Условие:

Написать программу для создания динамической конструкции из элементов структуры, каждый из которых ссылается на два других элемента. Ссылка осуществляется через поля-указатели. Элементы организуются по следующему принципу. Начальный (первый) элемент ссылается на два элемента (второй и третий), каждый из которых ссылается друг на друга и на еще один элемент (четвертый и пятый). Четвертый и пятый элементы ссылаются друг на друга и на один общий элемент (шестой), который ссылается на два элемента, и т.д.

Исходный код программы:

#include **<iostream>  
using namespace** std;  
**int** global\_counter = 0;  
  
**struct** DemkaTreeStruct {  
  
 DemkaTreeStruct\* p1;  
 DemkaTreeStruct\* p2;  
 DemkaTreeStruct\* p3;  
 DemkaTreeStruct\* p4;  
 **int** n;  
  
};  
  
DemkaTreeStruct\* MakeNewDemkaTree(**int** \_n);  
  
**int** main(){  
  
 DemkaTreeStruct\* first;  
 first = MakeNewDemkaTree(3);  
 cout << **"Элементов в дереве: "** << global\_counter <<**"\n"**<<first->n<<**" ["**<<&first->n<<**"]\n"**;  
 cout << first->p1->n<<**" ["**<<&first->p1->n<<**"]\n"**;  
 cout << first->p1->p3->p2->n<<**" ["**<<&first->p1->p3->p2->n<<**"]\n"**;  
 cout << first->p2->p4->p1->n<<**" ["**<<&first->p2->p4->p1->n<<**"]\n"**;  
 **return** 0;  
}  
  
DemkaTreeStruct\* MakeNewDemkaTree(**int** \_n)  
{  
  
 DemkaTreeStruct\* p;  
 p = **new** DemkaTreeStruct;  
 global\_counter++;  
 p->n = global\_counter;  
 **if** (\_n > 1){  
  
 p->p1 = MakeNewDemkaTree(\_n - 1);  
 p->p2 = MakeNewDemkaTree(\_n - 1);  
 p->p1->p3 = MakeNewDemkaTree(\_n - 1);  
 p->p2->p4 = MakeNewDemkaTree(\_n - 1);  
 }  
  
 **return** p;  
  
}

Скриншоты программы:

