**Организация списков с помощью указателей и структур.**

Дан список идентификаторов. Длина каждого иден тификатора не более 8 символов. Идентификаторы в списке расположены в лексикографическом порядке. Составить функции (подпрограммы) для следующих операций:

1.Удалить из списка все элементы, начиная с k-го по порядку;

2.Заменить на заданный идентификатор значение предпоследнего элемента списка.

3.Определить количество идентификаторов в списке следующие после заданного идентификатора.

Код:

#include <iostream>

#include<string>

#include <windows.h>

using namespace std;

struct EL\_SP

{

string id;

EL\_SP \*sled;

};

void Vkl(EL\_SP \*\*p, string t\_id)

{

EL\_SP \*pt;

EL\_SP \*k = NULL;

EL\_SP \*j = NULL;

pt = new EL\_SP();

pt->id = t\_id;

if (\*p == NULL || pt->id<(\*p)->id)

{

pt->sled = \*p;

\*p = pt;

}

else

{

k = \*p;

while (k != NULL&&pt->id >= k->id)

{

j = k;

k = k->sled;

}

j->sled = pt;

pt->sled = k;

}

}

void PechSp(EL\_SP \*p)

{

EL\_SP \*i;

cout << "Результат: " << endl;

for (i = p; i != NULL; i = i->sled)

cout << i->id << endl;

}

void zamenapredposlednego(EL\_SP \*p, int kol1)

{

EL\_SP \*i;

int k=0;

string zd\_id;

cout << "ВВедите на какой элемент заменить";

cin >> zd\_id;

int j = 1;

for (i = p,j; i != NULL; i = i->sled,j++)

k++;

j=1;

k--;

for (i = p,j; i != NULL&&j<k; i = i->sled,j++)

continue;

i->id=zd\_id;

cout<<"замена выполнена"<<endl;

}

void udaleniesk(EL\_SP \*\*p)

{

int s=0,kelem;

cout << "ВВедите с какого элемента удалить: ";

cin>>kelem;

EL\_SP \*i = \*p;

while (i != NULL)

{

if (s<kelem-1)

{

i = i->sled;

}

else

{

delete i;

cout << "удаление выполнено " << endl;

}

s++;

if (s>15)

break;

}

}

void kolvoposle(EL\_SP \*\*p)

{

string zd\_id;

int k=0, kolvo=0;

cout << "ВВедите идентификатор: ";

getline(cin, zd\_id);

EL\_SP \*i = \*p;

while (i != NULL)

{

if (i->id == zd\_id)

{

k++;

i = i->sled;

}

else

{

i = i->sled;

}

if(k>0)

{

kolvo++;

}

}

cout<<"количество идентификаторов после заданного элемента = "<<kolvo-1<<endl;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int kol;

char n;

EL\_SP \*p;

unsigned i;

string t\_id;

p = NULL;

cout << "Введите индентифактор" << endl << "После каждого нажмите ENTER" << endl;

for (i = 1;; i++)

{

getline(cin, t\_id);

cin.sync();

cin.clear();

if (t\_id.size() > 8)

{

i--;

cout << "Ошибка!" << endl << "Введите индентификатор не более 8 символов" << endl;

}

else if (t\_id.size() == 0)

{

i--;

cout << "Ввод индентификаторов закончен" << endl;

break;

}

else

Vkl(&p, t\_id);

}

kol = i;

cout << kol;

do

{

cout << "========================================================" << endl;

cout << "Выберите номер пункта меню:" << endl;

cout << "1 - вывод идентификаторов в алфавитном порядке" << endl;

cout << "2 - удалить идентификаторы начиная с k-ого" << endl;

cout << "3 - заменить на заданный идентификатор предпоследний элемент" << endl;

cout << "4 - кол-во идентификаторов после заданного" << endl;

cout << "5 - выход" << endl;

cout << "------------------------------------------------------" << endl;

n = cin.get();

cin.sync();

switch (n)

