Министерство просвещения ПМР

ГОУ СПО «Тираспольский техникум информатики и права»

**КУРСОВАЯ РАБОТА**

**по дисциплине:** Разработка, внедрение и адаптация программного обеспечение отраслевой направленности

**на тему:** Разработка системы классов, описывающих автобусный парк.

**Выполнил обучающаяся**

Балинский Александр

**Специальность 2.09.02.05**

Прикладная информатика

в экономике.

**Руководитель**

преподаватель информационных

дисциплин

Балашова Юлия Владимировна

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, подпись руководителя)

Тирасполь, 2021

**Оглавление**

Введение **3.**

Исследование предметной области **4.**

1.1 История автобусов. **4.**

1.2 Описание предметной области 5

1.3 Анализ существующих разработок 6

1.4. Достоинство и недостатки существующих разработок…….....……8

1.5 Общая постановка задачи **.**10

2 Реализация программного обеспечения 11

2.1 Требования к программе 11

2.2 Руководство программиста 12

2.3 Руководство пользователя 29

Заключение 36

Список литературы 37

**Введение**

В настоящее время развитие информационных технологий позволяет избавить человека от необходимости выполнения огромного количества простых и сложных расчетов, что позволяет повысить точность и продуктивность труда. Поэтому возникает потребность в написании программного продукта с использованием информационных систем, которые решили бы существующие проблемы производительности.

В первую очередь, автоматизация автопарков связана с внедрением систем. Они позволяют решать следующие важные задачи:

* планировать маршруты и оптимизировать их;
* следить за тем, чтобы соблюдались графики движения;
* отслеживать автомобили в реальном времени;
* контролировать расход топлива;
* информировать клиентов о местонахождении груза и так далее.

Отдельно решаются вопросы удаленного взаимодействия с водителями, отслеживания их действий и обеспечения безопасности. Все более важным становится направление, связанное с разработкой и внедрением мобильных приложений по заказу транспорта. Это тоже часть автоматизации, потому что работа диспетчеров во многом заменяется алгоритмами и действиями пользователей.

Объектом исследования является автобусный парк

Целью курсового проекта является разработка автоматизации автобусного парка.

1. **Исследование предметной области**

**1.1 История автобусов**

Ежедневно множество людей по всему миру использует для поездок один из самых распространенных видов общественного транспорта – автобус. История этого транспортного средства, которое ежедневно перевозит миллионы жителей земли началась более двухсот лет назад. Самый первый автобус был сконструирован в 1801 году. На потеху толпе по улицам английского города Кэмборна проехала невиданная машина на паровом двигателе. Его изобретатель – британец Ричард Тревитик – сконструировал и первый английский паровоз. Транспортное средство вмещало до 8 пассажиров, а сзади на подножке стоял кочегар. Новинка, выбрасывая в воздух клубы черного дыма, уверенно завоевывала право на существование. 12 апреля 1903 г. в Лондоне состоялся первый в истории рейс автобуса, как городского общественного транспорта.

Немцы развили идеи англичан. В 1895 году германский завод «Бенц» представил миру первый автобус с двигателем внутреннего сгорания. Внешне он был похож скорее на увеличенную извозчичью коляску, вмещал в салоне 8 пассажиров и набирал до 13 км/ч. Такие машины использовали для сообщения между тремя немецкими городами: Дойц, Зиген и Нетфен.

До царской России автобус с двигателем внутреннего сгорания дошел спустя 7 лет. Петербургский завод «Фрезе» в 1903 г. сконструировал даже более прогрессивный экземпляр, чем немецкий. Он мог вместить уже 10 человек и разогнаться до 16 км/ч, однако имел существенный недостаток: кабину открытого типа. Поэтому в непогоду поездки либо отменялись, либо были крайне неприятными для водителя и пассажиров.

Автобус пока еще не носил глобального значения в роли общественного транспорта, но постепенное повышение уровня технического потенциала и комфортабельности не могло не наделить его основной задачей – регулярно перевозить туда и обратно потоки людей. 12 апреля 1903 г. в Лондоне состоялся первый в истории рейс автобуса, как городского общественного транспорта. В России это судьбоносное событие случилось немного позже.

Летом 1907 г. в Архангельск из Германии был доставлен автобус марки «NAG» вместимостью 26 человек, который незамедлительно обрел колоссальную востребованность и существенно облегчил передвижение по городу. К ноябрю того же года пассажирские автобусы появились в столице – Санкт-Петербурге, а в августе 1908 г. дошли и до Москвы. Но стабильно наладилось движение здесь только после революции и обретения Москвой статуса столицы СССР. В 1924 г. был организован первый регулярный автобусный маршрут между Тверской заставой и Каланчевской площадью.

С тех пор автобус непрестанно преображался. Его вместимость становилась 50, 100, 200 человек. Сейчас строятся специальные транспортные средства для доставки людей в аэропорты, проведения экскурсий, автобусных туров между странами. Некоторые из них достигают 40 метров в длину и вмещают до 500 пассажиров!

## Описание предметной области

Актуальность темы исследования обосновано тем, что упрощает продажу билетов, позволяет узнать информацию о водителях и автобусах автопарка. С возможностью коррекции данных.

Целью работы является разработка автоматизации процесса работы автопарка.

В качестве предметной области выбрано обработка учёта списков водителей, маршрутов и транспорта.

Автобусный парк содержит определенное количество подвижных единиц. На маршруты выходят иногда одни и те же автобусы. Автобусы из автобусного парка ездят по определённым, заранее спроектированным маршрутам. Все автобусы должны быть в исправном стоянии. За техническим состоянием автобусов наблюдают тех. служащие или механики.

