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 ск 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 %;

разделы 5, 6 ,7 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.*

Задание принял к исполнению_____ *Шарай П. Ю.*

СОДЕРЖАНИЕ

Оглавление пустое, так как стили абзацев, выбранные для отображения в оглавлении, не используются в документе.

ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ СОКРАЩНИЙ

ID – идентификационный номер.
ОС – операционная система.
ООП – Объектно-ориентированное программирование.
STL – (*Standard Template Library*) – Библиотека стандартных шаблонов.

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 стандартых 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 плияние на них извне класса

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

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

Универсальность и гибкость языка C++ позволяют использовать его в самых разных сферах. Мощные инструменты языка необходими 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 Анализ аналогов программного средства

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

2. Постановка задачи

Реализация проекта будет использована интегрированныя среда разработки CLion и система сборки проектов CMake.

В программе будут реализованы функции:

- Выбор нужной услуги.
- Регистрация аккаунта пользователя.
- Возможность редактирования информации.
- Возможность просмотра всех данных программы администратором.
- ОС - macOS.

Для реализации данного программного обеспечения используется объектно-ориентированный язык программирования C++ (стандарт C++14).

2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ

2.1 Структура программы

В структуре программы можно выделить несколько основных модулей: модуль интерфейса, модуль обработки ошибок и модуль функционирования программы, блок работы с базой данных автомобилей.

Блок интерфейса предназначен для взаимодействия пользователя с программой. Пользователь может быть как рядовым потребителем, так и администратором.

Модуль функционирования программы включает в себя все те механизмы, позволяющие комфортно взаимодействать с программой и выполнять требуемые задачи.

Модуль обработки ошибок предназначен для борьбы с возможными исключениями и обеспечивает надежную работу всей программы.

Модуль работы с базой данных автомобилей предназначен для сериализации объектов различных типов автомобилей, поскольку в языке C++ прямая запись объектов в файл может в определенных ситуациях привести к ошибкам чтения и потери доступа к части информации.

2.2 Структура файлов программы

Вся информация будет хранится в бинарных файлах.

Файлы, содержащие информацию о различных типах услуг имеют крайне схожую структуру, запись и чтение из них происходит через класс Services, служащий для корректного считывания и записи в файлы.

Данные файлы имеют названия Autoservice.bin (содержит информацию об услугах автосервиса), BodyRepair.bin (о кузовном ремонте), TireFitting.bin (о шиномонтаже), Diagnostics.bin (о диагностике), CarWash.bin (об автомойке).

Далее рассмотрим их структуру и поля. Некоторые поля могут содержатся только в некоторых файлах, если об этом ничего не сказано, то следует считать, что поле содержится во всех файлах.

– string _name. Название услуги .

– double _price . Цена услуги.

Для реализации функций регистрации и входа в аккаунт были созданны файлы User.bin, Admin.bin.

Файлы User.bin и Admin.bin содержат информацию о пользователях и администраторах соответственно. Файлы User.bin и Admin.bin соответствующие объекты классов, содержащие следующие общие поля:

– string Car. Марка автомобиля.

– string name и string surname. Имя и фамилия пользователя.

– string PhoneNumber. Номер телефона пользователя.

Файл User.bin содержит так же свои собственные поля:
– wchar_t userLogin[30].
– wchar_t userPassword[30].
– int id. ID пользователя, которому принадлежат логин и пароль.
– string PhoneNumber. Номер телефона пользователя.

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

3.1 Диаграмма классов

Диаграмма классов является важнейшим элементом любого проекта. Она позволяет графически представить связь классов между собой, избегая просмотра кода всей программы, который может быть очень объемным.

Диаграмма классов программы представлена в приложении А.

3.2 Обзор используемых классов

3.2.1 Класс Services и его наследники

Поскольку главной услугой автосервиса, очевидно, является ремонт автомобилей, то в программе реализовано 6 классов. Базовым классом для всех остальных является класс Services, от него наследуются классы Autoservice, BodyRepair, TireFittig, Diagnostics и CarWash, которые представляют категории предоставляемых услуг. В свою очередь от класса Autoservice наследуются классы Filters, Engine и BrakeSystem, которые представляют услуги по замене фильтров, ремонту двигаеля и тормозной системы. От класса BodyRepair наследуются классы GlassWork и BodyWork. От класса TireFittig наследуются классы LowProfile и RunFlat. От класса Diagnostics наследуются классы EngineDiagnostics и BrakeSystemDiagnostics. От класса CarWash наследуются классы Complex и DryCleaning.

Методы класса Services:
– Services(). Конструктор.
– ~Services(). Деструктор.
– void DeleteService(). Удаляет услугу.
– string* GetCarName(). Возвращает марку.
– void DisplayAutoservice(). Отображение услуг автосервиса.
– void DisplayBodyWork(). Отображение услуг кузовного ремонта.
– void DisplayTireFitting(). Отображение услуг шиномонтажа.
– void DisplayDiagnostics(). Отображение услуг диагностики.
– void DisplayCarWash(). Отображение услуг автомойки.
–void ExitToMenu(). Выход в меню.

