17. Использование ресурсов стандартных системных модулей.

Некоторые интересные задачи и их решения.

17.1. Задачи по рекурсии.

17.1.1. Графическое построение бинарного дерева.

Постройте рекурсивную процедуру, которая чертит бинарное дерево до определенного уровня и определенного места локации в окне вывода. В компьютерной науке бинарные деревья начинаются со ствола, и ветви расходятся по 45-градусным углам. Ветви на каждом новом уровне имеют длину в два раза короче предыдущей. На уровне «один» дерево состоит из двух сегментных линий, на уровне «два» к дереву присоединяются две новые ветки и т.д.

Величины уровня (глубина) рекурсии и локации ствола дерева должны передаваться процедуре в качестве параметров. Вам понадобятся несколько дополнительных параметров для определения длины ветвей. Схема вашего решения должна содержать следующие пункты:

* Корректный заголовок процедуры со всеми параметрами;
* Пошаговое описание алгоритма, с выделением основного и рекурсивного вариантов.

17.1.2. Рисунок из вложенных ромбов.

Постройте рекурсивную процедуру, которая чертит рисунок из квадратов. Параметрами процедуры в качестве параметров должны иметь: уровень рекурсии, а также 8 целочисленных параметров, представляющих горизонтальные и вертикальные координаты 4-х точек в графическом окне вывода.

✓На первом уровне процедура должна начертить 4 отрезка, связанных в циклическом порядке.

✓На втором уровне процедура дополнительно чертит 4 связанных отрезка, соединяющих средние точки предыдущих отрезков.

✓На третьем уровне процедура соединяет средние точки новых четырех сегментов, и т.д.

✓Используйте рекурсию хвоста. Схема вашего решения должна содержать те же пункты (элементы), что и предыдущее упражнение 2.

Теперь рассмотрим полный программный код:

```
Type point =. Record x,y:Integer End;
Var dl,angle,a,b,c,d,x0,y0,level,limit:Integer;

Procedure drawSquares(a,b,c,d:point;level,limit:Integer);
Var al,bl,cl,dl:Integer;
Begin
                { Соединить линиями 4 точки: a,b,c,d }
                Line(a.x,a.y, b.x,b.y);
                Line(b.x,b.y, c.x,c.y);
                Line(c.x,c.y, d.x,d.y);
                Line(d.x,d.y, a.x,a.y);
                {Увеличить уровень рекурсии}
                Inc(level);
                {Условие прерывания рекурсии (граничное
                  условие)}
                If level > limit Exit;
                { Вычислить координаты для следующего
                уровня}
                al.x:=(a.x+b.x)Div2;
                al.y:=(a.y+b.y) Div 2;
                bl.x:=(b.x+c.x)Div2;
                bl.y:=(b.y+c.y)Div2;
                cl.x:=(c.x+d.x)Div2;
                cl.y:=(c.y+d.y)Div2;
                dl.x:=(d.x+a.x)Div2;
                dl.y:=(d.y+a.y)Div2;
```

```
                    {Рекурсивный самовызов }
                    drawSquares(a1,b1,c1,d1,level,limit);
                        End; {drawSquares}
{ Обратите внимание, что значение переменной «level» обновляется
при каждом рекурсивном вызове }
    Begin {Main}
        Writeln(' Введите длину квадрата:'); Read(dl);
        Writeln(' Введите глубину рекурсии:'); Read(limit);
        { Вычисление координат центра квадрата }
        x0:=GetMaxX Div 2;
        y0:=GetMaxY Div 2;
    { Вычисление координат четырех вершин первого квадрата }
        a.x:= x0 - dl Div 2;
        a.y:= y0 - dl Div 2;
        b.x:= x0 - dl Div 2;
        b.y:= y0 + dl Div 2;
        c.x:= x0 + dl Div 2;
        c.y:= y0 - dl Div 2;
        d.x:= x0 + dl Div 2;
        d.y:= y0 + dl Div 2;
        { Вызов рекурсивной процедуры }
        drawSquares(a,b,c,d,0,limit);
        { Начальное значение уровня рекурсии level=0 }
    End.
```