Министерство цифрового развития, связи и   
массовых коммуникаций Российской Федерации

Сибирский государственный университет телекоммуникаций и информатики

# Лабораторно-практическая работа №1

# по дисциплине: «Программирование и обработка графического интерфейса»

# «Компоненты и события»

**Выполнил**: Сливинский Р.О

**Группа**: ДИМ-35

**Вариант:** 4

**Проверил**: Голованчиков С.А.

Новосибирск, 2025

**Задание** 1. Разработайте и реализуйте приложение WPF, которое: - содержит 2 тестовых поля подписанных как «А» и «Б» - четыре кнопки подписанных как «+», «-», «\*» и «/» - поле для вывода текста - пользователь может ввести числа в текстовые поля, нажать кнопку и получить в текстовом

поле результат, соответствующей арифметической операции над введёнными числами в текстовом поле.

Описание.

**1. Назначение программы**

Программа представляет собой простой калькулятор, выполняющий четыре базовые арифметические операции:

Сложение (+)

Вычитание (-)

Умножение (\*)

Деление (/)

Программа разработана на платформе WPF (Windows Presentation Foundation) с использованием языка C#.

**2. Интерфейс пользователя**

Интерфейс состоит из следующих элементов:

- Два текстовых поля для ввода чисел (А и Б)

- Четыре кнопки с арифметическими операциями

- Поле для вывода результата

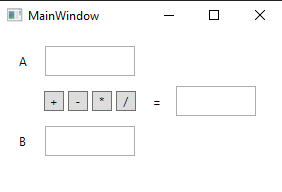


Рис1. Основной интерфейс программы.

3. Алгоритм работы.

3.1 Алгоритм обработки ввода

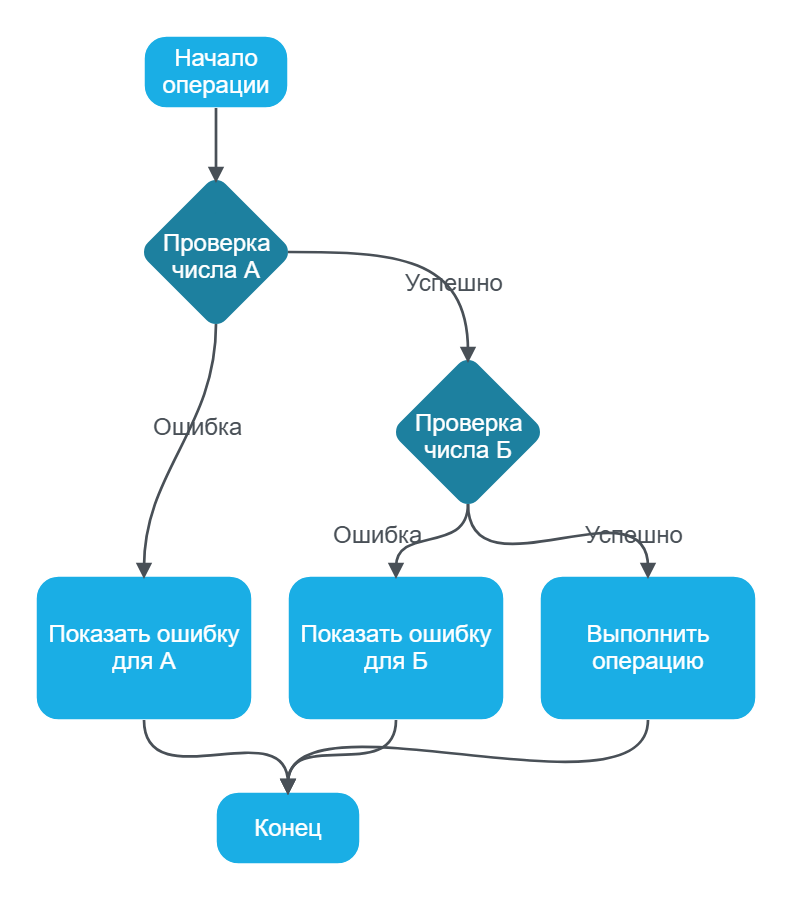


Рис2. Блок-схема алгоритма обработки ввода.