Поля класса Services:
– int _choice. Выбор услуги.
– string _name. Название услуги.
– double _price. Цена услуги.

Методы класса Autoservice:
– Autoservice(). Конструктор.
– ~ Autoservice(). Деструктор.
– void DisplayEngine().
– void DisplayFilters().
– void DisplayBrakeSystem().
Методы класса BodyRepair:
– BodyRepair(). Конструктор.
– ~ BodyRepair(). Деструктор.
– void DisplayGlassWork(). Отображение услуг ремонта стёкол.
– void DisplayBodyWork(). Отображение услуг кузовного ремонта.
Методы класса TireFitting:
– TireFitting(). Конструктор.
– ~TireFitting(). Деструктор.
– void DisplayLowProfile(). Отображение услуг для низкопрофильной резины.
– void DisplayRunFlat(). Отображение услуг для RunFlat резины.
Поля класса TireFitting:
– int TireSize. Радиус шин.
Методы класса Diagnostics:
– Diagnostics(). Конструктор.
– ~Diagnostics(). Деструктор.
– void DisplayEngineDiagnostics(). Отображение услуги диагностики двигателя.
– void DisplayBrakeSystemDiagnostics(). Отображение услуги диагностики тормозной системы.
Методы класса CarWash:
– CarWash()
– ~CarWash()
– void DisplayComplex(). Отображение услуг комплексной автомойки.
– void DisplayDryCleaning(). Отображение услуг химчистки.

3.2.2 Класс User и его наследники

Класс User и его наследник Admin предназначен для описания аккаунтов людей, которые взаимодействуют с программой.
Поля класса User:
– string _login. Логин пользователя.
– string _password. Пароль пользователя.
– string name. Имя пользователя.
– string surname. Фамилия пользователя.
– string PhoneNumber. Номер телефона пользователя.
– string Car. Марка автомобиля пользователя.

Методы класса User:
– User(). Конструктор.
– ~User(). Деструктор.
– void SetLogin(). Задаёт логин пользователя.
– void SetPassword(). Задаёт пароль пользователя.
– void SetName(). Задаёт имя пользователя.
– void SetSurname(). Задаёт фамилию пользователя.
– void SetCar(). Задаёт автомобиль пользователя.
– void SetPhoneNUmber(). Задаёт номер телефона пользователя.
– string GetLogin(). Возвращает логин пользователя.
– string GetPassword(). Возвращает пароль пользователя
– string GetName(). Возвращает имя пользователя.
– string GetSurname(). Возвращает фамилию пользователя.
– string GetCar(). Возвращает автомобиль пользователя.
– string GetPhoneNUmber(). Возвращает номер телефона пользователя.
Методы класса Admin:
– Admin(). Конструктор.
– ~Admin(). Деструктор.
Поля класса Admin:
– string password. Пароль администратора.
– Методы класса Admin:
– Admin(). Конструктор.
– – ~Admin(). Деструктор.
– void UserFile(). Просмотр данных всех пользователей.

3.2.3 Класс Exception

Класс Exception реализовывает работу с исключительными ситуациями. От класса Exception наследуются классы IncorrectInput, IncorrectPasswordLength, IncorrectFileOpen, IncorrectStringInput.

Поля класса Exception:
– string ErrorMessage. Сообщение об ошибке.
Методы класса Exception:
– Exception(). Конструктор.
– ~ Exception(). Деструктор.
– string Exception(messg). Метод сообщения об ошибке.
– void DisplayMessage(). Отобразить сообщение об ошибке.
Методы класса IncorrectInput:
– string IncorrectInput(). Неправильный ввод.
Методы класса IncorrectPasswordLength:
– string IncorrectPasswordLength(). Неправильна длина пароля.
Методы класса IncorrectFileOpen:
– string IncorrectFileOpen(). Неправильное открытие файла.
Методы класса IncorrectStringInput:
– string IncorrectStringInput(). Неправильный ввод строки.

4 РАЗРАБОТКА АЛГОРИТМОВ

4.1 Алгоритмы по шагам

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

1. Начало.
2. Входные данные:

userID – ID заказчика;

carID – ID автомобиля;

costPerDay – стоимость аренды автомобиля на день;

user – объект класса User, содержит информацию о заказчике.

3. Проверяем, не равна ли costPerDay нулю, поскольку стоимость 0 означает, что автомобиль удален. Если не равна, то продолжаем, если равна то выполнение метода прерывается и возвращается false.

4. Полям this->userID и this->carID присваиваются значения carID и userID.

5. Пользователь вводит дату начала заказа.

6. Введенная дата преобразуется в юлианскую дату, представляющую собой число дней с 1 января 4713 года до н. э.

7. Если дата введена неверно, то нужно ее ввести еще раз.

8. Аренда не может начаться больше, чем через 3 дня с текущей даты, поэтому если это условие не выполняется, то выполнение метода прерывается.

9. Пользователь вводит дату окончания заказа и повторяются пункты 6 и 7 алгоритма.

