МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Модели безопасности компьютерных систем»

Тема: Информационный поток по памяти

Шеин В.М.

Студенты гр. 0362 _____ Циулин В.Т.

Преподаватель _____ Шкляр Е.В.

Санкт-Петербург
2023

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

Тема работы: информационный поток по памяти.

Исходные данные:

1. Написать программу, реализующую следующий функционал:

- Ввод и сохранение строки текста в файл в приватную папку (создание ценного объекта).

- Копирование по запросу пользователя данных из файла приватной папки в файл общедоступной папки.

2. Написать программу нарушителя, реализующего следующий функционал:

- Определение факта появления в общедоступной папке нового файла с информацией.

- Чтение данных из файла в буфер обмена.

- Сохранение считанных данных в свою папку (объект доступный нарушителю).

Рис. 1.1. Информационный поток по памяти:

u_1 — нарушитель; u_2 — пользователь, обрабатывающий ценную информацию; o_1 — объект, доступный нарушителю на запись; o_2 — общедоступный объект; o_3 — ценный объект

Рис. 1 – Информационный поток по памяти

ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Существуют три основные угрозы информационной безопасности: угрозы доступности, целостности и конфиденциальности.

В рамках данной работы рассматривается нарушение конфиденциальности, причиной которой являются неблагоприятные информационные потоки по памяти.

В данной работе будут фигурировать два субъекта: «пользователь», имеющий доступ к своей и общедоступной директориям, и «злоумышленник», имеющий доступ к своей и общедоступной директории.

Угроза безопасности информации – воздействие на систему, которое прямо или косвенно могут нанести ущерб ее безопасности.

Основные угрозы:

1) конфиденциальности информации – кому доступно;
2) целостности информации – повреждение при передаче;
3) доступности информации – доступно физически на заданное время;
4) раскрытия параметров компьютерной системы.

Основные виды политик безопасности:

1) Дискретная политика безопасности;
2) Мандатная политика безопасности;
3) Политика безопасности информационных потоков;

Цель: разделение потоков на множества: множество благоприятных и неблагоприятных инф-х потоков.

4) Политика ролевого разграничения доступа;
5) Политика изолированной программной среды.

РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

При запуске пользовательской программы открывается окно (рис.2), где находятся кнопки для выбора папок, поле для ввода текста, кнопки для сохранения и копирования файла. При нажатии кнопки «Сохранить» файл создается в локальной папке пользователя, ему присваивается уникальное имя используя GUID. Все файлы пользовательской папки отображаются во втором окне. При нажатии кнопки «Копировать» выбранный из списка файл копируется в общедоступную папку, файлы общедоступной папки отображаются в третьем окне.

