

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

Кафедра систем управления и информационных технологий в строительстве

КУРСОВОЙ ПРОЕКТ

по дисциплине: «Объектно-ориентированное программирование»  
на тему: «Разработка приложения для отслеживания работы ГОКа»

Выполнил: студент группы БПИЭ-202

\_\_\_\_\_ Богданов В.Ю.  
(подпись)

Руководитель: д.т.н., проф. Кононов А.А.

Работа защищена « \_\_\_\_ » \_\_\_\_\_ г.

с оценкой \_\_\_\_\_  
(подпись)

Члены комиссии

\_\_\_\_\_  
Подпись, дата                      Инициалы, фамилия

\_\_\_\_\_  
Подпись, дата                      Инициалы, фамилия

Нормоконтролер

\_\_\_\_\_  
Подпись, дата                      Инициалы, фамилия

2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИ-  
ТЕТ»  
(ФГБОУ ВО «ВГТУ», ВГТУ)

Кафедра систем управления и информационных технологий в строительстве

ЗАДАНИЕ  
на курсовой проект

по дисциплине: «Объектно-ориентированное программирование»  
Тема: «Разработка приложения для отслеживания работы ГОКа»

Студент бПИЭ-202, Богданов Вадим Юрьевич  
Группа, фамилия, имя, отчество

Вариант \_\_\_\_\_

Технические условия операционная система Windows 10, ОЗУ 2048 МБ

Сроки выполнения этапов: анализ и постановка задачи (15.09-10.10);  
изучение теоретического обоснования работы (10.10-10.11);  
реализация программного решения (10.11-10.12);  
оформление пояснительной записки (10.12-20.12).

Срок защиты курсового проекта: декабрь 2022 года

Руководитель \_\_\_\_\_  
Подпись, дата А.А. Кононов  
Инициалы, фамилия

Задание принял студент \_\_\_\_\_  
Подпись, дата В.Ю.Богданов  
Инициалы, фамилия

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ.....	3
1. Описание предметной области.....	4
2. Постановка задачи .....	5
3. Выбор языка программирования разработки.....	5
4. Выбор среды разработки.....	10
5. Проектирование структуры приложения .....	10
6. Разработка Базы данных.....	12
7. Разработка приложения.....	14
8. Демонстрация разработанного приложения .....	24
ЗАКЛЮЧЕНИЕ .....	27
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	28
ПРИЛОЖЕНИЕ А .....	30

## ВВЕДЕНИЕ

Объектно-ориентированное программирование (сокр. ООП) — методология программирования, основанная на представлении программы в виде совокупности взаимодействующих объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

C# — объектно-ориентированный язык программирования (ООП). Всё взаимодействие в нём происходит через объекты. Это в целом похоже на то, что творится в реальном мире. Все эти сущности описывают в коде и учат взаимодействовать друг с другом. В итоге программа в стиле ООП состоит из отдельных блоков, которые хорошо расширяются и масштабируются. Поэтому язык C# подходит для разработки программ, которые планируют долго использовать и постоянно развивать.

C# берёт лучшее из компилируемых и интерпретируемых языков. Чтобы разобраться в этом свойстве, нужно шагнуть ещё немного назад. Язык программирования — это язык, на котором программист и процессор договариваются, как выполнять команды. Так вот процессор не полиглот и не обязан знать все языки, на которых им хотят покомандовать. Поэтому язык программирования нужно переводить на язык процессора. Делается это двумя способами — интерпретированием и компилированием.

Целью данной курсовой работы является разработка программы с использованием принципов объектно-ориентированного программирования на ООП языке – C# средствами среды разработки Visual Studio.

Задачами данной курсовой работы являются:

- Выделение классов, необходимых для решения задачи
- Выделение основного действия в задаче и построение алгоритма его реализации

## 1. Описание предметной области

Концепции ООП являются основополагающими элементами и составляют основу языка программирования С#. В рамках данного подхода выделяют следующие термины: абстракция, инкапсуляция, наследование и полиморфизм. Понимание данных принципов служит ключом к построению целостной картины того, как работают программы, написанные на С#. По большому счету, объектно-ориентированный подход позволяет нам описывать классы, определять методы и переменные таким образом, чтобы затем использовать их вновь, частично либо полностью, без нарушения безопасности.

**Абстракция.** Абстракция означает использование простых вещей для описания чего-то сложного. В С# под абстракцией подразумеваются такие вещи, как объекты, классы и переменные, которые в свою очередь лежат в основе более сложного кода. Использование данного принципа позволяет избежать сложности при разработке ПО.

**Инкапсуляция.** Под инкапсуляцией подразумевается сокрытие полей внутри объекта с целью защиты данных от внешнего, неконтролируемого изменения со стороны других объектов. Доступ к данным (полям) предоставляется посредством публичных методов (геттеров/сеттеров).

**Наследование.** Это особая функциональность в объектно-ориентированных языках программирования, которая позволяет описывать новые классы на основе уже существующих. При этом поля и методы класса-предка становятся доступны и классам-наследникам. Данная фича делает классы более чистыми и понятным за счет устранения дублирования программного кода.

**Полиморфизм.** Данный принцип позволяет программистам использовать одни и те же термины для описания различного поведения, зависящего от контекста. Одной из форм полиморфизма в С# является переопределение метода, когда различные формы поведения определяются объектом, из которого данный метод был вызван.

## **2. Постановка задачи**

В рамках данной курсовой работы требуется разработать приложение, программный код которого будет основываться на парадигме объектно ориентированного программирования и его основных принципах (наследование, инкапсуляция, полиморфизм, абстракция).

Основная задача проектируемого приложения заключается в проверка деятельности ГОКа.

Программа должна быть отказоустойчивой, т.е корректно реагировать и обрабатывать ошибки, которые могут возникнуть ходе использования приложения (некорректно введенные данные, деление на ноль, ошибка чтения системных файлов и.т.д)

Разрабатываемое приложение должно предоставлять своим пользователям простой и понятный результат для своего корректного и эффективного использования.

## **3. Выбор языка программирования разработки**

Приложение, которое создается сейчас должно полностью соответствующего поставленным требованиям, поэтому необходимо выбрать один из объектно-ориентированных языков программирования. Много языков применяют парадигму ООП, но у каждого есть свои достоинства и недостатки.

К наиболее популярным на данный момент язык программирования, реализующих принципы ООП можно отнести:

- C++
- Java
- C#

## C++

### Достоинства языка:

1. Масштабируемость. На языке C++ разрабатывают программы для самых различных платформ и систем.
2. Возможность работы на низком уровне с памятью, адресами, портами. Что, при неосторожном использовании, может легко превратиться в недостаток.
3. Возможность создания обобщенных алгоритмов для разных типов данных, их специализация, и вычисления на этапе компиляции, используя шаблоны.

### Недостатки языка:

1. Наличие множества возможностей, нарушающих принципы тип безопасности приводит к тому, что в C++-программы может легко закрасться трудноуловимая ошибка. Вместо контроля со стороны компилятора разработчики вынуждены придерживаться весьма нетривиальных правил кодирования. По сути эти правила ограничивают C++ рамками некоего более безопасного подязыка. Большинство проблем тип безопасности C++ унаследовано от C, но важную роль в этом вопросе играет и отказ автора языка от идеи использовать автоматическое управление памятью (например, сборку мусора). Так визитной карточкой C++ стали уязвимости типа "переполнение буфера".
2. Плохая поддержка модульности. Подключение интерфейса внешнего модуля через препроцессорную вставку заголовочного файла (`#include`) серьезно замедляет компиляцию, при подключении большого количества модулей. Для устранения этого недостатка, многие компиляторы реализуют механизм прекомпиляции заголовочных файлов `Precompiled Headers`.
3. Недостаток информации о типах данных во время компиляции (CTTI).

4. Язык C++ является сложным для изучения и для компиляции.
5. Некоторые преобразования типов неинтуитивны. В частности, операция над без знаковых и знаковых чисел выдаёт без знакового результата.

