

## 4. PROCEDURY

### 4.1. Wiadomości ogólne o procedurach

Jednym z najczęściej stosowanych uproszczeń w programowaniu jest wykorzystanie faktu, że pewne fragmenty programu pojawiają się wielokrotnie. Takie fragmenty staramy się zaprogramować tylko raz, tworząc z nich pewne całości, z których następnie korzystamy wielokrotnie, jako z czegoś znanego, poprzednio przygotowanego. Nazywamy je wtedy podprogramami. Pojęcie *p r o c e d u r y* (procedure; czytaj: prosj-dzer) jest rozszerzeniem pojęcia podprogramu. Rozszerzenie to polega głównie na wprowadzeniu możliwości rekurencji, zarówno samej procedury, jak i sposobu jej wywoływania, oraz udoskonaleniu wywoływania procedury poprzez tzw. parametry formalne.

Procedury dzielą się na standardowe i niestandardowe. Procedury standardowe wchodzi w skład translatora i dlatego nie wymagają żadnych deklaracji; wystarczy znać sposób ich wywoływania. Procedury niestandardowe muszą być deklarowane. Fragment programu stanowiący treść procedury znajduje się w translatorze, gdy procedura jest standardowa, natomiast w procedurach niestandardowych tworzy tzw. *c i a ł o p r o c e d u r y* (procedure body) wchodzące zawsze do deklaracji procedury. Lista procedur standardowych GIER-ALGOLu jest podana w paragrafie 5.1. Wszystkie inne procedury traktujemy w GIER-ALGOLu jako niestandardowe.

W szczególnym przypadku procedury mogą służyć do obliczania wartości funkcji dla danych wartości argumentów. Będziemy je wtedy krótko nazywali funkcjami. Procedury standardowe będące funkcjami nazywamy *f u n k c j a m i s t a n d a r d o w y m i*. W GIER-ALGOLu jest ich 9 i poznaliśmy je w paragrafie 1.15. Procedury niestandardowe będące funkcjami będziemy nazywać *p r o c e d u r a m i f u n k c y j n y m i*.

Procedury standardowe nie będące funkcjami będą omówione szczegółowo w rozdziale 5. Dla uproszczenia wysławiania się przyjmujemy w dalszym ciągu niniejszego rozdziału, że przez procedury będziemy w tym rozdziale rozumieć procedury niestandardowe.

W programach napotykamy procedury dwojako: deklaracje procedur i wywoływanie procedur.

### 4.2. Deklaracje procedur niefunkcyjnych

Deklaracja procedury niefunkcyjnej ma następującą budowę:

procedure  
nazwa procedury  
(lista parametrów formalnych);

value

lista parametrów formalnych wywoływanych przez wartość;  
specyfikacje wszystkich parametrów formalnych;  
ciało procedury;

Nazwy procedur tworzymy na tych samych zasadach co nazwy zmiennych (patrz paragraf 1.8). Po nazwie procedury następuje na ogół lista tzw. parametrów formalnych, ujęta w nawiasy okrągłe. Bywają procedury bez parametrów formalnych, wtedy nie piszemy również nawiasów okrągłych i po nazwie procedury następuje bezpośrednio średnik.

Parametry formalne służą jedynie do formalnego określenia działań w ciele procedury. Dla wykonania procedury jest konieczne podstawienie w miejsce parametrów formalnych parametrów aktualnych, tzn. parametrów właściwych w miejscu wywołania procedury (patrz paragraf 4.3). Dlatego parametrów formalnych nie potrzeba uprzednio deklarować. Parametry formalne w liście oddzielamy albo przecinkami, albo następującym ciągiem znaków

) nazwa złożona z samych liter : (

stanowiącym rodzaj komentarza. Na przykład listę parametrów formalnych

(A, n, s)

możemy napisać w postaci

(A) Rzad: (n) Wynik: (s)

Parametrami formalnymi mogą być tylko nazwy, a więc nazwy zmiennych (bez indeksów), nazwy tablic, etykiety, nazwy przełączników, nazwy procedur, jak również nazwy tzw. łańcuchów (strings), o których będzie mowa w rozdziale 5.