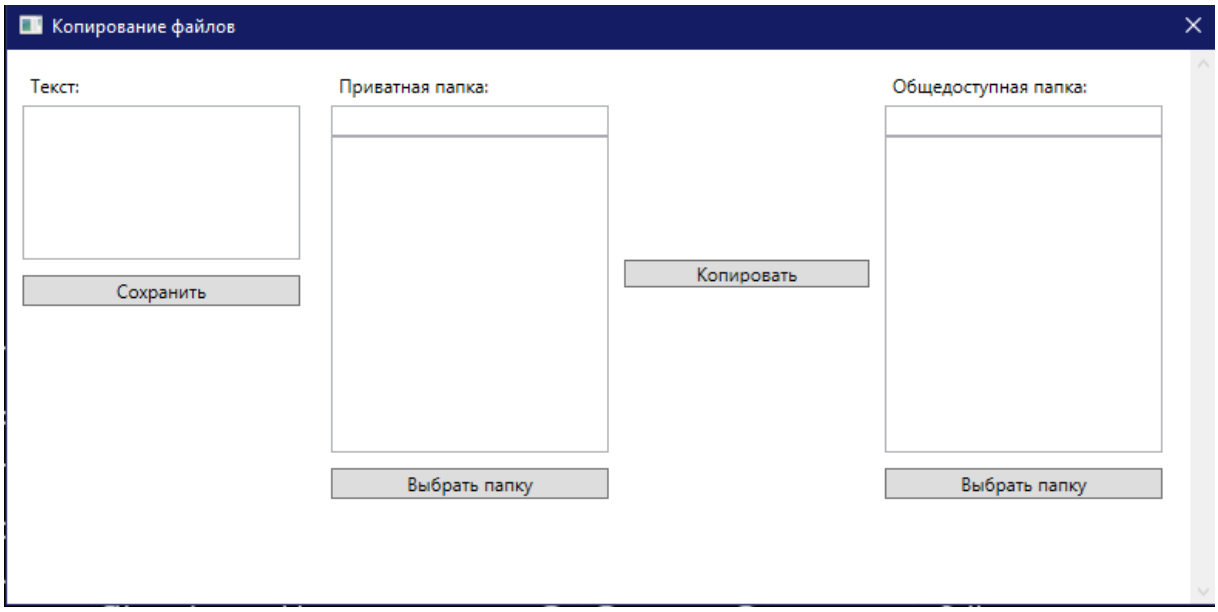

Рис.2 – Начало работы программы пользователя.

При запуске программы перехватчика открывается окно (рис. 3), оно содержит три окна и кнопки выбора папок, а также запуска слежения.

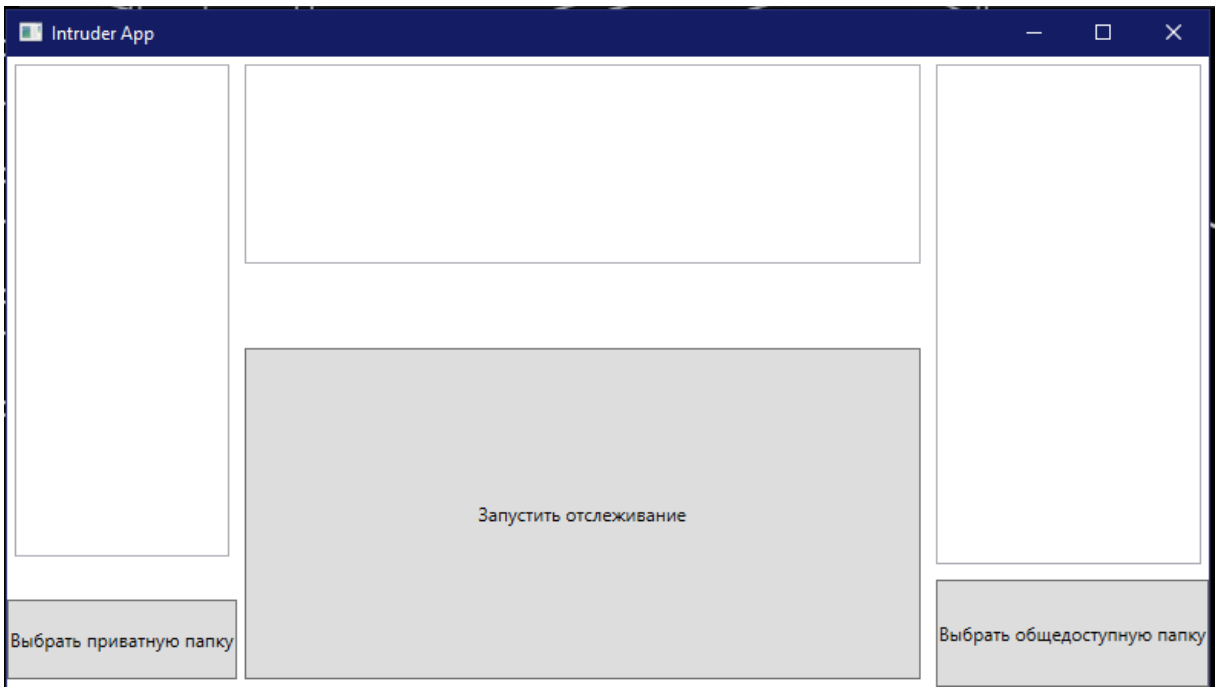

Рис.3 – Начало работы программы перехватчика

Далее отображены этапы работы программы пользователя. На рисунке 4 показано создание файла из программы. На рисунке 5 показано копирование файла из локальной директории в общедоступную.

