REFERAT LA INFORMATICA

TEMA:FUNCȚII ȘI PROCEDURI

Autor: Malai Marius cl. 11B
Profesor: Guțu Maria

Cuprins

SUBPROGRAM

Prin subprogram vom înțelege un ansamblu alcătuit din tipuri de date, varibile și instrucțiuni scrise în vederea unei anumite prelucrări (calcule, citiri, scrieri) şi care poate fi utilizat (rulat) doar dacă este apelat de un program sau de alt subprogram.

Pentru proiectarea unei aplicaţii complexe, este necesară descompunerea problemei care trebuie rezolvată în subprobleme. Pentru fiecare dintre aceste subprobleme se vor scrie module de program mai simple (subprograme), ce vor fi apelate ori de câte ori este nevoie, fie în programul principal, fie în alte subprograme.

În limbajul Pascal subprogramele se împart în două categorii:
1. funcţii
2. proceduri

Deosebirea dintre ele rezultă din numărul de valori calculate şi returnate programului apelant:
1. funcţiile calculează şi returnează o singură valoare, asociată numelui funcţiei ;
2. procedurile calculează mai multe valori, eventual nici una.

FUNCȚII

Funcțiile sunt subprograme care calculează și returnează o singură valoare. Limbajul PASCAL conține un set de funcții predefinite, cunoscute oricărui program: sin, cos, eof etc. În completare, programatorul poate defini funcții proprii, care se apelează în același mod ca și funcțiile-standard.

La realizarea unui program ce conține un subprogram „funcție” ținem cont de:

- Numărul de caractere să fie același ;
- Tipul parametrilor să coincidă ;
- Tipul funcției trebuie să coincidă cu tipul variabilei.

Textul PASCAL al unei declarații de funcție are următoarea forma:

- F - numele funcției;
- (x1, x2, ..., xn)- lista opțională de paremetri formali reprezintînd argumentele funcției;
- tr- tipul rezultatului; acesta trebuie să fie numele unui tip simplu sau tip referință.
- Antetul este urmat de corpul funcției, format din declarațiile locale opționale D și instrucțiunea compusă begin ... end.
- Numele f al funcției apare cel puțin o datî în partea stîngă a unei instrucțiuni de atribuire care se execută: f:=e. Ultima valoare atribuită lui f va fi introdusă în programul principal.

Parametri sunt de doua tipuri:

Formali-apar în antetul subprogramului şi sunt utilizaţi de subprogram pentru descrierea abstractă a unui proces de calcul

Actuali- apar în instrucţiunea de apelare a uni subprogram şi sunt folosiţi la execuţia unui proces de calcul pentru valori concrete.

Exemple

```
program P1;
var x, y: integer ;
function F (a : integer) : integer
begin
F := a mod 10;
a:= a+1;
end;
begin
x := 11; y:=12;
writeln (‘rezultat:’, F(x)+F(x)+F(y));
end.
```

Tipul rezultatului

numele funcției

parametru formal

tipul de datw

Corpul funției

```
program P2;
function Q(s: string): string;
var i: integer; var c: char;
begin
for i := 1 to length(s) div 2 do
begin
c := s[i];
s[i] := s[length(s) - i + 1];
s[length(s) - i + 1] := c;
end;
Q := s;
end;
var p: integer;
var s: string;
begin
writeln; writeln('Dati un cuvant: ');
readln(s);
s:= Q(s);
writeln(s);
readln;
end.
```

Declaram o functie Q cu un singur argument S de tip STRING care returneaza un cuvant. function pr3(s: string): string;a

Declaram 2 variabile locale intregi: c – pentru o litera auxiliara, i – pentru a parcurge variabila S.

