|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

**Отчет по лабораторной работе**

|  |  |  |  |
| --- | --- | --- | --- |
| **Структуры и алгоритмы обработки данных** | | | |
| *( наименование дисциплины )* | | | |
| Направление подготовки | | | 09.03.01 Информатика и вычислительная техника |
|  | | |  |
| Тема | Линейные списки | | |
| Институт | | информационных технологий (ИТ) | |

Выполнил студент 3 курса

Группа ИВБО-06-17

Фамилия Имя Ушакова

Анастасия

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Оглавление

[Оглавление 2](#_Toc21508760)

[1. Задание 1 3](#_Toc21508761)

[1.1 Вариант задания 3](#_Toc21508762)

[1.2 Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания. 3](#_Toc21508763)

[1.3 Реализация АТД 3](#_Toc21508764)

[1.4 Таблица тестов 5](#_Toc21508765)

[1.5 Код основной программы с комментариями 6](#_Toc21508766)

[1.6 Скриншоты выполнения тестов 14](#_Toc21508767)

[2. Задание 2 15](#_Toc21508768)

[2.1 Структура узла 15](#_Toc21508769)

[2.2 Структура класса 15](#_Toc21508770)

[2.3 Таблица тестов 15](#_Toc21508771)

[2.4 Код основной программы с комментариями 17](#_Toc21508772)

[2.5 Скриншоты выполнения тестов 25](#_Toc21508773)

[Заключение 26](#_Toc21508774)

# Задание 1

## Вариант задания

Разработать многомодульную программу, которая демонстрирует выполнение всех операций, определённых вариантом, над линейным однонаправленным динамическим списком.

Разработайте функции для выполнения операции над линейным списком:

* создание списка из n узлов
* вывод списка
* поиск узла с заданным значением (операция должна возвращать указатель на заданный узел).

Оформите операции предложенных вариантов в виде функций и включите в отдельный файл с расширением cpp. Подключите к этому файлу заголовочный файл с определением структуры узла.

Разработайте программу, управляемую текстовым меню, согласно заданию и включите в меню демонстрацию выполнения всех операций задания и варианта.

## Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания.

Марка автомобиля, страна изготовитель, год выпуска

Упорядочить исходный список по номеру группы.

Вставить новый узел по автомобилю так, чтобы он разместился в списке своего модельного ряда, перед узлом, год выпуска которого меньше.

Сформировать новый список, узел которого содержит данные: марка автомобиля, указатель на начало модельного ряда этой марки в исходном списке.

Удалить информацию обо всех автомобилях заданной марки из исходного списка, поиск первого узла модельного ряда, осуществить с помощью нового списка.

## Реализация АТД

* Определение типа информационной части узла (typedef)

typedef struct spisok{

string type;

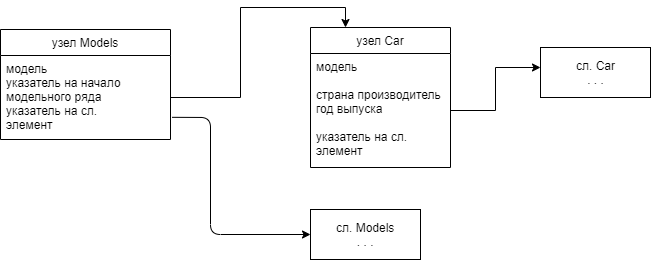
string country;

int year;

spisok\* next;

} Car;

* Структура узла



* Алгоритмы задач варианта, которые не включены в АТД на псевдокоде и использующие операции из АТД.

Вставить новый узел по автомобилю так, чтобы он разместился в списке своего модельного ряда, перед узлом, год выпуска которого меньше:

пока( следующий объект существует){

если (модель следующего объекта совпадает с введённой моделью){

если (год следующего объекта меньше года введённого объекта){

конец цикла;

}

} иначе если(модель объекта совпадает с введённой моделью){

Конец цикла;

}

передвинуться на один объект дальше;

}

сохранить в буферной переменной указатель на следующий объект;

указателю на следующий объект присвоить указатель на введённый объект;

указателю на следующий после введённого объект присвоить значение буферной переменной;

вернуть указатель на первый объект;

Сформировать новый список, узел которого содержит данные: марка автомобиля, указатель на начало модельного ряда этой марки в исходном списке:

Для (каждого i от 0 до m) {

создание нового объекта;

ввод информации о новом объекте;

если (лист пуст) {

присвоить указатель на начало списка;

} иначе {

пока (существует следующий объект) {

присвоить последнему указателю указатель на новый объект;

}

перейти к следующему объекту;

}

присвоить указателю на начало модельного ряда результат функции нахождения элемента по определённой модели;

}

Удалить информацию обо всех автомобилях заданной марки из исходного списка, поиск первого узла модельного ряда, осуществить с помощью нового списка:

поиск указателя на первый удаляемый объект;

если (нашелся объект, который надо удалить) {

пока (модель объекта совпадает с заданным значением и существует следующий объект) {

трюк Вирта;

}

если (текущий объект последний или модель следующего объекта не равна заданному значению) {

присваиваем объекту указатель на следующий объект;

если (удаляемый объект стоит в начале) {

присваиваем указателю на первый объект указатель на следующий объект после удаляемого;

}

}

}

возвращаем указатель на начало списка;

* Список модулей реализации АТД (или описать где расположена реализация АТД)

Для реализации использовались:

Структура Car

Структура Models

## Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| №  теста | Эталон результата | Результат программы | Результат анализа  теста (+ или -) |
|
| 1 | Вывод списка:  Information.  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:jhgf  Year:1980  Type:BMW  Country:hgfd  Year:2000  Type:BMW  Country:hgfd  Year:2010    Поиск элемента:  ELEMENT:  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:jhgf  Year:1980  Вставка информации о новой машине:  Information.  Type:Nissan  Country:hgf  Year:1999  **Type:Nissan**  **Country:hgf**  **Year:1990**  Type:Nissan  Country:jhgf  Year:1980  Type:BMW  Country:hgfd  Year:2000  Type:BMW  Country:hgfd  Year:2010  Вывод списка моделей машин:  Information about cars.  Type:Nissan  Type:BMW  Удаление элементов:  Information.  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:hgf  Year:1990  Type:Nissan  Country:jhgf  Year:1980  Выход:  Process returned 0 (0x0) execution time : 387.774 s  Press any key to continue. | Вывод списка:  Information.  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:jhgf  Year:1980  Type:BMW  Country:hgfd  Year:2000  Type:BMW  Country:hgfd  Year:2010  Поиск элемента:  ELEMENT:  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:jhgf  Year:1980  Вставка информации о новой машине:  Information.  Type:Nissan  Country:hgf  Year:1999  **Type:Nissan**  **Country:hgf**  **Year:1990**  Type:Nissan  Country:jhgf  Year:1980  Type:BMW  Country:hgfd  Year:2000  Type:BMW  Country:hgfd  Year:2010  Вывод списка моделей машин:  Information about cars.  Type:Nissan  Type:BMW  Удаление элементов:  Information.  Type:Nissan  Country:hgf  Year:1999  Type:Nissan  Country:hgf  Year:1990  Type:Nissan  Country:jhgf  Year:1980  Выход:  Process returned 0 (0x0) execution time : 387.774 s  Press any key to continue. | **+** |

## Код основной программы с комментариями

#include "Structers.h"

#include "windows.h"

using namespace std;

//создание списка

void createList(Car\*& List, int n){

Car\* newCar;

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

newCar=new Car;

cout<< "Enter information about a car.\n";

cout<<"Type:";

cin>>newCar->type;

cout<<"Country:";

cin>>newCar->country;

cout<<"Year:";

cin>>newCar->year;

newCar->next = 0;

if(List==NULL){

List=newCar;

}

else{

Car\* newCar1 = List;

while(newCar1->next){

newCar1=newCar1->next;

}

newCar1->next = newCar;

}

}

}

//вывод списка

void outList(Car\* List){

Car\* newCar = List;

cout<<"Information.\n";

while (newCar){

cout<<"Type:"<<newCar->type<<"\nCountry:"<<newCar->country<<"\nYear:"<<newCar->year<<endl;

newCar = newCar->next;

}

}

//поискк элемента по значению

Car\* findElement(Car\* List,string value){

Car\* pointer= List;

while(pointer){

if(pointer->type==value)

break;

pointer=pointer->next;

}

return pointer;

}

//вставка элемента

Car\* pushEl(Car\* List){

Car \*temp, \*ptr, \*pointer=List;

temp=new Car;

cout<< "Enter information about a new car.\n";

cout<<"Type:";

cin>>temp->type;

cout<<"Country:";

cin>>temp->country;

cout<<"Year:";

cin>>temp->year;

while(pointer->next){

if (pointer->next->type==temp->type){

if(pointer->next->year<temp->year){

break;

}

}else if(pointer->type==temp->type){

break;

}

pointer=pointer->next;

}

//вставка

ptr=pointer->next;

pointer->next=temp;

temp->next = ptr;

return temp;

}

//создание списка моделей машин

void createNewList(Models\*& List, int m, Car\* L){

Models\* ptr;

for(int i=0;i<m;i++){

ptr=new Models;

cout<< "Enter information.\n";

cout<<"Type:";

cin>>ptr->type;

ptr->next = 0;

if(List==NULL){

List=ptr;

}

else{

Models\* ptr1 = List;

while(ptr1->next){

ptr1=ptr1->next ;

}

ptr1->next = ptr;

}

ptr->begining = findElement(L,ptr->type);

}

}

//вывод списка моделей машин

void outNewList(Models\* List){

Models\* ptr = List;

cout<<"Information about cars.\n";

while (ptr){

cout<<"Type:"<<ptr->type<<endl;

ptr = ptr->next;

}

}

//удаление элементов

Car\* deleteModel(Models\* List, Car\* L, string value){

Models\* pointer= List;

Car\* temp;

//поиск указателя на первый удаляемый элемент

while(pointer){

if(pointer->type==value){

break;

}

pointer=pointer->next;

}

//Если нашелся элемент, который нужно удалить

if (pointer && pointer->begining)

{

temp = pointer->begining;//для удобства

while (temp->type == value && temp->next) {

//Трюк Вирта

temp->country = temp->next->country;

temp->type = temp->next->type;

temp->year = temp->next->year;

temp->next = temp->next->next;

}

//удаление

if (temp->next == 0 || temp->next->type != value) {

temp = temp->next;

if (temp == L) {

L = temp->next;

free(temp);

}

}

}

return L;

}

int main()

{ SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Car\* List = NULL;

Models\* Spisk = NULL;

int var;

while(var){

system("cls");

cout<<"Меню:\n"<<"1.Создание модельного списка машин.\n2.Вывод модельного списка машин.\n3.Поиск элемента в модельном списке\n4.Вставка информации о новой машине.\n5.Создание списка моеделей машин\n6.Вывод списка машин\n7.Удаление информации о машинах определённой модели\n0.Выход\n";

cout<<"Выберите действие:";

cin>>var;

switch(var){

case 1:{

int n=4;

