РЕКУРСИЯ: СЛОЖНО, НО ИНТЕРЕСНО

Доцент ф-та ВМК МГУ

Грацианова Татьяна Юрьевна

РЕКУРСИВНОЕ ОПРЕДЕЛЕНИЕ

- Факториал

При N=1 N!=1,

При N>1

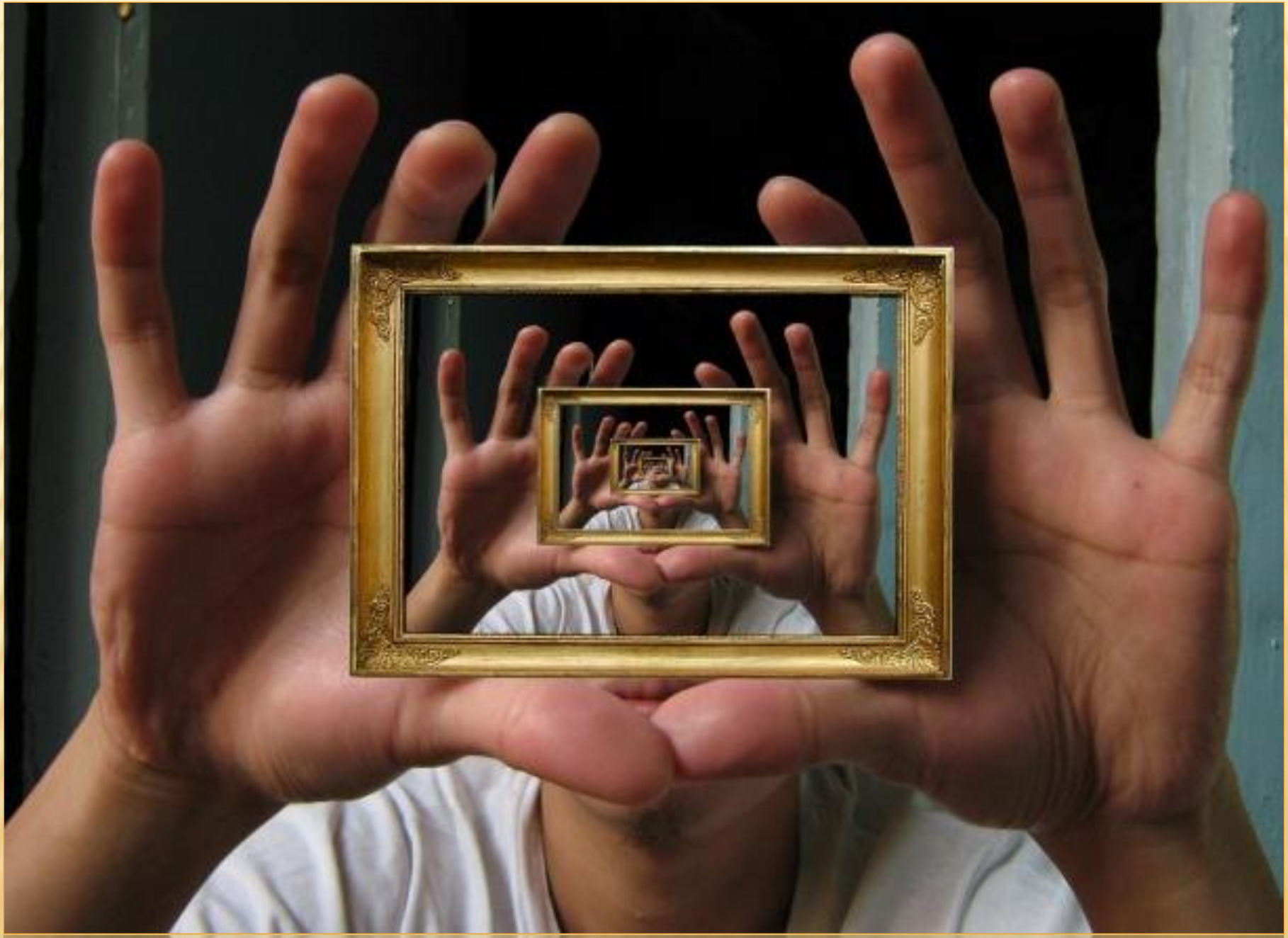
N!= (N-1)! *N

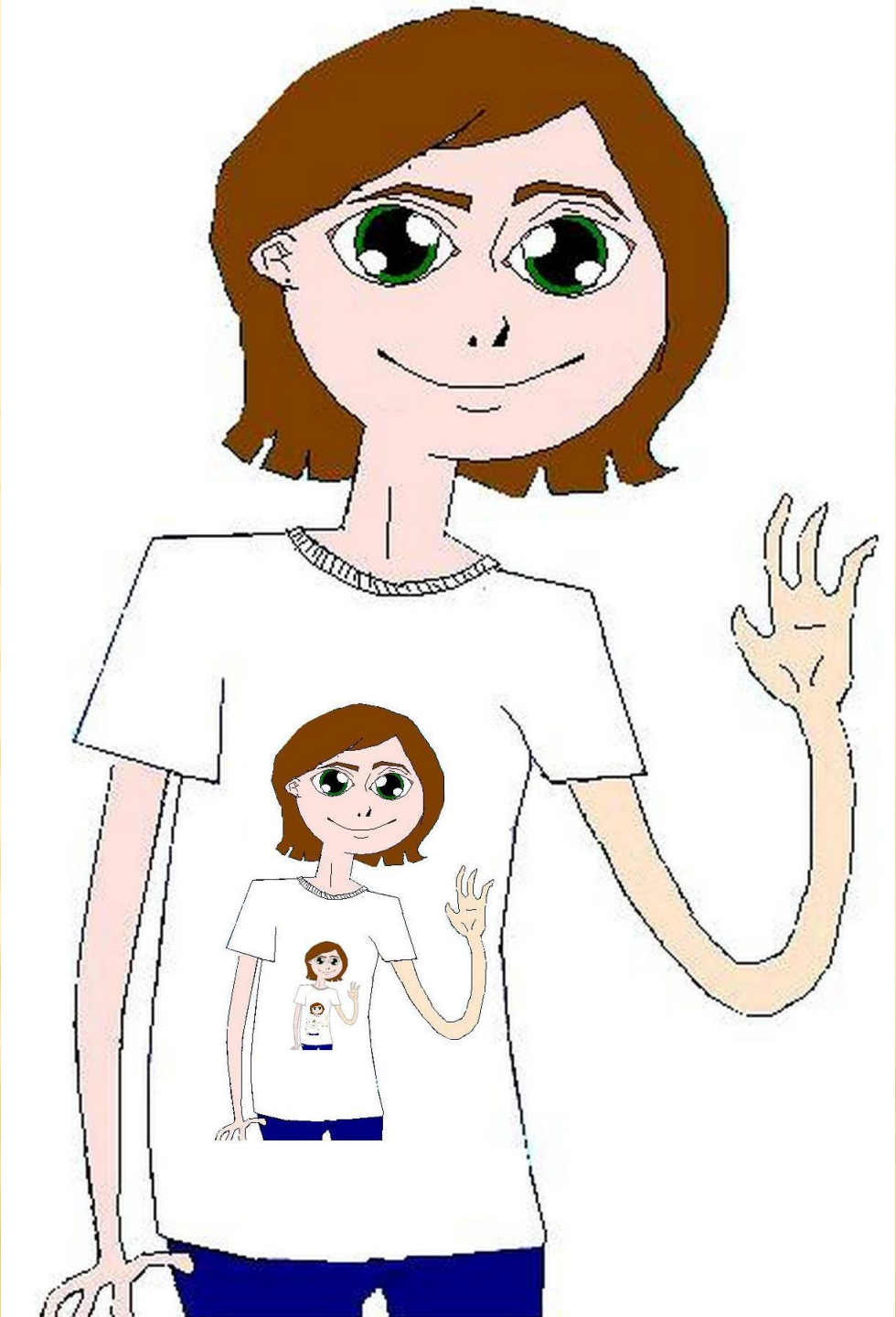
ЧИСЛО ФИБОНАЧЧИ

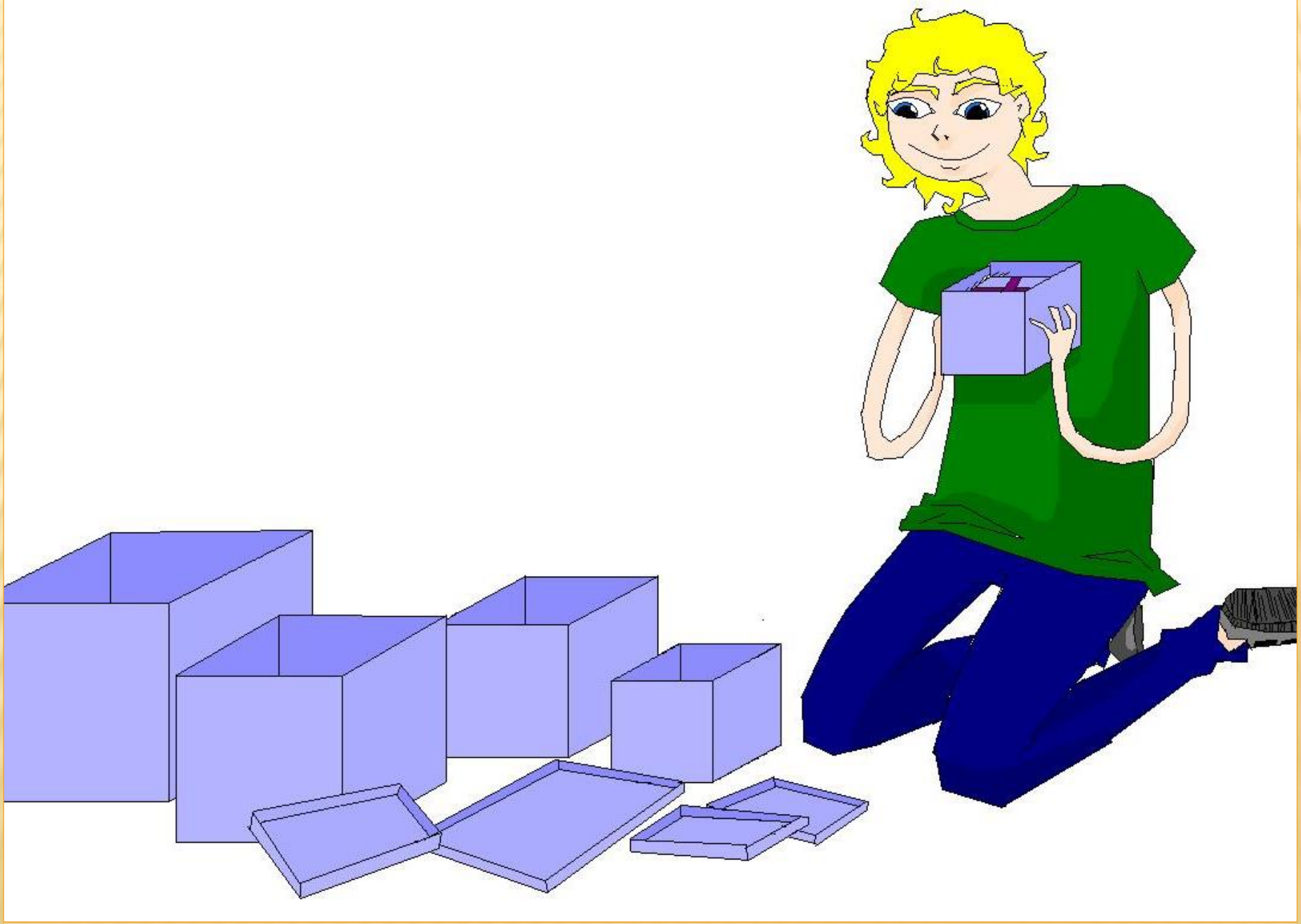
$F(1)=1,\ F(2)=1,$

$F(n)=F(n-1)+F(n-2)$

для $n>2$







РЕКУРСИВНЫЕ ОПРЕДЕЛЕНИЯ ИЗВЕСТНЫХ ФУНКЦИЙ

Показательная функция a^n:

$F(1)=a$

$F(n)=F(n-1)*a$ для $n>1$

Линейная функция $k*n+b$

$F(0)=b$

$F(n)= F(n-1)+k$ для $n>0$

ОПРЕДЕЛЕНИЯ БЕЗ РЕКУРСИИ И С РЕКУРСИЕЙ

Арифметическая прогрессия (первый член a, разность d

$a_n=a+(n-1)*d,$

$a_1=a;$

$a_n=a_{n-1}+d$ для $n>1$

