

Министерство образования Республики Беларусь

Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Программирование на языках высокого
уровня

К ЗАЩИТЕ ДОПУСТИТЬ

_____ Марзалюк А.В.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему

МОБИЛЬНЫЙ ОПЕРАТОР

БГУИР КР 1-40 02 01 329 ПЗ

Студент:

Шарай П.Ю.

Руководитель:

Марзалюк А.В.

МИНСК 2022

Учреждение образования
«Белорусский государственный университет информатики
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

_____ Б. В. Никульшин

(подпись)

_____ 2022г.

ЗАДАНИЕ

по курсовому проектированию

Студенту Шараю Петру Юрьевичу.

1. Тема проекта Информационная система автосервиса.
2. Срок сдачи студентом законченного проекта 15 декабря 2022 г.
3. Исходные данные к проекту Файлы: Services – содержит информацию о всех услугах, Autoservice – Автосервис, BodyRepair – кузовной ремонт, TireFitting – шиномонтаж, Diagnostics – диагностика, CarWash – автомойка User – информация о пользователях. Admin – информация об администраторах. Контейнер: STL – vector.
4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)
Введение. Содержание. 1. Постановка задачи 2. Структурное проектирование. 3. Функциональное проектирование. 4. Описание классов. 5. Разработка алгоритмов. 5.1 Разработка алгоритмов по шагам. 5.1 Разработка сх- алгоритмов. 6. Заключение. Список использованной литературы.

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. Диаграмма классов

2. Схема алгоритма метода void ShowUserServices()

3. Схема алгоритма метода bool UserLogin(User&)

6. Консультант по проекту Марзалюк А. В.

7. Дата выдачи задания 16 сентября 2022 г.

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

разделы 1, 2 к 10 октября 2022 г. – 20 %;

разделы 3, 4 к 10 ноября 2022 г. – 40 %;

раздел 5 к 10 декабря 2022 г. – 20 %;

оформление пояснительной записки и графического материала к 15 декабря 2022 г. – 20 %

Защита курсового проекта с 19 декабря 2022 г. по 30 декабря 2022 г.

РУКОВОДИТЕЛЬ

Марзалюк А. В.

(подпись)

Задание принял к исполнению

Шарай П. Ю.

(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
1 ОБЗОР ИСТОЧНИКОВ.....	6
2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ.....	7
3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ.....	8
4 РАЗРАБОТКА АЛГОРИТМОВ.....	14
5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ.....	16
ЗАКЛЮЧЕНИЕ.....	18
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	19
ПРИЛОЖЕНИЕ А.....	20
ПРИЛОЖЕНИЕ Б.....	21
ПРИЛОЖЕНИЕ В.....	22
ПРИЛОЖЕНИЕ Г.....	23
ПРИЛОЖЕНИЕ Д.....	24

ВВЕДЕНИЕ

Сфера услуг в настоящее время является одной из важных отраслей народного хозяйства призванной удовлетворять индивидуальные запросы и потребности населения страны в различных видах услуг. Сфера услуг как отрасль экономической деятельности представляет собой совокупность организаций, цель которых – оказание разнообразных платных услуг по индивидуальным заказам населения. Таким образом, сфера услуг решает важнейшие социально-экономические задачи и ее значение в жизни общества неуклонно возрастает. Одним из видов таких услуг являются услуги автосервиса.

Объектно-ориентированное программирование представляет собой парадигму, в которой программа предствляется в виде связанных между собой объектов и методов, описывающих их поведение. Данная парадигма вытекла из необходимости писать более расширяемый и поддерживаемый код.

C++ — компилируемый, статически типизируемый язык общего назначения. Преимущество языка C++ в том, что он жестко не привязан к объектно-ориентированному программированию и позволяет писать программы в различных стилях, например в процедурном. Данная особенность обусловлена тем, что язык является развитием языка C и, следовательно, полностью поддерживает его и позволяет работать его командами и конструкциями. Язык предоставляет и возможность работы на низком уровне (ассемблерные вставки, управление памятью).

Язык C++ имеет огромное количество стандартных функций и возможностей. Это и стандартная библиотека шаблонов (STL), потоки ввода-вывода, поддержка многопоточности, регулярные выражения.