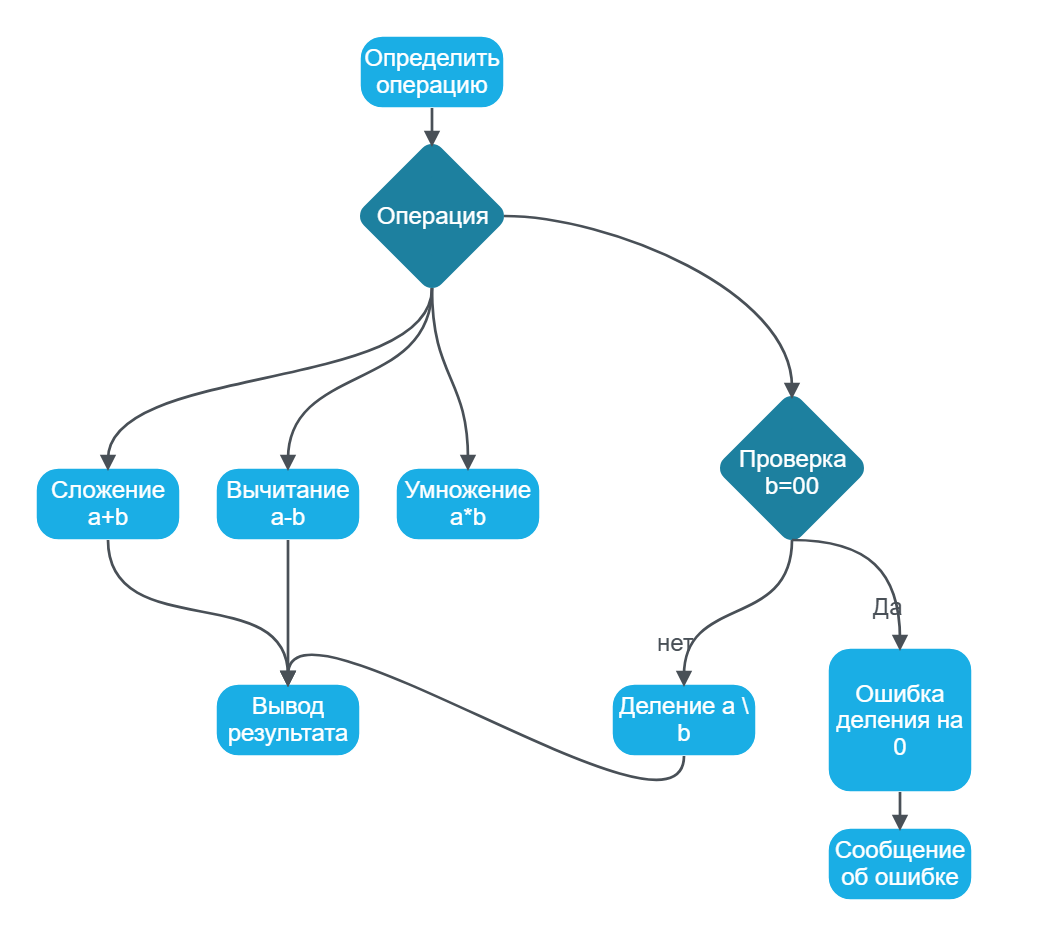
3.2 Алгоритм выполнения операций

Рис3. Блок-схема алгоритма выполнения операций.

**4. Примеры выполнения программы.**

Пример сложения:

-Вводим в поле А: 15

-Вводим в поле Б: 7

-Нажимаем кнопку «+»

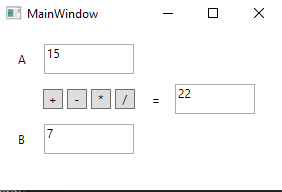


Рис4. Результат сложения.

Пример деления с ошибкой:

-Вводим в поле А: 10

-Вводим в поле Б: 0

-Нажимаем кнопку «/»

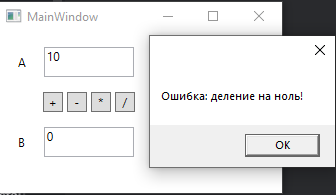


Рис5. Ошибка при делении на ноль

Пример с неверным вводом:

-Вводим в поле А: двадцать

-Вводим в поле Б: 5

-Нажимаем любую кнопку операции

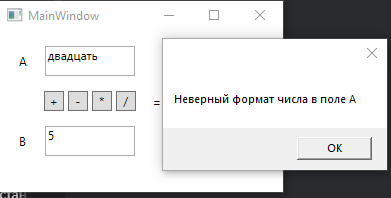


Рис 6. Сообщение о неверном формате числа

**Вывод:**

Программа представляет собой простой, но функциональный калькулятор, демонстрирующий:

- Основы работы с WPF

- Принципы обработки пользовательского ввода

- Правильную работу с ошибками

- Чистую и понятную архитектуру кода

**Задание 2.**Разработайте и реализуйте приложение WPF, которое: - содержит текстовое поле - содержит текстовый список - содержит кнопку, подписанную как “Добавить” - при нажатии на кнопку, текст, записанный в текстовом поле, должен добавляться как новая строка в текстовый список, после добавления, текстовое поле должно быть очищено.