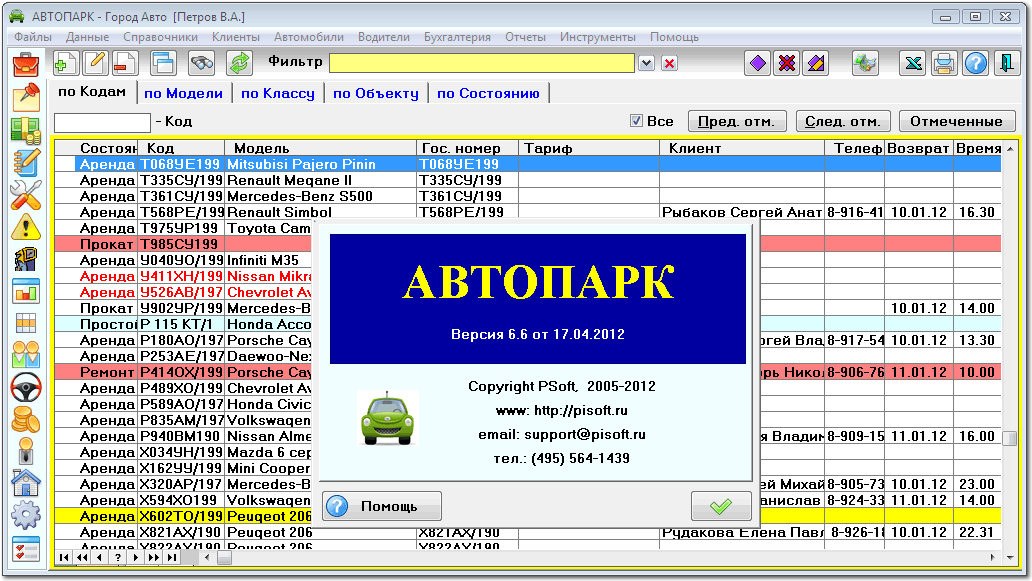
Водители отвечают за работоспособность автобуса, закрепленного за ними. Каждому водителю администрация автобусного парка должна предоставить автобус - транспортное средство, при помощи которого водитель должен исполнять свои служебные обязанности. Каждому водителю назначается конкретный автобус, за которым водитель обязан следить, но на практике, это не всегда так. В случае поломки автобуса, водителю могут предоставить другой автобус, в то время как, в случае не выхода на работу водителя, на его автобусе, по распоряжение начальства, может ездить другой водитель.

Диспетчер закрепляет за автобусами маршруты, определяет в какую смену выходит водитель и на каком автобусе. Указывает маршрут и время возвращения в парк. Каждый маршрут имеет пункт отправления и назначения.

**1.3. Анализ существующих разработок**

После изучение предметной области были найдены следующие аналоги программ, программа “Автопарк” и “ТМ корпоративные поездки”.

Программа "Автопарк" предназначена для комплексной автоматизации авто предприятия. Программа позволяет вести учет всего, что связано с автопарком: ТО, ремонты, ДТП, ГСМ, резина, комплектующие, документы на автомобили, путевые листы; прием и обработку поступающих заказов, учет работы и взаиморасчеты с водителями, учет работы операторов и диспетчеров. Она ведет учет договоров проката, платежей и взаиморасчетов с клиентами, историю каждого автомобиля, формирует все необходимые для работы документы и отчеты.



(Рис. 1) Процесс работы программы.

Программа ТМ: Корпоративные поездки — современное решение, которое позволяет произвести автоматизацию собственного автопарка предприятия. Благодаря цифровизации процесса заказа можно в реальном времени собирать информацию о положении автомобилей и сократить холостой пробег. Внедрение программного комплекса позволяет оптимизировать работу и осуществлять управление корпоративным автопарком эффективно



(Рис 2.) ТМ корпоративные поездки.

**1.4. Достоинство и недостатки существующих разработок**

Внешний вид и содержание печатаемых документов может настраиваться конечными пользователями. Предусмотрено резервирование автомобилей; имеется наглядный график изменения состояний автомобилей (шахматка). Гибкая настройка тарифов: тарифы, зависящие от числа дней проката, почасовые тарифы, тарифы за перепрокат, тарифы выходного дня и пр. Имеется поддержка тарифных планов для корпоративных клиентов, автоматизация взаиморасчетов с клиентами. Программа имеет развитую систему разделения доступа сотрудников к различным функциям и отчетам.

Программа Автопарк является сетевой и позволят работать с общей базой данных одновременно с нескольких рабочих мест: операторы, диспетчера, бухгалтеры, руководители. Возможен доступ к базе данных программы из удаленных филиалов через интернет.

|  |  |
| --- | --- |
|  | -Модели автомобилей |
|  | -Состояния автомобилей |
|  | -График занятости автомобилей |
|  | -Учет ремонтов, ТО, ДТП автомобилей |
|  | -Учет путевых листов |
|  | -Учет оплаты ГСМ |
|  | -Водители**,**Графики работы водителей |
|  | -Взаиморасчеты с водителями |
|  | -Тарифы**,**Тарифные планы |
|  | -**Клиенты** |
|  | --Категории клиентов |
|  | -Учет контактов с клиентами |
|  | -Работа с корпоративными клиентами |
|  | -Улицы и районы |
|  | -Дополнительные услуги |
|  | -Сотрудники и разделение доступа |
|  | -Категории сотрудников |
|  | -Журнал операций |
|  | -Фирмы |
| - | Договора проката |
|  | -Резервирование автомобилей |
| - | Кассовая книга |
|  | -Задания сотрудников |
| - | Отчеты по автомобилям |
| - | Отчеты по договорам |
| - | **Отчеты по кассе** |
| - | Настройка печатаемых документов |
| - | **Параметры настройки программы** |
| - | Использование фильтров |
| - | Автоматическое резервирование данных  Программа ТМ корпоративные поездки имеет следующие преимущества:  - Мобильное приложение для сотрудников;  - Мобильное приложение для водителей;  - Кабинет организации;  - GPS-трекинг;  - Аналитическая платформа;  - Заказ поездок;  -Кабинет клиента;  -Карты и поиск адресов;  -Лимит на поездки;  -Прием звонков роботом;  -Рабочее место ТМ;  -Управление составом автопарка. |

**1.5. Общая постановка задачи**

Основной задачей данной работы является реализация программы обработки информации автобусного парка.Программа должна быть простой, удобной для использования неопытных пользователей компьютера.

В программе должны быть реализованы следующие основные функции:

- Создание, удаление, редактирование данных;

-Вывод данных;

-Сохранение информации;

-Редактирование информации о водителях;

-Расчет и продажа билетов;

-Редактирование информации о автобусах;

-Сброс данных программы.

# 2.Исследование предметной области

## 2.1Требования к программе

Минимальная конфигурация аппаратного и программного обеспечения компьютера, на котором будет использоваться программа:

- процессор мощностью *1.0 ГГц* и выше;

- видеокарта совместимый с *DirectX 9;*

- оперативная память 512 МБ;

- операционная система *Windows 7* и выше.

Так же программа должна нормально функционировать при бесперебойной работе компьютера. При возникновении сбоя в работе аппаратуры должно производится восстановлении работоспособности программы, а также ее резервное сохранение данных.

Программный продукт разработан с помощью интегрированной среды разработки *Visual Studio 2019* на языке программирования C# на целевую платформу *.Net Framework 4*.