Parametry formalne dzielą się na dwie kategorie: parametry wołane przez wartość i parametry wołane przez nazwę. Lista tych wszystkich parametrów formalnych danej procedury, które są wołane przez wartość, musi być podana po wyrażeniu value (w liście tej parametry oddzielamy tylko przecinkami). Gdy procedura nie posiada parametrów formalnych wołanych przez wartość, opuszczamy również wyraz value (znaczy on "wartość"; czytaj: welju) i najbliższy średnik (gdyż mielibyśmy inaczej dwa średniki bezpośrednio po sobie). Wszystkie parametry formalne nie wymienione w liście po value są traktowane jako wołane przez nazwę. Różnica między parametrami wołanymi przez wartość, a parametrami wołanymi przez nazwę jest następująca: parametry formalne wołane przez wartość pozostają w ciele procedury jako parametry robocze, którym w chwili wejścia do bloku stanowiącego ciało procedury nadaje się aktualną wartość aktualnego parametru podstawionego w wywołaniu procedury na miejsce danego parametru formalnego. Efekt jest taki, jak gdyby ciało procedury zanurzono w szerszym bloku dodatkowo utworzonym, w nagłówku którego byłyby deklarowane wszystkie parametry formalne danej procedury wołane przez wartość i w którym najpierw byłyby instrukcje podstawienia na parametry formalne odpowiednich parametrów aktualnych, po czym następowałoby ciało procedury. Jeśli np. w deklaracji procedury występuje parametr formalny A wołany przez wartość, na którego miejsce w wywołaniu procedury podstawiono wyrażenie W, to w ciele procedury parametr A pozostanie nadal jako zmienna robocza, a tylko w momencie wejścia do bloku stanowiącego ciało procedury będziemy jakby mieli otwarty dodatkowy blok np.:

begin real A;

A:= W;

po czym nastąpi ciało procedury.

Natomiast parametry formalne wołane przez nazwę zostają przed wejściem do bloku stanowiącego ciało procedury zastąpione wszędzie przez parametr aktualny podany w wywołaniu procedury. W tym przypadku w czasie wykonywania procedury parametr formalny już w ogóle nie występuje. Jeśli np. w deklaracji procedury występuje parametr formalny A wołany przez nazwę, a w wywołaniu procedury na jego miejsce podstawiono wyrażenie W jako parametr aktualny, to w każdym miejscu ciała procedury, w którym w deklaracji figuruje parametr A, zostaje podstawione wyrażenie W, z ewentualnym ujęciem go w nawiasy, jeśli tego wymaga składnia. Należy wyraźnie podkreślić, że zostaje tu podstawione samo wyrażenie W, a nie jego wartość. Zostaje ono wprowadzone w miejsce nazwy stanowiącej parametr formalny. Stąd pochodzi określenie: wołanie przez nazwę.

Wywoływanie parametru formalnego przez wartość daje następujące korzyści:

- zachowuje w czasie wykonania procedury niezmienną wartość odpowiedniego parametru aktualnego,
- daje oszczędność czasu pracy maszyny, ponieważ wartość parametru aktualnego podstawiana na parametr formalny zostaje jednorazowo na wstępie przeliczona i nie ma potrzeby powtarzania tego obliczenia w następnych miejscach, w których w ciele procedury pojawia się dany parametr formalny,
- mamy cały czas w ciele procedury możliwość korzystania z danego parametru formalnego jako ze zmiennej roboczej.

Niemniej jednak używanie parametrów formalnych wołanych przez nazwę - jak to zobaczymy na przykładach - daje w wielu przypadkach znaczne rozszerzenie możliwości wykorzystywania procedur.

W GIER-ALGOLu nie można wywoływać tablic przez wartość. Jest to bodaj najistotniejsza różnica między ALGOLEM i GIER-ALGOLEM.