**1. Описание:**

это простое WPF-приложение, позволяющее пользователям добавлять текстовые строки в список. Программа имеет минималистичный интерфейс и выполняет базовые функции добавления и отображения текстовых данных.

**2. Интерфейс программы**



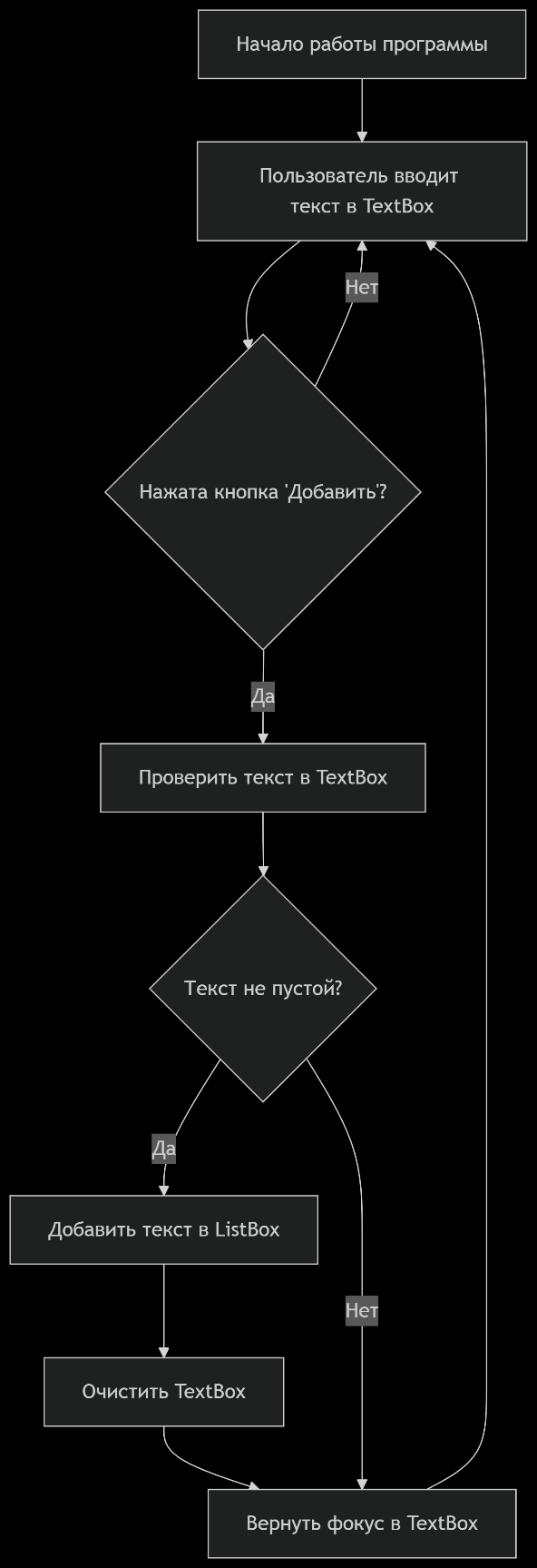
Программа состоит из трех основных элементов:

- Текстовое поле для ввода

- Список добавленных элементов

- Кнопка "Добавить"

**3. Алгоритм работы.**



**4. Логика работы:**

- Проверка, что поле ввода не пустое

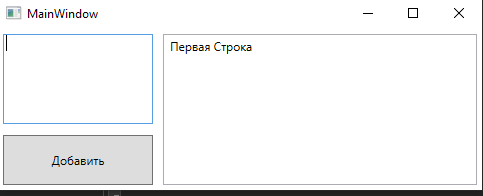
- Добавление текста в ListBox

- Очистка поля ввода

- Возврат фокуса в поле ввода

**5. Примеры выполнения**

Пример : Добавление первой строки.



