

ГУАП

КАФЕДРА № 42

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

канд. техн. наук, доцент  
\_\_\_\_\_  
должность, уч. степень, звание

\_\_\_\_\_  
подпись, дата

Н. В. Богословская  
\_\_\_\_\_  
инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 1

КЛАССЫ И ОБЪЕКТЫ. ПОЛЯ КЛАССОВ, КОНСТРУКТОР ПО  
УМОЛЧАНИЮ

по курсу:

ОСНОВЫ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4326

\_\_\_\_\_  
подпись, дата

Г. С. Томчук  
\_\_\_\_\_  
инициалы, фамилия

Санкт-Петербург 2024

## СОДЕРЖАНИЕ

1	Цель работы .....	3
2	Задача.....	3
3	Ключевые позиции.....	3
3.1	Разработка интерфейса формы.....	3
3.2	Диаграмма классов .....	4
3.3	Класс сущности файловой системы.....	4
3.4	Класс файла .....	5
3.5	Класс директории.....	5
4	Тестирование программы .....	6
	ВЫВОДЫ .....	7

## 1 Цель работы

Целью лабораторной работы является создание функционирующего приложения Windows Forms на определенную тематику.

Вопросы, изучаемые в работе:

- Классы, объекты, методы, поля и работа с ними;
- C#, Windows Forms.

## 2 Задача

11. В приложении пользователь может создать объект класса Текстовый файл, используя классы Файл, Директория. Методы для работы с файлами должны полностью обеспечить пользователя возможностями создания, удаления, изменения, переименования файлов.

## 3 Ключевые позиции

### 3.1 Разработка интерфейса формы

Интерфейс был исполнен в виде трех текстовых полей, необходимых для ввода пути, имени и содержимого файла, нескольких кнопок, отвечающих за главный функционал и одной надписи, отображающей текущий рабочий файл.

