Iterativitate si recursivitate

Cuprins:

Aspecte teoretice, avantaje si dezavantaje

Un algoritm iterativ face o repetitie constanta a unui sir de instructiuni, pe cand unul recursiv face uz de sine insusi, in mod repetat, autoapelandu-se. Pentru ca un algoritm recursiv sa deruleze este necesar sa existe: cazuri elementare, care se rezolv direct, cazuri care nu se rezolva direct, insa procesul de calcul in mod obligatoriu progreseaza spre un caz elementar.Astfel, avem recursii directe (if), si indirecte (while, repeat).

	avantaje	*dezavantaje*
iterativitate	*necesarul de memorie mic*	*volumul de munca mare*
	testatrea si depanarea simpla	*structura programului complicata*
recursivitate	*volumul de munca mic*	*necesarul de memorie mare*
	structura programului simpla	*testatrea si depanarea complicata*
Timpul de executie este acelasi		

Probleme folosind subprograme iterative

```
Program P1;
type vector=array [1..100] of integer;
 var a:vector;  s,i,n:integer;  med:real;
  function media(a1:vector; n1:integer):real; {functia ce calculeaza media}
   var s,i:integer;
  begin   s:=0;
    for i:=1 to n1 do
      s:=s+a1[i];  media:=s/n1;
   end;
begin {blocul programului principal unde se va apela functia}
 read(n);  for i:=1 to n do readln(a[i]);
                med:=media(a,n);
  writeln('Media este: ', med);
end.


  Program P2;
 type vector=array [1..100] of integer;
 var a:vector;  s,i,n:integer;
 function suma(b:vector; nm:integer):integer; {f-tia ce va calcula suma
elementelor pare de pe locurile impare din tabel}
     var s,i:integer;
   begin     s:=0;
    for i:=1 to m do
       if (i mod 2=1) and ( b[i]mod 2=0) then s:=s+b[i];
       suma:=s;
    end;
begin  {blocul programului principal unde se va apela functia iterativ}
  read(n);  for i:=1 to n do readln(a[i]);
   s:= suma(a,n); writeln('Suma este ',s);
end.
```

```
Program P3;
var a,b,c,d,s:integer;
 procedure sum(a1,b1,c1,d1:integer; var suma:integer);
 begin
   suma:=a1+b1+c1+d1;
 end; {sfarsitul procedurii  ce calculeaza suma a 4 termeni}
begin  {blocul programului unde se va folosi procedura}
 readln(a,b,c,d);  suma(a,b,c,d,s); writeln('s=',s);
end.

Program P4;
var a,b,e:integer;
 function f(x, y:integer):integer; {f-tia ce calculeaza x^y}
  var i,r:integer;
 begin      r:=1;
  for i:=1 to y do
    r:=r*x;
  f:=r; writeln('Rezultatul x^y=',f);
 end;
begin {blocul programului unde se introduc valori pentru calcule}
  write('nr ce va fi ridicat la putere:'); readln(a);
  write('puterea:'); readln(b);
  e:=f(a,b);
end.

Program P5;
var a,b,c,x1,x2:real;
  procedure p(a,b,c:real; var x1,x2:real);
    var d:real;
  begin
```

```
    d:=sqr(b)-4*a*c;
     x1:=(-b-sqrt(d))/(2*a);   x2:=(-b+sqrt(d))/(2*a);
   end;
begin  {blocul progr. unde se introduc parametrii si se afiseaza rezultatul}
  write('Introduceti parametrii:'); readln(a,b,c);
   p(a,b,c,x1,x2);
   writeln('Radacinile: ', x1, ' ', x2);
end.
```

Probleme folosind subprograme recursive

```
   Program P1;
  var s:string;
   function f(x:string; n:integer):string;
    begin   {functie pentru a schimba recursiv termenii unui sir}
     if n=0 then f:='' else f:=x[n]+f(x,n-1);
   end;
begin
  write('s='); readln(s);
  writeln(f(s,length(s)))
end.
```

```
  Program P2;
  var s,b:string; n:integer;
    procedure p(a: string; n: integer; var b: string);
    begin  {procedura pentru a schiba termenii unui sir}
       if n=0 then b:='' else begin p(a,n-1,b);  b:=a[n]+b;   end;
    end;
begin
   write('s='); readln(s);    n:=length(s);   p(s,n,b);    writeln (b);
```

```
end.
 Program P3;
 var n: integer;
  function f(m: integer):integer; {calcularea factorialului recursiv}
  begin
    if m=0 then f:=1  else f:=f(m-1)*m;
  end;
begin
  write ('n='); readln(n);   writeln(f(n));
end.

 Program P4;
 var a: integer; c,b:real;
  procedure p(x:real; y: integer; var c:real);  {proced. ce calculeaza b^a}
  begin
     if y=0 then c:=1 else  begin  p(x, y-1,c); c:=c*x  end;
  end;
begin
  writeln ('exponent:'); readln(a);
   write('puterea:'); readln(b);
    p(b,a,c); writeln(c);
end.

 Program P5;
 var  n,a: integer;
   procedure p(n: integer; var x: integer);
   begin  {procedura ce calculeaza factorialul recursiv}
      If n=0 then x:=1 else begin p(n-1,x); x:=x*n; end;
    end;
begin
```

```
   write('n='); readln(n);
    p(n,a); writeln(a);
end.
```

Concluzii

Atat algoritmele iterative, cat si cele recursive sunt utile la rezolvarea problemelor si trebuiesc utilizate in functie de caz.

Bibliografie

https://www.slideshare.net/Vytamin/iterativitate-sau-recursivitate