//создание списка машин

createList(List,n);

system("pause");

break;

}

case 2:{

//вывод списка машин

outList(List);

system("pause");

break;

}

case 3:{

//поиск элемента

cout<<"ELEMENT:\n";

Car\* Lst = findElement(List, "Nissan");

while (Lst && Lst->type=="Nissan"){

cout<<"Type:"<<Lst->type<<"\nCountry:"<<Lst->country<<"\nYear:"<<Lst->year<<endl;

Lst = Lst->next;

}

system("pause");

break;

}

case 4:{

//вставка элемента

pushEl(List);

system("pause");

break;

}

case 5:{

int m=2;

createNewList(Spisk,m, List);

system("pause");

break;

}

case 6:{

outNewList(Spisk);

system("pause");

break;

}

case 7:{

//удаление элементов

deleteModel(Spisk,List, "BMW");

system("pause");

break;

}

case 0:{

exit(0);

system("pause");

break;

}

default:

cout<<"Выберите число от 0 до 7\n";

system("pause");

break;

}

}

return 0;

}

Файл structers.h

#include <iostream>

#include <cstdlib>

using namespace std;

typedef struct spisok{

string type;

string country;

int year;

spisok\* next;

} Car;

typedef struct spisok2{

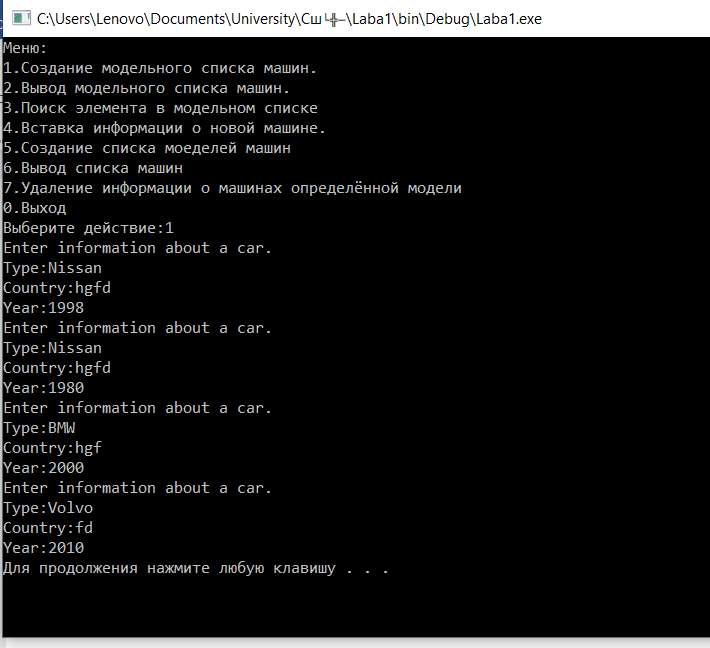
string type;

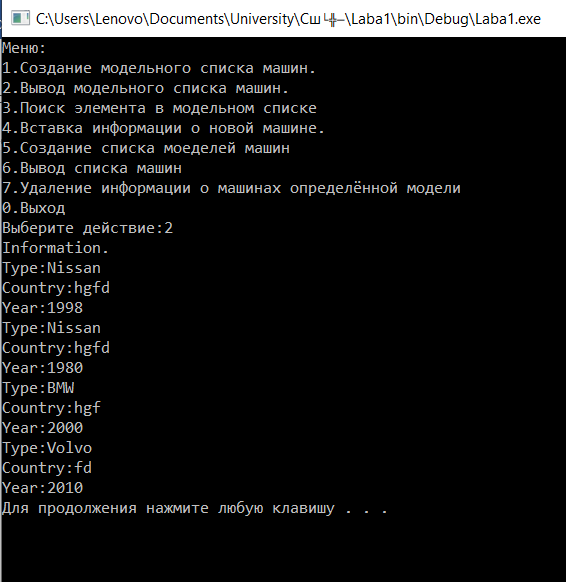
spisok\* begining;

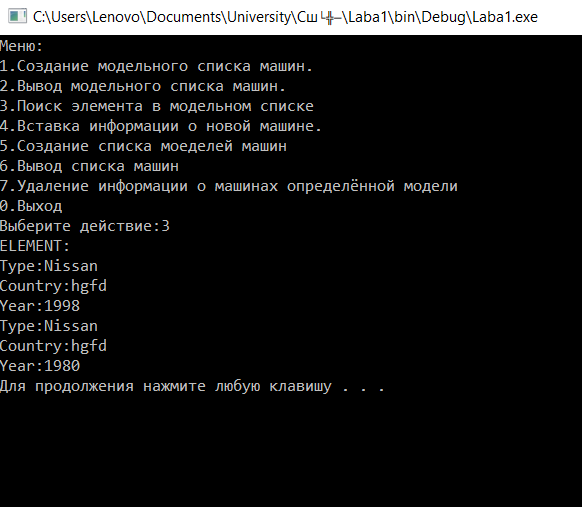
spisok2\* next;

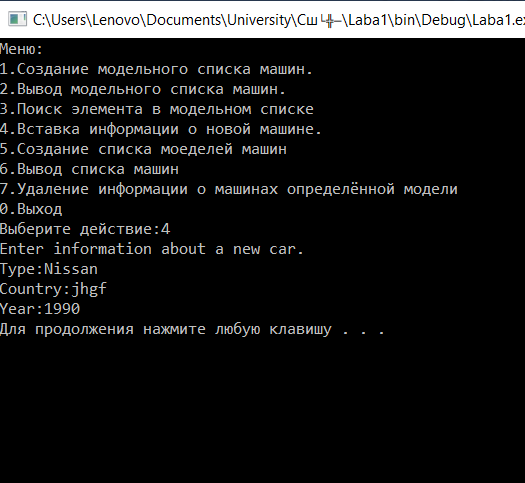
} Models;