Конфигурация аппаратного и программного обеспечения компьютера, на котором велась разработка программного продукта:

- процессор AMD *Ryzen 3 3200g* с тактовой частотой 3.6 Ггц;

- оперативная память 8 ГБ типа *DDR4*;

- видеокарта Radeon RX 580, видеопамять 8 ГБ;

- операционная система *Windows 10 LTSC 64* бит.

## 

## 2.2. Руководство программиста

Программный продукт представляет собой консольное приложение для упрощения работы автобусного парка. В разработанном программном продукте реализовано множество алгоритмов. Наиболее важные из них кратко описаны в данной главе.

Класс ***Voditel*** является программным представлением работников автопарка. Данный класс содержит в себе несколько полей: *Fio*- ФИО участника, *contact-*Номер телефона сотрудника, *ZP-*заработная плата сотрудника.

Класс ***Machine*** является программным представлением автобусов в автопарке. Данный класс содержит в себе несколько полей: *marka-*марка автобуса, *model*-марка автобуса, *koll*-количество автобусов в автопарке.

Класс ***Marshrut*** является программным представление маршрутов.

Класс ***Program*** – главный класс приложения, в котором определена точка входа для приложения. Класс имеет несколько методов, рассмотрим каждый метод:

Метод ***Main*** - это главный метод программы, ее входная точка где и реализуется логика работы команд при помощи оператора выбора *switch*.

**Исходный код**

using System;

using System.IO;

using System.Linq;

using System.Collections.Generic;

using System.Threading.Tasks;

namespace ConsoleApp9

{

class Voditel

{

struct Voditel1

{

public string fio;

public string spec;

public string Contact;

public double ZP;

public void DisplayInfo()

{

Console.WriteLine("ФИО: {0}", this.fio);

Console.WriteLine("Специальность: {0}", this.spec);

Console.WriteLine("Контактные данные: {0}", this.Contact);

Console.WriteLine("Зарплата: {0}\n", this.ZP);

}

}

internal void Ar()

{

Console.Clear();

Voditel1[] arr = new Voditel1[6];

arr[0] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

arr[1] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

arr[2] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

arr[3] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

arr[4] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

arr[5] = new Voditel1 { fio = "Введите данные", spec = "Введите данные", Contact = "Введите данные", ZP = 0 };

do

{

Console.WriteLine("Выберите операцию:");

Console.WriteLine("1 - Изменить/Добавить");

Console.WriteLine("2 - Показать");

Console.WriteLine("4 - Сброс");

Console.WriteLine("5 - Назад");

Console.Write("Команда: ");

int v = Convert.ToInt32(Console.ReadLine());

if (v == 1)

{

Console.Clear();

Console.WriteLine();

Console.Write("Введите номер сотрудника: (1-5) для редактирования или добавления информации. ");

int n = Convert.ToInt32(Console.ReadLine());

Random rnd = new Random();

if (rnd.Next(2) == 0)

A(ref arr[n - 1]);

else

arr[n - 1] = A1(arr[n - 1]);

}

if (v == 2)

{

Console.Clear();

Console.WriteLine();

foreach (var Voditel1 in arr)

Voditel1.DisplayInfo();

}

if (v == 4)

{

Console.Clear();

Program.Main();

}

if (v == 5)

{

Console.Clear();

}

if (v >= 6)

{

Console.Clear();

Program.Error("Такой вариант отсутствует, проверьте правильность ввода");

}

} while (true);

}

static void A(ref Voditel1 a)

{

Console.WriteLine("ФИО: {0}", a.fio);

Console.Write("Новое ФИО: ");

a.fio = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Специальность: {0}", a.spec);

Console.Write("Новая специальность: ");

a.spec = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Контактные данные: {0}", a.Contact);

Console.Write("Новые контактные данные: ");

a.Contact = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Зарплата: {0}", a.ZP);

Console.Write("Новая зарплата: ");

a.ZP = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("");

}

static Voditel1 A1(Voditel1 a)

{

Console.WriteLine("ФИО: {0}", a.fio);

Console.Write("Новый ФИО: ");

a.fio = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Специальность: {0}", a.spec);

Console.Write("Новая специальность: ");

a.spec = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Контактные данные: {0}", a.Contact);

Console.Write("Новые контактные данные: ");

a.Contact = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Оклад: {0}", a.ZP);

Console.Write("Новая зарплата: ");

a.ZP = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("");

return a;

}

}

class machine

{

struct machine1

{

public string marka;

public string model;

public string koll;

public void DisplayInfo()

{

Console.WriteLine("Марка: {0}", this.marka);

Console.WriteLine("Модель: {0}", this.model);

Console.WriteLine("Колличество: {0}\n", this.koll);

}

}

internal void Ar2()

{

Console.Clear();

machine1[] arr = new machine1[4];

arr[0] = new machine1 { marka = "Mercedes Benz", model = "Conecto", koll = "5" };

arr[1] = new machine1 { marka = "Man", model = "MAN Lion’s Top Coach", koll = "5" };

arr[2] = new machine1 { marka = "Setra", model = "Setra S415 GT; ", koll = "5" };

arr[3] = new machine1 { marka = "Neoplan", model = "Neoplan Tourliner", koll = "5" };

do

{

Console.WriteLine("Выберите операцию:");

Console.WriteLine("1 - Изменить");

Console.WriteLine("2 - Показать");

Console.WriteLine("4 - Назад");

Console.WriteLine("5 - Сброс");

int v = Convert.ToInt32(Console.ReadLine());

Console.Clear();

if (v == 1)

{

Console.WriteLine();

Console.Write("Выберите марку автомобиля,чью информацию вы хотите изменить:\n 1-Mercedes Benz\n 2-Man\n 3-Setra\n 4-Neoplan\n" );

Console.Write("Команда: ");

int n = Convert.ToInt32(Console.ReadLine());

Random rnd = new Random();

if (rnd.Next(2) == 0)

A(ref arr[n - 1]);

else

arr[n - 1] = A1(arr[n - 1]);

}

if (v == 2)

{

Console.WriteLine();

foreach (var machine1 in arr)

machine1.DisplayInfo();

}

if (v == 4)

{

Console.Clear();

Program.Main();

}

if (v == 5)

{

Console.Clear();

}

if (v >= 6)

{

Program.Error("Такой вариант отсутствует");

}

} while (true);

}

static void A(ref machine1 a)

{

Console.WriteLine("Марка: {0}", a.marka);

Console.Write("Новая марка: ");

a.marka = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Модель: {0}", a.model);

Console.Write("Новая модель: ");

a.model = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Колличество: {0}", a.koll);

Console.Write("Колличество: ");

a.koll = Console.ReadLine();

Console.WriteLine("");

}

static machine1 A1(machine1 a)

{

Console.WriteLine("Марка: {0}", a.marka);