10. Расчет количества дней, на которое арендуется автомобиль. Если разница между началом и окончанием заказа 0, то стоимость будет браться за 1 день.

11. Расчет стоимости. Реализовано снижение стоимости, если заказ на много дней, если заказ на неделю, то плата берется за 6 дней, если на месяц, то за 26 дней.

12. Проверяется, достаточно ли у пользователя средств, если нет, то выполнение метода прерывается и возвращается false.

13. К расчитанной стоимости заказа применяются скидки, в зависимости от статуса пользователя, чем больше он тратит, тем больше у него скидка и выше статус.

14. Запрашиваем подтверждение заказа.

15. Если пользователь отказался, то прерываем выполнение метода и возвращаем false, иначе отнимаем с баланса пользователя стоимость заказа и заносим арендованный автомобиль в список недоступных для заказа.

16. Записываем информацию о заказе в файл Session.bin.

17. Возвращаем true и завершаем выполнение метода.

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

1. Начало.

2. Входные данные:

3. Создаем новый объект NewUser типа User.

4. Открываем для чтения файл User.bin и считываем информацию из него в объект NewUserd.

5. Закрываем файл.

6. Пользователь вводит новый логин. Происходит обработка ошибок, если пользователь ввел пустую строку, либо строку с пробелами.

7. Проверяем доступность логина методом IncorrectInput(message).

9. Открываем файл User.bin для чтения и создаем временный файл tmp.bin для записи.

10. Создаем временный объект tmp типа Userd, служащего в качестве буфера.

11. Считываем из файла User.bin объекты типа User используя объект tmp.

12. Проверяем, равны ли ID объекта tmp и ID объекта NewUser. Если совпадают, то в файл tmp.bin записывается объект NewUser, иначе tmp.

14. Закрываем файл User.bin и удаляем его.

15. Закрываем файл tmp.bin и переименовываем его в User.bin.

16. Конец.

4.2 Схемы алгоритмов

Схема алгоритма метода void User::NewUser() приведена в приложении Б. – создаёт нового пользователя.

Схема алгоритма метода bool UserAuthentication(User&) приведена в приложении В. – аутентификация пользователя в программе.

5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Для использования программы в качестве пользователя необходимо сначала зарегистрироваться. В процессе регистрации задаются логин и пароль. Они не должны содержать пробелы и не могут являться пустыми строками. Логин и пароль можно сменить в любой момент. На рисунке 5.1 главное меню программы.

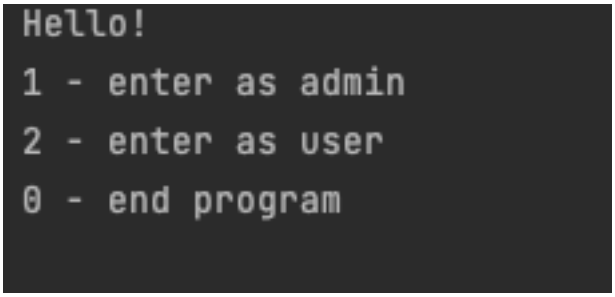

Рисунок 5.1 – Главное меню программы

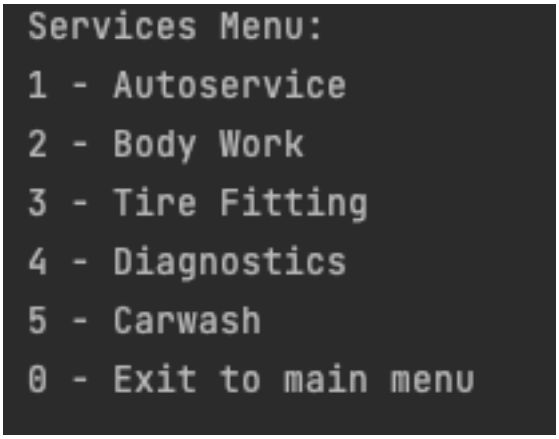

Рисунок 5.2 – Главное меню пользователя

Все пункты в меню программы выбираются с помощью цифр и нажатия клавиши Enter. Неверный ввод обрабатывается.

Пользователь выбирает тип желаемой услуги и выбирает сам услугу из соответствующего списка. На рисунке 5.3 представлен пример выбора услуги.

```
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 5.3 – Оформление заказа

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта было изучено и успешно использовано большое количество текстовой и графической информации, в результате чего стало возможным проектирование программного продукта, его тестирование и устранение ошибок.

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 C++/Ш. Герберт. Санкт-Петербург 2003г.

– [2] Дейтел, Х.М. Как программировать на C++ / Х.М. Дейтел, П.Д. Дейтел; пер. с англ. – М. : Бином, 2007. – 1152 с..

– [3] Страуструп, Б. Язык программирования C++ / Б. Страуструп; специальное издание. Пер. с англ. – СПб. : BHV, 2008. – 1098 с.

– [4] Элджер, Дж. C++: библиотека программиста / Дж. Элджер. – СПб. : Питер, 2001. – с.

– [5] Объектно-ориентированное программирование в C++/ 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