Министерство образования Республики Беларусь

Оршанский колледж ВГУ имени П.М. Машерова

**Отчет**

по учебной практике по программированию

«Модуль 7. Проектная работа»

Выполнил учащийся Казаченко Н.С.

группа 3ПОИС23 21.10.2025 г.

Проверил Алейников М.А.

\_\_\_\_\_\_\_\_2025 г.

Орша, 2025

**Задание:**

1. Выберите проект, который вы разработали в рамках предмета Основы алгоритмизации и программирования.
2. Сформулируйте требования к проекту и создайте план разработки.
3. Разработайте проект, следуя плану и используя полученные знания и навыки.
4. Проведите тестирование и отладку проекта.
5. Подготовьте документацию и презентацию для защиты проекта перед преподавателями.

**Решение:**

Для выполнения практического задания было взято выполнение задания из 25 лабораторной работы. Задания включает выполнение следующих аспектов:

* Разработать оконное приложение для работы с текстом с использованием компонента RichTextBox или аналога. функции (чтение файла txt, сохранение) реализовать как обработчики событий нажатия на кнопки Button.
* Добавить в приложение из задания 1 возможность настройки шрифтов в главном окне.

Составленная документация:

Требования к проекту текстового редактора.

Проект представляет собой приложение, реализующее функции текстового редактора с возможностью многодокументной работы. Основное назначение программы — обеспечение удобной работы с текстовыми документами формата TXT, включая создание новых файлов, открытие существующих, редактирование и сохранение.

Приложение разработано на языке C# с использованием технологии Windows Forms, что обеспечивает стабильность, совместимость с операционными системами Windows и простоту развёртывания. Главным элементом пользовательского интерфейса является компонент RichTextBox, предоставляющий возможность работы с текстом, поддержкой шрифтов, выделения и базового форматирования. Интерфейс приложения организован в виде главного окна, содержащего меню, статусбар и рабочую область.

Меню включает стандартные разделы: Файл, Правка, Формат, Вид и Справка. Все основные функции дублируются сочетаниями горячих клавиш. При запуске программы автоматически создаются два окна редактора, что обеспечивает многодокументный режим работы с самого начала. Функциональные возможности приложения включают полный набор стандартных операций редактирования: копирование, вырезание, вставку, удаление выделенного текста, отмену последнего действия и выделение всего содержимого.

Реализованы функции форматирования текста с помощью стандартного диалогового окна выбора шрифта, а также настройка переноса строк и цвета текста. Цвет текста применяется к выделенному фрагменту, а при отсутствии выделения — становится цветом по умолчанию для нового текста. Важной особенностью является возможность открытия нескольких окон редактора, каждое из которых работает независимо и может содержать собственный текст, шрифт, цвет и параметры отображения. При создании нового окна копируются содержимое, шрифт, цвет и текущие настройки исходного документа.

Работа с файлами реализована через стандартные диалоговые окна OpenFileDialog и SaveFileDialog, предоставляемые пространством имён System.Windows.Forms. Для открытия используется метод File.ReadAllText, который загружает содержимое файла в компонент RichTextBox. Для сохранения применяется метод File.WriteAllText, который сохраняет текущий текст документа в выбранный пользователем файл с кодировкой UTF-8.

Заголовок окна обновляется при каждой операции, чтобы отображать имя открытого или сохранённого файла либо статус «Новый документ». Меню Файл включает команды: Новый документ, Открыть, Сохранить, Новое окно. Меню Правка содержит пункты: Отменить, Вырезать, Копировать, Вставить, Удалить, Выделить всё. Меню Формат включает функции выбора шрифта, цвета текста и управления переносом строк. Меню Вид позволяет переключать тёмную и светлую темы оформления. Для настройки шрифта используется стандартный диалог FontDialog, после выбора шрифта его параметры применяются ко всему тексту в RichTextBox.