Основными концепциями ООП являются инкапсуляция, полиморфизм и наследование.

Инкапсуляция – механизм сокрытия и защиты данных, исключающий влияние на них извне класса

Наследование – механизм, позволяющий расширять и дополнять созданные классы и объекты.

Полиморфизм – механизм, позволяющий сущностям изменять свое поведение в зависимости от класса.

Универсальность и гибкость языка C++ позволяют использовать его в самых разных сферах. Мощные инструменты языка необходимы в написании высокопроизводительных систем и программ. Возможность работы на низком уровне востребована в написании драйверов и частей операционных систем. Язык широко используется в игровой индустрии.

Используя все то, что было приведено выше, считаю, что язык C++ более чем подходит для написания курсового проекта.

1 ОБЗОР ИСТОЧНИКОВ

1.1 Обзор аналогов

Автосервисы возникли практически одновременно с широким распространением автомобилей, и в течении следующего времени они только расширились и предлагали все больше услуг и возможностей.

Wdrive, FMotors, PitShop - это одни из самых лучших и популярных автосервисов города Минск, услугами которых пользуется большое количество людей.

1.2 Постановка задачи

Задача курсовой работы заключается в следующем: создать программу с удобным пользовательским интерфейсом с пунктами меню. Программа должна хранить информацию, хорошо обрабатывать информацию. Также должна быть организована работа с файлами. Должны присутствовать основные принципы ООП. А также работа с контейнерами (свой контейнер и встроенный STL). Предусмотреть обработку исключительных ситуаций.

Основные задачи:

- Создание пользователя
- Работа с существующим пользователем
- Админ, которые имеет ряд своих функций
- Удаление и запись в файл
- Поиск пользователей по параметрам

После того, как пользователь заведёт свою учётную запись, ему становится доступен ряд функций: добавление и смена услуг, сохранение изменений, удаление аккаунта.

Админ может просмотреть всех пользователей, производить поиск по параметрам(номер телефона), также может удалить всех пользователей.

2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ

Для хранения данных в данной работе используются файлы с расширением txt. Они наиболее используемые файлы по умолчанию текстовые файлы, которые обеспечивают всем, чем требуется для работы с программой.

В программе можно выделить несколько основных элементов: блок взаимодействия пользователя с программой, блок ввода, вывода, чтения и записи в файл, блок исключений, блок констант.

Блок взаимодействия пользователя с программой отвечает за выбор пользователем определённых действий, например, смена тарифа ,пополнение баланса.

В блоке исключений происходит обработка исключительных ситуаций, например, проверка на ввод чисел, проверка успешного открытия файла для записи или чтения.

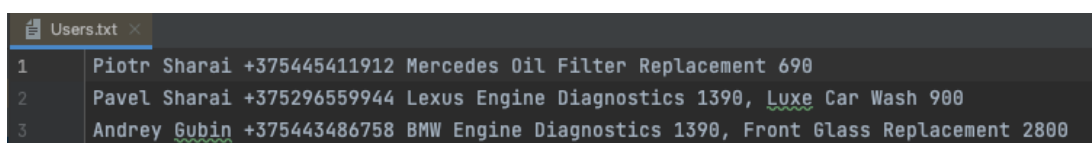
Блок констант используется для хранения всех констант, которые используются в программе.

В таблице 2.1 представлена структура данных, записи объекта в файл, хранящий информацию о клиентах.

Таблица 2.1 – Структура данных файла «Clients.txt»

Имя	Piotr
Фамилия	Sharai
Автомобиль	Mercedes
Номер телефона	+375445411912
Пароль	qwerty12345

На рисунке 2.1 представлен файл, который хранит в себе данные о клиентах компании.



```
1 Piotr Sharai +375445411912 Mercedes Oil Filter Replacement 690
2 Pavel Sharai +375296559944 Lexus Engine Diagnostics 1390, Luxe Car Wash 900
3 Andrey Gubin +375443486758 BMW Engine Diagnostics 1390, Front Glass Replacement 2800
```

Рисунок 2.1 – Файл с пользователями

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

3.1 Диаграмма классов

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов, методов, интерфейсов и взаимосвязей (отношений) между ними. Диаграммы позволяют быстро разобраться в сложных системах, а также служить справочными картами, позволяющими в них ориентироваться. Она отражает связи между классами и структурами.