Геометрическая прогрессия (первый член b,знаменатель q)

$b_n=b*q^{n-1}$

$b_1=b;$

$b_n=b_{n-1}*q$ для $n>1$

Формула сложного процента (начальный вклад R, ставка P процентов)

$S_n=R*(1+0.01*P)^n$

$S_0=R$

$S_n=S_{n-1}+S_{n-1}*P/100,\ n>0$

РЕКУРСИВНОЕ ОПРЕДЕЛЕНИЕ

База рекурсии

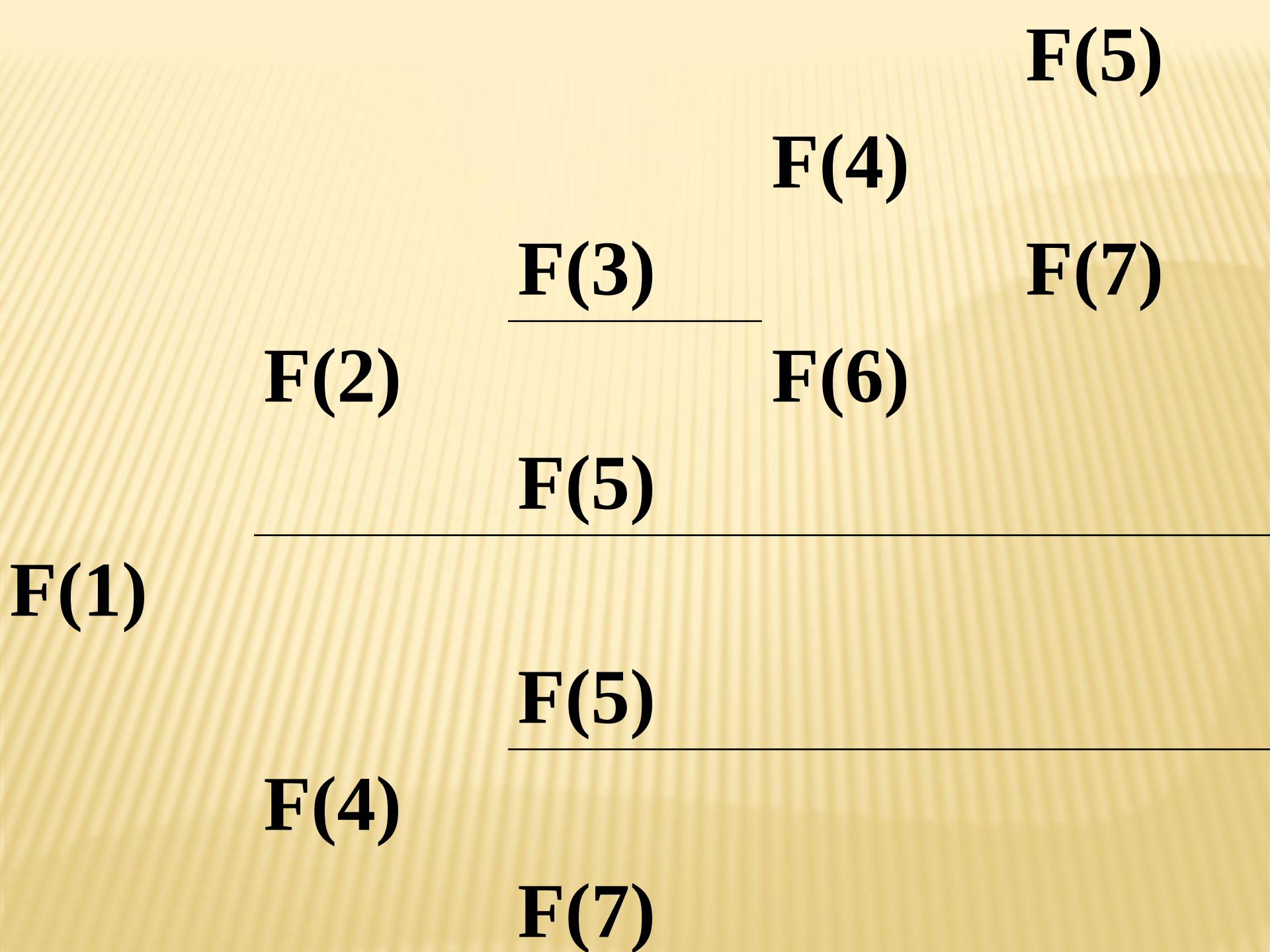
Шаг рекурсии

ДЕМО ЕГЭ

- Ниже на пяти языках программирования записан рекурсивный алгоритм F.

```
Procedure F(n: integer);
begin
   writeln(n);
   if n < 5 then
        begin
            F(n + 1);
            F(n + 3)
        end
end
```

- Чему равна сумма всех чисел, напечатанных на экране при выполнении
- вызова F(1)?


F(5)
F(4)
F(3)
F(7)
F(2)
F(6)
F(5)
F(1)
F(5)
F(4)
F(7)