## **Java**

Достоинства языка:

1. Независимый код. Любая платформа, которая поддерживает виртуальную машину Java, воспроизведет ваш код.
2. Надежный код. Строгая статистическая, типизация дает главное преимущество — надежность вашего кода.
3. Высокая функциональность. На Java можно написать практически все: от простого приложения на смартфон до программ по машинному обучению для беспилотных автомобилей.
4. Синтаксис средней сложности. Данный язык поддается изучению новичкам, которые раньше вообще не имели дела с программированием.
5. Java для Android. Android — самая популярная ОС для смартфонов, а Java — самый популярный язык для приложений на Android, соответственно, изучив Java, будет очень широкое поле для деятельности.

Недостатки языка:

1. Более низкая производительность. За счет своей специфики Java во многих случаях работает медленнее, чем другие языки, такие как: C, C#, C++, Python.
2. Потребляет память. Опять же, за счет своей специфики работы данный язык требует больше памяти, чем многие сторонние языки.
3. Платность. Буквально с 2019 года для коммерческо-юридических проектов язык Java стал платным, но для частного использования он абсолютно бесплатен.



## C#

### Достоинства языка:

1. Независимость от железа. Программисту не надо адаптировать программу под разные платформы и системы — за него это делает виртуальная машина, вшитая в .NET Framework. В итоге один и тот же код можно запускать на любых устройствах — смартфонах, компьютерах, серверах, банкоматах и даже умных часах.
2. Отличная совместимость с Windows. Не зря же язык разработали именно в Microsoft. Так же как Swift идеально подходит для программирования под экосистему Apple, C# прекрасно вписывается в экосистему Windows.
3. Управление памятью. Чтобы программа работала стабильно, её надо иногда чистить от ненужных объектов, ссылок, кэша и прочего мусора. В C# это происходит автоматически — разработчику не надо следить за расходом памяти, бороться с её утечками или удалять мёртвые куски кода.
4. Строгая типизация. Когда вы объявляете переменную в C#, надо сначала указать, что в ней лежит — строка, число или массив. Так разрабатывать чуть дольше, зато ваш код работает предсказуемо — числа взаимодействуют с числами, строки со строками и так далее. В языках со слабой типизацией свободы и драйва больше, но есть шанс пропустить ошибку, которая всплывёт в готовой программе.
5. Большое сообщество. На C# пишут более миллиона программистов по всему миру. В соцсетях полно чатов и сообществ «шарпистов», где можно задать вопрос, обсудить сложную тему или найти готовое решение. В теории можно даже найти ментора, который поделится знаниями и поможет быстрее освоить язык.

6. Синтаксический сахар. В C# есть много способов сократить код, не нарушая логику программы. Программисты называют такие приёмы «синтаксическим сахаром» — они помогают сделать код проще, понятнее и в целом симпатичнее. Сравните, например, как выглядит сложение чисел с «сахаром» и без.

Недостатки языка:

1. Скорость. Когда мы запускаем программу на C#, код выполняется не сразу, а сначала адаптируется под нужное железо. Так мы охватываем больше платформ, но теряем в скорости — программе нужно сделать двойную работу, чтобы просто стартовать. Из-за этого интерфейсы на C# иногда подтормаживают при первом запуске.
2. Безопасность. Эксперты говорят, что код на C# легко декомпилировать — то есть перевести из машинного обратно в человеческий. Проблема в том, что так программу может легко прочесть хакер или конкурент — и изучить её уязвимости, украсть фрагменты кода или написать для неё вредоносный софт.
3. Мало доступа к железу. Так как C# — язык высокого уровня, на нём редко пишут проекты, где нужно полное взаимодействие с железом, — игровые движки, операционные системы, авиационный софт и так далее. Та же Unity целиком написана на низкоуровневом языке C++, хотя и умеет исполнять C#-команды.

В ходе проведения анализа из всех представленных языков, учитывая плюсы и минусы был выбран язык C#(.net) и фреймворк WinForms(.net 6). На данном языке довольно просто реализовать принципы ООП, он безопасный — это обеспечивает автоматическое управление памятью, сборщики мусора.

#### 4. Выбор среды разработки

Действительно хорошим выбором из инструментов для владения языком C# будет среда разработки Visual Studio.

Visual Studio 2022 Community - лучшая комплексная среда IDE для разработчиков .NET и C++ в Windows. Полноценный набор инструментов и функций для улучшения и усовершенствования каждого этапа разработки программного обеспечения.

