1. Рекурсія. Випереджаюче описання. Приклади.

Рекурсія – такий спосіб організації обчислення процесу при якому підпрограму звертається сама до себе.

**1, if n=0**

**n! =**

**n(n-1)!, if n>0**

uses crt;

procedure PrintA;

begin

write(‘Рекурсія’);

PrintA;

end;

begin

PrintA;

end.

При написанні рекурсивної підпрограми особливу увагу треба приділяти виходу з підпрограми в потрібний момент.

Uses crt;

Function FAKT(i:integer):integer;

Begin

if i<0 then write(‘Error’)

else if i=0 then FAKT:=1

else FAKt:=Fakt(i-1)\*i;

end;

begin

writeln(‘Кількість поєднань з 2 по 1 =’,FAKT(2)/(FAKT(1)\*FAKT(2-1));

end.

Можна показати, що будь-яка задача, яка розв’язується рекрсивно може бути розв’язана також ітераційно, але ці 2 підходи ідуть до розв’язання з протилежних боків.

Рекурсивно: від складного до простого.

Ітераційно: від простого до складного.

Ітерація

Function FIB(n:integer):longint;

var a1,a2:longint;

i:integer;Result:integer;

begin

A1:=0;A2:=0;

If n=0 then Result:=0;

Else Result:=1;

For i:=2 to n do

Begin

Result:=A1+A2;

A1:=A2;

A2:=Result;

End;

FIB:=Result;

End;

Рекурсія

Function Fibr(n:integer):longint;

Begin

If n<2 then Result:=n;

Else Result:=Fibr(n-1)+Fibr(n-2);

Fibr:=Result;

End;

Кожен рекурсивний виклик вимагає додаткового часу. При кожному виклику для локальних змінних і для параметрів значень виділяється нова пам’ять.

Uses crt;

Const k=1;

Function SUMA(res,i,n:byte):byte;

Begin

Res:=res+(i+k);

If i=n then SUMA:=res;

Else SUMA:=Suma(res,i+1,n);

End;

Begin

Writeln(Suma(0,1,3));

End.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Рівень рекурсії | Res | i | n | SUMA |
| Виклик |  |  |  | SUMA(0,1,3) |
| 1 | 0+1+1 | 1 | 3 | SUMA(2,2,3) |
| 2 | 2+2+1 | 2 | 3 | SUMA(5,3,3); |
| 3 | 5+3+1 | 3 | 3 | SUMA:=9 |