Перенос строк управляется свойством WordWrap компонента RichTextBox. При изменении состояния пункта меню «Перенос по словам» переключается значение между true и false, что изменяет поведение прокрутки текста в окне: при включённом переносе отображается только вертикальная полоса прокрутки, при выключенном — обе. Многодокументная структура приложения реализуется путём создания новых экземпляров формы Form1. При выборе команды «Новое окно» создаётся новый объект Form1, которому передаётся содержимое текущего RichTextBox, шрифт, цвет текста и настройки переноса. Каждый экземпляр окна работает независимо, что позволяет редактировать несколько документов параллельно.

Система горячих клавиш реализована через свойства ShortcutKeys пунктов меню и обработку событий KeyDown. Сочетания клавиш: Ctrl+N — для создания нового документа, Ctrl+Shift+N — для создания нового окна, Ctrl+O — для открытия файла, Ctrl+S — для сохранения, Ctrl+C — для копирования, Ctrl+V — для вставки, Ctrl+X — для вырезания, Ctrl+Z — для отмены, Ctrl+A — для выделения всего текста, Del — для удаления выделенного. Дополнительно реализована статистика текста в статусбаре, обновляемая в реальном времени: количество символов, слов и строк. Реализовано автосохранение каждые 60 секунд во временный файл с предложением восстановления при запуске.

Добавлена смена темы оформления (тёмная/светлая) с сохранением визуальной согласованности всех элементов интерфейса. Архитектура приложения построена на использовании одного класса Form1, содержащего как элементы интерфейса, так и логику обработки. Основные функции вынесены в отдельные методы и обработчики событий, что обеспечивает удобство сопровождения и расширения программы. Р

азработка приложения выполняется в несколько этапов. На первом этапе создаётся основное окно с компонентом RichTextBox, меню и статусбаром. Настраиваются диалоги открытия, сохранения и выбора шрифта. На втором этапе реализуются функции работы с файлами: открытие, сохранение, создание нового документа и обновление заголовка окна.

На третьем этапе добавляются операции редактирования текста, буфер обмена и горячие клавиши. Четвёртый этап включает реализацию функций форматирования, цвета текста, переноса строк и многодокументного интерфейса. Пятый этап — добавление статистики, автосохранения и смены темы. На заключительном этапе выполняется тестирование, отладка и улучшение визуального оформления интерфейса. Процесс тестирования охватывает все основные аспекты работы приложения. Проверяется корректность открытия и сохранения файлов, включая поддержку UTF-8. Тестируется работа буфера обмена, операции копирования и вставки между несколькими окнами, а также корректность выполнения команд через горячие клавиши.

Особое внимание уделяется тестированию функций выбора шрифта, цвета выделенного текста, переноса строк и копирования настроек при создании нового окна. Проверяется обновление заголовка окна, работа автосохранения и восстановления, стабильность при открытии и редактировании нескольких документов одновременно, а также корректность переключения тем. Результаты тестирования подтверждают, что приложение корректно реализует все заявленные функции.

Программа успешно выполняет операции чтения и записи файлов, редактирования и форматирования текста, создания новых окон и работы с несколькими документами. Интерфейс отличается простотой и удобством, а использование Windows Forms обеспечивает стабильность и производительность. Текстовый редактор готов к эксплуатации и полностью соответствует требованиям технического задания.

Листинг 1. Код приложения текстового редактора.

using System;

using System.IO;

using System.Windows.Forms;

using System.Timers;

using System.Drawing;

namespace NotepadPlus

{

public partial class Form1 : Form

{

private System.Timers.Timer autoSaveTimer;

private readonly string autoSavePath = Path.Combine(Path.GetTempPath(), "NotepadPlus\_AutoSave.txt");

private bool isDarkMode = false;

public Form1()

{

InitializeComponent();

SetupAutoSave();

UpdateStatusBar();

}

private void SetupAutoSave()

{

autoSaveTimer = new System.Timers.Timer(60000); // каждые 60 сек

autoSaveTimer.Elapsed += (s, e) => AutoSave();

autoSaveTimer.AutoReset = true;

autoSaveTimer.Start();

// Восстановление при запуске

if (File.Exists(autoSavePath))