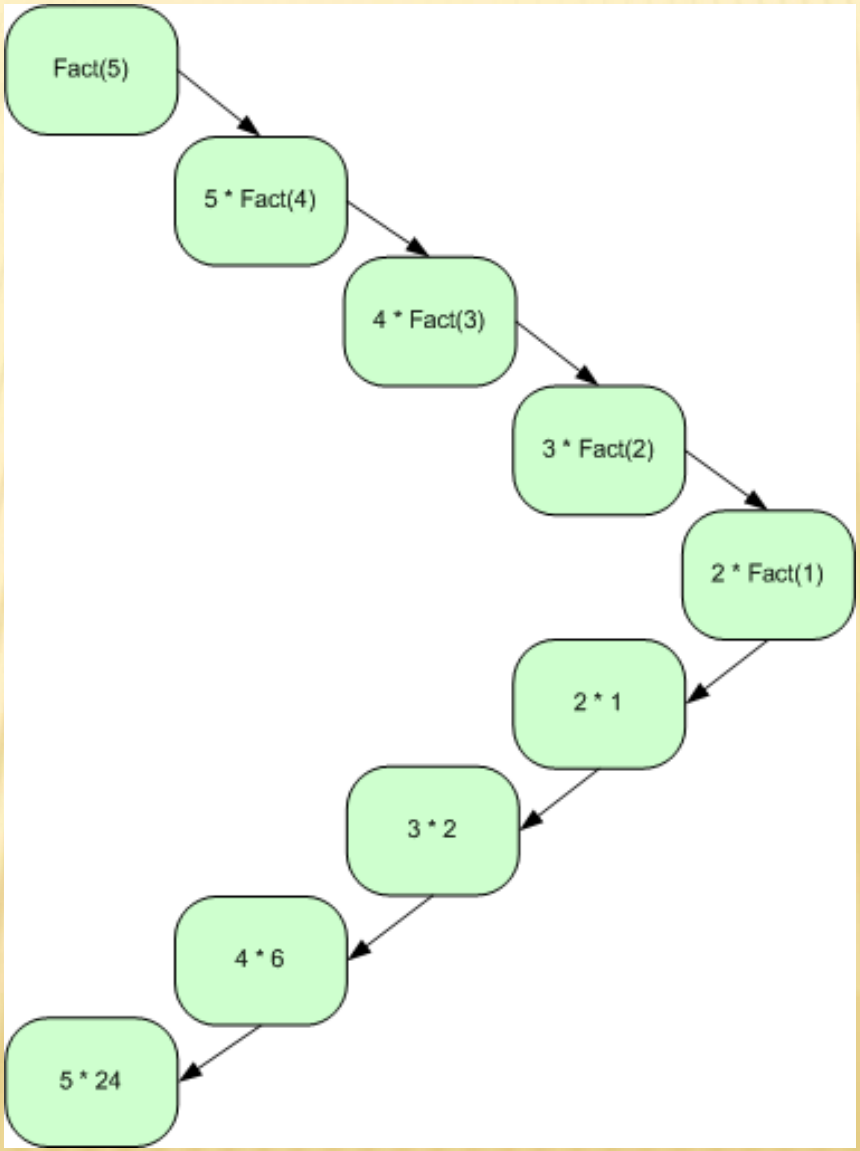
РЕАЛИЗАЦИЯ: НЕРЕКУРСИВНО И РЕКУРСИВНО

N!=1*2*…*(N-1)*N 1!=1, при N>1 N!=(N-1)! * N

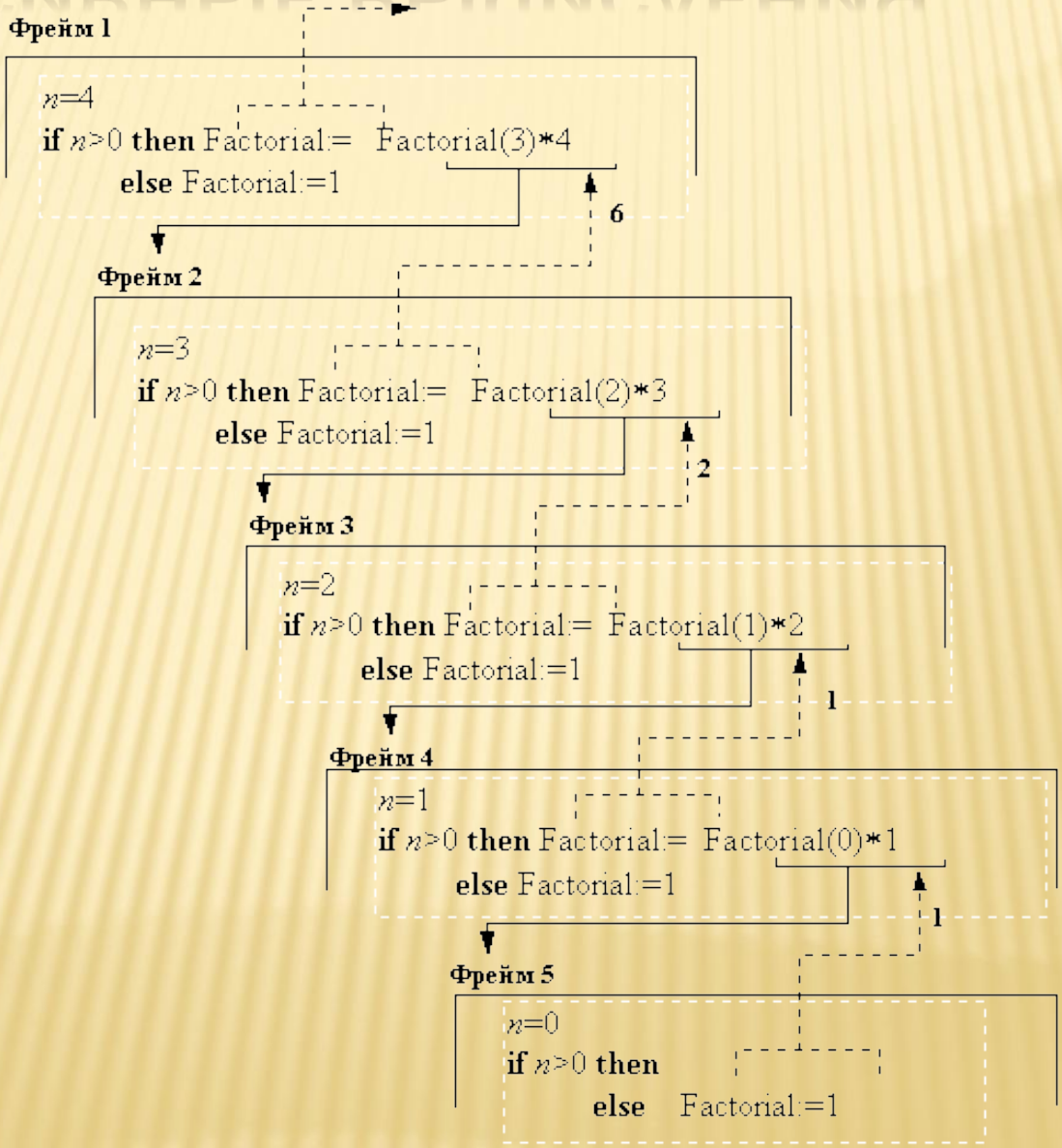
```
Function Factor(N:Integer) :
                        Integer;
Var I, F : Integer;
Begin F:=1;
    For I:=2 to N do
            F:=F*I;
    Factor:=F
End;
```

```
Function FactorR (N:Integer) :
                        Integer;

Begin
  If  N=1 Then  FactorR:=1
        Else
    FactorR:= FactorR (N-1)*N
End;
```

РЕКУРСИВНЫЕ ВЫЧИСЛЕНИЯ



РЕКУРСИВНЫЕ ВЫЧИСЛЕНИЯ



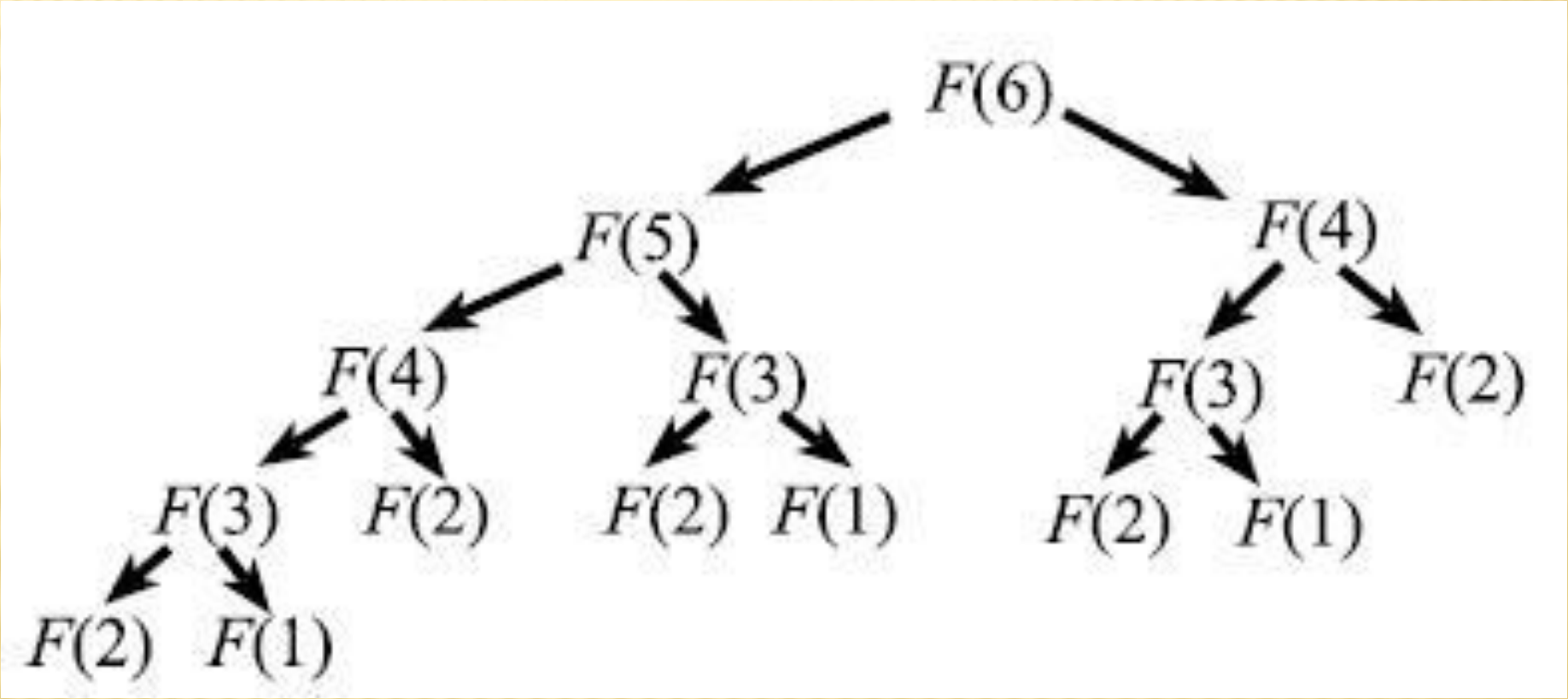
ЧИСЛО ФИБОНАЧЧИ: НЕРЕКУРСИВНО И РЕКУРСИВНО

F(1)=1, F(2)=1,
F(n)=F(n-1)+F(n-2) для n>2

```
Function
        Fib(N:Integer):Integer;
Var A,B,F, I:Integer;
Begin A:=1; B:=1;  F:=1;
    For I:=3 to N do
    Begin F:=A+B;
        A:=B;
        B:=F
    End;
    Fib:=F
End;
```

```
Function
        FibR(N:Integer):Integer;

Begin
  If N<=2 Then FibR:=1
        Else
    FibR:=FibR(N-1)+FibR(N-2)
End;
```

РЕКУРСИВНОЕ ДЕРЕВО



```
Procedure Print;
var a : integer;
Begin read(a);
    If a=0 then Writeln
           else Begin
                    Writeln(a);
                    Print
                End
End;
```

```
Program RecProc;
  Procedure Print;
    var a : integer;
  Begin read(a);
     If a=0 then Writeln
              else Begin
                     Print
                       Writeln(a);
                   End
End;
Begin {Основная программа}
   Writeln('Введите числа '); Print
End.
```

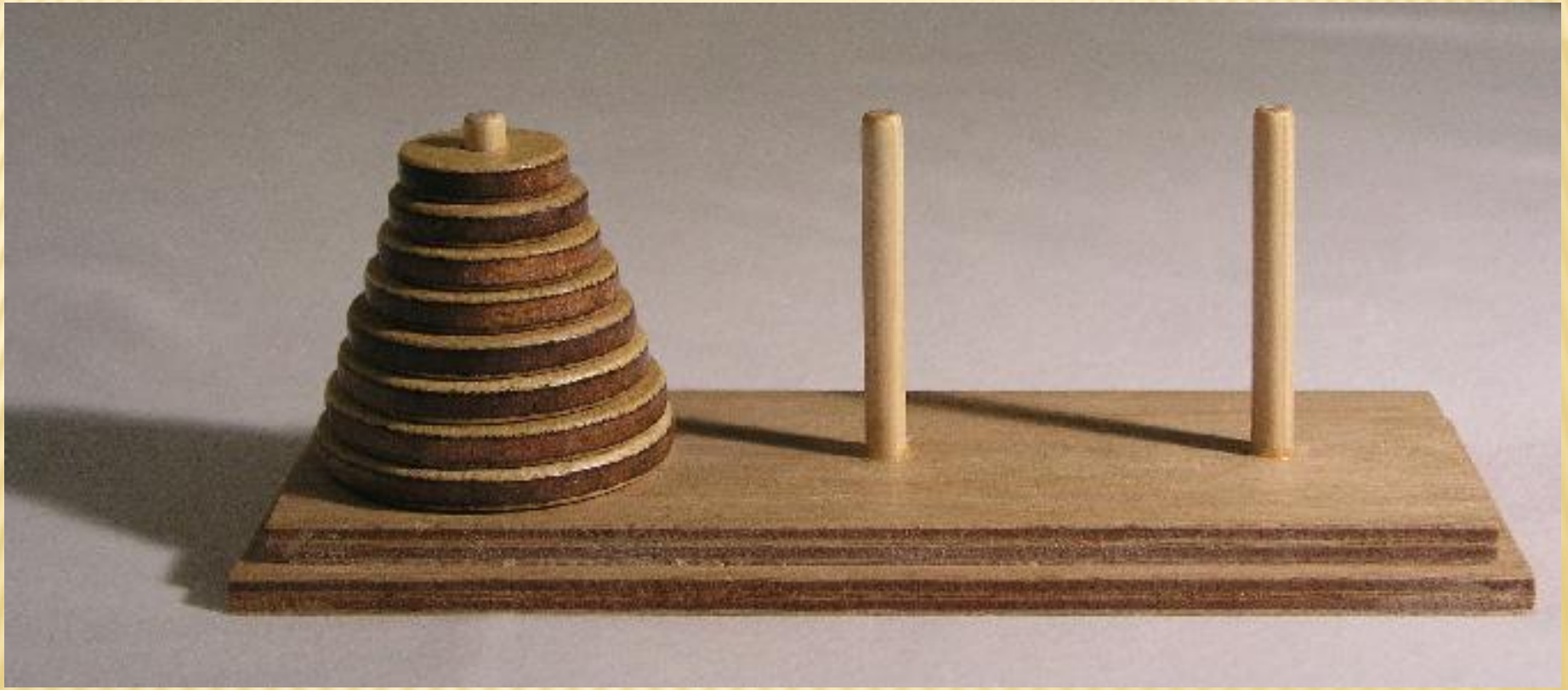
1 2 3 0

3

2

1

ХАНОЙСКАЯ БАШНЯ



ХАНОЙСКАЯ БАШНЯ: ОСНОВНАЯ ПРОГРАММА

```
Program Hanoy_B;
const Z=' Переложите колечко с '; {глобальные переменные и константы}
Var Kp:Integer;{количество перекладываний}
Procedure Hanoy(K,N1,N2:Integer); {Описание рекурсивной процедуры}

{Основная программа}
 Var K,N1,N2: Integer;
Begin
  Write('Сколько колечек в башне '); Readln(K);
  Write('Номер стержня, на котором колечки '); Readln(N1);
  Write('Номер стержня, на который надо переставить '); Readln(N2);
  Kp:=0; {Глобальная переменная для подсчета количества операций}
  Hanoy(K,N1,N2)
End.
```

ХАНОЙСКАЯ БАШНЯ: ПРОЦЕДУРА

```
Procedure Hanoy(K,N1,N2:Integer); {Описание рекурсивной процедуры}
Var Nd:Integer; {Локальная переменная, номер дополнительного стержня}
Begin  Nd:=6-N1-N2;
   If K=1 Then Writeln(Z,N1, ' на ', N2) {база рекурсии}
   Else {база рекурсии}
    If K=2 Then Begin kp:=Kp+1; Writeln(Kp,Z, N1,' на ',Nd);
                kp:=Kp+1; Writeln(Kp,Z, N1,' на ',N2);
                kp:=Kp+1; Writeln(Kp,Z, Nd,' на ',N2)
              End
          Else Begin {рекурсивный шаг}
                Hanoy(K-1,n1,nd);
                kp:=Kp+1; Writeln(Kp, Z,N1,' на ',N2);
                {kp – номер операции}
                Hanoy(K-1,Nd,N2);
              End
End;
```

Ханойская башня: количество перекладываний

N колечек - 2^N-1 операций

N=2 $3=2^2-1$

N+1

$$2^N-1 +1 +2^N-1 =2^{N+1}-1$$

ОШИБКИ В РЕКУРСИВНЫХ ФУНКЦИЯХ

```
Function FactorR (N:Integer) :
Integer;
Begin   If  N=1 Then  FactorR:=1
        Else          Write(N:8);
   FactorR:= FactorR (N-1)*N
End;
Begin Writeln(FactorR(5)) End.
```

Stack overflow error

«РЕКУРСИВНЫЕ» ОШИБКИ В НЕРЕКУРСИВНЫХ ФУНКЦИЯХ

```
Function Factor(N:Integer) : Integer;
Var I : Integer;
Begin Factor:=1;
    For I:=2 to N do
        Factor:=Factor(N)*I;
End.
```

Stack overflow error

СУММА ЦИФР ЧИСЛА

```
Function Sumc ( N:LongInt):Integer;
Begin
 If N<10 Then Sumc:=N {база}
         Else {рекурсивный шаг}
      Sumc:=N mod 10 + Sumc(N div 10)
 End;
```

АЛГОРИТМ ЕВКЛИДА

```
Function NOD (a, b: integer): integer;
begin
 if a = b then NOD := a
        else if a > b then
                              NOD := NOD(a - b, b)
                        else
                              NOD := NOD(a, b - a)
end;
```

КОРЕНЬ ФУНКЦИИ МЕТОДОМ ДЕЛЕНИЯ ПОПОЛАМ

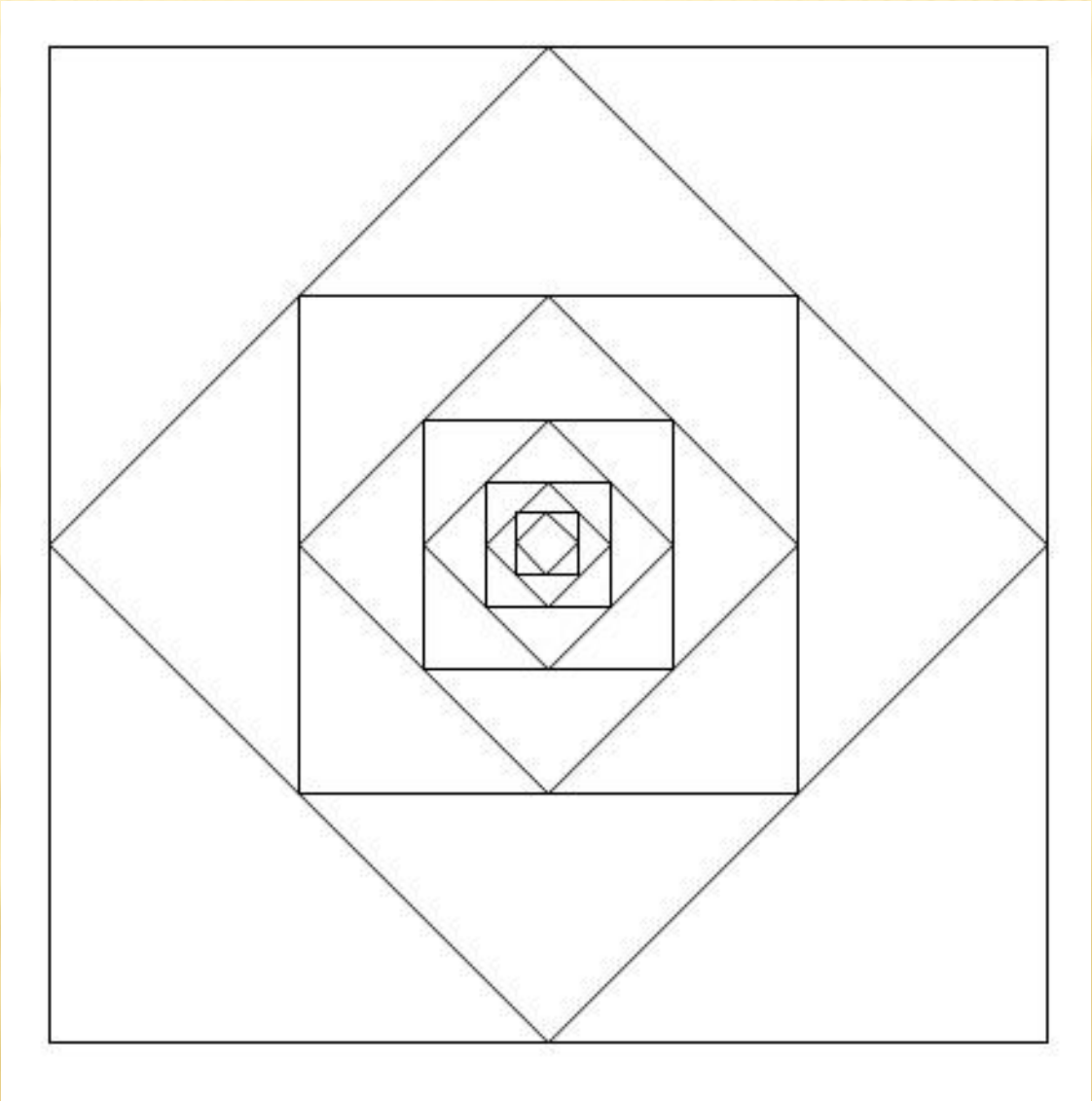
```
Function kor(A, B:real):real;
Var C:real;
Begin
        If B-A<Eps Then Kor:=(B+A)/2
                Else Begin
                                C:=(B+A)/2;
                        If f(c)<0 Then
                                                Kor:=Kor(c,b)
                                        Else
                                                Kor:=Kor(a,c)
                        End
End;
```