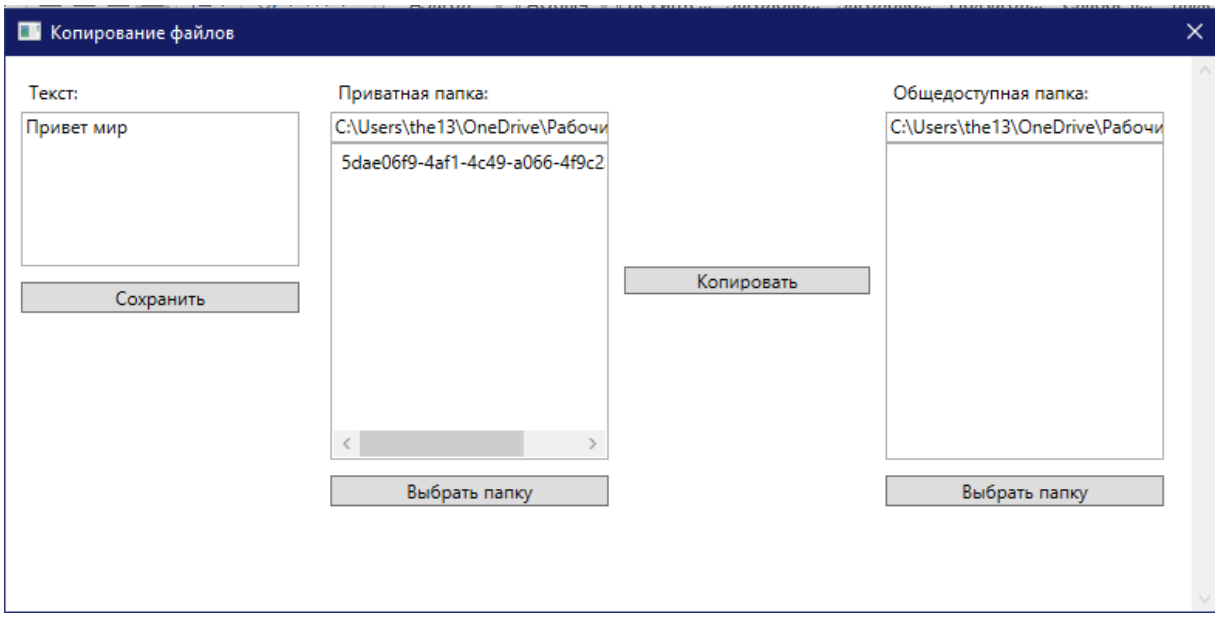

Рис.4 – Создание файла

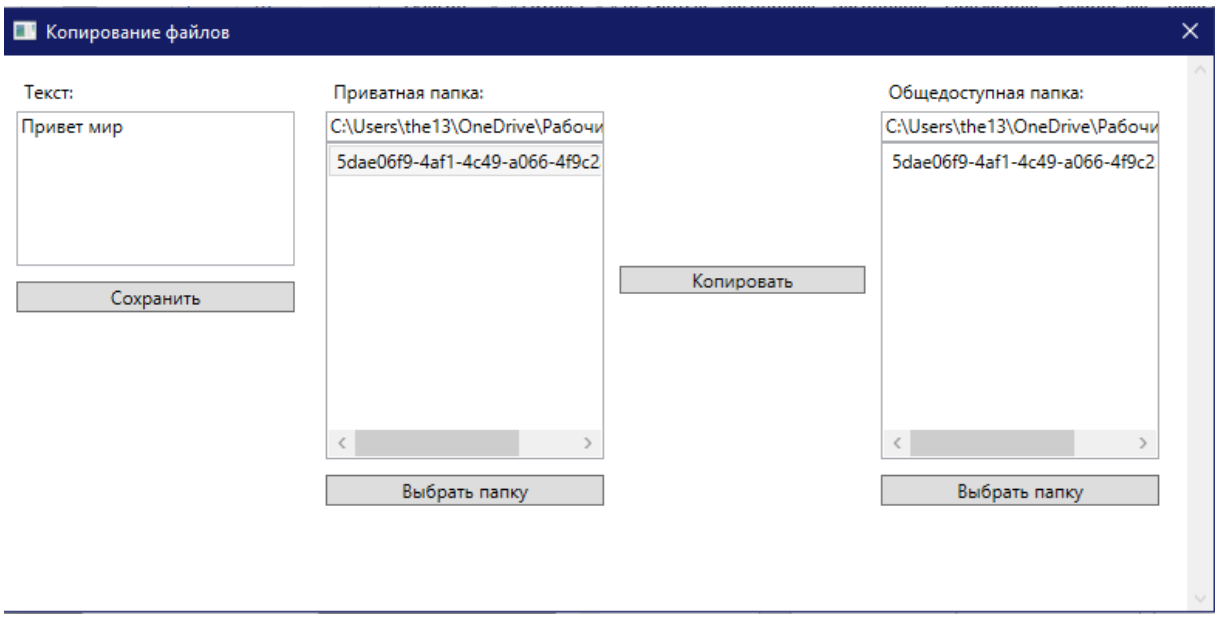

Рис.5 – Копирование файла

Далее отображены этапы работы программы перехватчика. На рисунке 6 показан запуск программы при нажатии соответствующей кнопки. На рисунке 7 показан результат работы программы.