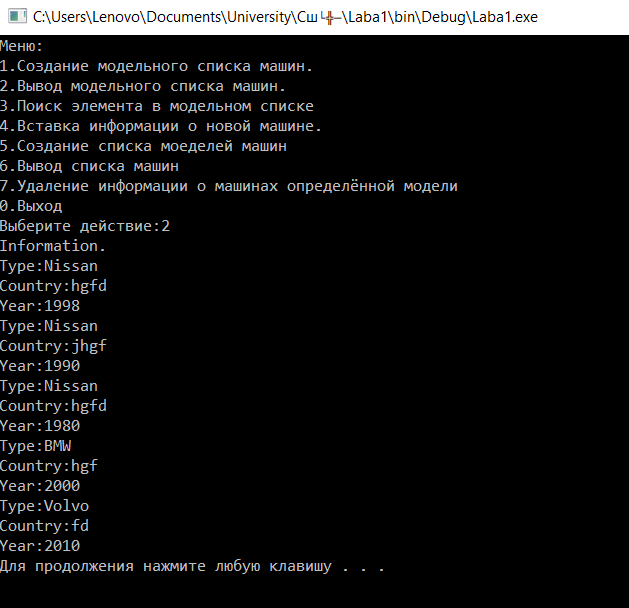
## Скриншоты выполнения тестов

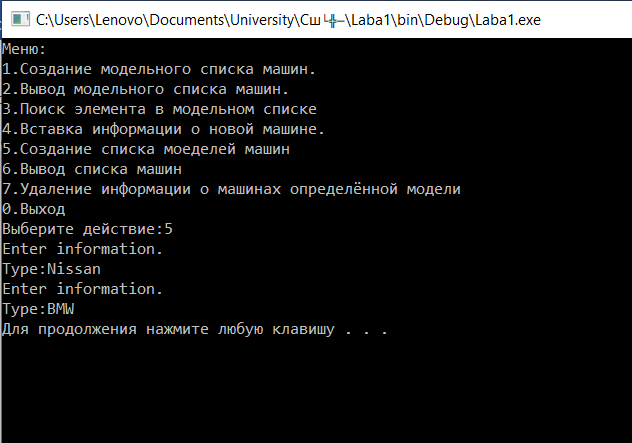


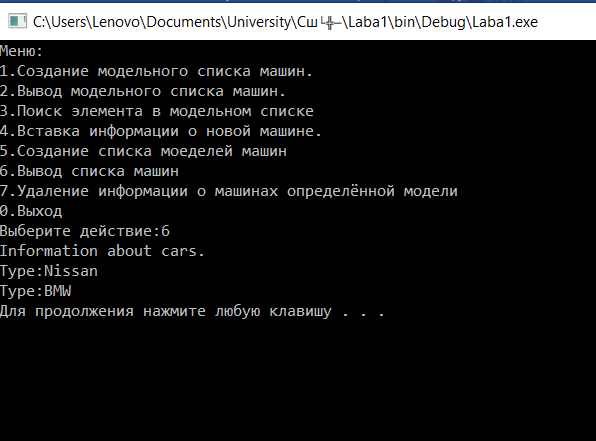


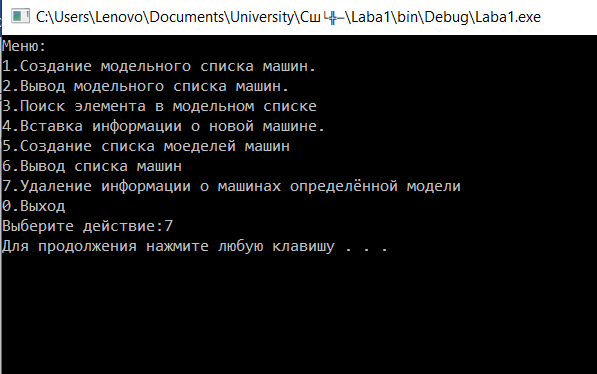


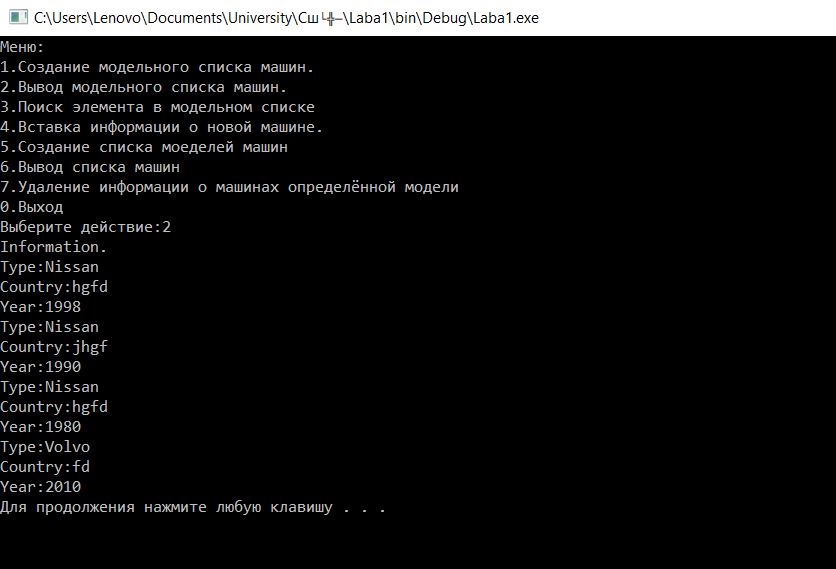
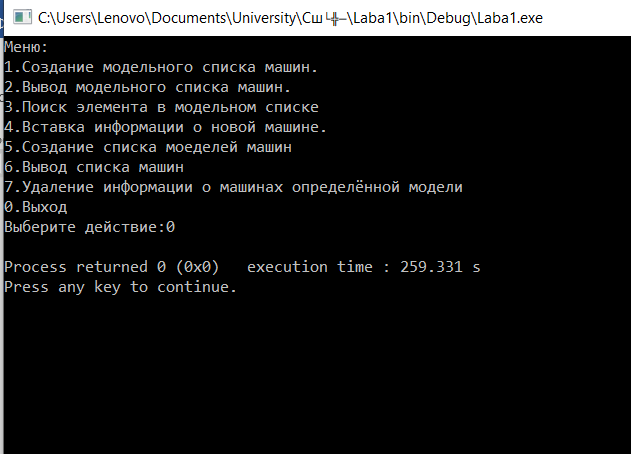