ABC - ГРАФИКА

uses GraphABC

Line(x1, y1, x2, y2);
WindowWidth WindowHeight
SetPenWidth SetPenColor
circle(x0,y0,L)
arc(x0,y0,L, 0, 360)

РЕКУРСИВНЫЕ УЗОРЫ



НАХОЖДЕНИЕ СЕРЕДИНЫ ОТРЕЗКА

```
Function Mid(A,B:Integer):Integer;
Begin
  Mid:=(A+B)div 2;
End;
```

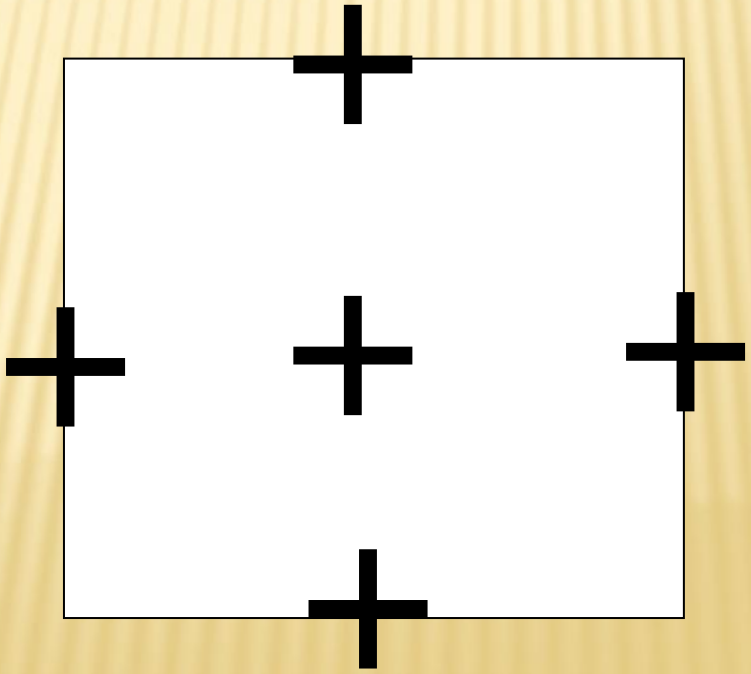
```
Procedure Uzor
 (x1, y1, x2, y2, x3, y3, x4, y4:Integer);

Begin
 Line(x1, y1, x2, y2);
 Line(x2, y2, x3, y3);
 Line(x3, y3, x4, y4);
 Line(x4, y4, x1, y1);
```

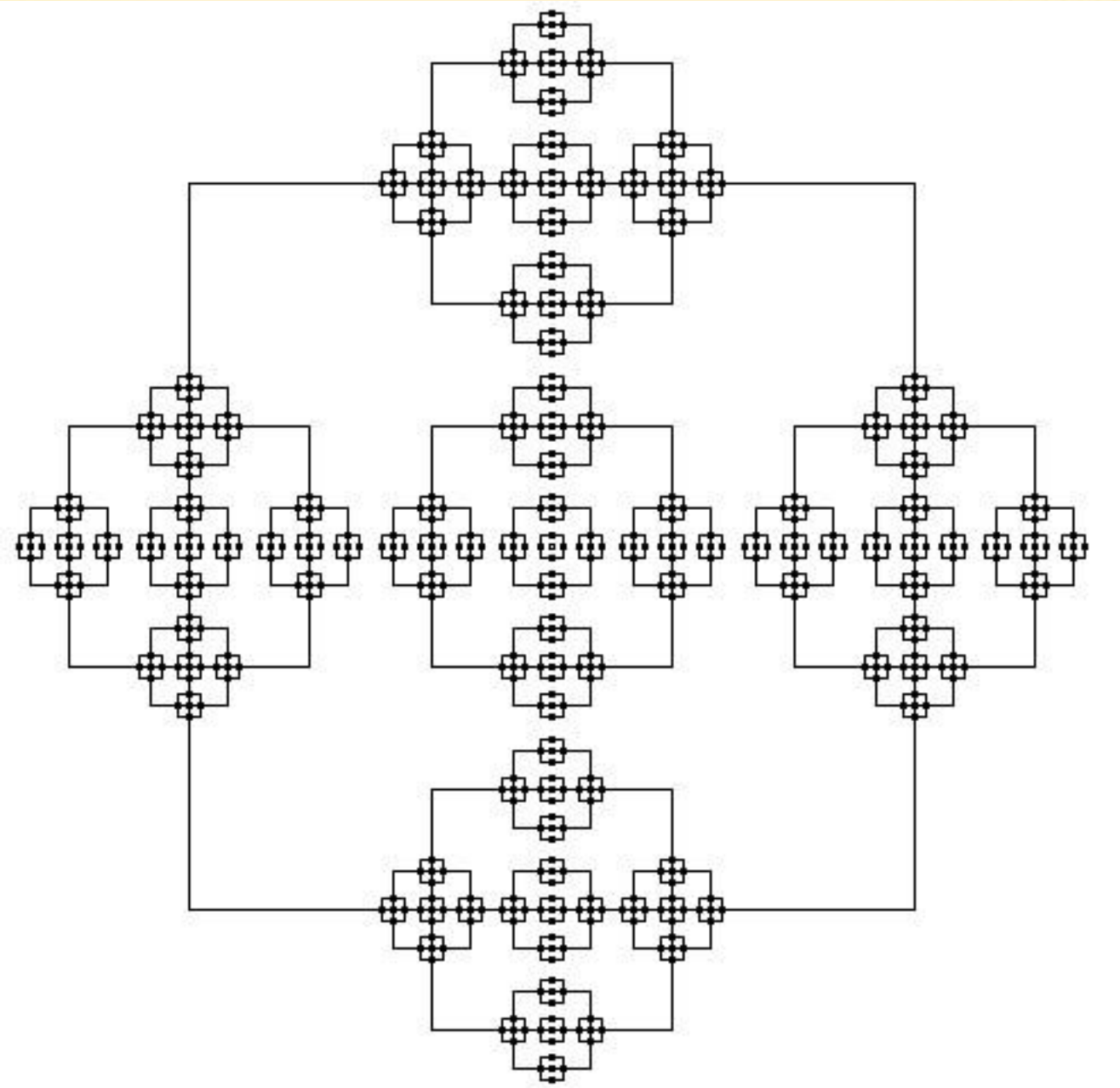
```
if (abs(y1-y2)>15) or
   (abs(x1-x2)>15)
 then Uzor(mid(x1,x2),mid(y1,y2),
                  mid(x2,x3), mid(y2,y3),
                  mid(x3,x4), mid(y3,y4),
                 mid(x4,x1), mid(y4, y1))
end;
```