#### 5. Проектирование структуры приложения

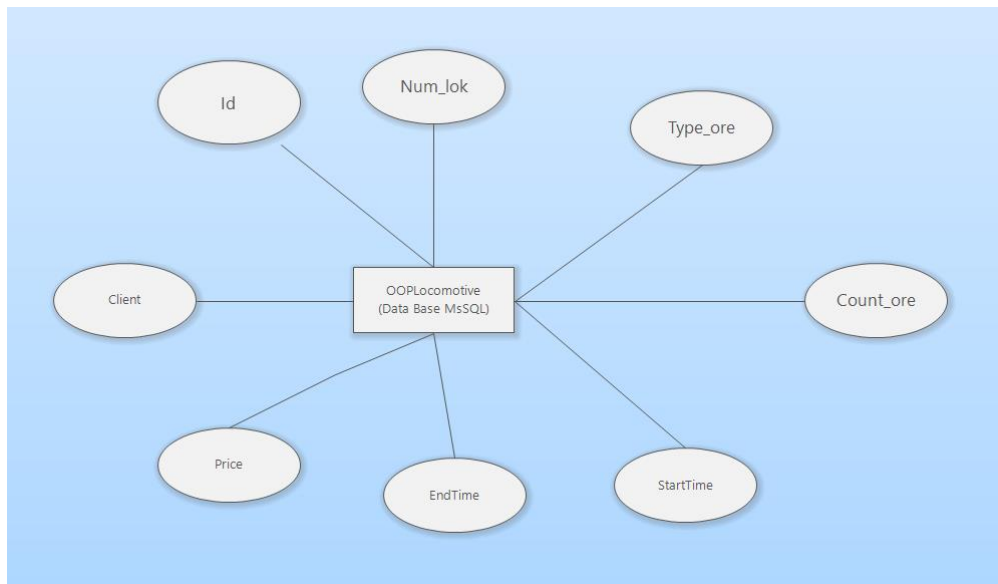


Рисунок 1 – концептуальная модель класса Базы данных

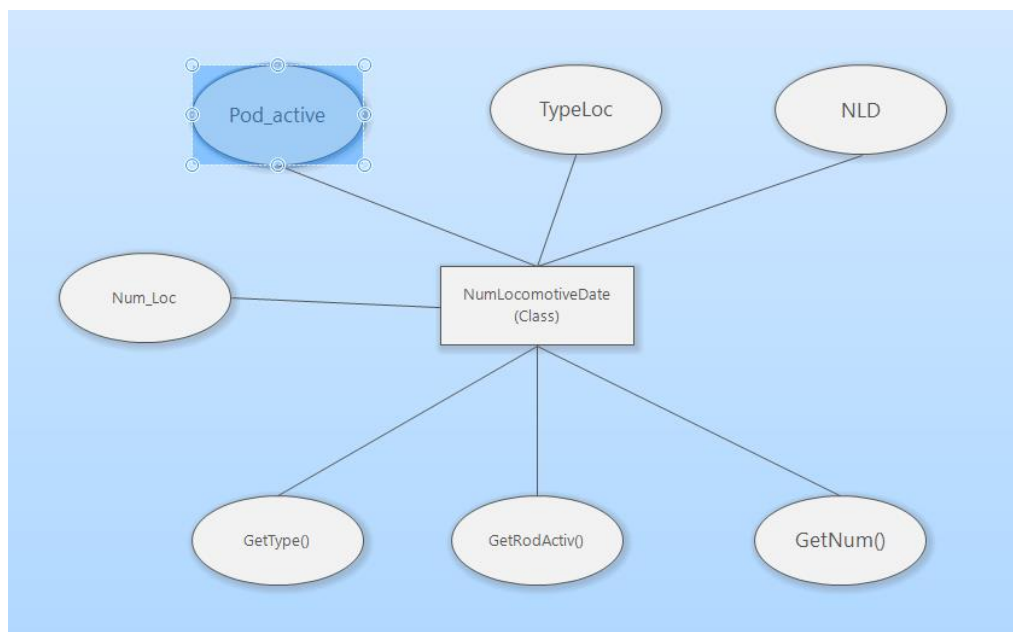


Рисунок 2 – концептуальная модель класса NumLocomotiveDate

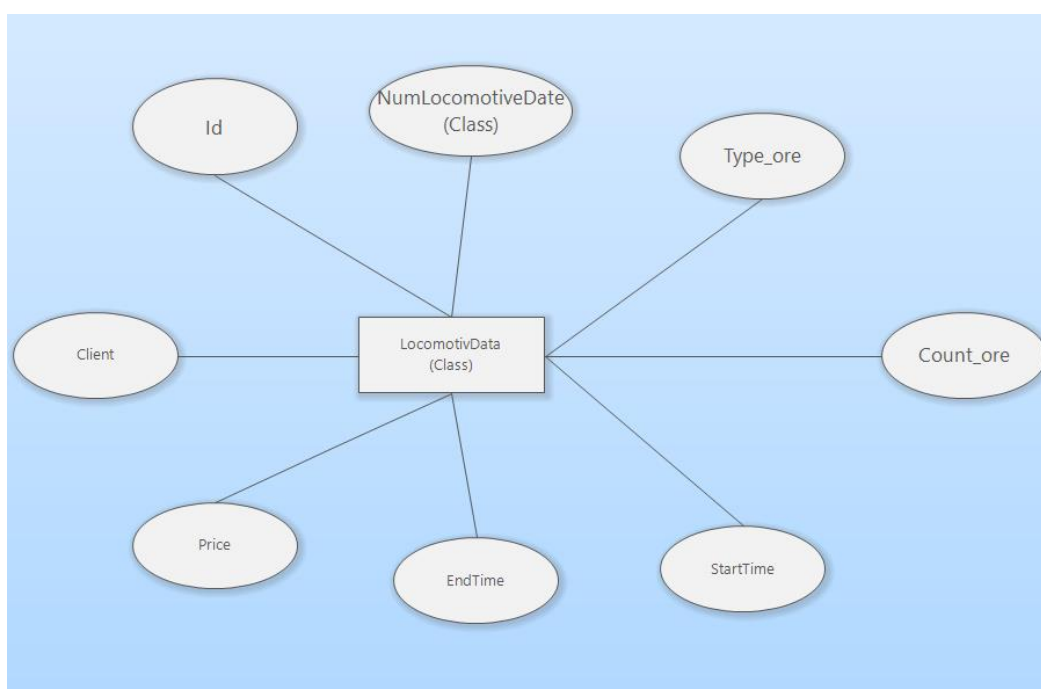


Рисунок 3 – концептуальная модель класса LocomotivData

На рисунках 1-3 показана концептуальная разрабатываемой программы. На основе данной модели происходит создание логической модели. Концептуальная модель помогает определить сущности и их атрибуты.

## 6. Разработка Базы данных

В нашем приложении необходима база данных которая будет содержать данные о заказах и локомотивах.

Для этого создадим таблицу OOPLocomotive

В таблице OOPLocomotive создадим поля (Id, Num\_lok, Type\_ore, Count\_ore, StartTime, EndTime, Price, Client).

```
13 CREATE TABLE OOPLocomotive (  
14     Id int identity(1,1) PRIMARY KEY,  
15     Num_lok varchar(8) NOT NULL CHECK (Num_lok > '1000000' AND Num_lok < '1999999'),  
16     --П Т Р XXXX К. -- П - идентификатор локомотива  
17     --Р - род службы  
18     --Т - тип локомотива  
19     --0 - паровозы; 1 - электровозы односекционные; 2 - электровозы многосекционные;  
20     --3 - электропоезда; 4 - метрополитен; 5 - тепловозы односекционные;  
21     --6 - тепловозы многосекционные; --7 - дизель-поезда и автомотрисы;  
22     --8 - специальный тяговый подвижной состав (мотовозы, автодрезины и т.д.);  
23     --9 - путевые машины.  
24     Type_ore varchar(255) NOT NULL,  
25     Count_ore float NOT NULL,  
26     StartTime DATETIME NOT NULL,  
27     EndTime DATETIME NOT NULL,  
28     Price int NOT NULL,  
29     Client varchar(255) NOT NULL,  
30 );  
31
```

Рисунок 4 – реализация таблицы OOPLocomotive

Так же для функционирования приложения нужно разработать sql скрипты который будет возвращать данные из таблицы в зависимости от выбранной фильтрации.

```

38 select distinct Num_lok from OOPLocomotive
39 union all (select '' as Num_lok)
40 order by Num_lok
41
42 select distinct Type_ore from OOPLocomotive
43 union all (select '' as Type_ore)
44 order by Type_ore
45
46 select * from OOPLocomotive
47 where (Num_lok = 1301347)
48
49 select * from OOPLocomotive where (Type_ore = 'Гранит')
50
51 select * from OOPLocomotive
52 where (Type_ore = 'Гранит') and (Num_lok = 1301347)

```

Рисунок 5 – Скрипты для отображения отфильтрованных таблиц

Ещё нужен скрип который будет вставлять данные в таблицу OOPLocomotive.

```

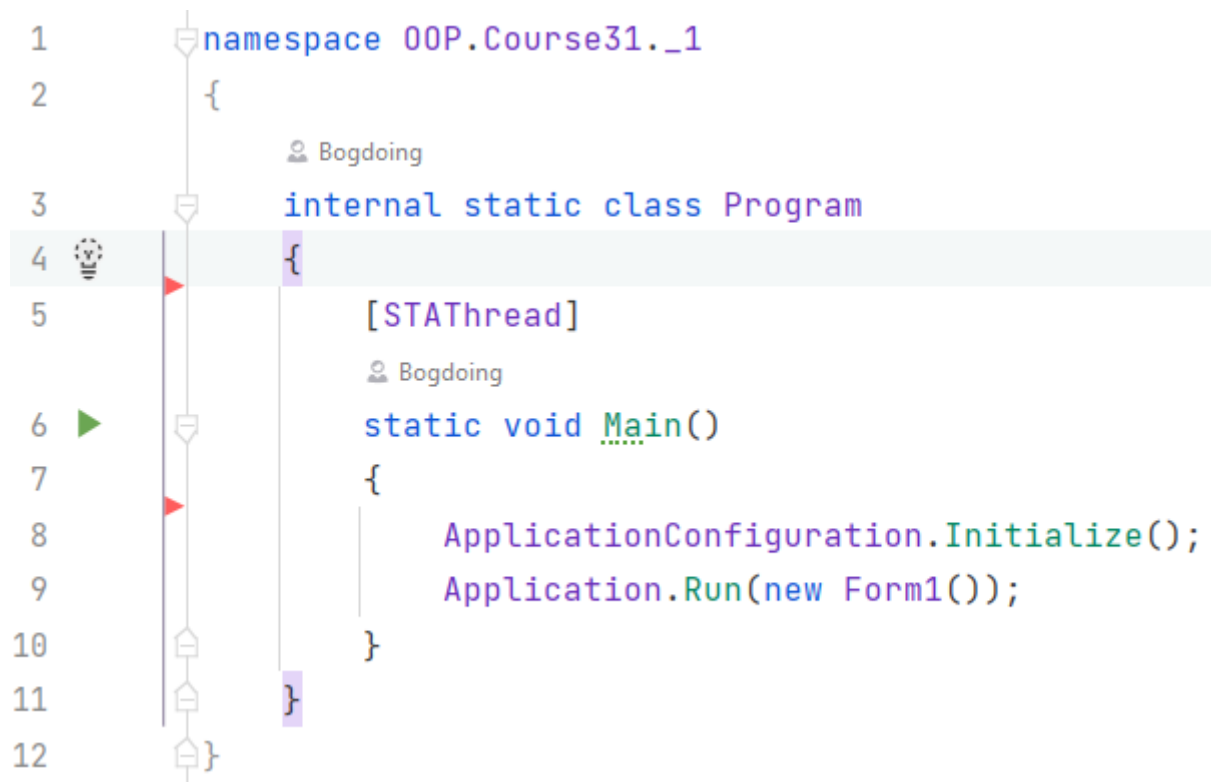
34 INSERT INTO dbo.OOPLocomotive
35 VALUES( '1301347', 'Гранит', '262', '25/9/2028', '25/10/2020', '4507947')
36 INSERT INTO dbo.OOPLocomotive
37 VALUES( '1054833', 'Гранит', '155', '15/10/2028', '15/10/2027', '6500199')
38 INSERT INTO dbo.OOPLocomotive
39 VALUES( '1336935', 'Базальт', '51', '29/06/2029', '29/07/2025', '6812877')

```

Рисунок 6 – Скрип для вставки данных в таблицу OOPLocomotive

## 7. Разработка приложения

Сначала реализуем класс Program для запуска программы.