Программная реализация представлена такими классами, как «User», «Admin», «Services», «Autoservice», «Engine», «Filters», «BodyRepair», «GlassWork», «BodyWork», «TireFittig», «RunFlat», «LowProfile», «Diagnostics», «EngineDiagnostics», «BrakeSystemDiagnostics», «CarWash», «Complex», «DryCleaning», «Container», «Exception», «IncorrectInput», «IncorrectFileOpen», «IncorrectPasswordLength».

Диаграммы классов программы «Автосервис» представлены в приложении А.

3.1 Описание классов

1. Класс User

```
class User {  
  
private:  
  
    string Name; - имя пользователя  
  
    string Surname; - фамилия пользователя  
  
    string PhoneNumber; - номер телефона пользователя  
  
    string ServiceName; - название услуги  
  
    string Password; - пароль  
  
public:  
  
    User() {} - конструктор  
  
    ~User(); - деструктор
```



```

void AddUser(User &Newuser) {}- создание клиента

void SetName() {}- сеттер имени

void SetSurname() {}- сеттер фамилии

void SetPhoneNumber() {} - сеттер номера

void SetPassword() {} - сеттер пароля

vector<User> AddUserToVector() {}- загрузка всех пользователей в
вектор

void SetUser() {} - сеттер пользователя

void ServicesList() {}- вывод информации об услугах

void setService() {} - сеттер услуги

string GetPassword() {} - геттер пароля

string GetName() {} - геттер имени

string GetSurname() {} - геттер фамилии


int GetAge() {} - геттер возраста

void GetCar() {} - геттер автомобиля

string GetService() {} - геттер услуги

string GetPhoneNumber() {}- геттер номера

void WriteFile(User NewUser) {} - запись клиента в файл

friend ostream& operator << (std::ostream &os, Client &p);
    - перегрузка оператора <<

friend istream& operator >> (std::istream& in, Client& p);

}; - перегрузка оператора >>

```

`void ForNewUser() {}` – метод для нового пользователя

`void UserUser() {}` – метод для существующего пользователя

`void DeleteUser(Client personDelete) {}` – удаление клиента

`void UserToFile(vector<User> users) {}` – запись клиента в файл

2. Класс Admin

```
class Admin:public User{
```

```
private:
```

```
string Password = "admin12345";
```

 – пароль для входа админа

```
public:
```

```
Admin() {}
```

 – конструктор

```
~Admin() {}
```

 – деструктор

```
void CheckAdmin() {}
```

 – проверка на правильность ввода пароля

```
void AdminFunc() {}
```

 – функционал админа

```
void FileAdmin() {}
```

 – чтение файла

```
void ReadFileAdminNumber() {}
```

 – чтение файла по номеру

```
void DeleteUsers() {}
```

 – удаление пользователя

```
};
```

3. Класс Services

```
class Services
```

```
{
```

```
protected:
```

```
string _name;
```

 – имя услуги

```
double _price;
```

 – цена услуги

```
public:
```

```
string virtual ServiceName() = 0; – виртуальный метод для  
возврата имени тарифа
```

```
double virtual ServicePrice() = 0; – виртуальный метод для  
возврата баланса
```

```
};
```

4. Классы «Autoservice», «BodyWork», «CarWash», «Diagnostics», «TireFitting»

Эти классы являются наследниками класса «Services» и имеют одинаковую структуру отличаются лишь своими индивидуальными характеристиками.