Po liście parametrów formalnych wołanych przez wartość następują w deklaracji procedury tzw. s p e c y f i k a c j e. Służą one do określenia rodzaju i typu parametrów formalnych, ale w przeciwieństwie do deklaracji nie rezerwują miejsca w pamięci maszyny, ponieważ w momencie wykonywania procedury zostają na miejsce parametrów formalnych podstawiane parametry aktualne, zawierające wielkości i wyrażenia normalnie przygotowane deklaracjami, a więc już z potrzebną liczbą zarezerwowanych miejsc w pamięci maszyny. Niektóre ze specyfikacji nie mają odpowiedników w deklaracjach. Na przykład specyfikuje się parametry formalne będące nazwami łańcuchów czy etykietami, podczas gdy ani etykiet, ani łańcuchów nigdy się nie deklaruje. W niektórych przypadkach specyfikacje nie różnią się formalnie od deklaracji. Różnica między nimi polega na sposobie ich interpretacji przez maszynę.

Każda ze specyfikacji składa się z tzw. s p e c y f i k a t o r a i listy parametrów formalnych podlegających danej specyfikacji. Parametry w tej liście mogą być oddzielane od siebie tylko przecinkami. Każda ze specyfikacji kończy się średnikiem.

A oto wszystkie możliwe specyfikatory:

real, integer, Boolean,  
array, real array, integer array, Boolean array,  
procedure, real procedure, integer procedure, Boolean procedure,  
label,  
switch,  
string.

Specyfikatorów procedure z poprzedzającą nazwą typu real, integer czy Boolean używa się tylko w przypadku procedur funkcyjnych, o czym będzie mowa w paragrafie 4.10.

Przykłady specyfikacji:

real Alfa,p,q; switch SKOK,Wl; label START,Koniec;

Każdy z parametrów formalnych występuje w specyfikacjach dokładnie jeden raz.

W ALGOLu muszą być specyfikowane tylko parametry formalne wołane przez wartość, natomiast w GIER-ALGOLu musimy specyfikować wszystkie parametry formalne. W przypadku procedury bezparametrowej odpada oczywiście w deklaracji również część poświęcona specyfikacji i tym samym bezpośrednio po nazwie procedury mamy wtedy średnik i ciało procedury.

Należy zauważyć, że w ciele procedury mogą poza parametrami formalnymi występować również wielkości nielokalne, wprowadzone deklaracją w bloku obejmującym deklarację danej procedury. Wielkości te mogą być równolegle wykorzystywane w parametrach aktualnych danej procedury.

Ponadto w ciele procedury mogą jeszcze występować parametry lokalne robocze, wprowadzane normalnymi deklaracjami wewnątrz ciała procedury.

Deklaracja procedury kończy się ciałem procedury, czyli jej programem. Ciało procedury jest albo instrukcją algolowską (najczęściej blokiem), albo programem napisanym w kodzie wewnętrznym maszyny. Użycie kodu wewnętrznego dla napisania programu danej procedury ma miejsce we wszystkich procedurach standardowych. Programy te - jak już powiedzieliśmy - wchodzić w skład translatora i programista może się nimi nie interesować. Skorzystanie z kodu wewnętrznego maszyny w ciele jakiegokolwiek innej procedury algolowskiej jest możliwe tylko za pośrednictwem specjalnych procedur standardowych, o czym będzie mowa w rozdziale 5. W rozdziale niniejszym nie będziemy się tym zajmowali, tzn. zakładamy, że przez ciało procedury będziemy tu rozumieli zawsze instrukcję algolowską. W tym miejscu należy jednak zapamiętać, że nawet w przypadku, gdy ciało procedury nie ma formy bloku, w ALGOLu traktuje się je jako blok, ze wszystkimi regułami dotyczącymi lokalności zmiennych, etykiet itd.

#### 4.3. Wywoływanie procedur niefunkcyjnych

Wywołanie procedury niefunkcyjnej jest szczególnym przypadkiem instrukcji podstawowej (por. paragraf 3.1) i ma następującą budowę:

nazwa procedury

(lista parametrów aktualnych)

W przypadku procedury bezparametrowej nawiasy okrągłe z listą parametrów aktualnych nie występują i wywołanie następuje przez samą nazwę procedury.

Lista parametrów aktualnych musi zawierać dokładnie tyle parametrów, ile parametrów formalnych zostało podanych w deklaracji tejże procedury. Kolejność napisania parametrów decyduje o odpowiedniości parametrów formalnych i aktualnych. Pierwszy z kolei parametr aktualny odpowiada pierwszemu z kolei parametrowi formalnemu, drugi drugiemu itd. Tak jak parametry formalne, tak i parametry aktualne mogą być w liście oddzielane przecinkami albo ciągiem znaków

) nazwa złożona z samych liter: (

nie ma natomiast żadnej odpowiedniości między sposobem oddzielania parametrów formalnych a sposobem oddzielania parametrów aktualnych. Dwa parametry formalne mogą być na liście w nagłówku deklaracji procedury oddzielone przecinkami, a odpowiadające im parametry aktualne na liście w wywołaniu procedury oddzielone wspomnianymi ciągami znaków i na odwrót. Parametrem aktualnym może być: wyrażenie (w szczególności zmienna lub etykieta), nazwa tablicy, nazwa przełącznika, nazwa procedury, łańcuch.

Wykonanie procedury następuje na skutek jej wywołania. Można to sobie wyobrazić w ten sposób, że w momencie, gdy maszyna wykonując program napotyka na wywołanie procedury, przedstawia w miejsce tego wywołania ciało procedury i je wykonuje. Natomiast sama deklaracja procedury nie wywołuje wykonywania procedury przez maszynę, a tylko je określa.

Wywołanie procedury musi odpowiadać następującym warunkom:

- rodzaj i typ parametru aktualnego musi być zgodny z rodzajem i typem odpowiadającego mu parametru formalnego,
- parametrowi formalnemu wołanemu przez nazwę, który w ciele procedury występuje po lewej stronie instrukcji podstawienia, może odpowiadać tylko zmienna jako parametr aktualny,
- parametrowi formalnemu będącemu nazwą tablicy musi odpowiadać parametr aktualny będący nazwą tablicy o tej samej liczbie wymiarów,
- jeśli parametr formalny jest wołany przez wartość, nie może mu odpowiadać parametr aktualny nie posiadający wartości, jak np. przełącznik, łańcuch, nazwa procedury niefunkcyjnej, a w GIER-ALGOLu również tablica,
- łańcuch może być użyty jako parametr aktualny tylko w procedurach, których ciało jest programem w kodzie wewnętrznym maszyny, albo w procedurach, w których ciele będzie on użyty jedynie jako parametr aktualny procedury w kodzie wewnętrznym.

Do procedur, których ciało jest programem w kodzie wewnętrznym maszyny, zaliczamy - jak to już było powiedziane - procedury standardowe.

#### 4.4. Przykłady procedur niefunkcyjnych

- 1) Procedura na wyciąganie pierwiastka kwadratowego z liczby zespolonej.

```

procedure sqrtcplx(a,b) wynik:(x,y);
comment x+yi jest tym z dwu pierwiastkow kwadratowych liczby a+bi,
        dla ktorego x jest nieujemne, drugim jest -x-yi;
value a,b; real a,b,x,y;
begin real u;
    u:= (sqrt(a2+b2)/2+a)/2;
    x:= sqrt(u);
    y:= (if b<0 then -1 else 1)xsqrt(u-a)
end;

```

Procedura powyższa opiera się na znanych z algebry wzorach:

$$x + yi = \sqrt{a+bi} = \pm \left( \sqrt{\frac{a^2+b^2+a}{2}} + \epsilon i \sqrt{\frac{a^2+b^2-a}{2}} \right)$$

gdzie  $\epsilon = -1$  dla  $b < 0$  i  $\epsilon = 1$  dla  $b \geq 0$ .

Wywołanie powyższej procedury w przypadku, gdybyśmy chcieli na przykład obliczyć

$$\alpha + \beta i = \sqrt{7-5i},$$

będzie miało postać instrukcji:

```
sqrtcplx(7,-5,alfa,beta)
```

2) Zakładając następującą deklarację jakiejś procedury MIG:

```

procedure MIG(a,b); value a; real a,b,
begin integer i;
    for i:= 1 step 1 until 5 do b:= b+a
end;

```

prześledzimy wykonanie wywołania tej procedury

```
MIG(X+2XZ,Z)
```

w momencie, gdy zmienne X i Z mają wartości odpowiednio 1 i 3.

Wykonanie będzie równoznaczne z wykonaniem bloku:

```

begin real a;
    a:= X+2XZ;

```

```

begin integer i;
for i:= 1 step 1 until 5 do Z:= Z+a
end end;

```

Otrzymamy stąd  $X=1$  i  $Z=38$ .

Gdybyśmy w deklaracji procedury opuścili value a, wykonanie przebiegłoby według programu:

```

begin integer i;
for i:= 1 step 1 until 5 do Z:= Z+(X+2×Z)
end;

```

i otrzymalibyśmy  $X=1$  i  $Z=850$ .

Gdybyśmy natomiast w deklaracji procedury zamiast value a dali value b, wykonanie przebiegłoby według programu:

```

begin real b;
b:= Z;
begin integer i;
for i:= 1 step 1 until 5 do b:= b+(X+2×Z)
end end;

```

i otrzymalibyśmy  $X=1$ ,  $Z=3$ , czyli program powyższy przebiegłoby zupełnie jałowo.

### 3) Zakładając następującą deklarację procedury

```

procedure Transpozycja (M) Rzad:(n); value n;
array M; integer n;
begin real x; integer i,k;
for i:= 1 step 1 until n do
for k:= 1+1 step 1 until n do
begin x:= M[i,k];
M[i,k]:= M[k,1];
M[k,1]:= x
end
end Transpozycji;

```



prześledzimy wykonanie następującego fragmentu programu

```

. . . . .
m:= 2;
.   Transpozycja (TAB,m2+m+2);
. . . . .

```

Wykonanie to będzie równoważne wykonaniu następującego fragmentu:

```

m:= 2;
begin integer n;
n:= m2+m+2;
begin real x; integer i,k;
  for i:= 1 step 1 until n do
    for k:= 1+1 step 1 until n do
      begin x:= TAB[i,k];
        TAB[i,k]:= TAB[k,1]; TAB[k,1]:= x
      end
    end end;

```

Gdyby w deklaracji procedury opuścić value n, wykonanie miałoby następujący przebieg:

```

m:= 2;
begin real x; integer i,k;
  for i:= 1 step 1 until m2+m+2 do
    for k:= 1+1 step 1 until m2+m+2 do
      begin x:= TAB[i,k];
        TAB[i,k]:= TAB[k,1]; TAB[k,1]:= x
      end
    end
  end;

```

4) Procedura dla rozwiązywania równania kwadratowego o współczynnikach rzeczywistych:

```

procedure RownKw(A,B,C,r1,u1,r2,u2,ZDEGENEROWANE);
comment A,B,C współczynniki rzeczywiste równania  $Ax^2+Bx+C=0$ ,

```



$r_1, r_2$  części rzeczywiste pierwiastków,  
 $u_1, u_2$  części urojone pierwiastków,  
 ZDEGENEROWANE nieokreślona w procedurze etykieta dla  
 skoku w przypadku zdegenerowanym  $A=B=0$ ;  
value A,B,C; real A,B,C,r1,u1,r2,u2; label ZDEGENEROWANE;  
begin real delta;  
if A $\neq$ 0 then go to normalnie;  
if B=0 then go to ZDEGENEROWANE;  
 $r_1 := r_2 := -C/B$ ; go to zerowe;  
 normalnie:  $\text{delta} := B\sqrt{2-4AC}$ ;  
if delta  $> 0$  then go to rzeczywiste;  
 $r_1 := r_2 := -B/2/A$ ;  
 $u_1 := \text{sqrt}(-\text{delta})/2/A$ ;  
 $u_2 := -u_1$ ;  
go to KONIEC;  
 rzeczywiste:  $r_2 := \text{sqrt}(\text{delta})$ ;  
 $r_1 := (-B-r_2)/2/A$ ;  
 $r_2 := (-B+r_2)/2/A$ ;  
 zerowe:  $u_1 := u_2 := 0$ ;  
 KONIEC: end RownKw;