{

if (MessageBox.Show("Восстановить автосохранённый текст?", "Восстановление",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

richTextBox1.Text = File.ReadAllText(autoSavePath);

this.Text = "Текстовый редактор - Восстановлено";

}

File.Delete(autoSavePath);

}

}

private void AutoSave()

{

if (!string.IsNullOrWhiteSpace(richTextBox1.Text))

{

try { File.WriteAllText(autoSavePath, richTextBox1.Text); }

catch { }

}

}

protected override void OnFormClosed(FormClosedEventArgs e)

{

autoSaveTimer?.Stop();

if (File.Exists(autoSavePath)) File.Delete(autoSavePath);

base.OnFormClosed(e);

}

private void UpdateStatusBar()

{

int chars = richTextBox1.Text.Length;

int words = richTextBox1.Text.Split(new[] { ' ', '\n', '\r', '\t' }, StringSplitOptions.RemoveEmptyEntries).Length;

int lines = richTextBox1.Lines.Length;

statusLabel.Text = $"Символов: {chars} | Слов: {words} | Строк: {lines}";

}

private void richTextBox1\_TextChanged(object sender, EventArgs e) => UpdateStatusBar();

private void InitializeColorMenu()

{

var colors = new (string name, Color color)[]

{

("Чёрный", Color.Black),

("Синий", Color.Blue),

("Красный", Color.Red),

("Зелёный", Color.Green),

("Фиолетовый", Color.Purple),

("Оранжевый", Color.Orange)

};

foreach (var (name, color) in colors)

{

var item = new ToolStripMenuItem(name) { Tag = color };

item.Click += (s, e) =>

{

richTextBox1.ForeColor = color;

foreach (ToolStripMenuItem mi in textColorMenu.DropDownItems)

mi.Checked = false;

item.Checked = true;

};

if (name == "Чёрный") item.Checked = true;

textColorMenu.DropDownItems.Add(item);

}

}

// === Меню ===

private void NewFile() { richTextBox1.Clear(); Text = "Текстовый редактор - Новый документ"; }

private void OpenFile()

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

richTextBox1.LoadFile(openFileDialog1.FileName, RichTextBoxStreamType.PlainText);

Text = $"Текстовый редактор - {Path.GetFileName(openFileDialog1.FileName)}";

}

}

private void SaveFile()

{

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

richTextBox1.SaveFile(saveFileDialog1.FileName, RichTextBoxStreamType.PlainText);

Text = $"Текстовый редактор - {Path.GetFileName(saveFileDialog1.FileName)}";

}

}

private void NewWindow() => new Form1 { Text = "Текстовый редактор - Новое окно" }.Show();

private void ChangeFont()

{

fontDialog1.Font = richTextBox1.Font;

if (fontDialog1.ShowDialog() == DialogResult.OK)

richTextBox1.Font = fontDialog1.Font;

}

private void ToggleWordWrap()

{

richTextBox1.WordWrap = wordWrapItem.Checked;

richTextBox1.ScrollBars = wordWrapItem.Checked

? RichTextBoxScrollBars.Vertical

: RichTextBoxScrollBars.Both;

}

private void ToggleDarkMode()

{

isDarkMode = darkModeItem.Checked;

Color back = isDarkMode ? Color.FromArgb(30, 30, 30) : SystemColors.Window;

Color fore = isDarkMode ? Color.LightGray : SystemColors.WindowText;

Color menu = isDarkMode ? Color.FromArgb(45, 45, 45) : SystemColors.Control;

this.BackColor = back;

richTextBox1.BackColor = back;

richTextBox1.ForeColor = fore;

menuStrip1.BackColor = menu;

menuStrip1.ForeColor = fore;

statusStrip1.BackColor = menu;

statusStrip1.ForeColor = fore;

}

private void ShowAbout()

{

MessageBox.Show(

"Блокнот+ 2.0\n" +

"Разработано на C# Windows Forms\n\n" +

"• Статистика текста\n" +

"• Автосохранение\n" +

"• Цвет текста\n" +

"• Тёмная тема\n\n" +

"Автор: Казаченко Никита\n\n" +

"О программе");