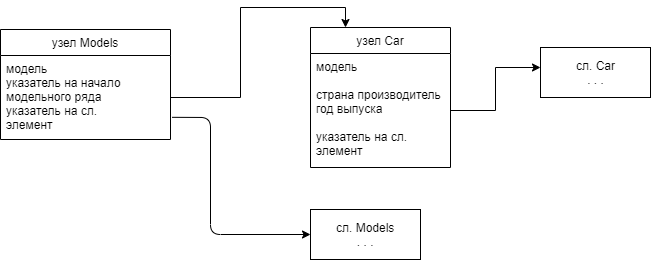




# Задание 2

## Структура узла



## Структура класса

class Car{

Node\* start; //указатель на начало списка

Node\* tail;

public:

Car() : start(0), tail(0) {};

~Car();

//добавление узла в список

void pushBack(Node &node);

//вывод односвязного списка

void outList();

//поиск элемента

Node\* findElement(string value);

//вставка элемента

void insertEl(Node &node);

//удаление элементов

void delEl(Node &node, string value, Node\* pointer);

};

## Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| №  теста | Эталон результата | Результат программы | Результат анализа  теста (+ или -) |
|
| 1 | Создание и вывод списка:  Type:Nissan  Country:jhg  Year:1998  0xb0a530  Type:Nissan  Country:hgf  Year:1980  0xb0a570  Type:BMW  Country:jhgf  Year:2000  0xb0a5b0  Type:BMW  Country:hgf  Year:2010  0  Поиск элемента:  ELEMENT:  Type:Nissan  Country:jhg  Year:1998  Type:Nissan  Country:hgf  Year:1980  Вставка элемента:  Type:Nissan  Country:jhg  Year:1998  0x81a6f0  **Type:Nissan**  **Country:uhgv**  **Year:1990**  **0x81a530**  Type:Nissan  Country:hgf  Year:1980  0x81a570  Type:BMW  Country:jhgf  Year:2000  0x81a5b0  Type:BMW  Country:hgf  Year:2010  Создание и вывод списка моделей:  Type:Nissan  Указатель на ряд:0x816d08  Type:BMW  Указатель на ряд:0x81a570  Удаление элементов:  Type:Nissan  Country:jhg  Year:1998  0x81a6f0  Type:Nissan  Country:uhgv  Year:1990  0x81a530  Type:Nissan  Country:hgf  Year:1980  0  Выход из меню:  Process returned 0 (0x0) execution time : 387.774 s  Press any key to continue. | Создание и вывод списка:  Type:Nissan  Country:jhg  Year:1998  0xb0a530  Type:Nissan  Country:hgf  Year:1980  0xb0a570  Type:BMW  Country:jhgf  Year:2000  0xb0a5b0  Type:BMW  Country:hgf  Year:2010  0  Поиск элемента:  ELEMENT:  Type:Nissan  Country:jhg  Year:1998  Type:Nissan  Country:hgf  Year:1980  Вставка элемента:  Type:Nissan  Country:jhg  Year:1998  0x81a6f0  **Type:Nissan**  **Country:uhgv**  **Year:1990**  **0x81a530**  Type:Nissan  Country:hgf  Year:1980  0x81a570  Type:BMW  Country:jhgf  Year:2000  0x81a5b0  Type:BMW  Country:hgf  Year:2010  Создание и вывод списка моделей:  Type:Nissan  Указатель на ряд:0x816d08  Type:BMW  Указатель на ряд:0x81a570  Удаление элементов:  Type:Nissan  Country:jhg  Year:1998  0x81a6f0  Type:Nissan  Country:uhgv  Year:1990  0x81a530  Type:Nissan  Country:hgf  Year:1980  0  Выход из меню:  Process returned 0 (0x0) execution time : 387.774 s  Press any key to continue. | **+** |

## Код основной программы с комментариями

Файл main.cpp

#include <iostream>

#include "Car.h"

#include <windows.h>

#include "Models.h"

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Car List;

Node node;

Element elem;

Models Spisk;

int var;

while(var){

system("cls");

cout<<"Меню:\n"<<"1.Создание модельного списка машин.\n2.Вывод модельного списка машин.\n3.Поиск элемента в модельном списке\n4.Вставка информации о новой машине.\n5.Создание списка моеделей машин\n6.Вывод списка машин\n7.Удаление информации о машинах определённой модели\n0.Выход\n";

cout<<"Выберите действие:";

cin>>var;

switch(var){

case 1:{

for(int i=0; i<4;i++){

node.input(node);