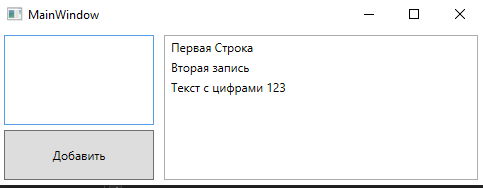
- Пользователь вводит "Первая строка"

- Нажимает кнопку "Добавить"

-Строка появляется в списке

-Поле ввода очищается

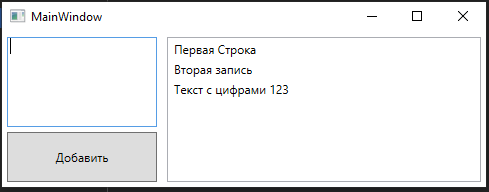
Пример : Добваление нескольких строк



-Добавлены строки: "Первая строка", "Вторая запись", "Текст с цифрами 123"

-Все строки отображаются в списке по порядку добавления

Пример : Попытка добавить пустую строку



-Пользователь оставляет поле ввода пустым

-Нажатие кнопки «Добавить» не приводит к изменениям

-В список не добавляеются пустые элементы

**6. Вывод**

Разработанная программа успешно реализует следующие функции:

-Ввод текстовых данных через интерфейс

-Добавление строк в список с проверкой на пустые значения

-Автоматическую очистку поля ввода после добавления

-Удобное отображение всех добавленных элементов

**Задание 3.** Разработайте и реализуйте приложение WPF, которое: - содержит текстовый список, содержащий названия планет солнечной системы - содержит поле для вывода текста - при выборе названия планеты из текстового списка, выводит краткую информацию о ней в поле для вывода текста.

**1. Описание:**

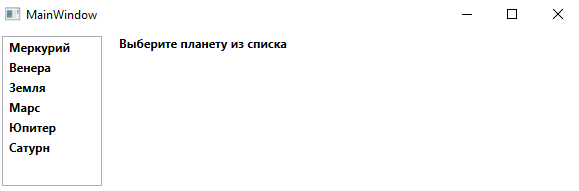
Программа представляет собой интерактивный справочник по планетам Солнечной системы с графическим интерфейсом. Основные функции:

-Отображение списка планет

-Показ детальной информации о выбранной планете

-Простота использования с интуитивным интерфейсом

**2. Интерфейс программы**

****

Интерфейс программы состоит из двух элементов:

- список планет

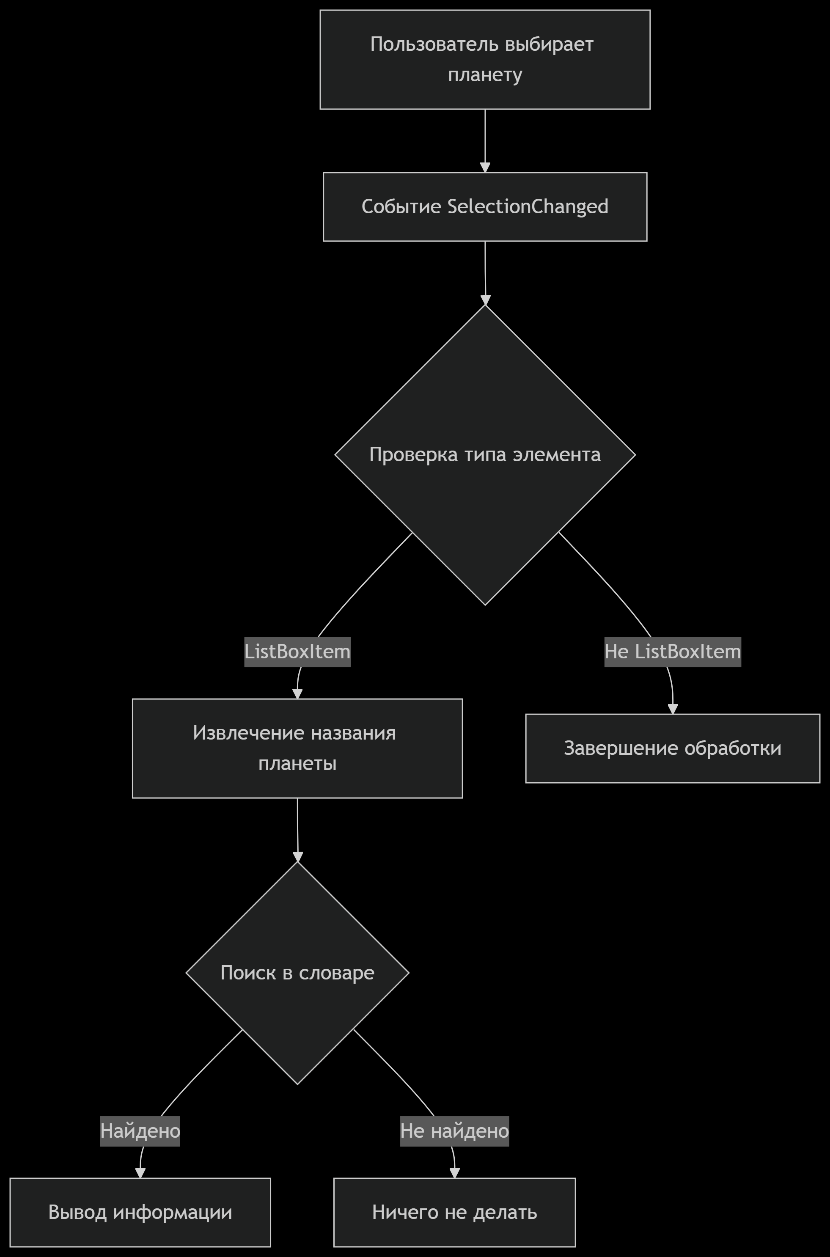
- текстовое поле с подсказкой и последующем выводе информации о планете.