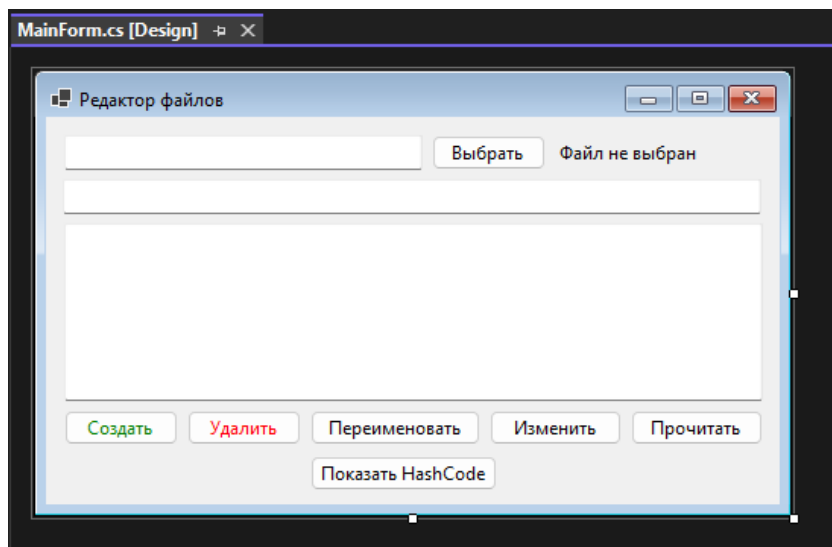


Рисунок 1

## 3.2 Диаграмма классов

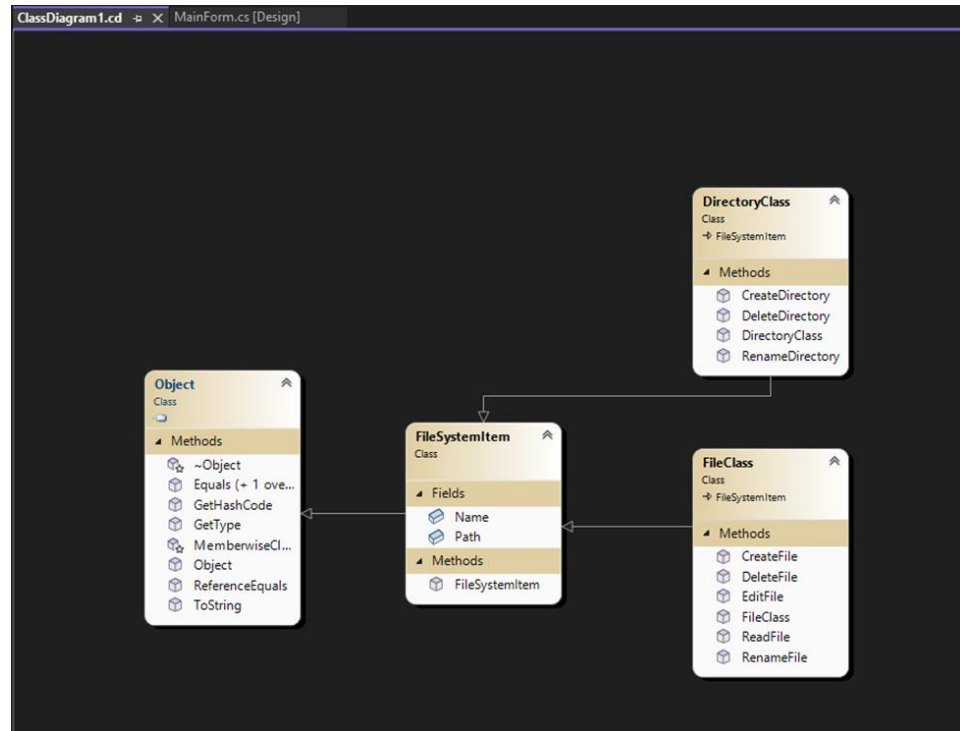


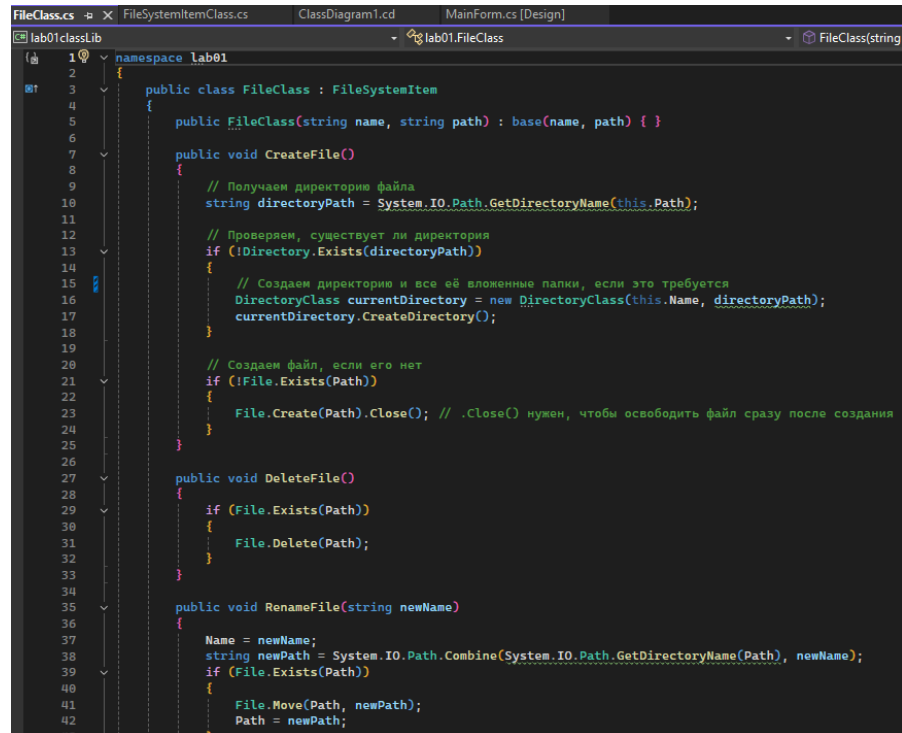
Рисунок 2

## 3.3 Класс сущности файловой системы

```
FileSystemItemClass.cs  ClassDiagram1.cd  MainForm.cs [Design]
lab01.classLib
1 namespace lab01
2 {
3     // Обобщенный класс сущности файловой системы
4     public class FileSystemItem
5     {
6         public string Name;
7         public string Path;
8
9         public FileSystemItem(string name, string path)
10        {
11            this.Name = name;
12            this.Path = path;
13        }
14    }
15 }
16
```