Console.Write("Новая марка: ");

a.marka = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Модель: {0}", a.model);

Console.Write("Новая модель: ");

a.model = Console.ReadLine();

Console.WriteLine("");

Console.WriteLine("Колличество: {0}", a.koll);

Console.Write("Колличество: ");

a.koll = Console.ReadLine();

Console.WriteLine("");

return a;

}

}

class Marshrut

{

internal void Ar1()

{

Marshrut post = new Marshrut();

Console.Clear();

do

{

Console.WriteLine("Выберите нужный маршрут");

Console.WriteLine("1 - Тирасполь-Дубоссары");

Console.WriteLine("2 - Тирасполь-Григориополь");

Console.WriteLine("3 - Тирасполь-Рыбница");

Console.WriteLine("4 - Тирасполь-Бендеры");

Console.WriteLine("5 - назад");

Console.WriteLine("6 - очистить");

int v = Convert.ToInt32(Console.ReadLine());

if (v == 1)

{

int p = 1000;

int p1 = 1000;

int p2 = 1000;

int p3 = 1000;

Console.WriteLine("\nВыберите время для поездки: ");

Console.WriteLine("1) 09:00: " + p + " руб");

Console.WriteLine("2) 13:00: " + p1 + " руб.");

Console.WriteLine("3) 15:00: " + p2 + " руб.");

Console.WriteLine("4) 17:00: " + p3 + " руб.");

int v1 = Convert.ToInt32(Console.ReadLine());

switch (v1)

{

case 1:

Console.Write("\n09:00 укажите колличество билетов: ");

int g = Convert.ToInt32(Console.ReadLine());

int r = g \* p;

Console.Write("Стоимость: " + r + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 2:

Console.Write("\n13:00 укажите колличество билетов: ");

int g1 = Convert.ToInt32(Console.ReadLine());

int r1 = g1 \* p1;

Console.Write("Стоимость: " + r1 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 3:

Console.Write("\n15:00 укажите колличество билетов: ");

int g2 = Convert.ToInt32(Console.ReadLine());

int r2 = g2 \* p2;

Console.Write("Стоимость: " + r2 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 4:

Console.Write("\n17:00 укажите колличество билетов: ");

int g3 = Convert.ToInt32(Console.ReadLine());

int r3 = g3 \* p3;

Console.Write("Стоимость: " + r3 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

default:

break;

}

}

if (v == 2)

{

int p = 700;

int p1 = 700;

int p2 = 700;

int p3 = 700;

int p4 = 700;

Console.WriteLine("\nВыберите время для поездки: ");

Console.WriteLine("1) 07:00: " + p + " руб.");

Console.WriteLine("2) 09:00: " + p1 + " руб.");

Console.WriteLine("3) 12:00: " + p2 + " руб.");

Console.WriteLine("4) 14:00: " + p3 + " руб.");

Console.WriteLine("5) 17:00: " + p3 + " руб.");

int v1 = Convert.ToInt32(Console.ReadLine());

switch (v1)

{

case 1:

Console.Write("\n07:00 укажите колличество билетов: ");

int g = Convert.ToInt32(Console.ReadLine());

int r = g \* p;

Console.Write("Стоимость: " + r + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 2:

Console.Write("\n09:00 укажите колличество билетов: ");

int g1 = Convert.ToInt32(Console.ReadLine());

int r1 = g1 \* p1;

Console.Write("Стоимость: " + r1 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 3:

Console.Write("\n12:00 укажите колличество билетов: ");

int g2 = Convert.ToInt32(Console.ReadLine());

int r2 = g2 \* p2;

Console.Write("Стоимость: " + r2 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 4:

Console.Write("\n14:00 укажите колличество билетов: ");

int g3 = Convert.ToInt32(Console.ReadLine());

int r3 = g3 \* p3;

Console.Write("Стоимость: " + r3 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 5:

Console.Write("\n17:00 укажите колличество билетов: ");

int g4 = Convert.ToInt32(Console.ReadLine());

int r4 = g4 \* p4;

Console.Write("Стоимость: " + r4 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

default:

break;

}

}

if (v == 3)

{

int p = 3000;

int p1 = 3000;

int p2 = 3000;

int p3 = 3000;

int p4 = 3000;

Console.WriteLine("\nВыберите время для поездки: ");

Console.WriteLine("1) 10:00: " + p + " руб.");

Console.WriteLine("2) 13:00: " + p1 + " руб.");

Console.WriteLine("3) 15:00: " + p2 + " руб.");

Console.WriteLine("4) 17:00: " + p3 + " руб.");

Console.WriteLine("5) 19:00: " + p4 + " руб.");

int v1 = Convert.ToInt32(Console.ReadLine());

switch (v1)

{

case 1:

Console.Write("\n10:00 укажите колличество билетов: ");

int g = Convert.ToInt32(Console.ReadLine());

int r = g \* p;

Console.Write("Стоимость: " + r + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 2:

Console.Write("\n13:00 укажите колличество билетов: ");

int g1 = Convert.ToInt32(Console.ReadLine());

int r1 = g1 \* p1;

Console.Write("Стоимость: " + r1 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 3:

Console.Write("\n15:00 укажите колличество билетов: ");

int g2 = Convert.ToInt32(Console.ReadLine());

int r2 = g2 \* p2;

Console.Write("Стоимость: " + r2 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 4:

Console.Write("\n17:00 укажите колличество билетов: ");

int g3 = Convert.ToInt32(Console.ReadLine());

int r3 = g3 \* p3;

Console.Write("Стоимость: " + r3 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 5:

Console.Write("\n19:00 укажите колличество билетов: ");

int g4 = Convert.ToInt32(Console.ReadLine());

int r4 = g4 \* p4;

Console.Write("Стоимость: " + r4 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

default:

break;

}

}

if (v == 4)

{

int p = 500;

int p1 = 500;

int p2 = 500;

int p3 = 500;

int p4 = 500;

Console.WriteLine("\nВыберите время для поездки: ");

Console.WriteLine("1) 10:00: " + p + " руб.");

Console.WriteLine("2) 12:00: " + p1 + " руб.");

Console.WriteLine("3) 14:00: " + p2 + " руб.");

Console.WriteLine("4) 16:00: " + p3 + " руб.");

Console.WriteLine("5) 18:00: " + p4 + " руб.");

int v1 = Convert.ToInt32(Console.ReadLine());

switch (v1)

{

case 1:

Console.Write("\n10:00 укажите колличество билетов: ");

int g = Convert.ToInt32(Console.ReadLine());

int r = g \* p;

Console.Write("Стоимость: " + r + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 2:

Console.Write("\n12:00 укажите колличество билетов: ");

int g1 = Convert.ToInt32(Console.ReadLine());

int r1 = g1 \* p1;

Console.Write("Стоимость: " + r1 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 3:

Console.Write("\n14:00 укажите колличество билетов: ");

int g2 = Convert.ToInt32(Console.ReadLine());

int r2 = g2 \* p2;

Console.Write("Стоимость: " + r2 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 4:

Console.Write("\n16:00 укажите колличество билетов: ");

int g3 = Convert.ToInt32(Console.ReadLine());

int r3 = g3 \* p3;

Console.Write("Стоимость: " + r3 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

case 5:

Console.Write("\n18:00 укажите колличество билетов: ");

int g4 = Convert.ToInt32(Console.ReadLine());

int r4 = g4 \* p4;

Console.Write("Стоимость: " + r4 + " руб.\n");

Sale.S("Заказ оформлен!\n");

break;

default:

break;

}

}

if (v == 5)

{

Console.Clear();

Program.Main();

}

if (v >= 6)

{

Console.Clear();

}

if (v >= 7)

{

Program.Error("Такой вариант отсутсвует");

}

} while (true);

}

}

class Sale

{

internal void Ar2()

{

Console.Clear();

Sale sal = new Sale();

}

public static void S(string info)

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(info);

Console.ResetColor();

}

}

class Program

{

public static void Error(string error)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine(error);

Console.ResetColor();

}

internal static void Main()

{

Voditel mast = new Voditel();

Marshrut post = new Marshrut();

Sale sal = new Sale();

machine mach = new machine();

int variant = 0;

do

{

Console.WriteLine("Вас приветствует автопарк Etherial");

Console.WriteLine("1) Работники\n2) Продажа билетов\n3) Машины в автопарке\n4) Сброс\n5) Выход");

Console.Write("Команда: ");

variant = Int32.Parse(Console.ReadLine());

switch (variant)

{

case 1:

mast.Ar();

Console.ReadKey();

Console.Clear();

break;

case 2:

post.Ar1();

break;

case 3:

mach.Ar2();

Console.ReadKey();

Console.Clear();

break;

case 4:

Console.Clear();

break;

case 5:

Environment.Exit(0);

break;

default:

Program.Error("Такой вариант отсутствует");

break;

}

} while (true);

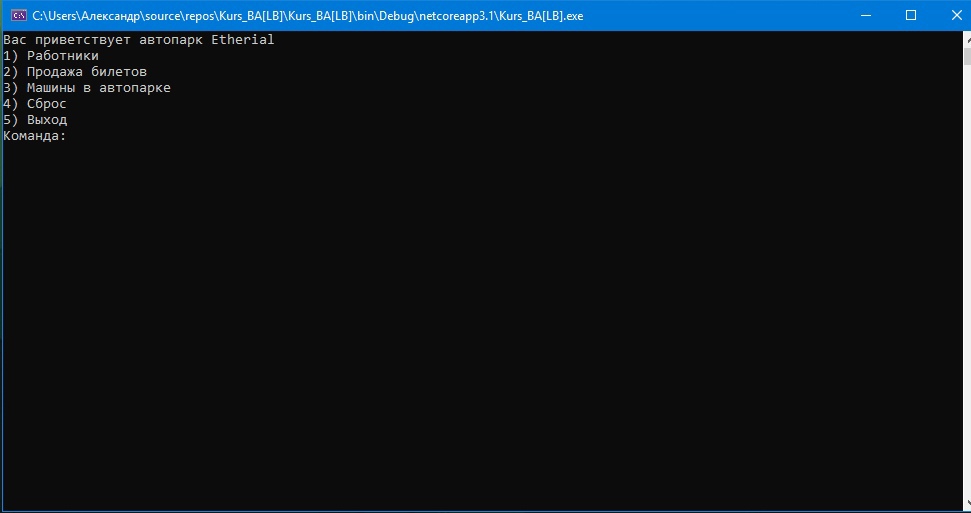
}

}

}

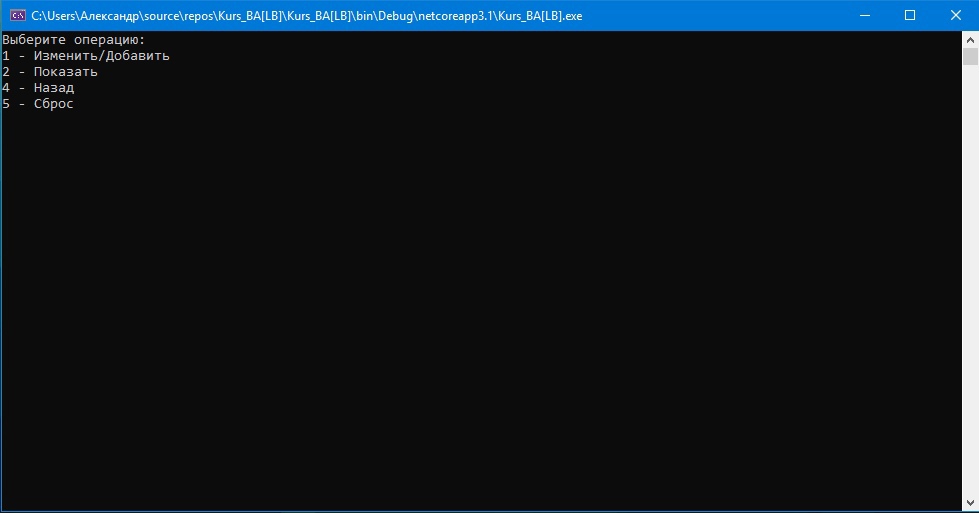
**2.3. Руководство пользователя**

При запуске программы появляется меню программы. ( Рис. 3)