**3. Алгоритм работы.**

**3.1 Алгоритм инициализации данных**

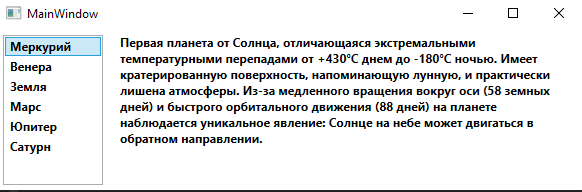
****

**3.2 Алгоритм обработки выбора планеты**

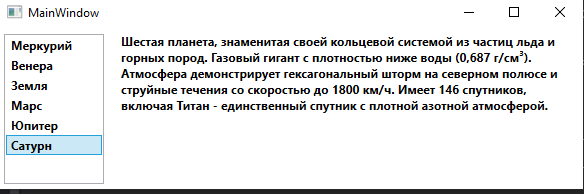
****

**4. Примеры выполнения**

Пример1: выбор планеты «Меркурий»

****

Пример 2: выбор планеты «Сатурн»



**5. Вывод**

Программа успешно решает поставленную задачу по предоставлению информации о планетах Солнечной системы. Использование WPF обеспечивает современный интерфейс, а грамотная архитектура кода позволяет легко расширять функциональность.

**Задание 4.** Разработайте и реализуйте приложение WPF, которое: - содержит три выпадающих списка, с помощью которых можно выбрать год, месяц и день - количество дней в месяце определяется только после выбора года и месяца, до этого,

выпадающий список с выбором дня должен быть не активен - после выбора всех трёх параметров, должно появляться сообщение с информацией о том,

сколько лет, месяцев и дней прошло с выбранной даты до текущего момента

**1. Описание:**

Программа позволяет:

- Выбирать произвольную дату с помощью трех выпадающих списков (год, месяц, день)

- Автоматически рассчитывать количество прошедшего времени от выбранной даты до текущего момента

Основные функциональные возможности:

1.Динамическая загрузка данных:

-Годы: от текущего года до 100 лет назад

-Месяцы: 12 месяцев с локализованными названиями

-Дни: автоматически рассчитываются в зависимости от выбранного месяца и года

2.Интеллектуальная валидация:

-Блокировка выбора дня до выбора года и месяца

-Корректная обработка високосных лет

-Проверка на будущие даты

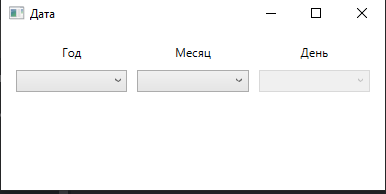
3.Точный расчет временного интервала:

-Учет разного количества дней в месяцах

-Корректный перенос месяцев/дней при отрицательных значениях

-Грамматически правильное отображение результатов

**2. Интерфейс программы.**

****

Интерфейс программы состоит из 3 блоков:

-выпадающий список года

-выпадающий список месяца

-выпадающий список дня

**3. Алгоритм работы.**

**3.1 Алгоритм расчета разницы между датами**

DateTime today = DateTime.Today;

int years = today.Year - selectedDate.Year;

int months = today.Month - selectedDate.Month;

int days = today.Day - selectedDate.Day;

if (days < 0)

{

months--;

days += DateTime.DaysInMonth(today.Year, today.Month);