Рисунок 3

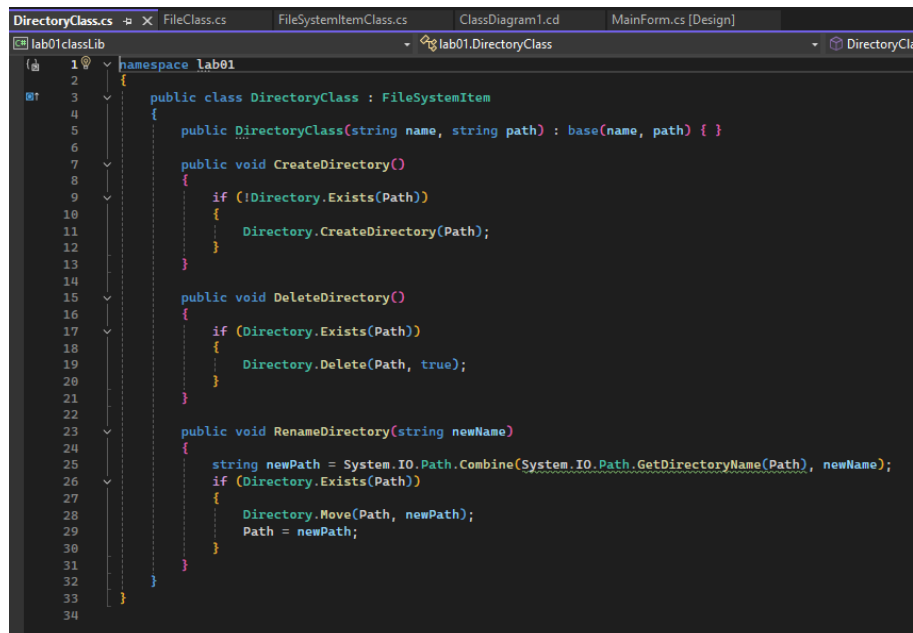
### 3.4 Класс файла



```
1 namespace lab01
2 {
3     public class FileClass : FileSystemItem
4     {
5         public FileClass(string name, string path) : base(name, path) { }
6
7         public void CreateFile()
8         {
9             // Получаем директорию файла
10            string directoryPath = System.IO.Path.GetDirectoryName(this.Path);
11
12            // Проверяем, существует ли директория
13            if (!Directory.Exists(directoryPath))
14            {
15                // Создаем директорию и все её вложенные папки, если это требуется
16                DirectoryClass currentDirectory = new DirectoryClass(this.Name, directoryPath);
17                currentDirectory.CreateDirectory();
18            }
19
20            // Создаем файл, если его нет
21            if (!File.Exists(Path))
22            {
23                File.Create(Path).Close(); // .Close() нужен, чтобы освободить файл сразу после создания
24            }
25        }
26
27        public void DeleteFile()
28        {
29            if (File.Exists(Path))
30            {
31                File.Delete(Path);
32            }
33        }
34
35        public void RenameFile(string newName)
36        {
37            Name = newName;
38            string newPath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(Path), newName);
39            if (File.Exists(Path))
40            {
41                File.Move(Path, newPath);
42                Path = newPath;
43            }
44        }
45    }
46 }
```

Рисунок 4

### 3.5 Класс директории



```
1 namespace lab01
2 {
3     public class DirectoryClass : FileSystemItem
4     {
5         public DirectoryClass(string name, string path) : base(name, path) { }
6
7         public void CreateDirectory()
8         {
9             if (!Directory.Exists(Path))
10            {
11                Directory.CreateDirectory(Path);
12            }
13        }
14
15        public void DeleteDirectory()
16        {
17            if (Directory.Exists(Path))
18            {
19                Directory.Delete(Path, true);
20            }
21        }
22
23        public void RenameDirectory(string newName)
24        {
25            string newPath = System.IO.Path.Combine(System.IO.Path.GetDirectoryName(Path), newName);
26            if (Directory.Exists(Path))
27            {
28                Directory.Move(Path, newPath);
29                Path = newPath;
30            }
31        }
32    }
33 }
34 }
```