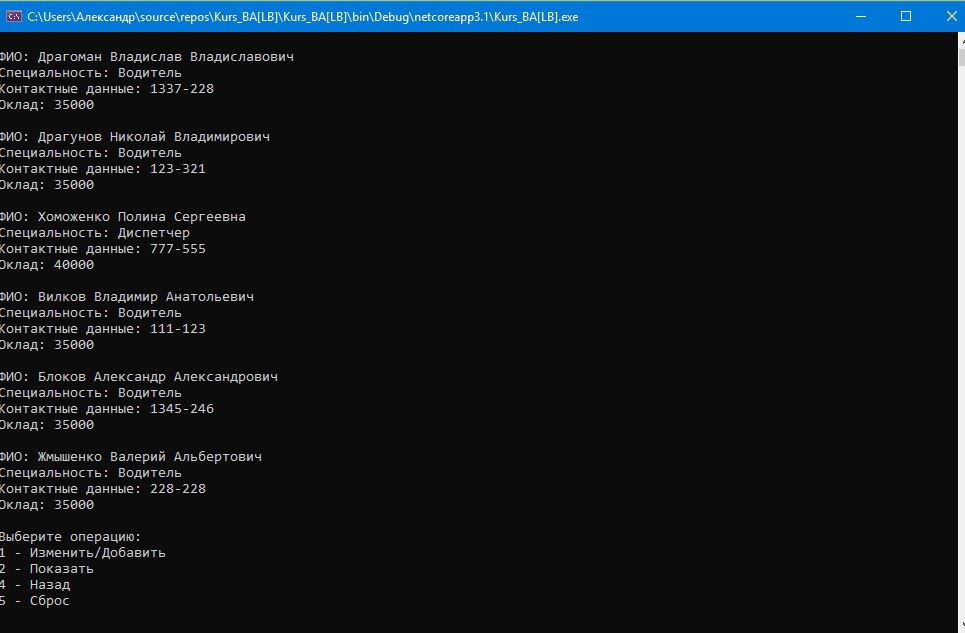


(Рис. 3). Меню программы.

1 Пункт-работники. Позволяет узнать или изменить список работников. (См. рис. 4,5)

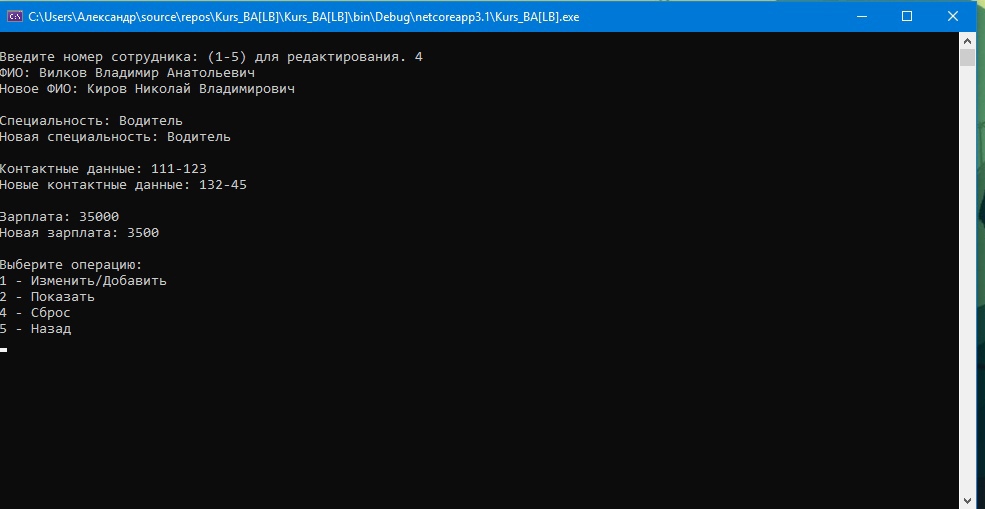


(Рис.4) Меню водителей.



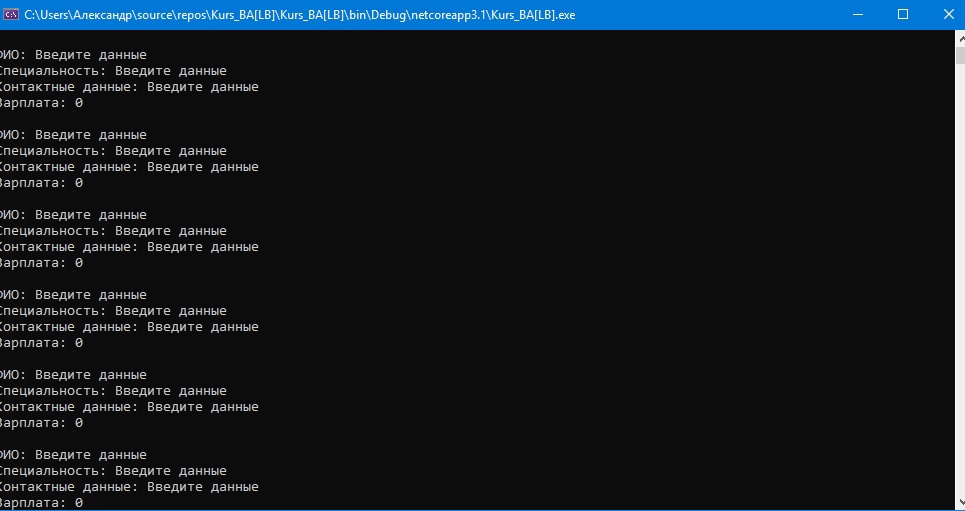
(Рис. 5) Вывод информации о водителях.

1.2. Если мы хотим редактировать информацию о сотрудниках, нам нужно выбрать 1 операцию, после чего нам предложат выбрать сотрудника, чью информацию мы хотим изменить. ( См. рис. 6)



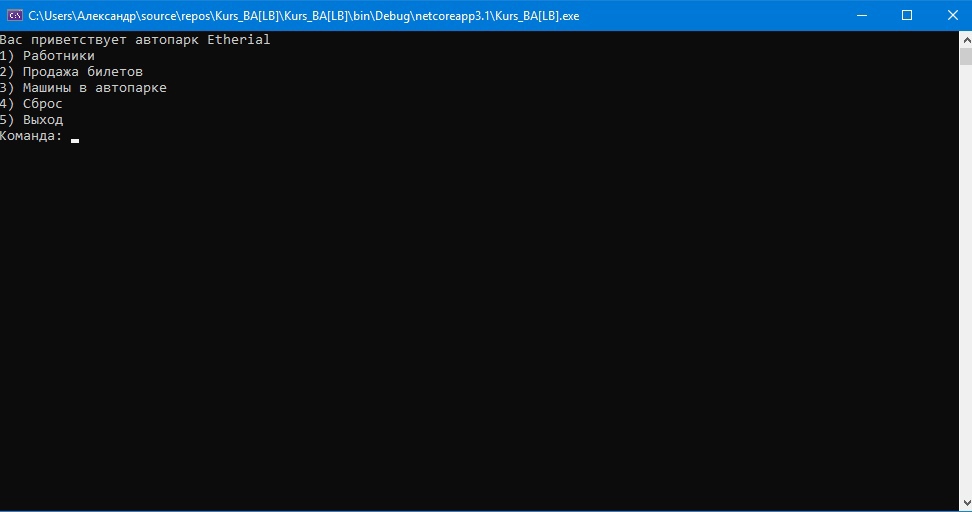
(Рис.6) Процесс изменения информации.

