Условие:

Написать программу для создания дерева объектов, каждый из которых ссылается на два других объекта. Ссылка осуществляется через поля-указатели. Объекты организуются по следующему принципу. Начальный (первый) объект ссылается на два объекта (второй и третий), каждый из которых ссылается на четвертый общий объект и друг на друга. Четвертый объект ссылается на два объекта и т.д.

Исходный код программы:

#define LINE cout << **"----------------------\n"**#include **<iostream>  
using namespace** std;  
  
**int** global\_counter = 0;  
  
**struct** ExampleClass {  
 **int** value\_int;  
 ExampleClass\* object1;  
 ExampleClass\* object2;  
};  
  
ExampleClass\* a = NULL;  
  
**void** add\_item(ExampleClass\*& abstract\_val, **int** transfer\_int);  
**void** struct\_outer(ExampleClass\*& abstract\_val);  
  
**int** main()  
{  
 **int** buf\_inputer;  
 cout<<**"Введите кол-во значений для записи => "**;  
 cin>>buf\_inputer;  
 **for** (**int** i = 1; i <= buf\_inputer; i++)  
 add\_item(a, i);  
 LINE;  
 cout << **"\*Выборочный вывод\*\n"**;  
 cout<<**"--> "**<<a->object1->object1->value\_int<<**" ["**<<&a->object1->object1->value\_int<<**"]\n"**;  
 cout<<**"--> "**<<a->object2->object2->value\_int<<**" ["**<<&a->object2->object2->value\_int<<**"]\n"**;  
 cout<<**"--> "**<<a->object1->object2->value\_int<<**" ["**<<&a->object1->object2->value\_int<<**"]\n"**;  
 cout<<**"--> "**<<a->object2->object1->object2->value\_int<<**" ["**<<&a->object2->object1->object2->value\_int<<**"]\n"**;  
 LINE;  
 cout << **"\*Вывод всей структуры\*\n"**;  
 struct\_outer(a);  
 **return** 0;  
}  
  
**void** add\_item(ExampleClass\*& abstract\_val, **int** transfer\_int)  
{  
  
 **static** ExampleClass \*FirstStatic, \*SecondStatic;  
 **if** (abstract\_val == NULL) {  
 abstract\_val = **new** ExampleClass;  
 abstract\_val->value\_int = transfer\_int;  
 abstract\_val->object1 = NULL;  
 abstract\_val->object2 = NULL;  
 global\_counter++;  
 **if** (global\_counter == 1)  
 SecondStatic = abstract\_val;  
 **if** ((global\_counter + 1) % 3 == 0)  
 FirstStatic = abstract\_val;  
 **if** (((global\_counter + 2) % 3 == 0) && global\_counter != 1) {  
 SecondStatic = abstract\_val;  
 FirstStatic->object2 = abstract\_val;  
 }  
 **if** (global\_counter % 3 == 0) {  
 abstract\_val->object2 = SecondStatic;  
 SecondStatic->object2 = abstract\_val;  
 }  
 }  
 **else** {  
 add\_item(abstract\_val->object1, transfer\_int);  
 }  
}  
  
  
**void** struct\_outer(ExampleClass\*& abstract\_val)  
{  
 **if** (abstract\_val != NULL) {  
 cout<<**"--> "**<<abstract\_val->value\_int <<**" ["**<<&abstract\_val->value\_int<<**"]\n"**;  
 struct\_outer(abstract\_val->object1);  
 }  
}

Скриншоты программы:

