**Лабораторная работа №10**

**Работа с файлами**

**Задание 1**

1. Создать файл, записать в него текст, прочитать и вывести в консоль.
2. Проверить существование файла перед его удалением.
3. Получить информацию о файле (размер, даты).
4. Скопировать файл и убедиться, что копия существует.
5. Переместить файл в новую директорию.
6. Переименовать файл в файл familiya.io
7. Обработать ошибку при удалении несуществующего файла.
8. Сравнить два файла по размеру.
9. Удалить все файлы в папке, соответствующие определенному шаблону. С расширением ii (см. выше)
10. Вывести список всех файлов в заданной директории.
11. Запретить запись в файл и попытаться записать в него.
12. Проверить доступные права к файлу (чтение, запись, выполнение).

**Код**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Task1

{

internal class Program

{

static void Main(string[] args)

{

FileManager fileManager = new FileManager();

FileInfoProvider fileInfoProvider = new FileInfoProvider();

string filename = "Kozel.en";

// 1. Создать файл, записать в него текст, прочитать и вывести в консоль.

fileManager.CreateFile(filename, "Привет, мир!");

Console.WriteLine(File.ReadAllText(filename));

// 2. Проверить существование файла перед его удалением.

fileManager.DeleteFile(filename);

// 3. Получить информацию о файле (размер, даты).

fileManager.CreateFile(filename, "Тестовый файл");

var info = fileInfoProvider.GetFileInfo(filename);

if (info != null)

{

Console.WriteLine($"Размер: {info.Length} байт, Дата создания: {info.CreationTime}, Дата изменения: {info.LastWriteTime}");

}

// 4. Скопировать файл и убедиться, что копия существует.

string copyFilename = "Kozel\_copy.en";

fileManager.CopyFile(filename, copyFilename);

Console.WriteLine(File.Exists(copyFilename) ? "Копия существует." : "Копия не найдена.");

// 5. Переместить файл в новую директорию.

string newDirectory = "new\_folder";

Directory.CreateDirectory(newDirectory);

fileManager.MoveFile(copyFilename, Path.Combine(newDirectory, copyFilename));

// 6. Переименовать файл в файл familiya.io

fileManager.RenameFile(Path.Combine(newDirectory, copyFilename), Path.Combine(newDirectory, "familiya.io"));

// 7. Обработать ошибку при удалении несуществующего файла.

fileManager.DeleteFile("non\_existent\_file.ii");

// 8. Сравнить два файла по размеру.

string anotherFile = "another\_file.ii";

fileManager.CreateFile(anotherFile, "Другой файл");

if (new FileInfo(filename).Length == new FileInfo(anotherFile).Length)

{

Console.WriteLine("Файлы имеют одинаковый размер.");

}

else

{

Console.WriteLine("Файлы имеют разные размеры.");

}

// 9. Удалить все файлы в папке, соответствующие определенному шаблону. С расширением ii.

fileManager.DeleteFilesWithPattern(newDirectory, ".ii");

// 10. Вывести список всех файлов в заданной директории.

var files = fileManager.ListFiles(newDirectory);

Console.WriteLine("Список файлов:");

foreach (var file in files)

{

Console.WriteLine(file);

}

// 11. Запретить запись в файл и попытаться записать в него.

File.SetAttributes(filename, FileAttributes.ReadOnly);

try

{

File.AppendAllText(filename, "Попытка записи.");

}

catch (UnauthorizedAccessException)

{

Console.WriteLine("Ошибка: Запись запрещена.");

}

// 12. Проверить доступные права к файлу (чтение, запись, выполнение).

var attributes = File.GetAttributes(filename);

Console.WriteLine($"Права файла {filename}: {attributes}");

}

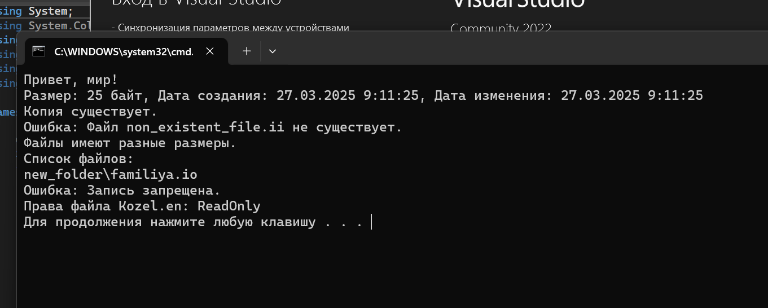
}

}

**Таблица 10.1 – Входные и выходные данные**

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
|  | **Привет, мир!**  **Размер: 25 байт, Дата создания: 27.03.2025 08:52:39, Дата изменения: 27.03.2025 08:52:39**  **Копия существует.**  **Ошибка: Файл non\_existent\_file.ii не существует.**  **Файлы имеют разные размеры.**  **Список файлов:**  **new\_folder\familiya.io**  **Ошибка: Запись запрещена.**  **Права файла Kozel.en: ReadOnly** |

**Анализ результата**

****

**Задание 2**

**Запись списка пользователей в текстовый файл**

**Классы:**

1. User – модель пользователя с полями Name, Age, Email.
2. UserFileWriter – записывает объекты User в file.data.

**Методы UserFileWriter:**

* WriteUsers(List<User> users) – записывает список пользователей в файл, каждую строку в формате:

graphql

КопироватьРедактировать

Иван,25,ivan@example.com

Ольга,30,olga@example.com

**Код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Task2

{

internal class Program

{

static void Main(string[] args)

{

List<User> users = new List<User>

{

new User("Иван", 25, "ivan@example.com"),

new User("Ольга", 30, "olga@example.com"),

new User("Петр", 22, "petr@example.com")

};

UserFileWriter userFileWriter = new UserFileWriter("file.data");

userFileWriter.WriteUsers(users);

Console.WriteLine("Данные пользователей успешно записаны в файл file.data.");

}

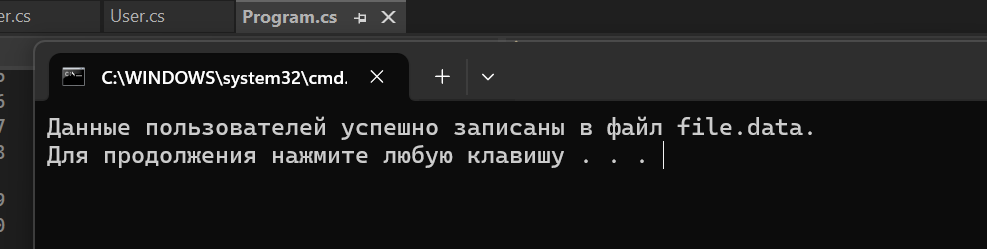
}

}

**Таблица 10.2 – Входные и выходные данные**

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
|  | **Данные пользователей успешно записаны в файл file.data.** |

**Анализ результата**

****