Лабораторная работа 4  
Вариант 3

Задание: Создать стек, каждый элемент которого является совокупностью двух целых значений. Заменить нулевыми все элементы с заданной суммой значений

#include <iostream>

#include<string>

using namespace std;

struct lish

{

int p; // информационный элемент

lish\* nExt, \* hEad; //Голова стека и указатель на следующий элемент

};

struct delte

{

int f; // информационный элемент

delte\* next, \* head; //Голова стека и указатель на следующий элемент

};

struct List

{

int x; //информационный элемент

List\* Next, \* Head; //Голова стека и указатель на следующий элемент

};

void Add(int x, List\*& MyList) //Принимаем элемент стека и указатель на стек, при этом говорим, что принимаемый указатель будет сам по себе указателем

{

List\* temp = new List; //Выделяем память для нового элемента

temp->x = x; //Записываем в поле x принимаемый в функцию элемент

temp->Next = MyList->Head; //Указываем, что следующий элемент это предыдущий

MyList->Head = temp; //Сдвигаем голову на позицию вперед

}

void Add1(int f, delte\*& mydelte)

{

delte\* Temp = new delte; //Принимаем элемент стека и указатель на стек, при этом говорим, что принимаемый указатель будет сам по себе указателем

Temp->f = f; //Выделяем память для нового элемента

Temp->next = mydelte->head; //Указываем, что следующий элемент это предыдущий

mydelte->head = Temp; //Сдвигаем голову на позицию вперед

}

void Add2(int p, lish\*& mylish)

{

lish\* temP = new lish; //Принимаем элемент стека и указатель на стек, при этом говорим, что принимаемый указатель будет сам по себе указателем

temP->p = p; //Выделяем память для нового элемента

temP->nExt = mylish->hEad; //Указываем, что следующий элемент это предыдущий

mylish->hEad = temP; //Сдвигаем голову на позицию вперед

}

void Show(List\* MyList, delte\*& mydelte, lish\*& mylish)

{

List\* temp = MyList->Head; //Объявляем указатель и Указываем ему, что его позиция в голове стека

delte\* Temp = mydelte->head; //Объявляем указатель и Указываем ему, что его позиция в голове стека

lish\* temP = mylish->hEad; //Объявляем указатель и Указываем ему, что его позиция в голове стека

int o;

o = 0;

while (temp != NULL) //выходим при встрече с пустым полем

{

o = Temp->f; //присваиваем о значение стека(на сейчас)

Temp = Temp->next; //Переходим к следующему элементу

if (o != temP->p)

{

cout << temp->x << " "; //Выводим на экран элемент стека

temp = temp->Next; //Переходим к следующему элементу

}

else

{

cout << "0 "; //выводим 0

temp = temp->Next; //идем дальше

}

}

}

void ClearList(List\* MyList)

{

while (MyList->Head != NULL) //Пока по адресу не пусто

{

List\* temp = MyList->Head->Next; //Временная переменная для хранения адреса следующего элемента

delete MyList->Head; //Освобождаем адрес обозначающий начало

MyList->Head = temp; //Меняем адрес на следующий

}

}

int main()

{

setlocale(LC\_ALL, "Russian"); //Подключаем русский язык

List\* MyList = new List; //Выделяем память для стека

delte\* mydelte = new delte; //Выделяем память для стека

lish\* mylish = new lish; //Выделяем память для стека

string s; //переменная S это строка

int g, a, y, n, t, m, j; // задаем переменные

a = 0; // инициализируем переменную

MyList->Head = NULL; //Во избежание ошибок инициализируем первый элемент

mydelte->head = NULL; //Во избежание ошибок инициализируем первый элемент

cin >> y; // вводим "сумму"

for (int i = 0; i < 10; i++)

{

cout << "Введите число:";

cin >> g; // вводим число

s = g; // задаем строку

n = s.length(); // длина строки

for (t = 0; t < n; t++)

{

Add(g, MyList); //Заносим данные в стек

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

{

m = g % 10; // берем остаток от заданного числа

a = a + m; // считаем сумму

g = g / 10; // сокращаем g на 10

}

Add1(a, mydelte); //записали сумму

a = 0;

}

Add2(y, mylish); // заносим "сумму"

} //тут работает

Show(MyList, mydelte, mylish); //показывает наш получившийся стек

ClearList(MyList); // отчищаем память

system("pause");

}



