#include "iostream"

#include "fstream"

#include <windows.h>

#include <stdio.h>

#include <io.h>

#include <conio.h>

#include <string.h>

using namespace std;

struct Stack {

int info;

Stack\* next;

} \*b, \* t;

struct Stack1 {

int info;

Stack1\* next;

} \*x;

struct Stack2 {

int info;

Stack2\* next;

} \*z;

Stack\* InStack(Stack\*, int);

void View(Stack\*);

void Del\_All(Stack\*\*);

void Sort\_a(Stack\*\* p);

void Sort\_info(Stack\* p);

void zadan(Stack\*);

void View1(Stack1\*);

void View2(Stack2\* z);

Stack1\* InStack1(Stack1\*, int);

Stack2\* InStack2(Stack2\*, int);

void Addlast(Stack\*, int);

void main()

{

Stack\* end = NULL;

setlocale(LC\_ALL, "ru");

int i, in, n, kod;

while (true) {

cout << "1-Создать\n""2-Добавить\n""3-Просмотр\n""4-Освобдить память\n""5-Разделение на положительные и отрицательные\n""6-Просмотр разделенных стэков\n""0-Выход\n";

cin >> kod;

switch (kod) {

case 1: case 2:

if (kod == 1 && b != NULL) {

cout << "Освободите память!" << endl;

break;

}

cout << "Введите количество = "; cin >> n;

for (i = 1; i <= n; i++) {

cout << "ELEMENT " << i << " =";

cin >> in;

b = InStack(b, in);

}

break;

case 3: if (!b) {

cout << "Стэк пуст" << endl;

break;

}

cout << "--- Стэк ---" << endl;

View(b);

break;

case 4:

Del\_All(&b);

cout << "Память свободна" << endl;

break;

case 5:

zadan(b);

break;

case 6:

if (!b) {

cout << "Стэк пуст" << endl;

break;

}

cout << "Положительные" << endl;

View1(x);

cout << "Отрицательные" << endl;

View2(z);

break;

}

}

}

void Del\_All(Stack\*\* b) {

while (\*b != NULL) {

t = \*b;

\*b = (\*b)->next;

delete t;

}

}

void View(Stack\* p) {

Stack\* t = p;

while (t != NULL) {

cout << " " << t->info << endl;

t = t->next;

}

}

Stack\* InStack(Stack\* p, int in) {

Stack\* t = new Stack;

t->info = in;

t->next = p;

return t;

}

Stack1\* InStack1(Stack1\* p, int in) {

Stack1\* pol = new Stack1;

pol->info = in;

pol->next = p;

return pol;

}

Stack2\* InStack2(Stack2\* p, int in) {

Stack2\* otr = new Stack2;

otr->info = in;

otr->next = p;

return otr;

}

void zadan(Stack\* b) {

if (b == NULL) {

cout << "Стек пуст" << endl;

return;

}

Stack\* t = b;

int r;

while (t != NULL) {

if (t->info >= 0) {

r = t->info;

x = InStack1(x, r);

t = t->next;

}

else {

r = t->info;

z = InStack2(z, r);

t = t->next;

}

}

cout << endl;

cout << "Стэк разделен ";

}

void View1(Stack1\* x) {

Stack1\* t = x;

while (t != NULL) {

cout << " " << t->info << endl;

t = t->next;

}

}

void View2(Stack2\* z) {

Stack2\* t = z;

while (t != NULL) {

cout << " " << t->info << endl;

t = t->next;

}

}

void Max(Stack\* b)

{

Stack\* t = b;

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

#include <iostream>

#include <ctime>

#include <cstdlib>

using std::cout;

using std::cin;

using std::endl;

struct Stack

{

int data = 0;

Stack\* next = NULL;

};

Stack\* Add(Stack\* s, int x)

{

Stack\* t = new Stack;

t->data = x;

t->next = s;

return t;

}

Stack\* Get(Stack\* s, int& x)

{

if (s == NULL)

return s;

x = s->data;

Stack\* t = s;

s = s->next;

delete t;

return s;

}

void View(Stack\* s)

{

while (s)

{

cout << s->data << ' ';

s = s->next;

}

cout << endl;

return;

}

void Delete(Stack\*& s)

{

while (s)

{

Stack\* t = s;

s = s->next;

delete t;

}

return;

}

int main()

{

srand(time(0));

int n = 0;

cout << "N = ";

cin >> n;

Stack\* s = NULL;

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

s = Add(s, rand() % 101 - 50);

cout << "Original stack:\n";

View(s);

Stack\* pos = NULL, \* neg = NULL;

int x = 0;

while (s != NULL)

{

s = Get(s, x);

if (x < 0)

neg = Add(neg, x);

else

pos = Add(pos, x);

}

cout << "Negative:\n";

View(neg);

cout << "Positive:\n";

View(pos);

Delete(neg);

Delete(pos);

return 0;

}