Krzysztof Regulski
AGH, WIMiIP

PASCAL – PROCEDURY I FUNKCJE

1. Procedury

- Procedura w Pascalu to pewna sekwencja czynności niezbędna do wykonania zadania.
- Poznaliśmy już szereg procedur zaimplementowanych standardowo w Pascalu tj.: `write, readln, ClrScr, Delay, TextColor`
- Użytkownik może tworzyć również własne procedury. Procedura pozwala stworzyć „podprogram”, który może być potem wykonywany wielokrotnie. Wywołuje się go podobnie jak wbudowane procedury – używając po prostu jego nazwy.
- Warto tworzyć procedury nawet jeśli dany fragment kodu wykonamy choćby tylko dwukrotnie – ewentualne poprawki nanosimy tylko w jednym miejscu.
- Procedurę deklarujemy w części deklaracyjnej programu:

```
program pisz_tutaj;
USES CRT;
PROCEDURE WriteXY(x,y:byte; s:string);
Begin
  GotoXY(x,y);
  Write(s);
End;
Begin
  Clrscr;
  WriteXY(10,15,'1');
  WriteXY(1,1,'2');
  WriteXY(35,25,'3');
  WriteXY(35,5,'4');
  WriteXY(55,14,'5');
  readln;
End.
```

- Procedura naprawdę się przydaje, jeśli fragment kodu jest skomplikowany i mało przejrzysty. Wtedy wygodniej napisać go raz (lub wykorzystać istniejący) i używać. Na przykład procedura rysująca kolorową ramkę wokół słowa w miejscu o zadanych współrzędnych:

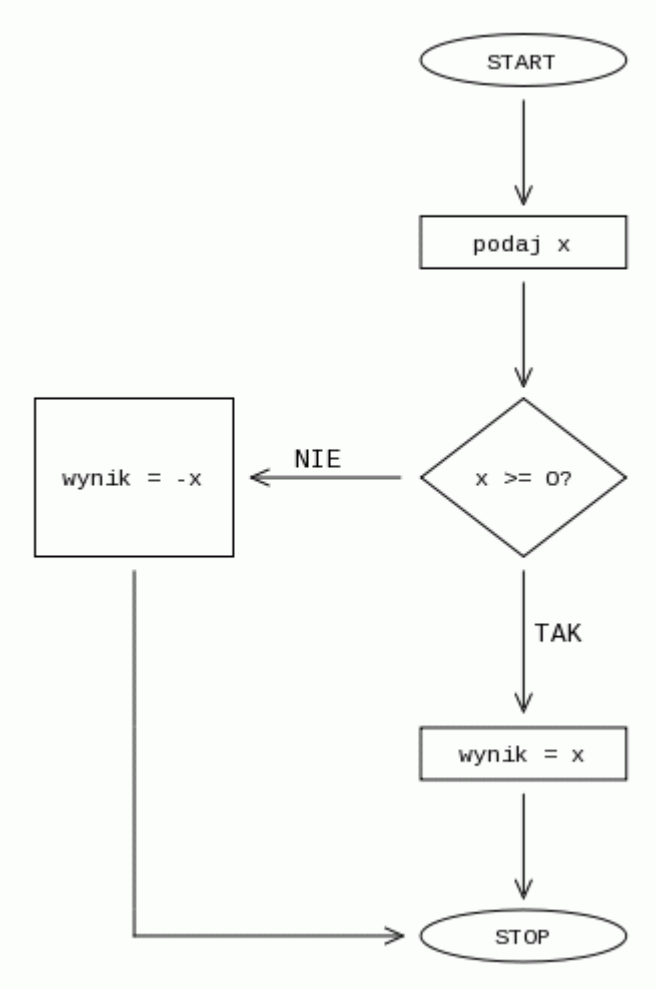
```
Program KOLOROWA_RAMKA;
USES CRT;

PROCEDURE WriteXY(x,y:byte; s:string);
Begin
  GotoXY(x,y);
  Write(s);
End;

PROCEDURE kolor_ramka_XY_slowo(x1,y1:byte; s1:string);
Var
 t:byte;
Begin
  TextColor(lightblue);
  TextBackGround(blue);
  WriteXY(x1,y1,chr(201));
  WriteXY(x1,y1+2,chr(200));
  WriteXY(x1+length(s1)+3,y1,chr(187));
  WriteXY(x1+length(s1)+3,y1+2,chr(188));
  For t:=x1+1 To x1+length(s1)+2 Do WriteXY(t,y1,chr(205));
  For t:=x1+1 To x1+length(s1)+2 Do WriteXY(t,y1+2,chr(205));
  TextColor(lightcyan);
  WriteXY(x1+1,y1+1,' '+s1+' ');
  TextColor(lightblue);
  For t:=y1+1 To y1+1 Do WriteXY(x1,t,chr(186));
  For t:=y1+1 To y1+1 Do WriteXY(x1+length(s1)+3,t,chr(186));
  TextColor(white);
  TextBackGround(black);
End;

Begin
  Clrscr;
  kolor_ramka_XY_slowo(30,5, 'KOLOROWA RAMKA nr 1');
  kolor_ramka_XY_slowo(1,1, 'KOLOROWA RAMKA nr 2');
  kolor_ramka_XY_slowo(14,20, 'KOLOROWA RAMKA nr 3');
  kolor_ramka_XY_slowo(7,12, 'KOLOROWA RAMKA nr 4');
  kolor_ramka_XY_slowo(50,12, 'KOLOROWA RAMKA nr 5');

  readln;
End.
```

- **Zadanie 1** (1pkt): Napisz program pobierający dane od użytkownika i uzupełniający rekord typu `czlowiek` (patrz poprzednie zajęcia), a następnie korzystając z przerobionej przez Ciebie odpowiednio procedury `kolor_ramka_XY_slowo` (poprzedni przykład) wypisuje dane w kolorowej ramce.

2. Funkcje

- Szczególnym rodzajem procedur są funkcje. Podobnie jak w matematyce, mają zwracać pewną wartość (niekoniecznie liczbową) w zależności od podanych im parametrów. Między definicją procedury a definicją funkcji są tylko trzy różnice. Po pierwsze zamiast słowa `procedure` używamy słowa `function`. Po drugie trzeba zaznaczyć, jakiego typu ma być zwracana wartość. Robi się to pisząc po nazwie funkcji dwukropek i typ zwracanej wartości. Po trzecie, w kodzie funkcji umieszczamy instrukcję przypisania, w której nazwie funkcji przypisujemy wartość którą ma ona zwrócić.

- Przykład: funkcją może być na przykład obliczanie wartości bezwzględnej z zadanej liczby, gdzie w wyniku podana zostanie wartość. Funkcja taka będzie miała następujący algorytm:




```
program funkcja1;
function wartosc_bezwzgl(x: real): real;
begin
  if x>=0 then wartosc_bezwzgl:=x
  else wartosc_bezwzgl:=-x;
end;

begin
  writeln(wartosc_bezwzgl (-100):3:0);
  readln;
end.
```

- **Zadanie 2**(1pkt): Już kiedyś na zajęciach pisałeś program, który podnosił podaną liczbę do wskazanej potęgi. Zapisz tą operację w funkcji, a następnie wywołaj ją w programie dla liczb 7 i 4:

$$A^B = e^{\ln(A^B)} = e^{B*\ln A}$$