List.pushBack(node);

}

system("pause");

break;

}

case 2:{

//вывод списка машин

List.outList();

system("pause");

break;

}

case 3:{

//поиск элемента

cout<<"ELEMENT:\n";

Node\* Lst = List.findElement("Nissan");

while (Lst && Lst->type=="Nissan"){

cout<<"Type:"<<Lst->type<<"\nCountry:"<<Lst->country<<"\nYear:"<<Lst->year<<endl;

Lst = Lst->pNext;

}

system("pause");

break;

}

case 4:{

//вставка элемента

node.input(node);

List.insertEl(node);

system("pause");

break;

}

case 5:{

//создание списка моделей

for(int i=0; i<2;i++){

elem.input(elem);

Node \*p = List.findElement(elem.model);

Spisk.pushBack(p, elem);

}

system("pause");

break;

}

case 6:{

//вывод списка моделей

Spisk.outList();

system("pause");

break;

}

case 7:{

//удаление элементов

List.delEl(node,"BMW",Spisk.findElement("BMW"));

system("pause");

break;

}

case 0:{

exit(0);

system("pause");

break;

}

default:

cout<<"Выберите число от 0 до 7\n";

system("pause");

break;

}

}

return 0;

}

Файл Car.h

#ifndef CAR\_H

#define CAR\_H

#include <iostream>

#include <cstdlib>

using namespace std;

struct Node{

string type;

string country;

int year;

void input(Node &node);

Node\* pNext;

};

class Car{

Node\* start; //указатель на начало списка

Node\* tail;

public:

Car() : start(0), tail(0) {};

~Car();

//добавление узла в список

void pushBack(Node &node);

//вывод односвязного списка

void outList();

//поиск элемента

Node\* findElement(string value);

//вставка элемента

void insertEl(Node &node);

//удаление элементов

void delEl(Node &node, string value, Node\* pointer);

};

#endif // CAR\_H

Файл Car.cpp

#include "Car.h"

Car::~Car(){

Node\* cur = start;

while (cur){

cur = start->pNext;

delete start;

start = cur;

}

}

void Car::pushBack(Node &node){

Node\* cur = new Node;

cur->type=node.type;

cur->country=node.country;

cur->year=node.year;

cur->pNext = 0;

if(!start){

start = cur;

}else{

tail->pNext = cur;

}

tail=cur;

}

void Car::outList(){

Node \*cur = start;

while(cur){

cout<<"Type:"<<cur->type<<"\nCountry:"<<cur->country<<"\nYear:"<<cur->year<<endl;

cout<<cur->pNext<<endl;

cur = cur->pNext;

}

}

void Node::input(Node &node){

cout<<"Заполните информацию о машине\n";

cout<<"Модель:";

cin>>type;

cout<<"\nСтрана:";

cin>>country;

cout<<"\nГод:";

cin>>year;

cout<<endl;

}

Node\* Car::findElement(string value){

Node\* strt = start;

while(strt){

if(strt->type==value)

break;

strt = strt->pNext;

}

return strt;

}

void Car::delEl(Node &node, string value, Node\* pointer){

Node\* cur1 = new Node;

cur1 = pointer;

while(cur1->pNext){

if(cur1->pNext->type != value){

break;

}

cur1 = cur1->pNext;

}

if (pointer == start){

start = cur1->pNext;

return;

}

Node\* cur = new Node;

cur = start;

while (cur->pNext != pointer)

cur = cur->pNext;

cur->pNext = cur1->pNext;

delete cur1;

}

void Car::insertEl(Node &node){

if(!start){

Node\* cur = new Node;

cur->type=node.type;

cur->country=node.country;

cur->year=node.year;

return;

}

Node\* c = start;

while(c->pNext){

if (c->pNext->type==node.type){

if(c->pNext->year<node.year){

break;

}

}else if(c->type==node.type){

break;

}

c=c->pNext;

}

Node\* temp = new Node;

temp->type=node.type;

temp->country=node.country;

temp->year=node.year;

//вставка

temp->pNext=c->pNext;

c->pNext = temp;

}

Файл Models.h

#ifndef MODELS\_H

#define MODELS\_H

#include "Car.h"

using namespace std;

struct Element{

string model;

void input(Element &elem);

Element\* next;

Node\* beg;

};

class Models{

Element\* head; //указатель на начало списка

Element\* ending;

public:

Models() : head(0), ending(0) {};

~Models();

//добавление узла в список

void pushBack(Node \*p, Element &elem);

//вывод односвязного списка

void outList();

//поиск элемента

Node\* findElement(string value);

};

#endif // MODELS\_H

Файл Models.cpp

#include "Models.h"

Models::~Models(){

Element\* cur = head;

while (cur){

cur = head->next;

delete head;

head = cur;

}

}

void Models::pushBack(Node \*p, Element &elem){

Element\* cur = new Element;

cur->model=elem.model;

cur->beg = p;

cur->next = 0;

if(!head){

head = cur;

}else{

ending->next = cur;

}

ending=cur;

}

void Models::outList(){

Element \*cur = head;

while(cur){

cout<<"Type:"<<cur->model<<"\nУказатель на ряд:"<<cur->beg<<endl;

cur = cur->next;

}

}

void Element::input(Element &elem){

cout<<"Заполните информацию о машине\n";

cout<<"Модель:";

cin>>model;

}

Node\* Models::findElement(string value){

Element\* strt = head;

while(strt){

if(strt->model==value)

return strt->beg;