Рассмотрим на примере «Autoservice»:

```
class Autoservice : public Services  
{  
public:  
    void DisplayAutoservice() – вывод меню автосервиса  
  
Autoservice(){} – конструктор  
~Autoservice(){} – деструктор  
};
```

5. Класс Container

Свой контейнер List.

```
template <typename T>  
class List {  
public:  
    List(); – конструктор  
    ~List(); – деструктор  
    void push_back(T data); – занесения объекта в контейнер
```

```

int getSize(); – геттер размера контейнера

T& operator[](const int index); – перегрузка оператора [ ]

void clear(); – чистка листа

bool isEmpty(); – проверка на пустой контейнер

private :

template<typename T>

    class Node {

public:

Node* next; – указатель на следующий элемент

T data; – данные

Node(T data = T(), Node* next = nullptr) {} – конструктор с
параметрами
    };

int size; - переменная размера контейнера

Node<T>* first; – первый элемент контейнера
Node<T>* last; – последний элемент контейнера
};

```

6. Класс Exception

```

class Exception{

public:

string message; – хранит название ошибки

public:

Exception() {} – конструктор

Exception(string messg) {} – конструктор с параметром

~Exception() {} – деструктор

void Display() {} – вывод ошибки

};

```

7. Классы «IncorrectInput»,
«IncorrectFileopen», «IncorrectPasswordLength».

Эти классы являются наследниками *Exception* и отличаются лишь названием своих конструкторов. Рассмотрим на примере одного класса.

Класс «IncorrectInput»:

```
class IncorrectInput:public Exception {  
public:  
IncorrectInput(string message){} – конструктор с параметрами  
};
```

8. Классы «Engine», «Filters», «GlassWork», «BodyWork», «RunFlat»,
«LowProfile», «EngineDiagnostics», «BrakeSystemDiagnostics»,
«Complex», «DryCleaning»,

Эти классы являются наследниками класса имеют одинаковую структуру отличаются лишь своими индивидуальными характеристиками.

Рассмотрим на примере «Engine»:

```
class Engine :public AutoService{  
class EngineDiagnostics{  
public:  
EngineDiagnostics(){} – конструктор  
  
_name = "Engine Diagnostics";  
_price = 1390;  
}  
string ServiceName(){} – название услуги  
  
double ServicePrice(){} – цена услуги  
  
~EngineDiagnostics(){}– деструктор  
};
```

4 РАЗРАБОТКА АЛГОРИТМОВ

4.1 Алгоритмы по шагам

4.1.1 Алгоритм по шагам метода void User::NewUser() – создание нового пользователя.

Алгоритм по шагам метода void User::NewUser() – создание пользователя.

1. Начало.
2. Создается объект класса User.
3. Вызывается метод из Admin::NewUser().
4. После чего вызывается метод User::NewUser(), где в свою очередь вызываются все сеттеры для установления полей объекта.
5. User::SetName() – ввод имени
6. User::SetSurname() – ввод фамилии
7. User::SetPhoneNumber() – ввод номера телефона
8. User::SetCar() – ввод марки автомобиля
9. После того, как мы все ввели, мы возвращаемся в меню метода void User::NewUser()
10. Выбираем 3 пункт меню, чтобы занести данные в файл и сохранить.
11. 3 пункт меню вызывает метод User::File().
12. В методе User::File() открывается файл.
13. Файл закрыт
14. Конец алгоритма

4.1.2 Алгоритм по шагам метода void Admin::FileAdminPhoneNumber() – поиск пользователя по номеру телефона.

1. Начало
2. Входные данные: номер телефона пользователя для поиска(identif)
3. Открытие файла fin.open(“Users.txt”)
4. Проверка на открытие файла
5. Создаем объект User “a” для считывания из файла.
6. Запускаем цикл while(!(fin.eof()))
7. Помещаем в объект “a” пользователя из файла
8. У объекта “a” вызывает геттер номера телефона(a.GetPhoneNumber)
9. Сравниваем с identif(if(a.GetPhoneNumber == identif))
10. Если совпадает, то печатаем этого пользователя a.GetUser()
11. После того, как цикл закончился, закрываем файл fin.close()
12. Конец

4.2 Схемы алгоритмов

Схема алгоритма метода `void User::NewUser()` приведена в приложении В – создание нового пользователя.

Схема алгоритма метода `bool UserAuthentication(User&)` приведена в приложении Г – поиск пользователя по номеру телефона.