Parcurgem cuvantul pana la jumatate. Litera curenta s[i] o vom schimba cu locul cu litera opusa (de la sfarsitul cuvantului)

Functia va returna cuvantul inversat

```
program P3;
nivel 0
var l,a,r,pi: real;
function ariecerc (pi1, r1 : real) : real;
-
begin
ariecerc := pi1*sqr(r1) ;
end ;
function lungimecerc (pi1 ; r1 :real) :real ;
begin
lungimecerc := 2*pi1*r1 ;
end ;
begin
readln (r) ;
pi := 3,14;
l:= lungimecerc (pi,r);
a:= ariecerc (pi, r);
writeln ('l=', l);
writeln ('a=',a);
end.

program p2;
function vocale(s: string): integer;
var v, i: integer;
begin
for i := 1 to length(s) do
if upcase(s[i]) in ['A', 'E', 'I', 'O', 'U'] then v := v + 1;
vocale:=v; end;
var p: integer;
var s: string;
begin
s:='Functie';
writeln;
writeln('In cuvantul ', s, ' sunt ', vocale (s), ' vocale.');
readln;
end.
```

Numele f-ției

parametri formali

tipul de date

tipul rezultatului

corpul funcției

Numele f-ției

Parametri formali

Tipul de date

tipul rezultatului

parametri actuali

Declaram o functie vocale cu un singur argument S de tip STRING care returneaza un numar intreg.

Declaram 2 variabile locale intregi: V - pentru numarul de vocale, I - pentru a parcurge variabila S.

Functia va returna numarul de vocale gasite

Parcurgem cuvantul si verificam fiecare litera. Fiecare litera o vom transforma in litera mare (cu functia upcase) si o vom verifica daca se include in multimea de vocale ['A', 'E', 'I', 'O', 'U']. Daca se include, incrementam variabila V.

```
program P4;

type tab=array[1..10] of real;

var a:tab;

    i,n:integer;
    s:real;
function suma(x:tab; n:integer):real;
var i:integer;
    z:real;
begin
z:=0;
for i:=1 to n do
z:=z+x[i];
suma:=z;
end;
begin
write('n='); readln(n);
write('dati elementele tabloului:');
for i:=1 to n do readln(a[i]);
s:=suma(a,n);
write('s=',s:5:2);
end.


progam P5;

var m1,m2,x1,x2,y1,y2:real;
function calcul(x,y:real):real;
begin
calcul:=(x+y)/2;
end;
begin
write('x1,x2=');
readln(x1,x2);
m1:=calcul(x1,x2);
write('y1,y2=');
readln(y1,y2);
m1:=calcul(y1,y2);
readln;
end.
```

PROCEDURI

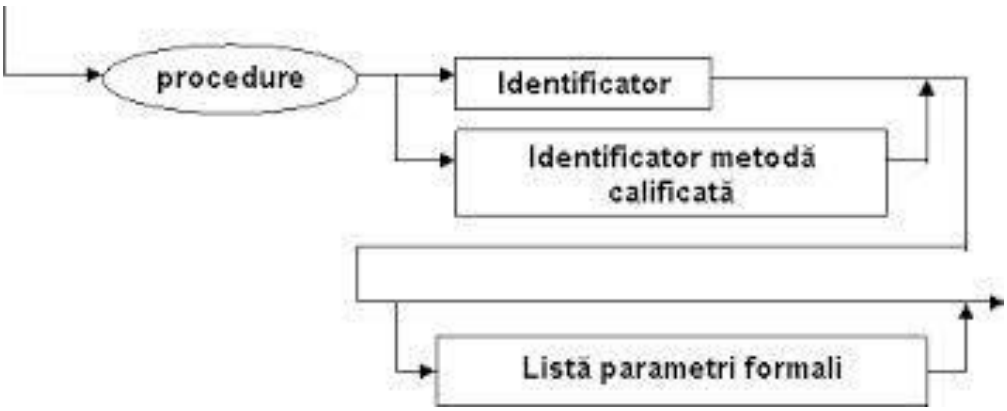
Procedurile sunt subprograme care efectuează prelucrarea datelor communicate în momentul apelului. Procedura poate întoarce mai multe rezultate, prin variabile desemnate (cu prefixul var) în lista parametrilor formali.

- var v1, v2... vn : <tip1> - parametri-variabilă
  (întoarcerea rezultatelor din procedură în programul principal)
- v1, v2… vn: <tip1>- parametric-valoare
  (transmiterea de valori din programul principal în procedură)

NB

- Parametru-valoare – drept parametru actual pot fi luate expresii, constant sau variabile.
- Parametru-variabilă – parametrii actuali sunt numai variabile.