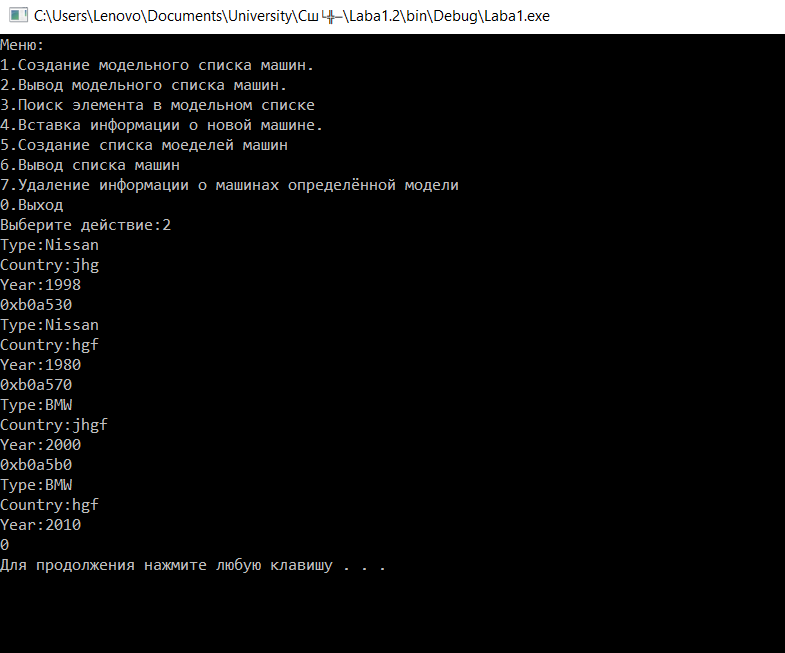
strt = strt->next;

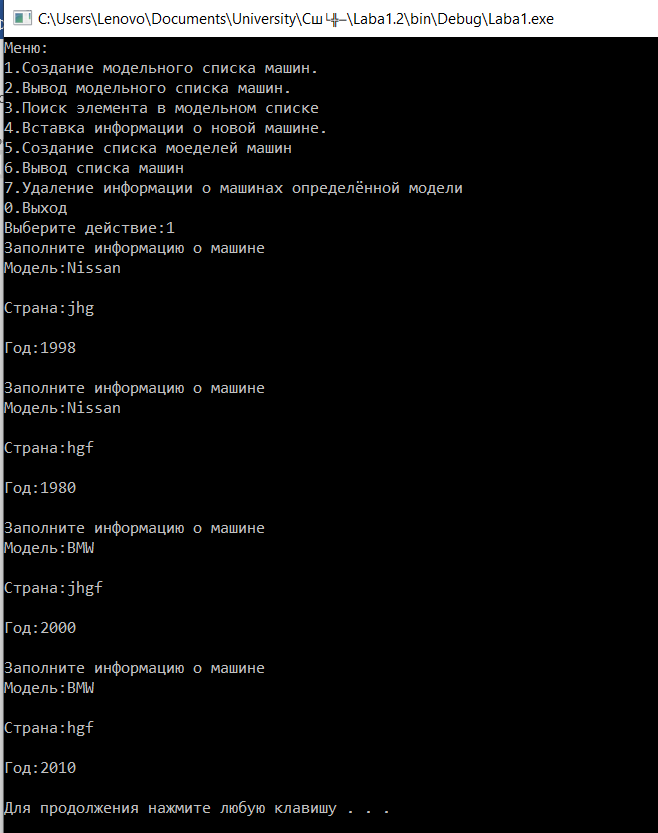
}

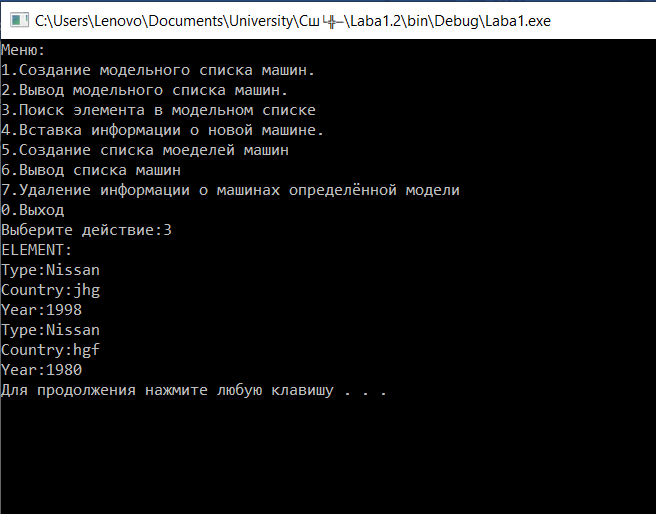
return 0;