1.3 Так же есть кнопка сбросить, которая позволяет вернуть данные в исходное состояние (Рис. 7)

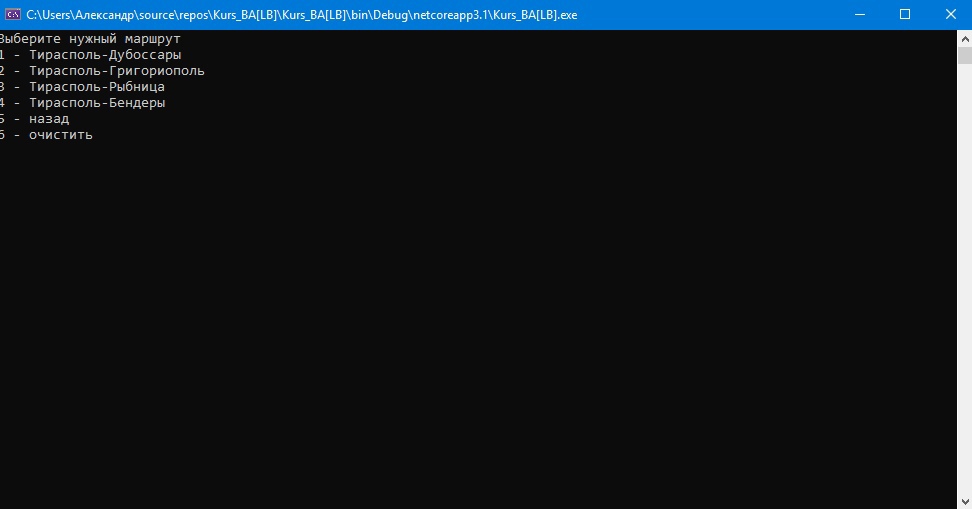


(Рис 7.) Информация после сброса данных.

1. 2 Пункт нашей программы, это продажа билетов. Перейдя в меню выбираем 2 пункт, мы попадаем в меню, где указаны маршруты. (Рис.8,9)

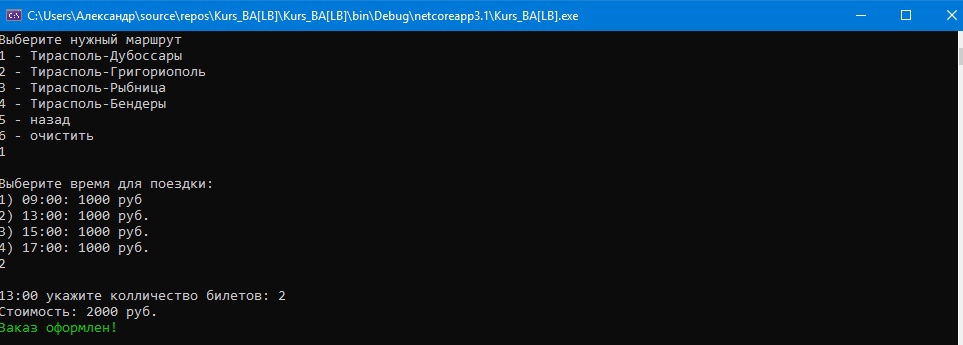


(Рис.8) Главное меню.



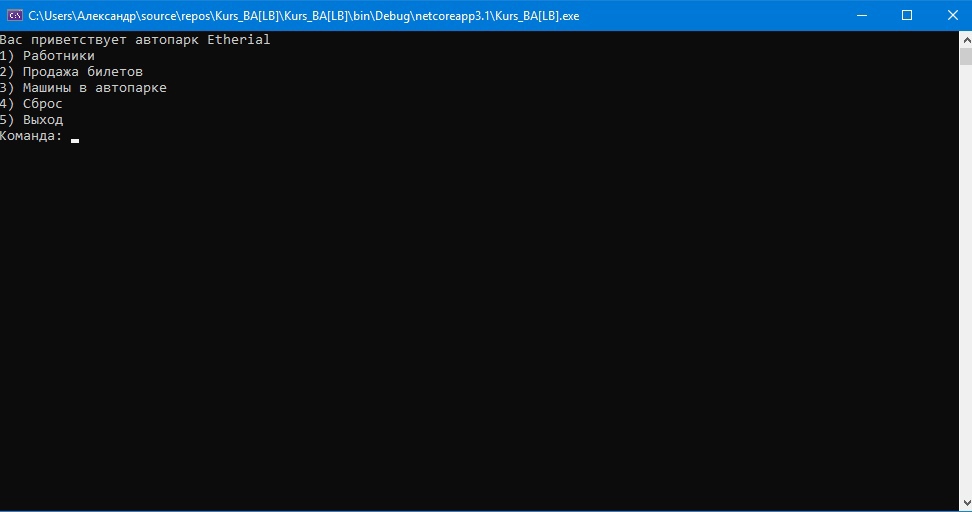
(Рис.9) Маршруты.

2.1 После выбора маршрута, нам предлагают выбрать время и количество билетов, после чего наш заказ будет оформлен. (Рис.10)



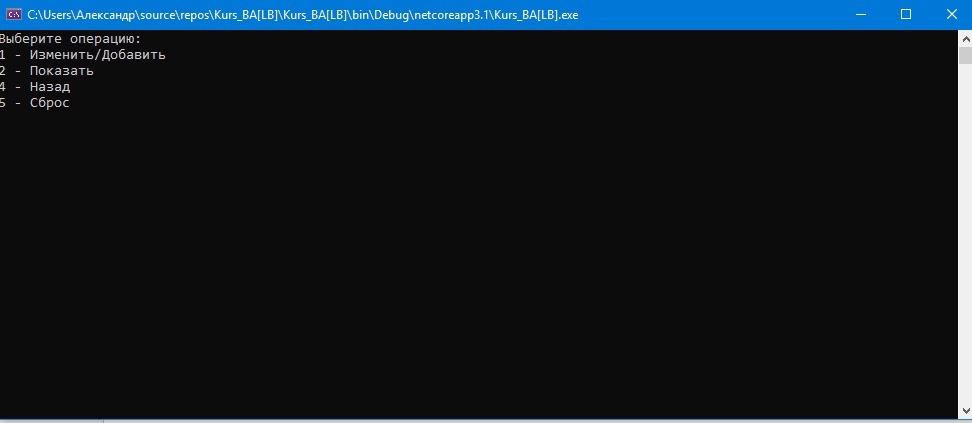
(Рис .10) Оформление заказа.