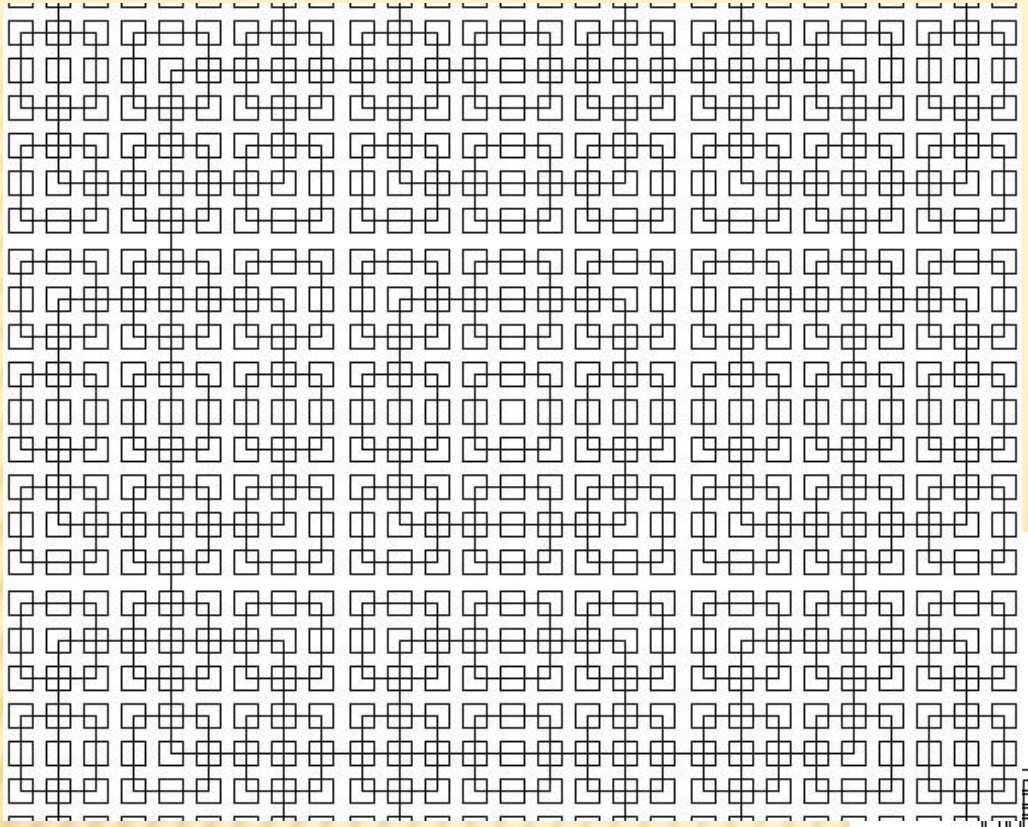
КВАДРАТИК С ЗАДАННЫМ ЦЕНТРОМ

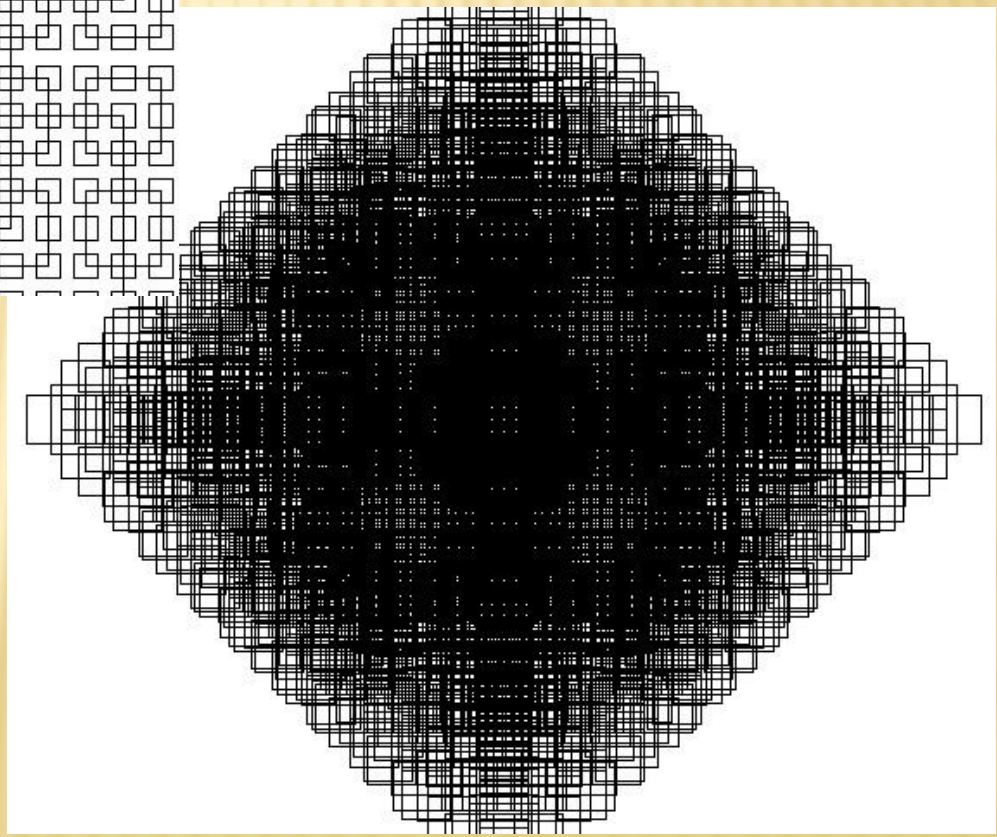
```
x1:=x0-L; y1:=y0+L;
x2:=x1;  y2:=y0-L;
x3:=x0+L;  y3:=y2;
x4:=x3;  y4:=y1;
Line(x1, y1, x2, y2);
Line(x2, y2, x3, y3);
Line(x3, y3, x4, y4);
Line(x4, y4, x1, y1);
L:=L div 3;
```

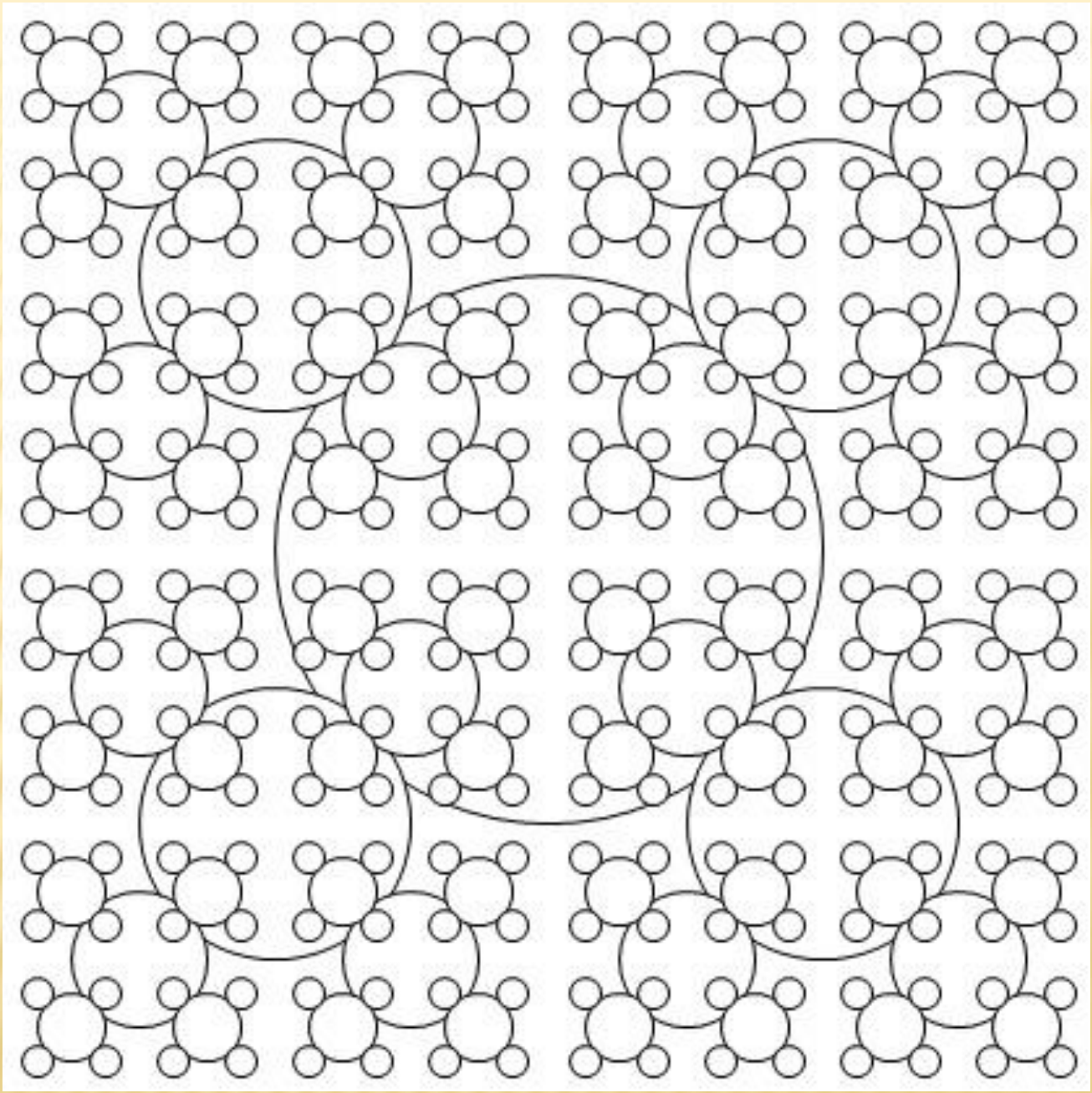
```
If L> 0 Then Begin   Uzor(x0,y0,L);
                     Uzor(mid(x1,x2), mid(y1,y2),L);
                     Uzor(mid(x2,x3), mid(y2,y3),L);
                     Uzor(mid(x3,x4), mid(y3,y4),L);
                     Uzor(mid(x4,x2), mid(y4,y1),L)
        End
```