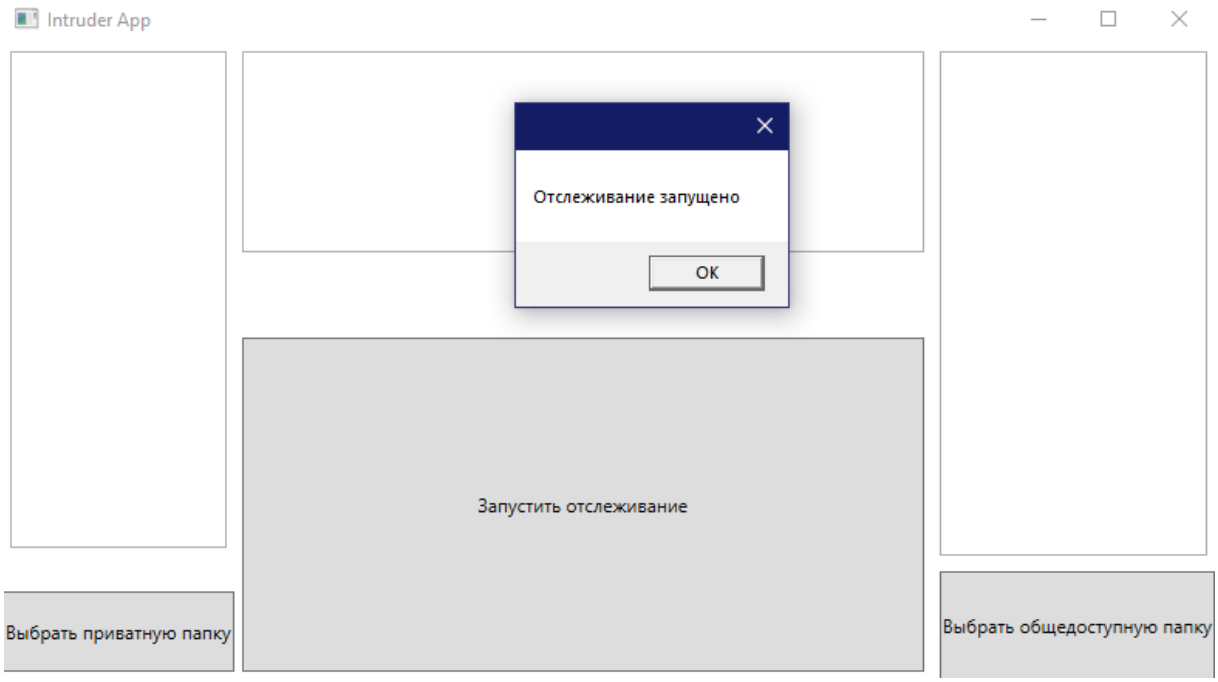

Рис.6 – Запуск программы перехватчика

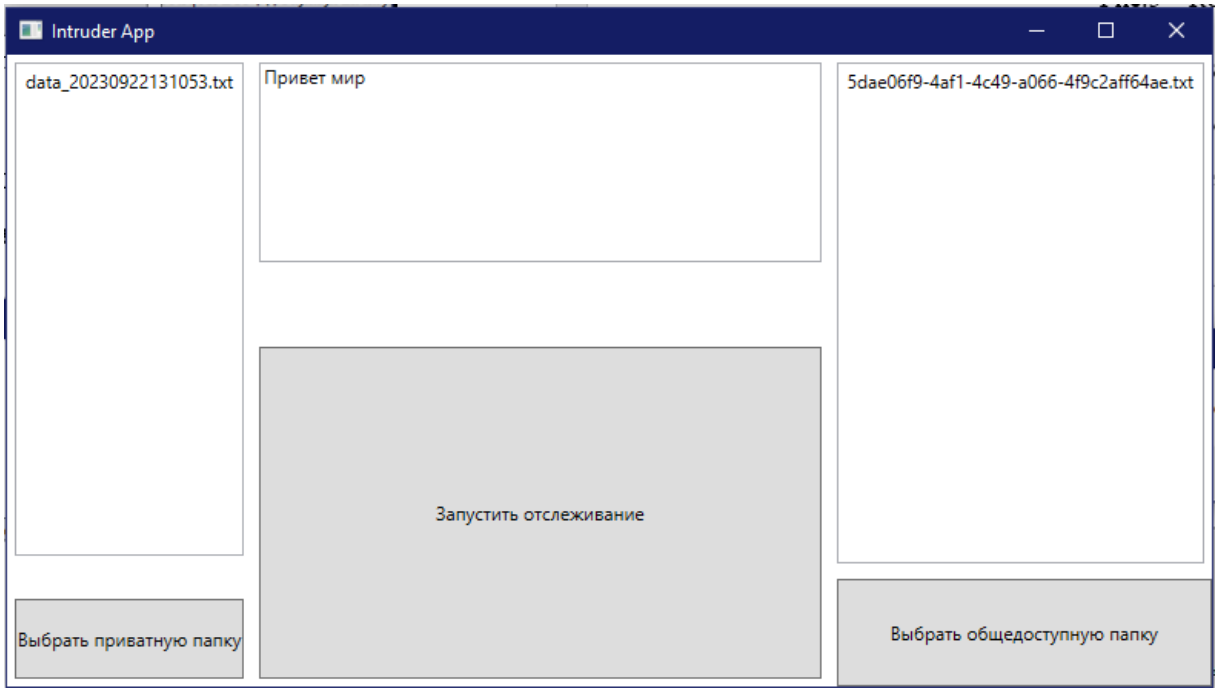

Рис.7 – Результат работы программы перехватчика

Программа перехватчик сохраняет файлы в свою папку присваивая им имя, состоящее из даты, когда был создан файл.

ВЫВОДЫ

В ходе выполнения данной лабораторной работы были изучены некоторые теоретические сведения об угрозах безопасности информации и основные виды политик безопасности. Была смоделирована ситуация, при которой неблагоприятный информационный поток по памяти приводит к нарушению конфиденциальности. Были написаны программы имитирующие действия законного пользователя и перехватчика.

На данном базовом примере хорошо показывается необходимость разграничивать права пользователей системы и правильно организовывать информационные потоки во избежание нарушения безопасности информации.