Wywołanie tej procedury dla rozwiązania równania kwadratowego

$$5xz\sqrt{2} - (2xa - 3xb)xz + (4xa\sqrt{2} + 9xb\sqrt{2}) = 0$$

o pierwiastkach  $z_1$  i  $z_2$ , których części rzeczywiste i urojone są odpowiednio:  $x_1$  i  $y_1$ ,  $x_2$  i  $y_2$ , w przypadku, gdy przewidujemy skok do etykiety OMYLKA jeżeli  $A=B=0$ , wygląda następująco:

RownKw(5, -2xa+3xb, 4xa $\sqrt{2}$ +9xb $\sqrt{2}$ , x1, y1, x2, y2, OMYLKA);

5) Procedura dla rozwiązywania równania algebraicznego trzeciego stopnia o współczynnikach rzeczywistych.

```

procedure Rown3st(A,B,C,D,r1,u1,r2,u2,r3,u3,ZDEGENEROWANE);
comment A,B,C,D współczynniki rzeczywiste rownania
           $Ax^3+Bx^2+Cx+D = 0$ ,
          r1,r2,r3 czesci rzeczywiste pierwiastkow,
          u1,u2,u3 czesci urojone pierwiastkow,
          ZDEGENEROWANE nieokreslona w procedurze etykieta
          dla skoku w przypadku zdegenerowanym;
value A,B,C,D; real A,B,C,D,r1,u1,r2,u2,r3,u3;
label ZDEGENEROWANE;
begin real b,w,v,s,t; integer p;
if A=0 then go to ZDEGENEROWANE;
v:= 3xA;
b:= B/v; t:= C/v; s:= D/v;
w:= bxb-t; v:= bxt-3xs; s:= 2xbw-v;
if s=0 then begin r1:= -b; go to Nastepne end;
t:= s^(1/3);
r1:= if w<0 then 7xs/(12xw-7xtxt)-b else
      0.05822xt-b-sign(t)xsqrt(3xw+txt);
for p:= 1,2,3,4 do begin real a;
          a:= (r1+b)^(1/2);
          r1:= ((2xr1xa+v)/(a-w)-b)/3
          end;
Nastepne: u1:= 0;
s:= 0.5x(r1+b);
w:= (w-sxs)x3;
s:= -b-s;
if w>0 then go to Rzeczywiste;
r2:= r3:= s;
u2:= sqrt(-w);
u3:= -u2;
go to KONIEC;

```

Rzeczywiste:  $r3 := \text{sqrt}(w)$ ;

89

$r2 := s - r3$ ;

$r3 := s + r3$ ;

$u2 := u3 := 0$ ;

KONIEC: end Rown3st;

W powyższej procedurze wykorzystano metodę kolejnych przybliżeń Newtona dla obliczenia pierwiastka rzeczywistego. Dwa następne, zarówno rzeczywiste jak i zespolone, oblicza się już z równania kwadratowego. Jeśli bowiem dla równania

$$Ax^3 + Bx^2 + Cx + D = 0,$$

gdzie  $A \neq 0$ , wprowadzić oznaczenia

$$b = \frac{B}{3A}, \quad c = \frac{C}{3A}, \quad d = \frac{D}{3A}, \quad w = b^2 - c,$$

a przez  $\bar{x}$  oznaczyć znany już pierwiastek tego równania, to można je napisać w postaci

$$(x - \bar{x})[x^2 + (3b + \bar{x})x + (\bar{x}^2 + 3b\bar{x} + 3c)] = 0$$

a pierwiastki równania kwadratowego

$$x^2 + (3b + \bar{x})x + (\bar{x}^2 + 3b\bar{x} + 3c) = 0$$

obliczać ze wzoru

$$x = -b - \frac{\bar{x} + b}{2} \pm \sqrt{3 \left[ w - \left( \frac{\bar{x} + b}{2} \right)^2 \right]}$$

Wzór Newtona na kolejne przybliżenia dla równania  $f(x)=0$ :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

w przypadku równania

$$\frac{1}{3}x^3 + bx^2 + cx + d = 0$$

ma postać

$$x_{n+1} = x_n - \frac{\frac{1}{3}x_n^3 + bx_n^2 + cx_n + d}{x_n^2 + 2bx_n + c}$$

którą po wprowadzeniu podstawienia

$$v = bc - 3d$$

można zastąpić przez

$$x_{n+1} = \frac{1}{3} \left( \frac{2x_n(x_n + b)^2 + v}{(x_n + b)^2 - w} - b \right)$$

Tak został on użyty w podanej procedurze.

Jako pierwszych przybliżeń użyto

$$x_0 = \frac{7s}{12w - 7t^2} - b \quad \text{dla } w < 0,$$

$$x_0 = 0.05822t - b - \text{sign}(t) \sqrt{3wt^2} \quad \text{dla } w > 0,$$

gdzie

$$s = 2bw - v, \quad t = \sqrt[3]{s}$$

Jak wykazał autor w jednej ze swych poprzednich publikacji\*, błąd takiego przybliżenia jest mniejszy niż  $0.06313|t|$ , a błąd przybliżenia otrzymanego - jak w podanej procedurze - po czterokrotnym powtórzeniu obliczeń wzorem Newtona mniejszy niż  $4 \cdot 10^{-20}|t|$ , nie uwzględniając błędów zaokrągleń.

Wywołanie danej procedury dla rozwiązania równania

$$z^3 - 2z + 5 = 0$$

o pierwiastkach  $x+yi$ ,  $p+qi$  i  $u+vi$  miałoby postać instrukcji

Rown3st(1,0,-2,5,x,y,p,q,u,v,KONIEC)

Ponieważ w tym przykładzie  $A=1 \neq 0$ , na parametr ZDEGENEROWANE można podstawić jakąkolwiek etykietę. Powyżej użyliśmy etykiety KONIEC występującej w ciele procedury Rown3st.

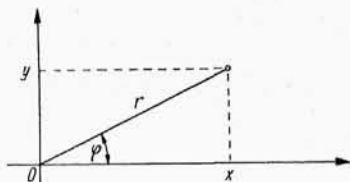
#### 4.5. Zadanie

Ułożyć procedurę na dzielenie liczb zespolonych według znanego wzoru:

$$(a+bi)/(c+di) = \frac{ac+bd}{c^2+d^2} + \frac{bc-ad}{c^2+d^2} i$$

#### 4.6. Zadanie

Opierając się na zadaniu 3.19 ułożyć procedurę na mnożenie macierzy, dając jej nazwę MnozMac.



Rys. 2

#### 4.7. Zadanie

Ułożyć procedurę na zamianę współrzędnych prostokątnych  $x, y$  na biegunowe  $r, \varphi$  (rys. 2).

\*M. Warmus: Rozwiązywanie numeryczne równań trzeciego i czwartego stopnia o współczynnikach rzeczywistych. Zastosowania Matematyki VI, 1961.

#### 4.8. Zadanie

Ułożyć procedurę na rozwiązywanie układu  $n$  algebraicznych równań liniowych, przyjmując jako parametry formalne: tablicę współczynników przy niewiadomych powiększoną o wektor wyrazów wolnych i liczbę równań. Zastosować metodę eliminacji Gaussa z wyborem największego elementu. Dla układu równań

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2$$

.....

$$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n$$

polega ona na:

- wyborze największego co do wartości bezwzględnej współczynnika przy niewiadomych,
- zamianie między sobą miejscami dwu równań i dwu niewiadomych, tak aby największy współczynnik stał się współczynnikiem  $a_{11}$ , z tym że należy zapamiętać zamianę niewiadomych, aby po zakończeniu obliczeń wrócić do wyjściowych oznaczeń,
- podzieleniu pierwszego równania przez  $a_{11}$  i sprowadzeniu go w ten sposób do postaci

$$x_1 + \alpha_{12}x_2 + \alpha_{13}x_3 + \dots + \alpha_{1n}x_n = \beta_1$$

która później umożliwi obliczenie niewiadomej  $x_1$ , gdy  $x_2, x_3, \dots, x_n$  będą już obliczone,