{

case '1': PechSp(p); break;

case '2': udaleniesk(&p); break;

case '3': zamenapredposlednego(p,kol); break;

case '4': kolvoposle(&p); break;

default: cout << "\nНужно вводить номер пункта от 1 до 5";

}

if (n != '5')

{

cout << "\nДля продолжения нажмите любую клавишу";

cin.get();

}

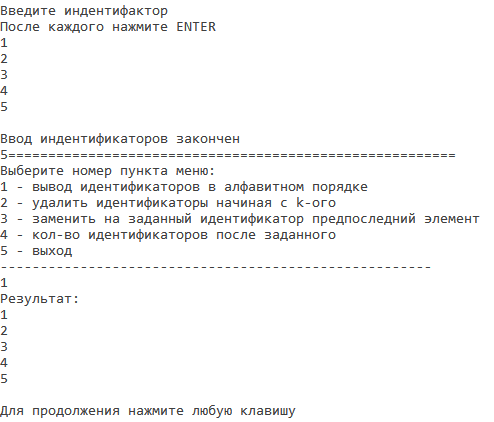
} while (n != '5');

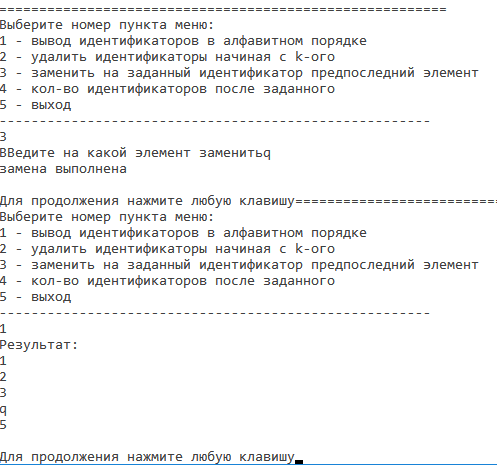
cout << endl;

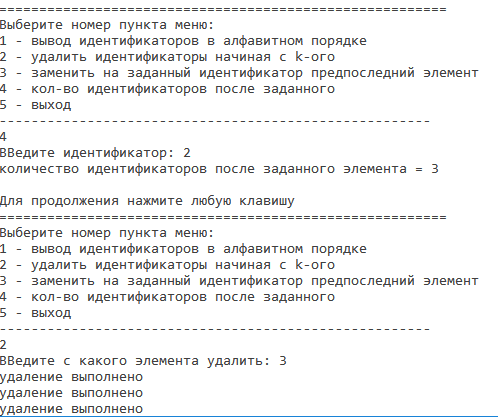
return 1;

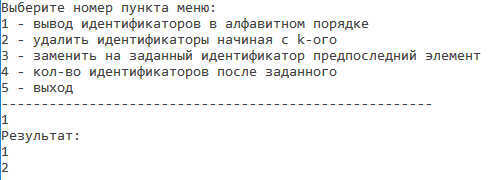
}

Результаты:









**Модульное программирование.**

Записать в отдельные файлы свои функции обработки списка. Добавить в проект имена своих файлов. Изменить функцию

main()

в файле «main.c», чтобы вызывались ваши функции и выводились

результаты их работы. Проверить результат работы программы.

Изменить в файле проекта расширения своих файлов на .obj

и снова выполнить программ

header.h:

#pragma once

#include <iostream>

#include<string>

#include <windows.h>

using namespace std;

struct EL\_SP

{

string id;

EL\_SP \*sled;

};

void Vkl(EL\_SP \*\*p, string t\_id);

void PechSp(EL\_SP \*p);

void zamenapredposlednego(EL\_SP \*p, int kol1);

void udaleniesk(EL\_SP \*\*p);

void kolvoposle(EL\_SP \*\*p);

main.cpp:

#include "header.h"

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int kol;

char n;