5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

При запуске программы мы попадаем в главное меню. Чтобы воспользоваться услугами в качестве пользователя требуется создать свой аккаунт и войти в него по номеру телефона и паролю, который был выбран при регистрации. Главное меню программы изображено на рис. 5.1.

```
Hello!
1 - enter as admin
2 - enter as user
0 - end program
```

Рисунок 5.1 – Главное меню программы

При выборе «2» мы попадаем в меню для пользователя рис. 5.2.

```
1 - log in
2 - new user
0 - main menu
```

Рисунок 5.2 – Главное меню для пользователя

Здесь мы можем войти в аккаунт или создать нового пользователя. После успешной регистрации клиент получает доступ к своему аккаунту, где он может просмотреть свои выбранные услуги. Также пользователь может удалить свой аккаунт.

После того, как мы правильно ввели данные, мы получаем информацию о пользователе и меню для пользователя рис. 5.3.

```
Enter your phone number: +375445411912
Enter your password: qwerty12345
*****
User: Piotr Sharai
Phone Number: +375445411912
Car: Mercedes
Services: Oil Filter Replacement 690, Luxe Car Wash 900, Engine Diagnostics 1390
*****
User menu:
1 - display user
2 - change phone number
3 - change car
4 - change services
5 - save changes and go to main menu
6 - delete account
```

Рисунок 5.3 – Меню для пользователя

В меню пункты можно выбрать цифрами и после чего нажать клавишу Enter. Пользователь может выбрать любой из пунктов меню, например, смена услуг - это пункт номер 4, на рисунке 5.4 показан результат работы.

```
Services Menu:
1 - Autoservice
2 - Body Work
3 - Tire Fitting
4 - Diagnostics
5 - Carwash
0 - Exit to main menu
1
Autoservice:
1 - Filters
2 - Engine
3 - BrakeSystem
0 - Exit to main menu
1
Filters:
Oil Filter Replacement: price 690
AirFilterReplacement: price 290
0 - Exit to main menu
Choose service you need:
1
Service Oil Filter Replacement chosen, price: 690
```

Рисунок 5.4 - Смена услуг

На рисунке 5.3 мы видим, что у нас было несколько услуг, на рисунке 5.4 мы меняем все эти услуги на услугу « Oil Filter Replacement». На рисунке 5.5 мы видим результат работы.

```
*****
User: Piotr Sharai
Phone Number: +375445411912
Car: Mercedes
Services: Oil Filter Replacement 690
*****
```

Рисунок 5.5 - Данные после смены услуг

Таким же образом мы можем выбирать и остальные пункты меню.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта были закреплены знания, полученные на лекционных и лабораторных занятиях. Также была проведена самостоятельная работа по изучению некоторого материала по различным источникам и литературе. В конечном результате была разработана программа «Автосервис».

Были разработаны функции ввода, хранения и редактирования информации о пользователях. Также реализован поиск пользователей по параметрам.

Данный проект может быть усовершенствован в следующих направлениях:

- Добавление поддержки баз данных для хранения информации.
- Разработка графического интерфейса.
- Разработка мобильного приложения и web-версии.
- Добавление локализации для различных рынков.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

- [1] Герберт, Ш. Самоучитель С++/Ш. Герберт. Санкт-Петербург 2003г.
- [2] Дейтел, Х.М. Как программировать на С++ / Х.М. Дейтел, П.Д. Дейтел; пер. с англ. – М. : Бином, 2007. – 1152 с..
- [3] Страуструп, Б. Язык программирования С++ / Б. Страуструп; специальное издание. Пер. с англ. – СПб. : ВHV, 2008. – 1098 с.
- [4] Элджер, Дж. С++: библиотека программиста / Дж. Элджер. – СПб. : Питер, 2001. – с.
- [5] Объектно-ориентированное программирование в С++/ Роберт Лафоре г. пер. с англ. – Санкт-Петербург, 2019. – 1152 с.

ПРИЛОЖЕНИЕ А
(обязательное)

Диаграмма классов

ПРИЛОЖЕНИЕ Б
(обязательное)

Схема алгоритма метода void ShowSpecificCarSessions()

ПРИЛОЖЕНИЕ В
(обязательное)

Схема алгоритма метода bool UserAuthentication(User&)

ПРИЛОЖЕНИЕ Г
(обязательное)

Листинг кода программы

ПРИЛОЖЕНИЕ Д
(обязательное)

Ведомость документов