#include <iostream>

#include <ctime>

using namespace std;

struct tree

{

int el;

tree \*left;

tree \*right;

};

tree\* List(int a);

void New\_List(int key);

void Watch(tree \*a, int level);

void Delete(tree \*t);

int menu();

void DopRand(int kol);

void Mission(tree \*o, int isk, tree \*papka);

tree \*Wurzel=NULL;

int main()

{

system("color 1a");

setlocale(0, "");

srand(time(0));

while (true)

{

int Number = menu();

switch (Number)

{

case 1: // Добавление рандомных элементов

{

int kol;

cout << "Введите приблизительное количество элементов для добавления:\t";

cin >> kol;

DopRand(kol);

}

break;

case 2: // Добавление нужного элемента

{

cout << "Введите элемент для добавления в дерево:\t";

int o;

cin >> o;

if (Wurzel == 0)

Wurzel = List(o);

else

New\_List(o);

}

break;

case 3: // Удаление дерева

{

Delete(Wurzel);

}

break;

case 5: // Просмотр дерева

{

Watch(Wurzel, 0);

system("pause");

}

break;

case 777: // Задание

{

cout << "Введите элемент, чью ветвь необходимо удалить:\t";

int isk;

cin >> isk;

Mission(Wurzel, isk, Wurzel);

}

break;

case 0: // Завершение

return 0;

}

}

}

tree\* List(int l)

{

tree \*list = new tree;

list->el = l;

list->left = list->right = NULL;

return list;

}

void New\_List(int key)

{

tree \*papa, \*t;

int Search = 1;

t = Wurzel;

while (t && Search)

{

papa = t;

if (key == t->el)

{

Search = 0;

cout << "Повторяющийся элемент!";

return;

}

else

{

if (key < t->el)

t = t->left;

else

t = t->right;

}

}

if (Search)

t = List(key);

if (key < papa->el)

papa->left = t;

if (key > papa->el)

papa->right = t;

}

void Watch(tree \*a, int level)

{

if (Wurzel == NULL)

{

cout << "Дерево пусто!!!" << endl;

return;

}

if (a)

{

Watch(a->right, level + 1);

for (int i = 0; i < level; i++)

cout << "=";

cout << a->el << endl;

Watch(a->left, level + 1);

}

}

void Delete(tree \*t)

{

if (t != NULL)

{

Delete(t->left);

Delete(t->right);

}

delete t;

Wurzel = NULL;

}

int menu()

{

system("cls");

cout << " Выберите операцию:\n";

cout << " 1. Добавление РАНДОМНЫХ элементов.\n";

cout << " 2. Добавление элемента в дерево.\n";

cout << " 3. Удаление дерева.\n";

cout << " 5. Просмотр дерева.\n";

cout << " 777. Выполнение задания.\n";

cout << " 0. Выход.\n";

cout << " --> ";

int t;

cin >> t;

system("cls");

return t;

}

void DopRand(int kol)

{

if (kol == 0)

return;

if (Wurzel == NULL)

{

Wurzel = List(0 + rand() % 30);

kol--;

}

for(int i=0; i < kol; i++)

New\_List(0 + rand() % 30);

}

void Mission(tree \*o, int isk, tree \*papka)

{

if (Wurzel == NULL)

{

cout << "Дерево пусто!!!" << endl;

return;

}

if (o)

{

if (o->el == isk)

{

system("pause");

if (papka->left == o)

papka->left = NULL;

if (papka->right == o)

papka->right = NULL;

return;

}

papka = o;

Mission(o->right, isk, papka);

Mission(o->left, isk, papka);

}

}