The image shows a code editor window with a file explorer on the left. The file explorer shows a project structure with a folder named '00P.Course31.\_1'. The code editor displays the following C# code:

```
1 namespace 00P.Course31._1
2 {
3     internal static class Program
4     {
5         [STAThread]
6         static void Main()
7         {
8             ApplicationConfiguration.Initialize();
9             Application.Run(new Form1());
10        }
11    }
12 }
```

The code is written in C# and defines a namespace '00P.Course31.\_1' containing a static class 'Program'. The 'Program' class has a single static method 'Main()' which is decorated with the '[STAThread]' attribute. The 'Main()' method calls 'ApplicationConfiguration.Initialize()' and 'Application.Run(new Form1())'.

Рисунок 7 – реализация класса Program

```

1 namespace OOP.Course31._1;
2
3 public class NumLocomotiveDate
4 {
5     public string NLD = "";
6     public string TypeLoc = "";
7     public string Pod_active = "";
8     public string Num_Loc = "";
9
10    public string GetType(char[] b)
11    {
12        string typeLoc = "";
13        // 0 - паровозы; 1 - электровазы односекционные; 2 - электровазы многосекционные;
14        // 3 - электропоезда; 4 - метрополитен; 5 - тепловозы односекционные;
15        // 6 - тепловозы многосекционные; 7 - дизель-поезда и автомотрисы;
16        // 8 - специальный тяговый подвижной состав (мотовозы, автодрезины и т.д.); 9 - путевые машины.
17        if (b[1] == '0') typeLoc = "Паровоз";
18        else if (b[1] == '1') typeLoc = "Электроваз односекционный";
19        else if (b[1] == '2') typeLoc = "Электроваз многосекционный";
20        else if (b[1] == '3') typeLoc = "Электропоезд";
21        else if (b[1] == '4') typeLoc = "Метрополитен";
22        else if (b[1] == '5') typeLoc = "Тепловоз односекционный";
23        else if (b[1] == '6') typeLoc = "Тепловозы многосекционный";
24        else if (b[1] == '7') typeLoc = "Дизель-поезд и автомотрис";
25        else if (b[1] == '8') typeLoc = "Специальный тяговый подвижной состав (мотовозы, автодрезины и т.д.)";
26        else if (b[1] == '9') typeLoc = "Путевые машины";
27
28        this.TypeLoc = typeLoc;
29        return typeLoc;

```

Рисунок 8 – реализация класса NumLocomotiveDate

```

32 public string GetRodActiv(char[] b)
33 {
34     string rodSl = "";
35     if (b[2] == '0') rodSl = "Пассажирский, скоростной";
36     else if (b[2] == '1') rodSl = "Грузовой, скоростной";
37     else if (b[2] == '2' || b[2] == '3' || b[2] == '4') rodSl = "Грузовой";
38     else if (b[2] == '5' || b[2] == '6' || b[2] == '7') rodSl = "Резерв";
39     else if (b[2] == '8') rodSl = "Пассажирский, двойного питания";
40     else if (b[2] == '9') rodSl = "Пассажирский, переменного тока";
41
42     this.Pod_active = rodSl;
43     return rodSl;
44 }
45
46 public string GetNum(char[] b)
47 {
48     string numLoc = b[3] + " " + b[4] + " " + b[5] + " " + b[6];
49     this.Num_Loc = numLoc;
50     return numLoc;
51 }
52 }

```

Рисунок 9 – реализация класса NumLocomotiveDate(методы конвертации)



Данный класс будет реализовывать поля нашей базы данных.

```
5 usages
3 public class LocomotivData
4 {
5     public string Id = "0";
6     public NumLocomotiveDate nld = new NumLocomotiveDate();
7     public string Type_ore = "";
8     public string Count_ore = "0";
9     public string StartTime = "0";
10    public string EndTime = "0";
11    public string Price = "0";
12    public string Client = "";
13 }
```

Рисунок 10 – реализация класса LocomotivData

На рисунке 11 создаётся приватные переменные для работы с базой данных

```
5 usages 1 usage Bogdoing
7 io
8 public partial class Form1 : Form
9 {
10     public string dbconnect =
11         "Data Source=DESKTOP-432U1GM\\SQLEXPRESS;" +
12         "Initial Catalog=OOP.Course31;Integrated Security=True;MultipleActiveResultSets=True";
13     private LocomotivData ld = new LocomotivData();
14     private string infoLoc = "";
15
16     public Form1()
17     {
18         InitializeComponent();
19     }
20
21 }
```

Рисунок 11 – реализация класса Form1

```

33     private void frmSport_Shown(object sender, EventArgs e)
34     {
35         string squery = @"select distinct Num_lok from OOPLocomotive
36                             union all (select '' as Num_lok)
37                             order by Num_lok";
38         try
39         {
40             SqlConnection sconn = new SqlConnection(dbconnect);
41             SqlCommand scomm = new SqlCommand(squery, sconn);
42             DataTable stable = new DataTable();
43             SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
44             sadapter.Fill(stable);
45             comboBox1.DataSource = stable;
46             comboBox1.DisplayMember = "Num_lok";
47             sconn.Close();
48         }
49         catch (Exception exception)
50         {
51             MessageBox.Show(text: exception.Message, caption: "error", MessageBoxButtons.OK, MessageBoxIcon.Information);
52         }
53     }

```

Рисунок 12 – реализация метода frmSport\_Shown

На рисунке 12 описана переменную, в которую вставляется sql запрос который используем для обряжения к БД в блоке try, и записывает результат запроса в таблицу Combobox1.

```

squery = @"select * from OOPLocomotive";
try
{
    SqlConnection sconn = new SqlConnection(dbconnect);
    SqlCommand scomm = new SqlCommand(squery, sconn);
    DataTable stable = new DataTable();
    SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
    sadapter.Fill(stable);
    dataGridView1.DataSource = stable;
    //dataGridView1.Columns[0].Visible = false;
    dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    dataGridView1.Columns[1].DefaultCellStyle.BackColor = Color.LightBlue;
    // dataGridView1.Columns[3].DefaultCellStyle.BackColor = Color.LightBlue;
    sconn.Close();
}
catch (Exception exception)
{
    MessageBox.Show(text: exception.Message, caption: "error", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

```

Рисунок 13 – реализация метода FormAppointment\_Load

На рисунке 13 аналогично рисунку 11, присваиваем значения sql запроса в котором достаём данные о докторах в dataGridView1.

```

1 usage  new *
private void button1_Click(object sender, EventArgs e)
{
    Form2add form2Add = new Form2add();
    form2Add.ShowDialog();
}

1 usage  Bogdoing *
private void button_cansel_Click(object sender, EventArgs e)
{
    Close();
}

1 usage  Bogdoing *
private void button2_Click(object sender, EventArgs e)
{
    Form3addrandom form3Addrandom = new Form3addrandom();
    form3Addrandom.ShowDialog();
}

```

Рисунок 14 – реализация метода button1\_Click, button\_cansel\_Click и button2\_Click