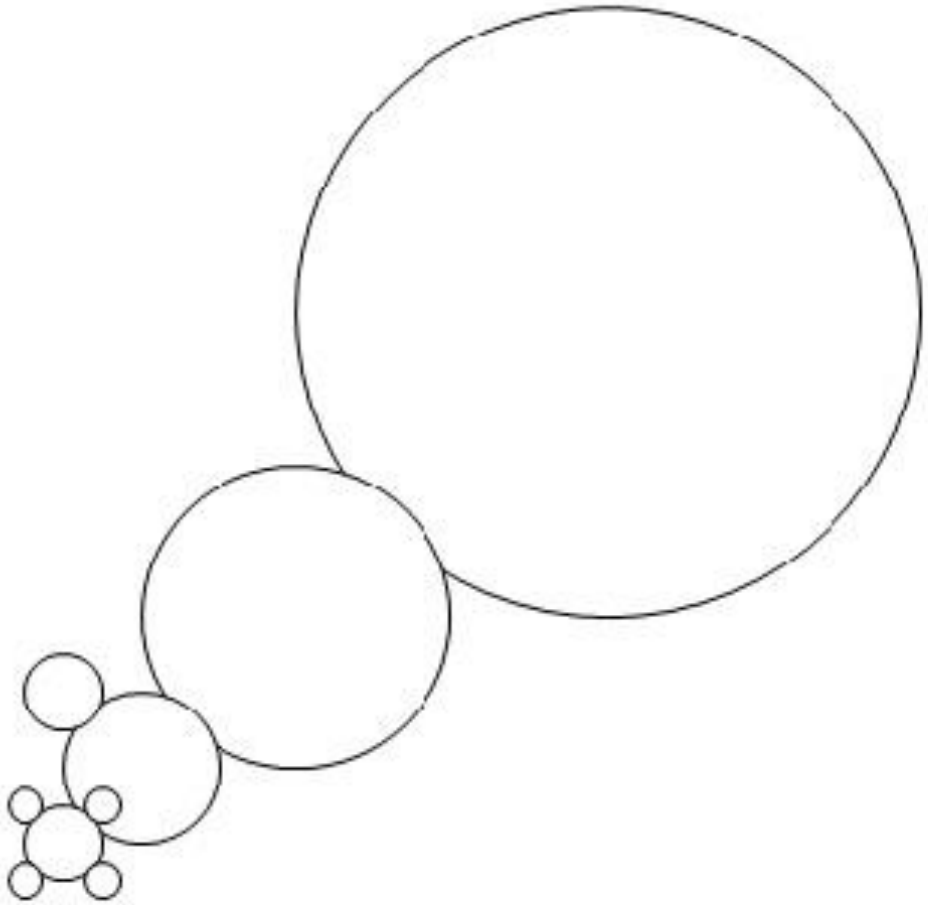


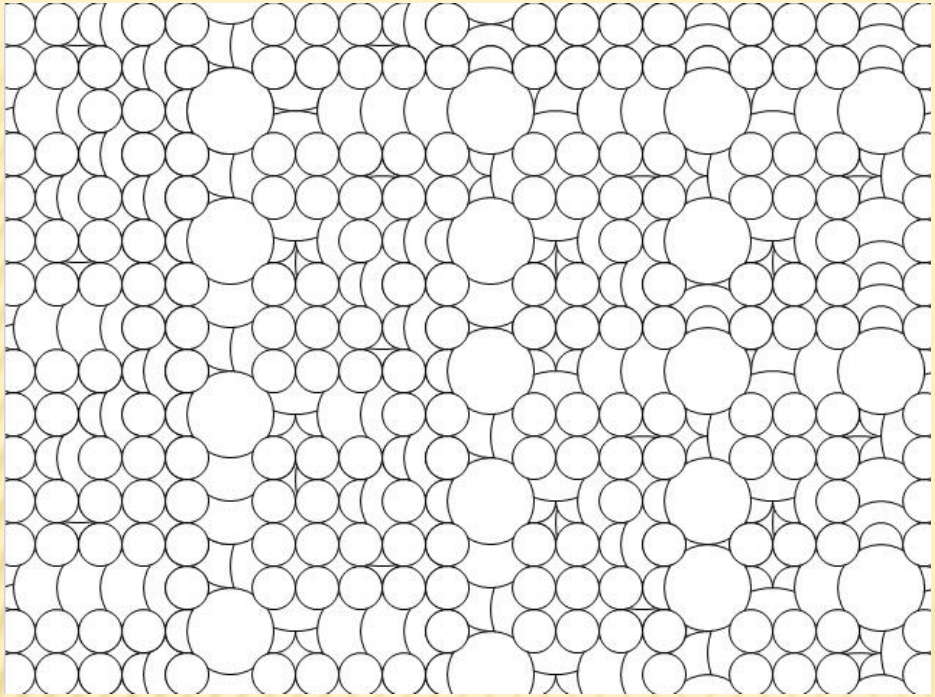


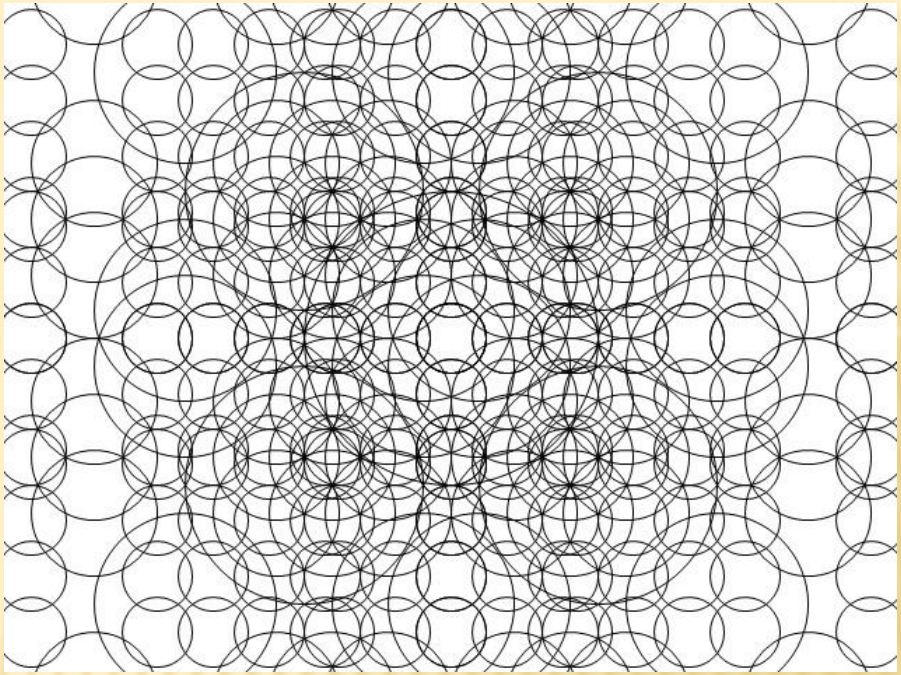


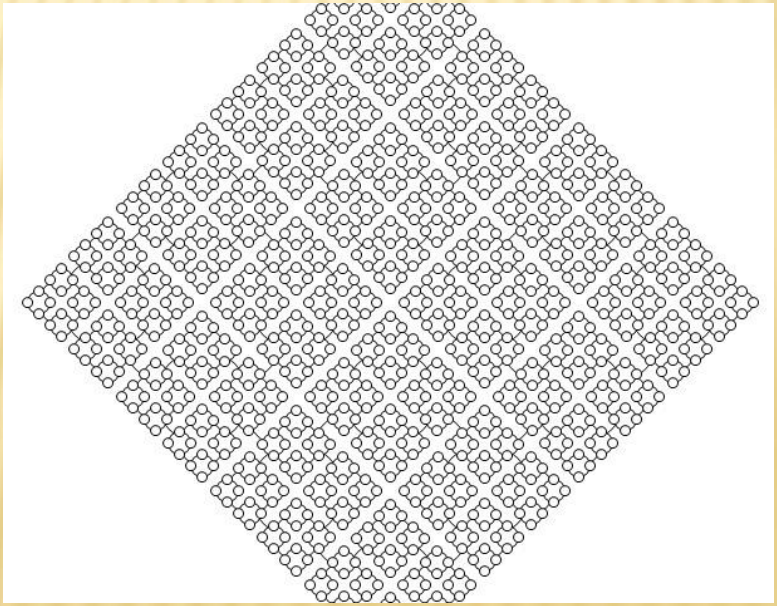


```
Procedure UzorKrug(x0,y0,l:Integer);
var x1,x2,x3,x4,y1,y2,y3,y4:Integer;
Begin
 circle(x0,y0,L);
 X1:=X0-L;Y1:=Y0+L;
 X2:=x1; Y2:=Y0-L;
 x3:=X0+L; Y3:=Y2;
 X4:=X3; Y4:=Y1;
 if L>10  then Begin UzorKrug (x1,y1,L div 2);
                     UzorKrug (x2,y2,L div 2);
                     UzorKrug (x3,y3,L div 2 );
                     UzorKrug (x4,y4,L div 2);

              End
End;
```

