funkcyjnej jest dozwolone tylko po uprzednim zadeklarowaniu tej procedury. Będzie o tym mowa w rozdziale poświęconym procedurom. Użycie funkcji standardowych nie wymaga żadnych deklaracji. Rzecz polega po prostu na tym, że programy służące do obliczania wartości funkcji standardowych są włączone na stałe do translatora, natomiast użycie innych funkcji wymaga uprzedniego wprowadzenia programu obliczania wartości tych funkcji. Do funkcji standardowych należą oczywiście funkcje najczęściej używane. Pozostałych używamy jako procedur funkcyjnych. Lista funkcji standardowych jest zależna od reprezentacji ALGOLu.

A oto 9 funkcji standardowych w GIER-ALGOLu:

abs(W)	wartość bezwzględna (moduł) wyrażenia W,
sign(W)	znak wartości wyrażenia W (+1 dla $W > 0$, 0 dla $W=0$ i -1 dla $W < 0$),
entier(W)	największą z liczb całkowitych nie większych od wartości wyrażenia W,
sqr(W)	pierwiastek kwadratowy z wartości wyrażenia W ($W \geq 0$),
sin(W)	sinus wartości wyrażenia W w radianach,
cos(W)	cosinus wartości wyrażenia W w radianach,
arctan(W)	arctangens wartości wyrażenia W

$$(-\frac{\pi}{2} < \arctan(W) < \frac{\pi}{2}),$$

ln(W)	naturalny logarytm z wartości wyrażenia W ($W > 0$),
exp(W)	funkcja e^W

W powyższych funkcjach W może być dowolnym wyrażeniem arytmetycznym (patrz paragraf 2.5). Wartości funkcji sign i entier są typu integer, wartości pozostałych 7 są typu real.

Translator GIER-ALGOLu sygnalizuje błędne użycie funkcji standardowych przez wydrukowanie na maszynie do pisania odpowiedniego sygnału. Oto możliwe sygnały:

sqr	oznacza, że funkcja sqr była wywołana dla ujemnego argumentu,
ln	oznacza, że funkcja ln była wywołana dla ujemnego argumentu (ln 0 nie powoduje w maszynie GIER sygnału alarmowego, natomiast daje wartość $-9.35_{10}49$),

`exp` oznacza, że wartość funkcji `exp` przekroczyła ograniczenie, o którym była mowa w paragrafie 1.7 dla liczb rzeczywistych (real);

U w a g a: sygnał alarmowy `ln` może powstać również w przypadku użycia potęgi o wykładniku typu real dla liczby ujemnej; podobnie sygnał `exp` może powstać, gdy wartość jakiegokolwiek potęgi o wykładniku real przekroczy zakres podany w paragrafie 1.7.

1.16. Zadanie

Napisać w Algolu:

- | | | |
|--------------------------------|-----------------------------|--|
| 1) $\sqrt{2}$, | 4) $\sin(\alpha + \beta)$, | 7) \sin/\bar{x} , |
| 2) $\sqrt{1 + \sqrt{3,325}}$, | 5) $\arctg z$, | 8) $\log \cos x $, |
| 3) $\sin \alpha$, | 6) e^{x+2} , | 9) $\sqrt{\log \sin(\arctg \sqrt{\cos \varphi}) }$, |

gdzie `log` oznacza logarytm naturalny.