В методах с рисунка 14 реализуются работа с окнами (открытие новых окон, выключение приложения).

```

1 usage  Bogdoing *
private void button4_Click(object sender, EventArgs e)
{
    string query = "";
    if (comboBox1.Text == "") // -
    {
        if (comboBox2.Text == "")
            query = @"select * from OOPLocomotive";
        else
            query = @"select * from OOPLocomotive where (Type_ore = '' + comboBox2.Text + '')";
    }
    else if (comboBox2.Text == "")
    {
        if (comboBox1.Text == "")
            query = @"select * from OOPLocomotive";
        else
            query = @"select * from OOPLocomotive where (Num_lok = '' + comboBox1.Text + '')";
    }
    else frmSport_Shown(sender, e); // Show default
}

```

Рисунок 15 – реализация метода button4\_Click

В методе button4\_Click создаётся условный оператор который будет в зависимости от состояния combobox создавать разные sql запросы в виде строк.

```

try
{
    SqlConnection conn = new SqlConnection(dbconnect);
    SqlCommand comm = new SqlCommand(query, conn);
    DataTable table = new DataTable();
    SqlDataAdapter adapter = new SqlDataAdapter(comm);
    adapter.Fill(table);

    if (tabControl1.SelectedIndex == 0)
        dataGridView1.DataSource = table;
    else if (tabControl1.Text == "Ожидание")
        dataGridView2.DataSource = table;
    else
        dataGridView3.DataSource = table;

    conn.Close();
}
catch (Exception ex)
{
    MessageBox.Show(text: ex.Message, caption: "error", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

```

Рисунок 16 – реализация метода

На рисунке 16 отправляется sql происходит подключение к БД и отображение необходимых данных в таблице в зависимости от активного элемента tobControl.

```
1 usage  Bogdoing *
private void button3_Click(object sender, EventArgs e) // Просмотр инф.о локомотиве
{
    try
    {
        FormInfo formInfo = new FormInfo(1d);
        formInfo.ShowDialog();
    }
    catch (Exception exception)
    {
        Console.WriteLine(exception);
        throw;
    }
}
```

Рисунок 17 – реализация метода button3\_Click

На рисунке 17 обработчик события нажатия кнопки для открытия окна с информацией о локомотиве.

```

1 usage  Bogdoing *
private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    button3.Text = "Просмотр информации";
    try
    {
        if (dataGridView1.Columns[dataGridView1.CurrentCell.ColumnIndex].HeaderText.ToString() ==
            "Num_lok")
        {
            ld.nld.NLD = dataGridView1.Rows[e.RowIndex].Cells[e.ColumnIndex].Value.ToString();
            //infoLoc = dataGridView1.Rows[e.RowIndex].Cells[e.ColumnIndex].Value.ToString();
            button3.Text = "Просмотр информации\r | " + ld.nld.NLD + " |";

            ld.Id = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();
            ld.Type_ore = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();
            ld.Count_ore = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();
            ld.StartTime = dataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString();
            ld.EndTime = dataGridView1.Rows[e.RowIndex].Cells[5].Value.ToString();
            ld.Price = dataGridView1.Rows[e.RowIndex].Cells[6].Value.ToString();
            ld.Client = dataGridView1.Rows[e.RowIndex].Cells[7].Value.ToString();
            MessageBox.Show(text: "cell content exp | " + ld.Id + " | " + ld.Type_ore + " | " + ld.Count_ore
                + " | " + ld.StartTime + " | " + ld.EndTime + " | " + ld.Price + " | " + ld.Client);
        }
    }
    catch (Exception exception)
    {
        MessageBox.Show(text: "cell content exp | " + exception);
        throw;
    }
}

```

Рисунок 18 – реализация метода button3\_Click

На рисунке 18 обработчик события выбора ячейки таблицы. При нажатии на ячейку таблицы столбца “Num\_lok” информация о локомотиве запишется в класс LocomotiveData.

3 usages

```
public partial class FormInfo : Form
{
    private LocomotivData ld = new LocomotivData();
    private DateTime dtstart;
    private DateTime dtend;
    1 usage
    public FormInfo(LocomotivData ld) // string infoLoc
    {
        InitializeComponent();

        textBox1.Text = ld.nld.NLD;
        char[] b = ld.nld.NLD.ToCharArray();

        string typeLoc = ld.nld.GetType(b);
        string rodSl = ld.nld.GetRodActiv(b);
        string numLoc = ld.nld.GetNum(b);

        ld.nld.TypeLoc = typeLoc;
        ld.nld.Pod_active = rodSl;
        ld.nld.Num_Loc = numLoc;

        textBox3.Text += b[1] + ld.nld.TypeLoc; //" Тип локомотива - "
        textBox4.Text += b[2] + ld.nld.Pod_active; //" Род службы - "
        textBox5.Text += ld.nld.Num_Loc;
```

Рисунок 19 – реализация конструктора класса FormInfo

На рисунке 19 в конструкторе FormInfo отображаем информацию о локомотиве в textbox.

```

string[] start = ld.StartTime.Split(separator: new char[] { ' ' }); // 0 - date / 1 - timestart
string[] end = ld.EndTime.Split(separator: new char[] { ' ' }); // 0 - date / 1 - timeend
string startTime = start[1];
string endTime = end[1];

var Sstart:string[] = start[0].Split(separator: new char[] { '.' });
var Send:string[] = end[0].Split(separator: new char[] { '.' });
string dateccS = Sstart[2] + "-" + Sstart[1] + "-" + Sstart[0];
string dateccE = Send[2] + "-" + Send[1] + "-" + Send[0];

try
{
    // "2022-12-17 14:40:52", "yyyy-MM-dd HH:mm:ss"
    dtstart = DateTime.ParseExact(SdateccS + " 14:40:52", format: "yyyy-MM-dd HH:mm:ss",
        System.Globalization.CultureInfo.InvariantCulture);
    dtend = DateTime.ParseExact(SdateccE + " 14:40:52", format: "yyyy-MM-dd HH:mm:ss",
        System.Globalization.CultureInfo.InvariantCulture);
}
catch (Exception ex)
{
    MessageBox.Show(text: ex.Message, caption: "error", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

textBox2.Text = ld.StartTime;
textBox6.Text = ld.EndTime;
textBox7.Text = dtend.Subtract(dtstart).ToString();

```

Рисунок 20 – конструктор класса FormInfo

На рисунке 20 выполняются преобразования данных о времени начала заказа и о его конце чтобы вывести эти данные в textbox.



## 8. Демонстрация разработанного приложения

Form1

Добавить запись

Создать случайную запись

Обновить

Просмотр информации | 1301347 |

Выход

Номер локомотива Тип Руды

Всё Ожидание История

	Id	Num_lok	Type_ore	Count_ore	StartTime	EndTime	Price	Client
▶	1	1301347	Гранит	262	25.09.2028	25.10.2020	4507947	TestClient
	2	1283191	Базальт	203	30.07.2022	30.08.2028	1256990	TestClient
	3	1433586	Базальт	22	10.06.2027	10.07.2029	6948986	TestClient
	4	1103952	Апатитов...	163	26.09.2023	26.10.2022	575433	TestClient
	5	1196993	Базальт	246	28.09.2025	28.10.2026	5817103	TestClient
	6	1117011	Медная р...	227	25.04.2021	25.05.2022	6007472	TestClient
	7	1176273	Марганце...	297	22.01.2022	22.02.2023	2548823	TestClient
	8	1687721	Гранит	77	11.05.2026	11.06.2028	864181	TestClient
	9	1881637	Медная р...	175	12.06.2023	12.07.2026	1579044	TestClient
*								

Рисунок 21 – пример работы формы FormAppointment

На рисунке 21 показана работа начальной формы Form1, на ней можно выбрать номер локомотива и нажатием на кнопку “Просмотр информации” открыть FormInfo.

Form1

Добавить запись

Создать случайную запись

Обновить

Просмотр информации

Выход

Номер локомотива Базальт Тип Руды

Всё Ожидание История

	Id	Num_lok	Type_ore	Count_ore	StartTime	EndTime	Price	Client
▶	2	1283191	Базальт	203	30.07.2022	30.08.2028	1256990	TestClient
	3	1433586	Базальт	22	10.06.2027	10.07.2029	6948986	TestClient
	5	1196993	Базальт	246	28.09.2025	28.10.2026	5817103	TestClient
*								

Рисунок 22 – пример работы сортировки формы FormAppointment

На рисунке 22 показана работа сортировки, формы Form1.