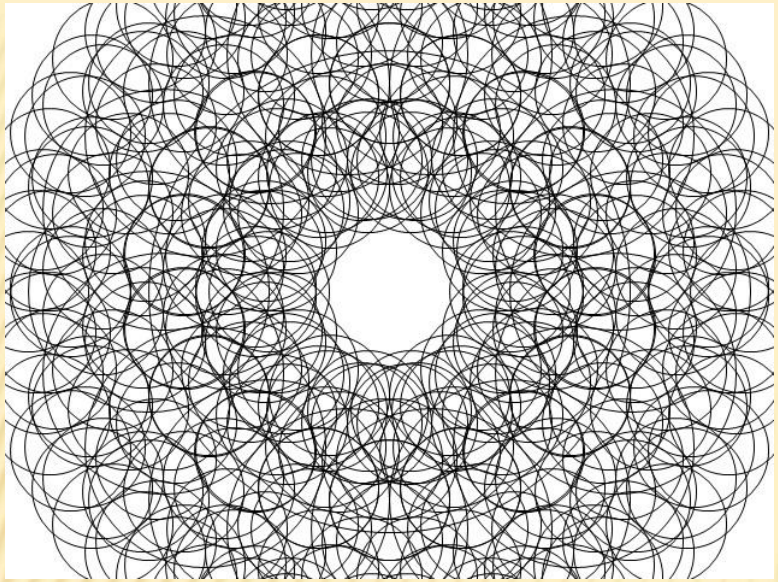


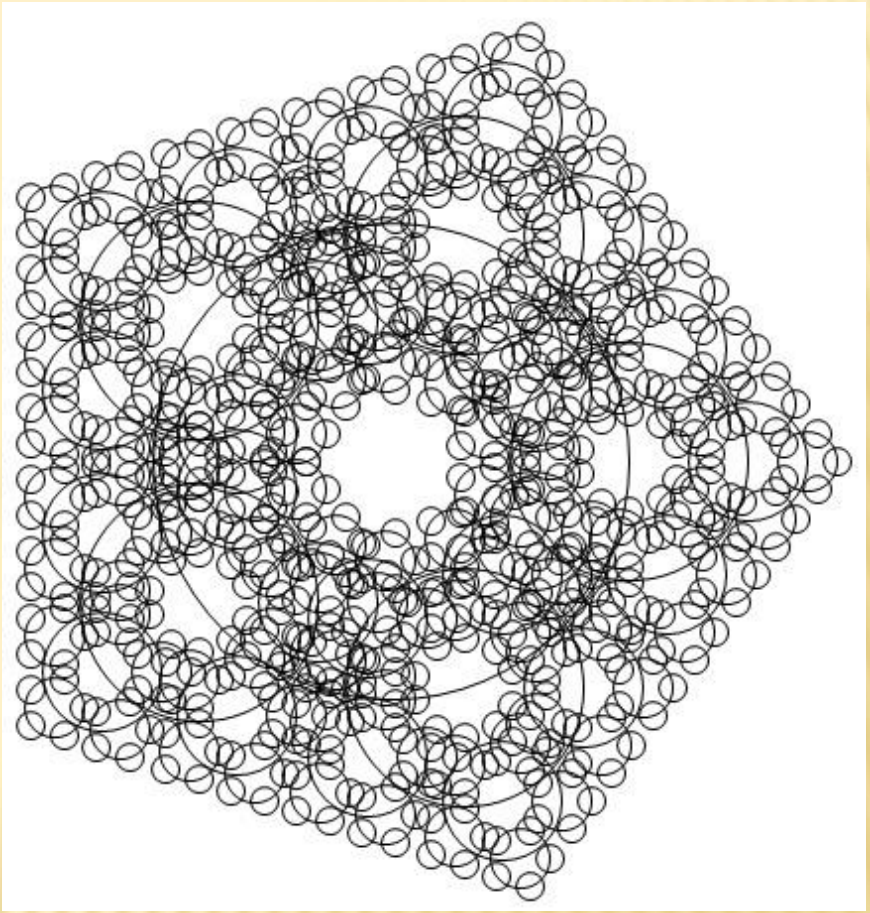


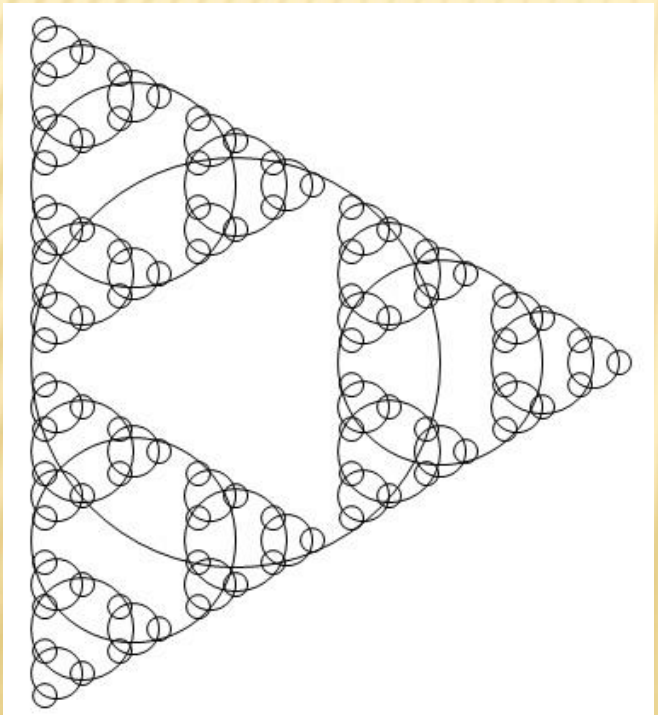


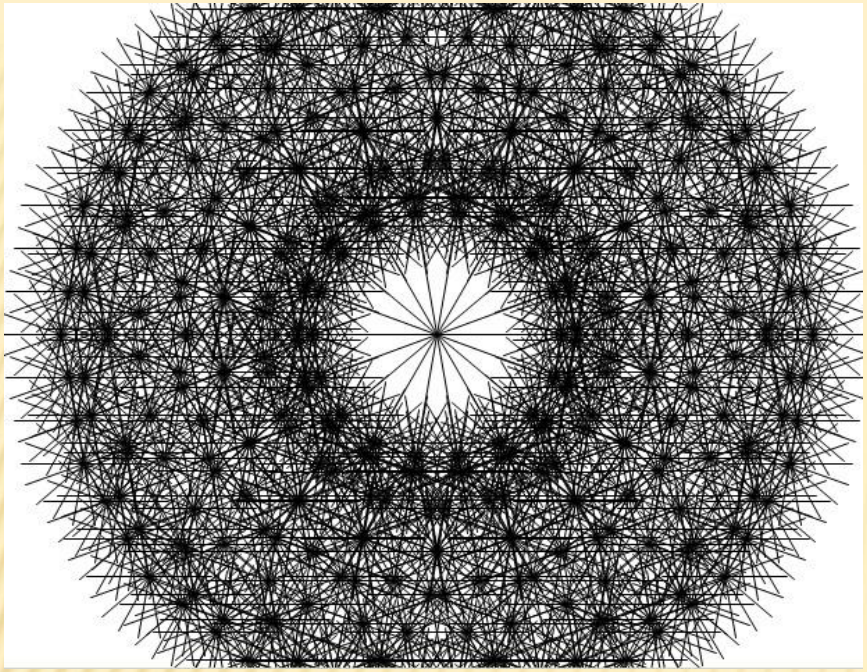


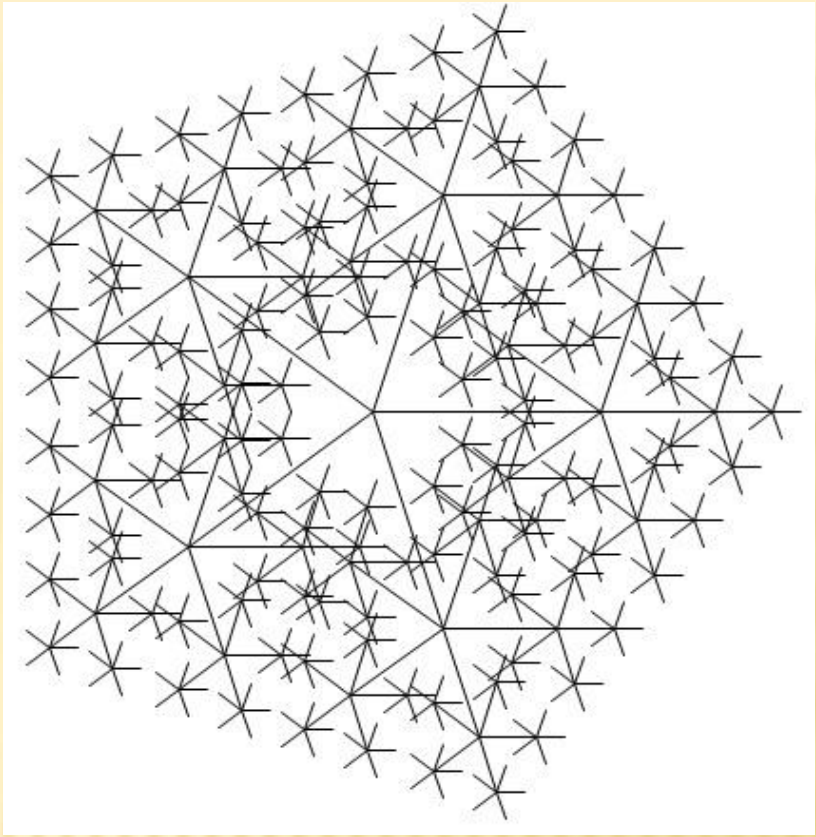
```
Procedure MnUg(x0,y0,l, n:Integer);
var x, y, i: Integer;
    F:Real;
Begin
 arc(x0,y0,L, 0, 360);
 if L>60 then Begin     F:=0;
      For I:=1 to N do
      Begin X:=round(X0+L* cos(F));
            Y:=round(Y0+L*sin (F));
            MnUg(x,y,L div 2, n );
            F:=F+2*Pi/n
      End
          End
End
```

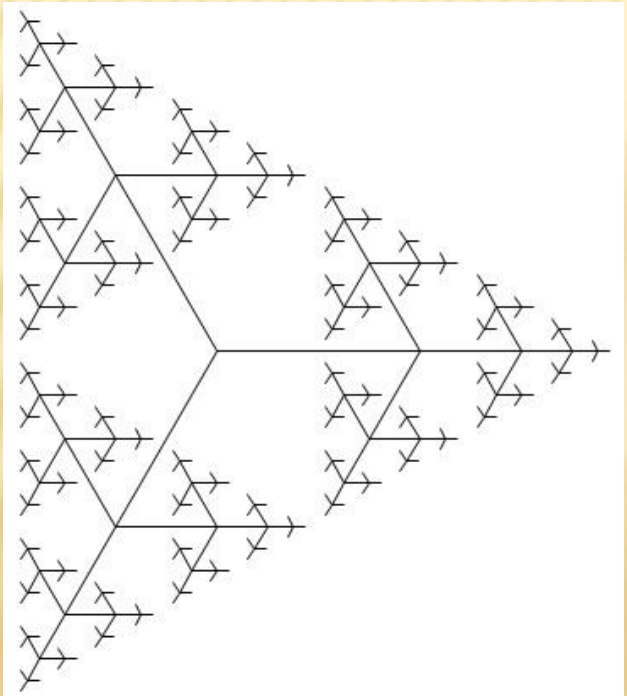












ДЕРЕВО

```
procedure Vetvi
            (x0, y0, a, L: real; N: integer);
const k = 0.6;
var x1, y1: real;
begin
if N > 0 then begin
            x1 := x0 + L*cos(a);
            y1 := y0 - L*sin(a);
            SetPenWidth(N);
            Line (round(x0), round(y0),
                  round(x1), round(y1));
```

```
SetPenColor (clblue);
Vetvi (x1, y1, a+pi/3, L*k, N-1);

SetPenColor (cllime);
Vetvi (x1, y1, a+3*pi/30, L*k, N-1);
SetPenColor (clred);
Vetvi (x1, y1, a-3*pi/30, L*k, N-1);
SetPenColor (clgreen);
Vetvi (x1, y1, a-pi/3, L*k, N-1);
```