}

private void Form1\_Load(object sender, EventArgs e)

{

string[] names = { "Чёрный", "Синий", "Красный", "Зелёный", "Фиолетовый", "Оранжевый" };

Color[] colors = { Color.Black, Color.Blue, Color.Red, Color.Green, Color.Purple, Color.Orange };

for (int i = 0; i < names.Length; i++)

{

ToolStripMenuItem item = new ToolStripMenuItem(names[i]);

item.Tag = colors[i];

item.Click += (s, ev) =>

{

if (richTextBox1.SelectionLength > 0)

{

richTextBox1.SelectionColor = (Color)item.Tag;

}

else

{

richTextBox1.ForeColor = (Color)item.Tag;

}

// Снимаем галочку с других

foreach (ToolStripMenuItem m in textColorMenu.DropDownItems)

m.Checked = false;

item.Checked = true;

};

if (i == 0) item.Checked = true;

textColorMenu.DropDownItems.Add(item);

}

}

private void richTextBox1\_KeyDown(object sender, KeyEventArgs e)

{

if (e.Control)

{

switch (e.KeyCode)

{

case Keys.Z: richTextBox1.Undo(); break;

case Keys.X: richTextBox1.Cut(); break;

case Keys.C: richTextBox1.Copy(); break;

case Keys.V: richTextBox1.Paste(); break;

case Keys.A: richTextBox1.SelectAll(); break;

case Keys.N:

if (e.Shift) NewWindow(); else NewFile();

break;

}

e.Handled = true;

}

else if (e.KeyCode == Keys.Delete)

{

richTextBox1.SelectedText = "";

e.Handled = true;

}

}

}

}

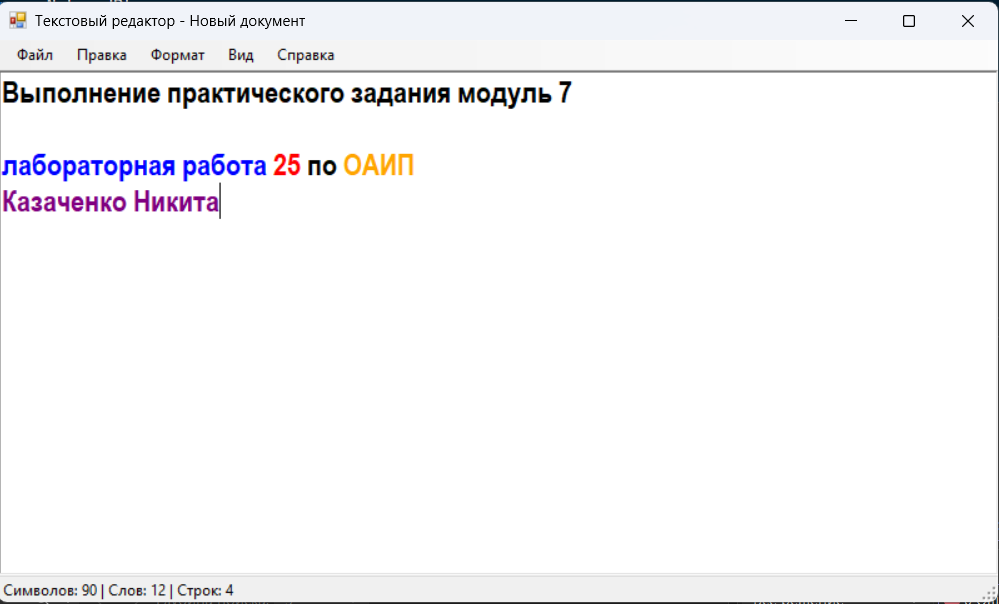


Рисунок 1 – Созданный текстовый редактор

Для завершения выполнения практического задания была разработана презентация, расположена которая в папке с проектом 7 модуля. Название презентации Редактор.pptx.

Ссылка на github: <https://github.com/Seyko00/PraktikaPoProgrammirovaniu>

**Вывод:** в ходе выполнения практического задания были получены умения создавать проектные работы.