2	1283191	Базальт	203	30.07.2022	30.08.2028	1
---	---------	---------	-----	------------	------------	---

FormInfo

1283191

Номер локомотива

2 Электровоз многосекционный

Тип локомотива

8 Пассажирский, двойного питания

Род службы

3 1 9 1

Номер локомотива

30.07.2022 0:00:00

Начальное время

30.08.2028 0:00:00

Конечное время

2223.00:00:00

Дней до конца заказа

Рисунок 23 – пример работы формы FormInfo

На рисунке 23 информацию о конкретном локомотиве и время выполнения его заказа.

Form1

Добавить запись

Создать случайную запись

Обновить

Просмотр информации | 1283191 |

Выход

Form3addrandom

```

INSERT INTO dbo.OOPLocomotive VALUES( '1784234', 'Марганцевая руда', '55', '16/02/2023', '16/03/2024',
'1487918', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1656620', 'Медная руда', '94', '17/9/2028', '17/10/2032', '2127147',
'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1261250', 'Апатитовая руда', '34', '26/11/2022', '26/11/2025',
'2654537', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1957072', 'Базальт', '187', '29/9/2028', '29/10/2032', '4085245',
'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1901951', 'Апатитовая руда', '9', '11/9/2022', '11/10/2024',
'4435489', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1728157', 'Базальт', '80', '22/03/2023', '22/04/2024', '861016',
'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1911030', 'Медная руда', '223', '10/05/2029', '10/06/2032',
'3366425', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1205360', 'Железная руда', '159', '25/10/2024', '25/10/2025',
'3454619', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1118878', 'Марганцевая руда', '0', '10/03/2025', '10/04/2027',
'1148781', 'TestClient')
INSERT INTO dbo.OOPLocomotive VALUES( '1620757', 'Марганцевая руда', '216', '15/02/2025', '15/03/2029',

```

10

button1

Рисунок 24 – пример работы формы Form3addrandom

На рисунке 24 нажатием кнопки “Создать случайную запись” открывается окно Form3addrandom и по нажатию кнопки генерируем такое количество записей, которое указываем в элементе в нижней левой части формы.

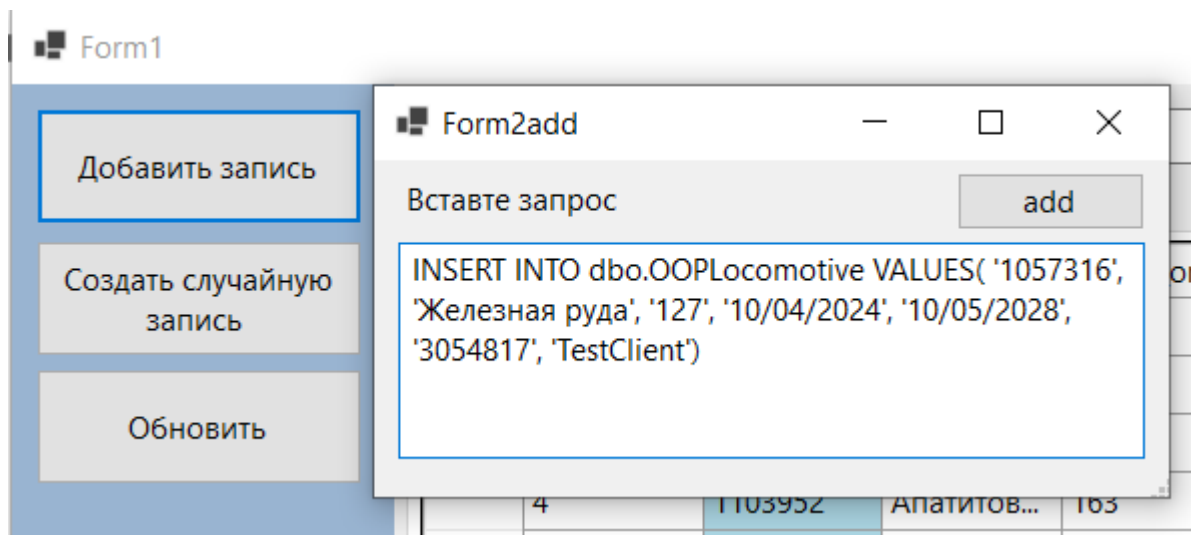


Рисунок 25 – пример работы формы Form2add

На рисунке 25 нажатием кнопки “Добавить запись” открывается окно Form2add где по нажатию кнопки генерируем запись о локомотиве записывается в базу данных.

## **ЗАКЛЮЧЕНИЕ**

В курсовой работе было разработано приложение на WinForms с использованием объектно-ориентированного программирования (ООП) и его столпов: наследования, инкапсуляция, полиморфизм, абстракция.

В ходе разработки приложения, была взята специфика десктопного приложения, средой разработки послужила Visual Studio, а технологией разработки выступила WinForms.NET(C#).

Разработанная программа обеспечивает просмотр подробную информацию о локомотивах и о их задачах.

Данная курсовая работа закрепляет знаний в области проектирования программ, а также навыков разработки и реализации пользовательских требований.

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Разработка пользователя программного обеспечения – Текст: электронный // CoderLessons.com [сайт] – URL: <https://coderlessons.com/tutorials/akademicheskii/programmnaia-inzheneriia/razrabotka-interfeisa-polzovatelia-programmnogo-obespecheniia> (дата обращения: 02.12.2022).
2. Плюсы и минусы C++ – Текст: электронный // cjblogr.blogspot.com [сайт] – URL: <https://cjblogr.blogspot.com/2015/02/blog-post.html> (дата обращения: 02.12.2022).
3. Логанов С.В. Объектно-ориентированное программирование : учебное пособие для СПО / Логанов С.В., Моругин С.Л.. — Саратов, Москва : Профобразование, Ай Пи Ар Медиа, 2022. — 215 с. — ISBN 978-5-4488-1355-9, 978-5-4497-1586-9. — Текст : электронный // IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/118969.html> (дата обращения: 02.10.2022). — Режим доступа: для авторизир. пользователей. - DOI: <https://doi.org/10.23682/118969>
4. Stack Exchange Q&A communities are different – Текст: электронный // <https://stackoverflow.com/> [сайт] – URL: <https://stackoverflow.com/> (дата обращения: 02.12.2022).
5. Adonet – Текст: электронный // metanit.com [сайт] – URL: <https://metanit.com/sharp/adonet/2.12.php> (дата обращения: 02.12.2022).
6. WITH в T-SQL // info-comp.ru [сайт] – URL: <https://info-comp.ru/obucheniist/495-the-with-in-t-sql-or-common-table-expression.html> (дата обращения: 02.12.2022).
7. Операции со строками – Текст: электронный // metanit.com [сайт] – URL: <https://metanit.com/sharp/tutorial/7.2.php#:~:text=Обрезать%20определенную%20часть%20строки%20позво>

ляет,%2F%2F%20результат%20%22ро-  
ший%20де%22%20Console.WriteLine(text) (дата обращения:  
02.12.2022).

8. With common table – Текст: электронный // learn.microsoft.com [сайт] – URL: <https://learn.microsoft.com/ru-RU/sql/t-sql/queries/with-common-table-expression-transact-sql?view=sql-server-linux-ver16> (дата обращения: 02.12.2022).
9. Sqlserver – Текст: электронный // metanit.com [сайт] – URL: <https://metanit.com/sql/sqlserver/3.3.php> (дата обращения: 02.12.2022).
10. SELECT предложение ORDER BY (Transact-SQL) – Текст: электронный // learn.microsoft.com [сайт] – URL: <https://learn.microsoft.com/ru-ru/sql/t-sql/queries/select-order-by-clause-transact-sql?view=sql-server-linux-ver16> (дата обращения: 02.12.2022).