EL\_SP \*p;

unsigned i;

string t\_id;

p = NULL;

cout << "Введите индентифактор" << endl << "После каждого нажмите ENTER" << endl;

for (i = 1;; i++)

{

getline(cin, t\_id);

cin.sync();

cin.clear();

if (t\_id.size() > 8)

{

i--;

cout << "Ошибка!" << endl << "Введите индентификатор не более 8 символов" << endl;

}

else if (t\_id.size() == 0)

{

i--;

cout << "Ввод индентификаторов закончен" << endl;

break;

}

else

Vkl(&p, t\_id);

}

kol = i;

cout << kol;

do

{

cout << "========================================================" << endl;

cout << "Выберите номер пункта меню:" << endl;

cout << "1 - вывод идентификаторов в алфавитном порядке" << endl;

cout << "2 - удалить идентификаторы начиная с k-ого" << endl;

cout << "3 - заменить на заданный идентификатор предпоследний элемент" << endl;

cout << "4 - кол-во идентификаторов после заданного" << endl;

cout << "5 - выход" << endl;

cout << "------------------------------------------------------" << endl;

n = cin.get();

cin.sync();

switch (n)

{

case '1': PechSp(p); break;

case '2': udaleniesk(&p); break;

case '3': zamenapredposlednego(p,kol); break;

case '4': kolvoposle(&p); break;

default: cout << "\nНужно вводить номер пункта от 1 до 5";

}

if (n != '5')

{

cout << "\nДля продолжения нажмите любую клавишу";

cin.get();

}

} while (n != '5');

cout << endl;

return 1;

}

Vkl.cpp:

#include "header.h"

void Vkl(EL\_SP \*\*p, string t\_id)

{

EL\_SP \*pt;

EL\_SP \*k = NULL;

EL\_SP \*j = NULL;

pt = new EL\_SP();

pt->id = t\_id;

if (\*p == NULL || pt->id<(\*p)->id)

{

pt->sled = \*p;

\*p = pt;

}

else

{

k = \*p;

while (k != NULL&&pt->id >= k->id)

{

j = k;

k = k->sled;

}

j->sled = pt;

pt->sled = k;

}

}

PechSp.cpp:

#include "header.h"

void PechSp(EL\_SP \*p)

{

EL\_SP \*i;

cout << "Результат: " << endl;

for (i = p; i != NULL; i = i->sled)

cout << i->id << endl;

}

zamenapredposlednego.cpp:

#include "header.h"

void zamenapredposlednego(EL\_SP \*p, int kol1)

{

EL\_SP \*i;

int k=0;

string zd\_id;

cout << "ВВедите на какой элемент заменить";

cin >> zd\_id;

int j = 1;

for (i = p,j; i != NULL; i = i->sled,j++)

k++;

j=1;

k--;

for (i = p,j; i != NULL&&j<k; i = i->sled,j++)

continue;

i->id=zd\_id;

cout<<"замена выполнена"<<endl;

}

udaleniesk.cpp:

#include "header.h"

void udaleniesk(EL\_SP \*\*p)

{

int s=0,kelem;

cout << "ВВедите с какого элемента удалить: ";

cin>>kelem;

EL\_SP \*i = \*p;

while (i != NULL)

{

if (s<kelem-1)

{

i = i->sled;

}

else

{

delete i;

cout << "удаление выполнено " << endl;

}

s++;

if (s>15)

break;

}

}

kolvoposle.cpp:

#include "header.h"

void kolvoposle(EL\_SP \*\*p)

{

string zd\_id;

int k=0, kolvo=0;

cout << "ВВедите идентификатор: ";

getline(cin, zd\_id);

EL\_SP \*i = \*p;

while (i != NULL)

{

if (i->id == zd\_id)

{

k++;

i = i->sled;

}

else

{

i = i->sled;

}

if(k>0)

{

kolvo++;

}

}

cout<<"количество идентификаторов после заданного элемента = "<<kolvo-1<<endl;

}