- pomnożeniu powyższego równania kolejno przez  $a_{21}, a_{31}, \dots, a_{n1}$  i odjęciu od kolejnych równań układu począwszy od drugiego, eliminując w nich w ten sposób niewiadomą  $x_1$ ,
- zastosowaniu do układu  $n-1$  równań z niewiadomymi  $x_2, x_3, \dots, x_n$  (tzn. do układu, jaki otrzymamy pomijając równanie pierwsze) analogicznego postępowania w celu wyrugowania następnej niewiadomej,
- po dojściu do układu równań

$$x_1 + \alpha_{12}x_2 + \alpha_{13}x_3 + \dots + \alpha_{1n}x_n = \beta_1$$

$$x_2 + \alpha_{23}x_3 + \dots + \alpha_{2n}x_n = \beta_2$$

.....

$$x_n = \beta_n$$

obliczeniu kolejno niewiadomych  $x_n, x_{n-1}, x_{n-2}, \dots, x_1$ ,

- powrocie do wyjściowych oznaczeń niewiadomych.

Gdyby się okazało w którymś z kroków, że największy współczynnik jest zerem, znaczyłoby to, że wszystkie współczynniki przy niewiadomych w rozpatrywanym układzie są zerami, czyli że mamy do czynienia z przypadkiem zdegenerowanym. Należy wte-

dy przewidzieć skok do etykiety, która w projektowanej procedurze nie musi być precyzowana.

Obliczane kolejno niewiadome umieszczać w jednej z kolumn tablicy współczynników.

#### 4.9. Zadanie

Zakładając, że procedura z poprzedniego zadania została już zadeklarowana w bloku zewnętrznym, napisać blok dla rozwiązania układu równań

$$2x - y - 3z = 4$$

$$x + y + z = -4$$

$$-x + y = 3$$

przyjmując również zmienne  $x, y, z$  jako nielocalne.

#### 4.10. Procedury funkcyjne

Główna idea, dla której procedury funkcyjne zostały wprowadzone jako oddzielna klasa procedur, polega na tym, że - skoro wykonanie procedury funkcyjnej daje w wyniku jedną liczbę jako wartość funkcji, dla której skonstruowano daną procedurę - wygodnie jest wywołanie procedury traktować nie jako instrukcję, ale jako część składową wyrażenia, podobnie jak funkcje standardowe, np.  $\sin(x)$  czy  $\ln(x)$ . Pociąga to jednak za sobą konieczność pewnych zmian w deklaracji procedury.

Skoro wywołanie procedury funkcyjnej ma stanowić część składową wyrażenia, to tym samym musi dawać pewną wartość, której typ musi być zadeklarowany. Robi się to w ten sposób, w deklaracji procedury przed wyrazem procedure umieszcza się deklarację typu, do którego należy wartość tej procedury, deklarację rozpoczyna się od real procedure, integer procedure lub Boolean procedure. W następującej po liście parametrów formalnych nie potrzeba już przewidywać osobnego parametru na wynik, ponieważ jest nim sama nazwa procedury. Natomiast w ciele procedury nazwa ta musi pojawić się co najmniej raz po lewej stronie instrukcji podstawienia, aby procedura otrzymała w wyniku jej wykonania jakąś wartość. Ponadto ciało procedury musi mieć taką konstrukcję, aby we wszystkich możliwych przypadkach wykonania procedury co najmniej jedna z takich instrukcji podstawienia z nazwą procedury po lewej stronie była wykonana. Natomiast każde pojawienie się nazwy procedury inaczej w roli zmiennej po lewej stronie instrukcji podstawienia dzie w ciele procedury powodowało wywołanie tejże procedury, o czym będzie mowa w paragrafie 4.16.

W pozostałych elementach deklaracja procedury funkcyjnej nie różni się od deklaracji procedury niefunkcyjnej.

Wywołanie procedury funkcyjnej różni się od wywołania procedury niefunkcyjnej tylko tym, że stanowi część składową wyrażenia, a nie osobną instrukcję.