## ПРИЛОЖЕНИЕ А

```
using OOP.Course31._1.Forms;
using System.Data;
using System.Data.SqlClient;

namespace OOP.Course31._1
{
    public partial class Form1 : Form
    {
        public string dbconnect =
            "Data Source=DESKTOP-432U1GM\\SQLEXPRESS;Initial Cata-
            log=OOP.Course31;Integrated Security=True;MultipleAc-
            tiveResultSets=True\r\n";

        private LocomotivData ld = new LocomotivData();

        private string infoLoc = "";

        public Form1()
        {
            InitializeComponent();
        }

        protected override void OnResizeBegin(EventArgs e)
        {
            SuspendLayout();
            base.OnResizeBegin(e);
        }

        protected override void OnResizeEnd(EventArgs e)
        {
            ResumeLayout();
            base.OnResizeEnd(e);
        }

        private void frmSport_Shown(object sender, EventArgs e)
        {
            string squery = @"select distinct Num_lok from OOPLocomotive
                                union all (select " as Num_lok)
                                order by Num_lok";

            try
            {
                SqlConnection sconn = new SqlConnection(dbconnect);
```

```

SqlCommand scomm = new SqlCommand(squery, sconn);
DataTable stable = new DataTable();
SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
sadapter.Fill(stable);
comboBox1.DataSource = stable;
comboBox1.DisplayMember = "Num_lok";
sconn.Close();
}
catch (Exception exception)
{
    MessageBox.Show(exception.Message, "|error", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

```

```

squery = @"select distinct Type_ore from OOPLocomotive
union all (select " as Type_ore)
order by Type_ore";

```

```

try
{
    SqlConnection sconn = new SqlConnection(dbconnect);
    SqlCommand scomm = new SqlCommand(squery, sconn);
    DataTable stable = new DataTable();
    SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
    sadapter.Fill(stable);
    comboBox2.DataSource = stable;
    comboBox2.DisplayMember = "Type_ore";
    sconn.Close();
}
catch (Exception exception)
{
    MessageBox.Show(exception.Message, "|error", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

```

```

squery = @"select * from OOPLocomotive";

```

```

try
{
    SqlConnection sconn = new SqlConnection(dbconnect);
    SqlCommand scomm = new SqlCommand(squery, sconn);
    DataTable stable = new DataTable();
    SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
    sadapter.Fill(stable);
    dataGridView1.DataSource = stable;
}

```



```

        //dataGridView1.Columns[0].Visible = false;
        dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeCol-
umnsMode.Fill;
        dataGridView1.Columns[1].DefaultCellStyle.BackColor =
Color.LightBlue;
        // dataGridView1.Columns[3].DefaultCellStyle.BackColor =
Color.LightBlue;
        sconn.Close();
    }
    catch (Exception exception)
    {
        MessageBox.Show(exception.Message, "|error", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }

```

```

        squery = @"select * from OOPLocomotive l WHERE (l.EndTime > GET-
DATE())";
        try
        {
            SqlConnection sconn = new SqlConnection(dbconnect);
            SqlCommand scomm = new SqlCommand(squery, sconn);
            DataTable stable = new DataTable();
            SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
            sadapter.Fill(stable);
            dataGridView2.DataSource = stable;
            //dataGridView1.Columns[0].Visible = false;
            dataGridView2.AutoSizeColumnsMode = DataGridViewAutoSizeCol-
umnsMode.Fill;
            dataGridView2.Columns[1].DefaultCellStyle.BackColor =
Color.LightBlue;
            //dataGridView2.Columns[3].DefaultCellStyle.BackColor =
Color.LightBlue;

            sconn.Close();
        }
        catch (Exception exception)
        {
            MessageBox.Show(exception.Message, "|error", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

```

```

        squery = @"select * from OOPLocomotive 1 WHERE (1.EndTime < GET-
DATE())";
        try
        {
            SqlConnection sconn = new SqlConnection(dbconnect);
            SqlCommand scomm = new SqlCommand(squery, sconn);
            DataTable stable = new DataTable();
            SqlDataAdapter sadapter = new SqlDataAdapter(scomm);
            sadapter.Fill(stable);
            dataGridView3.DataSource = stable;
            //dataGridView1.Columns[0].Visible = false;
            dataGridView3.AutoSizeColumnsMode = DataGridViewAutoSizeCol-
umnsMode.Fill;
            dataGridView3.Columns[1].DefaultCellStyle.BackColor =
Color.LightBlue;
            //dataGridView3.Columns[3].DefaultCellStyle.BackColor =
Color.LightBlue;
            sconn.Close();
        }
        catch (Exception exception)
        {
            MessageBox.Show(exception.Message, "|error", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        Form2add form2Add = new Form2add();
        form2Add.ShowDialog();
    }

    private void button_cansel_Click(object sender, EventArgs e)
    {
        Close();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        Form3addrandom form3Addrandom = new Form3addrandom();
        form3Addrandom.ShowDialog();
    }

```

```

private void button4_Click(object sender, EventArgs e)
{
    string query = "";
    if (comboBox1.Text == "") // -
    {
        if (comboBox2.Text == "")
            query = @"select * from OOPLocomotive";
        else
            query = @"select * from OOPLocomotive where (Type_ore = '" +
comboBox2.Text + "')";
    }
    else if (comboBox2.Text == "")
    {
        if (comboBox1.Text == "")
            query = @"select * from OOPLocomotive";
        else
            query = @"select * from OOPLocomotive where (Num_lok = '" +
comboBox1.Text + "')";
    }
    else frmSport_Shown(sender, e); // Show default

    try
    {
        SqlConnection conn = new SqlConnection(dbconnect);
        SqlCommand comm = new SqlCommand(query, conn);
        DataTable table = new DataTable();
        SqlDataAdapter adapter = new SqlDataAdapter(comm);
        adapter.Fill(table);

        if (tabControl1.SelectedIndex == 0)
            dataGridView1.DataSource = table;
        else if (tabControl1.Text == "Ожидание")
            dataGridView2.DataSource = table;
        else
            dataGridView3.DataSource = table;

        conn.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "error", MessageBoxButtons.OK, Mes-
sageBoxIcon.Information);
    }
}

```

```

    }

    private void button3_Click(object sender, EventArgs e) // Просмотр инф.о
ЛОКОМОТИВЕ
    {
        try
        {
            FormInfo formInfo = new FormInfo(Id);
            formInfo.ShowDialog();
        }
        catch (Exception exception)
        {
            Console.WriteLine(exception);
            throw;
        }
    }

    private void dataGridView1_CellContentClick(object sender, DataGridView-
CellEventArgs e)
    {
        button3.Text = "Просмотр информации";
        try
        {
            if (dataGridView1.Columns[dataGridView1.CurrentCell.Column-
Index].HeaderText.ToString() ==
                "Num_lok")
            {
                Id.nld.NLD = dataGridView1.Rows[e.RowIndex].Cells[e.Column-
Index].Value.ToString();
                //infoLoc = dataGridView1.Rows[e.RowIndex].Cells[e.Column-
Index].Value.ToString();
                button3.Text = "Просмотр информации\r | " + Id.nld.NLD + " |";

                Id.Id = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();
                Id.Type_ore = data-
GridView1.Rows[e.RowIndex].Cells[2].Value.ToString();
                Id.Count_ore = data-
GridView1.Rows[e.RowIndex].Cells[3].Value.ToString();
                Id.StartTime = data-
GridView1.Rows[e.RowIndex].Cells[4].Value.ToString();
                Id.EndTime = data-
GridView1.Rows[e.RowIndex].Cells[5].Value.ToString();
            }
        }
    }

```

```

        Id.Price = data-
GridView1.Rows[e.RowIndex].Cells[6].Value.ToString();
        Id.Client = data-
GridView1.Rows[e.RowIndex].Cells[7].Value.ToString();
        MessageBox.Show("cell content exp | " + Id.Id + " | " + Id.Type_ore +
" | " + Id.Count_ore
+ " | " + Id.StartTime + " | " + Id.EndTime + " | " + Id.Price
+ " | " + Id.Client);
    }
}
catch (Exception exception)
{
    MessageBox.Show("cell content exp | " + exception);
    throw;
}
}
}
}
}