ПРИЛОЖЕНИЕ 1 – ИСХОДНЫЙ КОД

Код файла intruder.MainWindow.xaml.cs

```
using System;
using System.IO;
using System.Threading;
using System.Windows;
using System.Windows.Forms;


namespace test
{
    public partial class MainWindow : Window
    {
        private string publicFolderPath;
        private string privateFolderPath;

        public MainWindow()
        {
            InitializeComponent();
        }


        private void SelectPublicFolder_Click(object sender, RoutedEventArgs e)
        {
            using (var dialog = new FolderBrowserDialog())
            {
                if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    publicFolderPath = dialog.SelectedPath;
                    FillPublicFilesListBox();
                }
            }
        }

        private void StartFileWatcher_Click(object sender, RoutedEventArgs e)
        {
            if (string.IsNullOrEmpty(publicFolderPath))
            {
                System.Windows.Forms.MessageBox.Show("Пожалуйста, выберите
общедоступную папку", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }

            var watcher = new FileSystemWatcher(publicFolderPath);
            watcher.Created += FileCreated;


            watcher.Filter = "*.txt";
            watcher.EnableRaisingEvents = true;

            System.Windows.Forms.MessageBox.Show("Отслеживание запущено");
        }

        private void FileCreated(object sender, FileSystemEventArgs e)
        {
            Thread.Sleep(3000);
            string data = File.ReadAllText(e.FullPath);

            SaveData(data);

            Dispatcher.Invoke(() => {
                ReadDataTextBox.Text = data;
            });
```

```
            Dispatcher.Invoke(new Action(FillPrivateFilesListBox));
        }

        private void SelectPrivateFolder_Click(object sender, RoutedEventArgs e)
        {
            using (var dialog = new FolderBrowserDialog())
            {
                if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    privateFolderPath = dialog.SelectedPath;
                    FillPrivateFilesListBox();
                }
            }
        }

        private void SaveData(string data)
        {
            if (string.IsNullOrEmpty(privateFolderPath))
            {
                System.Windows.Forms.MessageBox.Show("Пожалуйста, выберите
приватную папку", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }

            string fileName = $"data_{DateTime.Now:yyyyMMddHHmmss}.txt";

            string filePath = Path.Combine(privateFolderPath, fileName);

            File.WriteAllText(filePath, data);
        }

        private void FillPublicFilesListBox()
        {
            if (string.IsNullOrEmpty(publicFolderPath))
                return;

            string[] files = Directory.GetFiles(publicFolderPath);

            PublicFilesListBox.Items.Clear();

            foreach (string file in files)
            {
                PublicFilesListBox.Items.Add(Path.GetFileName(file));
            }
        }

        private void FillPrivateFilesListBox()
        {
            if (string.IsNullOrEmpty(privateFolderPath))
                return;

            string[] files = Directory.GetFiles(privateFolderPath);

            PrivateFilesListBox.Items.Clear();

            foreach (string file in files)
            {
                PrivateFilesListBox.Items.Add(Path.GetFileName(file));
            }
        }
    }
}
```

Код файла mbks1.MainWindow.xaml.cs

```
using System;
using System.IO;
using System.Windows;
using System.Windows.Forms;
namespace mbks1
{

    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void SelectFolder1_Click(object sender, RoutedEventArgs e)
        {
            using (var dialog = new FolderBrowserDialog())
            {
                if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    folder1PathTextBox.Text = dialog.SelectedPath;
                    var files = Directory.GetFiles(dialog.SelectedPath);

                    folder1ListBox.Items.Clear();
                    foreach (var file in files)
                    {
                        folder1ListBox.Items.Add(Path.GetFileName(file));
                    }
                }
            }
        }

        private void SelectFolder2_Click(object sender, RoutedEventArgs e)
        {
            using (var dialog = new FolderBrowserDialog())
            {
                if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    folder2PathTextBox.Text = dialog.SelectedPath;
                    var files = Directory.GetFiles(dialog.SelectedPath);

                    folder2ListBox.Items.Clear();
                    foreach (var file in files)
                    {
                        folder2ListBox.Items.Add(Path.GetFileName(file));
                    }
                }
            }
        }
        private void saveButton_Click(object sender, RoutedEventArgs e)
        {
            string text = textBox.Text;
            string folder1 = folder1PathTextBox.Text;

            string fileName = Guid.NewGuid().ToString() + ".txt";
            string fullPath = Path.Combine(folder1, fileName);

            File.WriteAllText(fullPath, text);

            folder1ListBox.Items.Add(fileName);
        }

        private void CopyButton_Click(object sender, RoutedEventArgs e)
        {
```

```
            string folder1 = folder1PathTextBox.Text;
            string folder2 = folder2PathTextBox.Text;

            if (folder1ListBox.SelectedItem != null)
            {
                var selectedFile = folder1ListBox.SelectedItem.ToString();
                var fileName = selectedFile;
                var sourceFile = Path.Combine(folder1, fileName);
                var destFile = Path.Combine(folder2, fileName);

                try
                {
                    File.Copy(sourceFile, destFile, true);

                    var files = Directory.GetFiles(folder2);
                    folder2ListBox.Items.Clear();
                    foreach (var file in files)
                    {
                        folder2ListBox.Items.Add(Path.GetFileName(file));
                    }
                }
                catch (Exception ex)
                {
                    System.Windows.MessageBox.Show(ex.Message);
                }
            }
            else
            {
                System.Windows.MessageBox.Show("Выберите файл для копирования");
            }
        }

    }

}
```