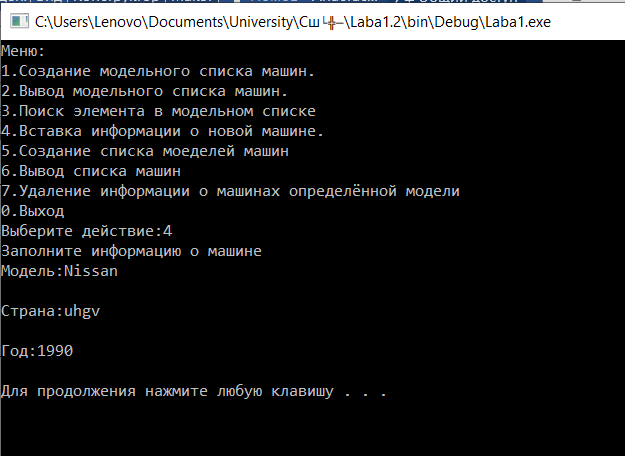
}

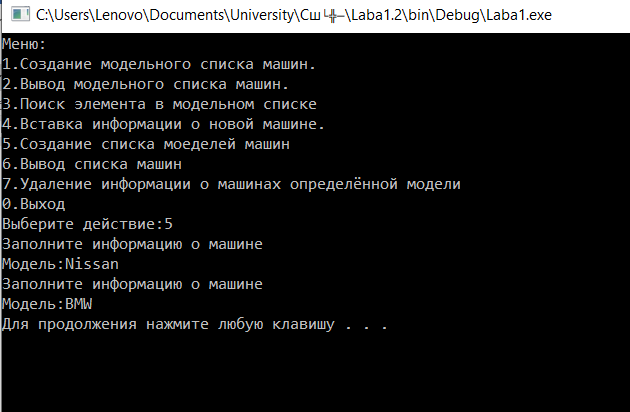
## Скриншоты выполнения тестов

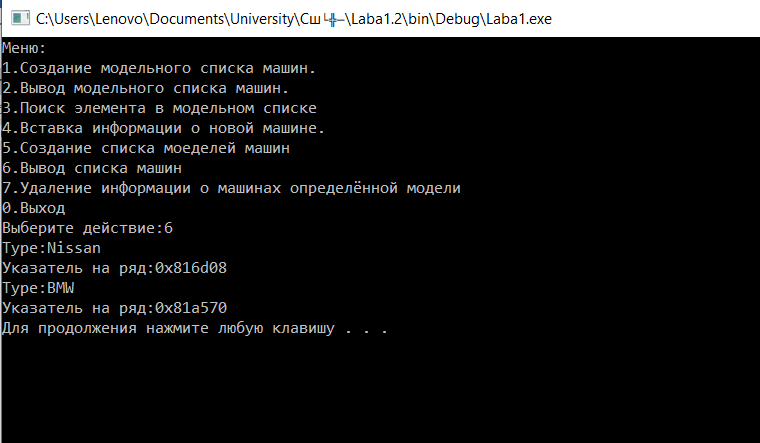


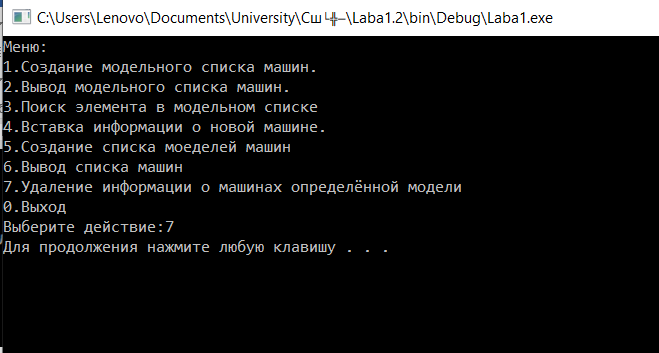


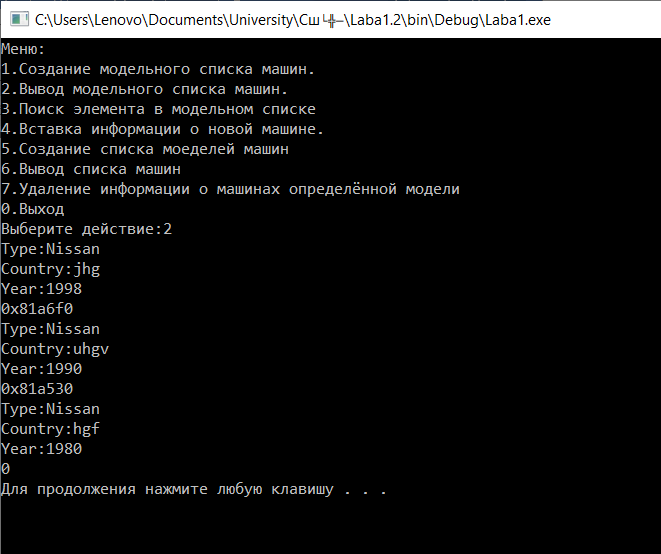


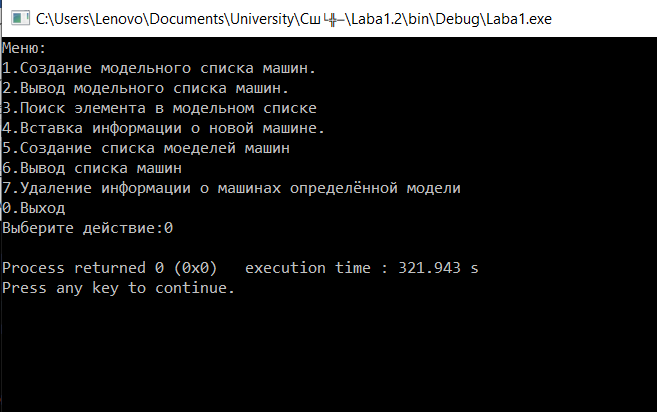












# Заключение

В данной лабораторной работе мы научились создавать односвязные элементарные списки посредством реализации на элементарных структурах и классах с использованием ООП на языке C++.

Для выполнения активно использовались указатели на объекты линейного списка с помощью которых, реализовывалась связь между элементами списка.

Полученные навыки и знания дают возможность выполнения последующих лабораторных работ и заданий, а так же создание собственных односвязных списков.