SCHEMĂ 0.1



Exemple

```
program P1;
procedure pr1(s: string);
var v, c, i: integer;
begin
for i := 1 to length(s) do
if upcase(s[i]) in ['A', 'E', 'I', 'O', 'U'] then v := v + 1
else c := c + 1;
if v > c then write('Sunt mai multe vocale')
else if c > v then write('Sunt mai multe consoane')
else write('Numarul de consoane si vocale este egal');
end;
end.
```

Declaram o procedura PR1 cu un singur argument S de tip STRING

Declaram 3 variabile locale intregi: V - pentru numarul de vocale, C - pentru numarul de consoane, I - pentru a parcurge variabila S

Parcurgem cuvantul si verificam fiecare litera. Fiecare litera o vom transforma in litera mare (cu functia upcase) si o vom verifica daca se include in multimea de vocale ['A', 'E', 'I', 'O', 'U']. Daca se include, incrementam variabila V. Daca nu - incrementam variabila C.

Comparam variabilele C si V si afisam mesajul respectiv.

```
program P2;
type Natural=0..MaxInt;
var a,b,c: Natural;
procedure DivizComuni(a,b,c:Natural);
```

numele procedurii

```
 var i: Natural;
 begin
 i:1;
 while (i*i<=a) do
 begin
 if (a mod i=0) then
begin
if (b mod i=0) and (c mod i=0)then write (I,' ');
if (b mod (a div i)=0) and (c mod (a div i)=0)
then write (a div i,'');
end;
i:=i+1;
end
```

instrucţiune compusă

```
program P3;
var s,p:string;
procedure cuvant(var sir:string);
var i:integer;
begin
sir:=` `;
for i:=1 to length(s)  do sir:=sir+s[i];
end;
begin
writeln (‘introduceti sirul ‘);
readln (s);
cuvant(p);
writeln (‘sirul obtinut ‘);
writeln (p);
end.

program P4;
procedure inversare (var sir: string);
var i:integer;
begin
sir:=` `;
for i:=1 to length (s) do sir:=s[i]+sir;
end;
begin
writeln (‘introduceti sirul ‘);
readln(s);
inversare (p);
writeln (‘sirul obtinut: ‘);
writeln (p);
end.

program P5;
var s:string;
k:integer;
procedure calcul(var sir:string; var k:integer);
var i:integer;
begin
for i:=1 to length(sir)-1 do if sir[i+1]=sir[i] then inc(k);
end;
begin
writeln (‘introduceti sirul’);
readln(s);
calcul(s,k);  writeln(k);
end.
```

DOMENIUL DE VIZIBILITATE

Corpul unui program sau subprogram se numește bloc. Deoarece subprogramele sînt incluse în programul principal și pot conține la rândul lor alte subprograme, rezultă că blocurile pot fi imbricate(incluse unul în altul).
Această imbricare de blocuri este denumită structură de bloc a programului PASCAL.

Prin Domeniul de vizibilitate al unei declarații se înțelege textul de program, în care numele introdus desemnează obiectul specificat de declarația în studiu. Domeniul de vizibilitate începe imediat după terminarea declarației și se sfîrșește o dată cu textul blocului respectiv.Domeniul de vizibilitate al unei declarații inclus acoperă domeniul de vizibilitate al declarației ce implică același nume din blocul exterior.

VARIABILE GLOBALE ȘI LOCALE – o variabilă este globală relativ la un subprogram atunci când ea este declarată în program sau subprogramul ce îl cuprinde, fără să fie redeclarată în subprogramul în studiu. Se utilizează în cazul în care mai multe subprograme prelucrează aceleași date.

Orice variabilă este locală în subprogramul în care a fost declarată.

Bibliografie

1. Manual de informatica + caietul
   file:///C:/Users/Admin/Downloads/XI_Informatica%20(in%20limba%20romana).pdf
2. https://cuochiipe.files.wordpress.com/2016/05/functii-si-proceduri-pdf1.pdf
3. https://infopascalfacts.wordpress.com/2016/11/09/proceduri/
4. https://sites.google.com/site/stanciu4info4tic/home/clasa-x/parametrii-actuali-si-formali
5. https://proiectinformaticamd.wordpress.com/2015/04/21/proceduri-pascal/