1. В 3 пункте мы можем посмотреть какие автобусы находятся во владении нашего автопарка. Для этого вернемся в главное меню и выберем 3 пункт. (Рис 11)

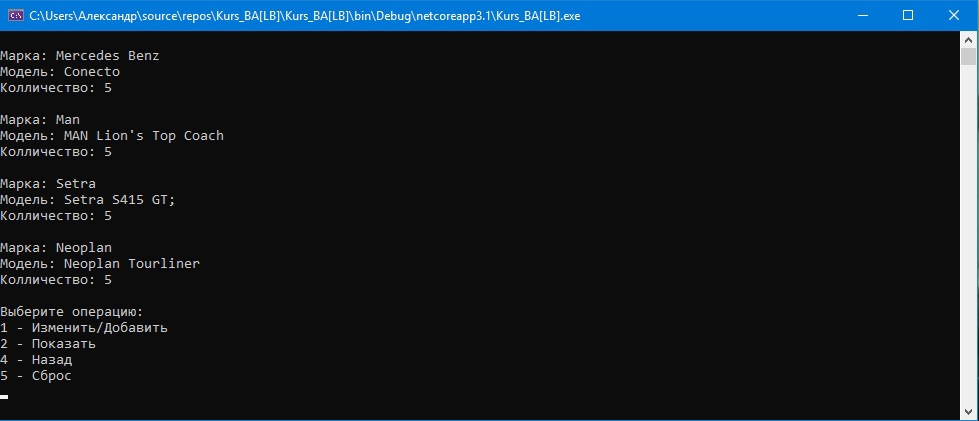


(Рис 11) Главное меню.

3.1 После выбора мы попадаем в меню, где можем посмотреть автобусы, которые находятся во владении нашего автопарка. (Рис. 12,13)

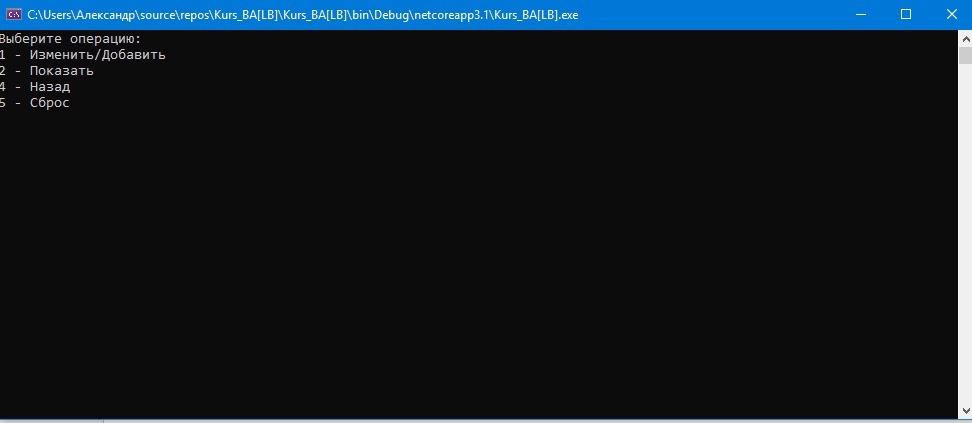


(Рис.12) Меню автобусов.

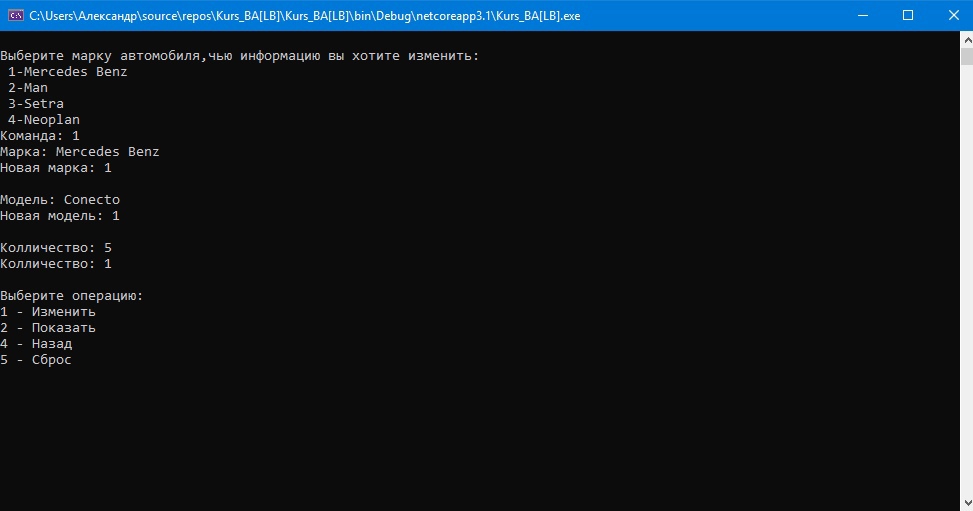


(Рис.13) Автобусы нашего автопарка.

3.2 Также мы можем изменить информацию о наших автобусах. Для этого вернемся назад и выберем 1 пункт (Рис.14,15).



(Рис.14) Меню автобусов.

(Рис.15) Редактирование информации.

1. В главном меню есть 4 пункт-сброс, который сбрасывает полностью программу.
2. Также имеется 5 пункт- выход, который инициализирует выход из программы. (Рис.16)



(Рис.16) 4 и 5 пункты.

# Заключение

В ходе выполнения курсовой работы на тему “Разработка системы классов, описывающих автобусный парк.” была изучена предметная область. Был произведён анализ существующих аналогов, выявлены недостатки существующих аналогов и принято решение создать собственный программный продукт.

При исследовании предметно области были выявлены требования к программе, а также составлены руководство пользователю и программисту.

Программный продукт удовлетворяет следующим требованиям:

1. Редактирование информации о водителях.

2. Расчет и продажа билетов.

3. Редактирование информации о автобусах.

4. Сброс данных программы.

Поставленные задачи были успешно выполнены, обозначенные цели достигнуты.

**Список использованной литературы**

1. Александров, В.А. Автотранспортные средства: Учебное пособие / В.А. Александров, Н.Р. Шоль. - СПб.: Лань П, 2016. - 336 c.
2. Бачурин, А.А. Планирование и прогнозирование деятельности автотранспортных организаций / А.А. Бачурин. - Вологда: Инфра-Инженерия, 2011. - 272 c.
3. Бачурин, А.А. Планирование и прогнозирование деятельности автотранспортных организаций: Учебное пособие / А.А. Бачурин. - М.: Академия, 2008. - 160 c.
4. Буров, А.Л. Проектирование автотранспортных предприятий / А.Л. Буров, А.А. Мылов. - М.: МГИУ, 2010. - 86 c