Рисунок 5

## 4 Тестирование программы

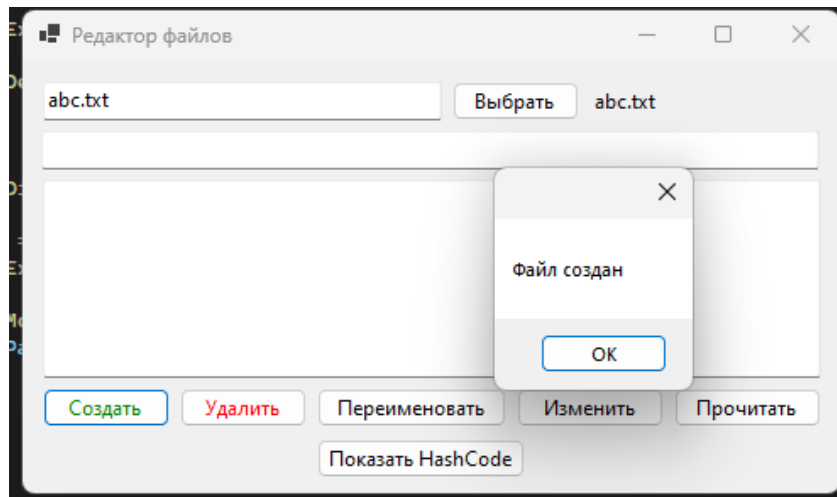


Рисунок 6

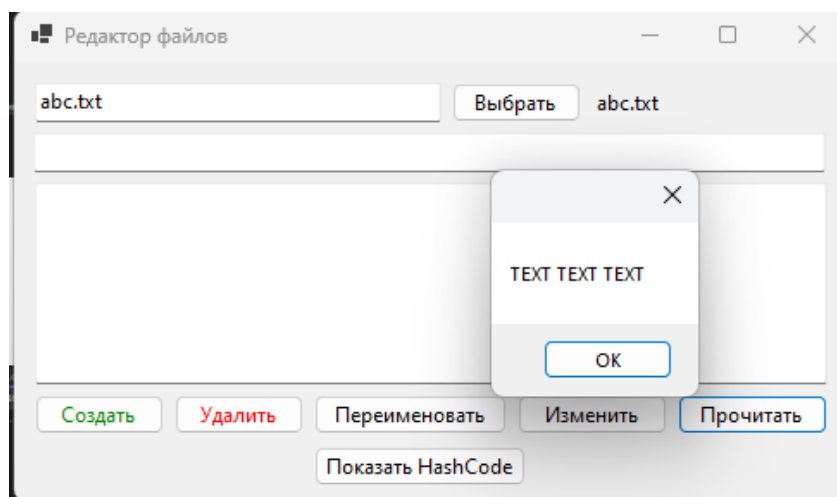


Рисунок 7

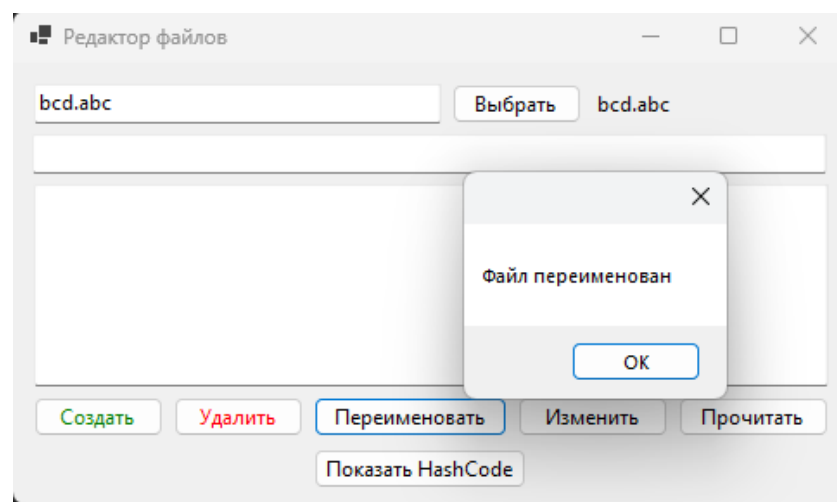


Рисунок 8

## ВЫВОДЫ

В C# классы являются основными строительными блоками объектно-ориентированного программирования. Класс — это шаблон, описывающий объект. Он включает в себя поля (данные), методы (поведение) и конструкторы для инициализации объектов. Классы помогают структурировать код и делать его более понятным. Выполняя данную работу, я усвоил следующее:

Класс может содержать поля (переменные), методы (функции), свойства, события и конструкторы.

Модификаторы доступа позволяют контролировать видимость полей и методов:

- `public`: доступен всем, кто использует объект данного класса. Поля и методы с этим модификатором могут быть вызваны из любого места программы.
- `private`: доступен только внутри самого класса. Используется для инкапсуляции данных и защиты их от внешнего вмешательства.
- `protected`: доступен внутри класса и его наследников.
- `internal`: доступен внутри текущей сборки (проекта).
- `protected internal`: доступен в пределах текущей сборки и в производных классах.

Конструктор — это специальный метод, который вызывается при создании объекта класса. Он используется для инициализации полей объекта. Конструктор по умолчанию — это конструктор без параметров, который создается автоматически, если не объявлен явно. Он инициализирует поля объекта значениями по умолчанию (например, для чисел — это 0, для ссылок — `null`).

Классы могут наследовать свойства и методы других классов с помощью ключевого слова `:`. Это позволяет использовать уже написанный функционал и расширять его.