}

if (months < 0)

{

years--;

months += 12;

}

**4.2. Алгоритм склонения слов**

private string GetRussianWord(int number, string form1, string form2, string form5)

{

number = Math.Abs(number) % 100;

int remainder = number % 10;

if (number > 10 && number < 20) return form5;

if (remainder > 1 && remainder < 5) return form2;

if (remainder == 1) return form1;

return form5;

}

**4.3. Алгоритм динамического заполнения дней**

int daysInMonth = DateTime.DaysInMonth(year, month);

cmbDay.Items.Clear();

for(int day = 1; day <= daysInMonth; day++)

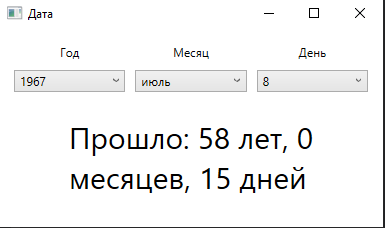
{

cmbDay.Items.Add(day);

}

**4. Пример использования программы:**

- Выбор Даты с выпадающего списка



**5. Вывод**

Разработанное приложение успешно выполняет поставленную задачу, обладает понятным интерфейсом и устойчиво к ошибкам ввода.

**Задание 5.** Разработайте и реализуйте приложение WPF, которое: - содержит меню и текстовый список - содержит в меню два пункта, позволяющие загрузить текст из выбранного, с помощью диалога открытия, файла текст в текстовый список, и сохранить текст из текстового списка в выбранный, при помощи диалога сохранения, файл

**1. Описание:**

WPF-приложение представляет собой простой текстовый редактор с возможностью:

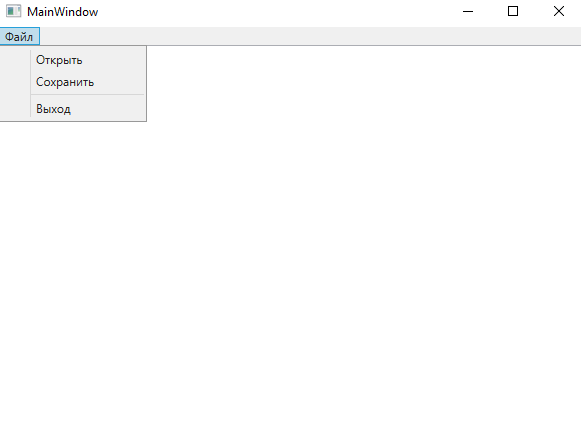
-Открытия текстовых файлов

-Редактирования текста

-Сохранения изменений в файл

-Выхода из приложения

**2. Интерфейс программы.**

****

Интерфейс программы состоит из двух элементов:

1. Меню «файл» в верхней части окна.

2. Основная область редактирование текста.

3.Полоса прокрутки (появляется при необходимости)

**3. Алгоритм работы.**

**3.1 Алгоритм открытия файла**

**1. Инициализация диалога**

-Создается объект OpenFileDialog

-Устанавливается фильтр для текстовых файлов (.txt) и всех файлов (.\*)

**2. Отображение диалога**

-Вызывается ShowDialog()

-Если пользователь нажал "Отмена" → алгоритм завершается

**3. Чтение файла**

try {

string fileContent = File.ReadAllText(openFileDialog.FileName);

TextContent.Text = fileContent;

}

catch (Exception ex) {

ShowError("Ошибка открытия файла", ex.Message);

}

**4. Обработка ошибок**

-Файл не существует

-Нет прав доступа

-Файл занят другим процессом

-Повреждение данных

**3.2. Алгоритм сохранения файла**

**1.Инициализация диалога**

-Создается объект SaveFileDialog

-Устанавливается фильтр и расширение .txt по умолчанию

**2. Отображение диалога**

-Вызывается ShowDialog()

-Если пользователь нажал "Отмена" → алгоритм завершается

**3.Запись в файл**

try {

File.WriteAllText(saveFileDialog.FileName, TextContent.Text);

}

catch (Exception ex) {

ShowError("Ошибка сохранения файла", ex.Message);

}

**4.Обработка ошибок**

-Нет прав на запись

-Диск переполнен

-Неверный путь

-Аппаратные сбои

**3.3 блок-схема основного алгоритма**

****

**Вывод:**

Разработано простое WPF-приложение для работы с текстовыми файлами. Программа позволяет открывать, редактировать и сохранять файлы через интуитивное меню с обработкой ошибок.