```

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

```

```

namespace OOP.Course31._1.Forms
{
    public partial class Form3addrandom : Form
    {
        public Form3addrandom()
        {
            InitializeComponent();
        }

        Random random = new Random();

        private void button1_Click(object sender, EventArgs e)
        {

```

```

const string chars = "ABCDEFGHJKLMNOPQRSTU-
VWXYZ0123456789";
const string chars_num = "0123456789";

int id = (int)numericUpDown1.Value;
string num_lok = "";
// string type_ore = "";
string[] type_ore = { "Гранит", "Базальт", "Железная руда",
"Марганцевая руда",
"Асбестовая руда", "Апатитовая руда", "Медная руда",
"Никелевая руда"};
int count_ore = 0;
string time_start = "";
string time_end = "";

for (int i = 0; i < id; i++)
{
    num_lok = new string(Enumerable.Repeat(chars_num, 6)
        .Select(s => s[random.Next(s.Length)]).ToArray());
    num_lok = "1" + num_lok;

    int type_ore_index = random.Next(0, 7);
    count_ore = random.Next(0, 300);

    int mount = random.Next(1, 12);
    string mount_str_start = "";
    string mount_str_end = "";

    if (mount < 9)
    {
        mount_str_start = "0" + mount;
        int mount_end = (int.Parse(mount_str_start) + 1);
        mount_str_end = "0" + mount_end;
    }
    else if (mount == 9)
    {
        mount_str_start = "" + mount;
        int mount_end = (int.Parse(mount_str_start) + 1);
        mount_str_end = "" + mount_end;
    }
    else
    {
        mount_str_start = "" + mount;

```

```

        mount_str_end = mount_str_start;
    }
    //SMALLDATETIME: хранит даты и время в диапазоне от
    01/01/1900 до 06/06/2079, то есть ближайшие даты. Занимает от 4 байта.
    int day = random.Next(10, 31);
    if (day > 28)
    {
        if ((int.Parse(mount_str_start) == 2) )
        {
            mount_str_start = "" + ((int.Parse(mount_str_start) + 1));
        }
        else if ((int.Parse(mount_str_end) == 2))
        {
            mount_str_end = "" + ((int.Parse(mount_str_end) + 1));
        }
    }

    int time = random.Next(2020, 2030);
    int tend = time + random.Next(1, 5);
    time_start = "" + day + "/" + mount_str_start + "/" + time;
    time_end = "" + day + "/" + mount_str_end + "/" + tend;

    textBox1.Text += "INSERT INTO dbo.OOPLocomotive VALUES( " +
num_lok + ", " + type_ore[type_ore_index] + ", " +
        count_ore + ", " + time_start + ", " + time_end + ", " + ran-
    dom.Next(50000, 7000000) + ", " + "TestClient" + ")\r\n";

    //INSERT INTO dbo.lokomotive VALUES('1230', 'QWE', 123.1,
    01/01/1905, 01/01/2000)
    }
    }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;

```

```

using System.Threading.Tasks;
using System.Windows.Forms;

namespace OOP.Course31._1.Forms
{
    public partial class FormInfo : Form
    {
        private LocomotivData ld = new LocomotivData();
        private DateTime dtstart;
        private DateTime dtend;
        public FormInfo(LocomotivData ld) // string infoLoc
        {
            InitializeComponent();

            textBox1.Text = ld.nld.NLD;
            char[] b = ld.nld.NLD.ToCharArray();

            string typeLoc = ld.nld.GetType(b);
            string rodSl = ld.nld.GetRodActiv(b);
            string numLoc = ld.nld.GetNum(b);

            ld.nld.TypeLoc = typeLoc;
            ld.nld.Pod_active = rodSl;
            ld.nld.Num_Loc = numLoc;

            textBox3.Text += b[1] + ld.nld.TypeLoc; //" Тип локомотива - "
            textBox4.Text += b[2] + ld.nld.Pod_active; //" Род службы - "
            textBox5.Text += ld.nld.Num_Loc;

            //

            string[] start = ld.StartTime.Split(new char[] { ' ' }); // 0 - date / 1 - timestart
            string[] end = ld.EndTime.Split(new char[] { ' ' }); // 0 - date / 1 - timeend
            string startTime = start[1];
            string endTime = end[1];

            var Sstart = start[0].Split(new char[] { '.' });
            var Send = end[0].Split(new char[] { '.' });
            string dateccS = Sstart[2] + "-" + Sstart[1] + "-" + Sstart[0];
            string dateccE = Send[2] + "-" + Send[1] + "-" + Send[0];

            try

```



```

        {
            // "2022-12-17 14:40:52", "yyyy-MM-dd HH:mm:ss"
            dtstart = DateTime.ParseExact(dateccS + " 14:40:52", "yyyy-MM-dd
HH:mm:ss",
                System.Globalization.CultureInfo.InvariantCulture);
            dtend = DateTime.ParseExact(dateccE + " 14:40:52", "yyyy-MM-dd
HH:mm:ss",
                System.Globalization.CultureInfo.InvariantCulture);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "error", MessageBoxButtons.OK, Mes-
sageBoxIcon.Information);
        }

        textBox2.Text = ld.StartTime;
        textBox6.Text = ld.EndTime;
        textBox7.Text = dtend.Subtract(dtstart).ToString();

    }
}

```

```
namespace OOP.Course31._1;
```

```
public class LocomotivData
```

```

{
    public string Id = "0";
    public NumLocomotiveDate nld = new NumLocomotiveDate();
    public string Type_ore = "";
    public string Count_ore = "0";
    public string StartTime = "0";
    public string EndTime = "0";
    public string Price = "0";
    public string Client = "";
}

```

```
namespace OOP.Course31._1;
```

```
public class NumLocomotiveDate
```

```

{
    public string NLD = "";
    public string TypeLoc = "";
    public string Pod_active = "";
    public string Num_Loc = "";
}

```

```

public string GetType(char[] b)
{
    string typeLoc = "";
    // 0 - паровозы; 1 - электровозы односекционные; 2 - электровозы много-
секционные;
    // 3 - электропоезда; 4 - метрополитен; 5 - тепловозы односекционные;
    // 6 - тепловозы многосекционные; 7 - дизель-поезда и автомотрисы;
    // 8 - специальный тяговый подвижной состав (мотовозы, автодрезины и
т.д.); 9 - путевые машины.
    if (b[1] == '0') typeLoc = " Паровоз";
    else if (b[1] == '1') typeLoc = " Электровоз односекционный";
    else if (b[1] == '2') typeLoc = " Электровоз многосекционный";
    else if (b[1] == '3') typeLoc = " Электропоезд";
    else if (b[1] == '4') typeLoc = " Метрополитен";
    else if (b[1] == '5') typeLoc = " Тепловоз односекционный";
    else if (b[1] == '6') typeLoc = " Тепловозы многосекционный";
    else if (b[1] == '7') typeLoc = " Дизель-поезд и автомотрис";
    else if (b[1] == '8') typeLoc = " Специальный тяговый подвижной состав
(мотовозы, автодрезины и т.д.)";
    else if (b[1] == '9') typeLoc = " Путевые машины";

    this.TypeLoc = typeLoc;
    return typeLoc;
}

public string GetRodActiv(char[] b)
{
    string rodSl = "";
    if (b[2] == '0') rodSl = " Пассажирский, скоростной";
    else if (b[2] == '1') rodSl = " Грузовой, скоростной";
    else if (b[2] == '2' || b[2] == '3' || b[2] == '4') rodSl = " Грузовой";
    else if (b[2] == '5' || b[2] == '6' || b[2] == '7') rodSl = " Резерв";
    else if (b[2] == '8') rodSl = " Пассажирский, двойного питания";
    else if (b[2] == '9') rodSl = " Пассажирский, переменного тока";

    this.Pod_active = rodSl;
    return rodSl;
}

public string GetNum(char[] b)
{
    string numLoc = b[3] + " " + b[4] + " " + b[5] + " " + b[6];

```

```
        this.Num_Loc = numLoc;  
        return numLoc;  
    }  
}
```