МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра систем автоматизации управления

Дата сдачи на проверку:

«06» апреля 2024 г.

Проверено:

«06» апреля 2024 г.

**Текстовые файлы, символы, строки**

Отчёт по лабораторной работе № 2  
по дисциплине

«Технологии программирования»

Вариант 10

Выполнил студент гр. УТб-1301-02-00 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Рабинович И. А./

(Подпись)

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Симонов А. Г. /

(Подпись)

Работа защищена «16» апреля 2024 г.

Киров 2024

**1 Индивидуальное задание**

Определить количество предложений текста и количество слов в каждом предложении; по нажатию произвольной клавиши поочередно выделить каждое предложение текста, а в выделенном предложении - поочередно все слова.

**2 Схема алгоритма**

Схемы алгоритма подпрограмм FindWord, которая возвращает начальный индекс найденного слова и SplitText, которая возвращает список списков, состоящий из предложений и слов представлены на рисунке 1. Схема алгоритма основной программы представлена на рисунке 2.



Рисунок 1 – Схемы алгоритма подпрограмм FindWord (a) и SplitText (б)



Рисунок 2 ­– Схема алгоритма основной программы

**3 Листинг**

int FindWord(String FWord, int beginValue, int n) // beginValue значение с которого начинается поиск

{

int LenWord;

String ComparText;

LenWord = FWord.Length;

for (int i = beginValue; i <= n - LenWord; i++)

{

ComparText = inputBox.Text.Substring(i, LenWord);

if (ComparText == FWord)

{

return i;

}

}

return -1;

}

static List<List<string>> SplitText(string inputText)

{

List<List<string>> sentences = new List<List<string>>();

// Разделители предложений:

string[] sentenceSeparators = { ".", "!", "?" };

// Второй аргумент указывает что метод сплит должен удалить все пустые подстроки из результирующего массива:

string[] inputSentences = inputText.Split(sentenceSeparators, StringSplitOptions.RemoveEmptyEntries);

foreach (var sentence in inputSentences)

{

List<string> words = new List<string>();

// Разделители слов, можно добавлять свои:

string[] wordSeparators = { " ", ",", ";" };

// Тоже самое, только для слов:

string[] inputWords = sentence.Split(wordSeparators, StringSplitOptions.RemoveEmptyEntries);

foreach (var word in inputWords)

{

words.Add(word);

}

sentences.Add(words);

}

return sentences;

}

string MyFName = "";

private void openFile\_Click(object sender, System.EventArgs e) // Открытие файла

{

try

{

string text = "";

openFileDialog1.Filter = "Текстовые файлы (\*.rtf; \*.txt; \*.dat) | \*.rtf; \*.txt; \*.dat";

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

{

MyFName = openFileDialog1.FileName;

inputBox.LoadFile(MyFName);

text = inputBox.Text.ToString();

}

logBox.Text += $"Файл '{MyFName}' открыт." + Environment.NewLine + Environment.NewLine;

// Подсчёт кол-ва предложений

var sentences = SplitText(text);

int countSentences = sentences.Count;

logBox.Text += $"Всего предложений в файле: {countSentences}." + Environment.NewLine;

for (int i = 0; i < countSentences; i++)

{

logBox.Text += $"В {i + 1}-м предложении слов: {sentences[i].Count}." + Environment.NewLine;

}

}

catch { logBox.Text += Environment.NewLine + $"Ошибка открытия файла!" + Environment.NewLine + Environment.NewLine; }

}

int words, sentences, prevPosBeginWord, prevPosBeginSent = 0;

private void buttonPressed(object sender, KeyPressEventArgs e)

{

try

{

int LenText = inputBox.Text.Length;

var text = SplitText(inputBox.Text.ToString()); // список списков предложений и слов

List<string> sentence = text[sentences]; // список слов в предложении

inputBox.SelectionStart = 0;

inputBox.SelectionLength = LenText;

inputBox.SelectionBackColor = Color.White;

int posBeginSent = FindWord(sentence[0], prevPosBeginSent, LenText); // Позиция начала предложения

int lenSent = FindWord(sentence.Last(), prevPosBeginSent, LenText) + sentence.Last().Length - posBeginSent - 1; // Длина предложения

int posBeginWord = FindWord(sentence[words], prevPosBeginWord, LenText); // Позиция начала слова

int lenWord = sentence[words].Length; // Длина слова

inputBox.SelectionStart = posBeginSent;

inputBox.SelectionLength = lenSent + 1; // +1 т.к. индекс не включает в себя последний символ

inputBox.SelectionBackColor = Color.Yellow;

inputBox.SelectionStart = posBeginWord;

inputBox.SelectionLength = lenWord;

inputBox.SelectionBackColor = Color.Green;

if (words < sentence.Count() - 1) words++;

else

{

words = 0;

if (sentences < text.Count() - 1)

{

sentences++;

prevPosBeginSent = posBeginSent + lenSent;

}

else sentences = 0;

}

prevPosBeginWord = posBeginWord + lenWord;

}

catch

{

words = sentences = prevPosBeginWord = prevPosBeginSent = 0;

logBox.Text += Environment.NewLine + "Произведён сброс. Для продолжения нажмите любую клавишу..." + Environment.NewLine;

}

}

**4 Экранные формы**

Экранные формы работающего приложения согласно индивидуальному заданию представлены на рисунках 3 – 5.

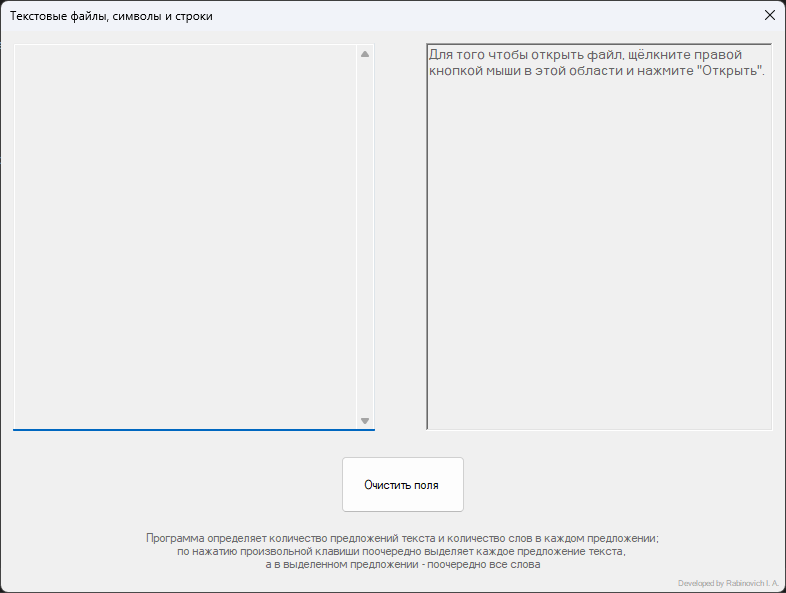


Рисунок 3 – Экранная форма

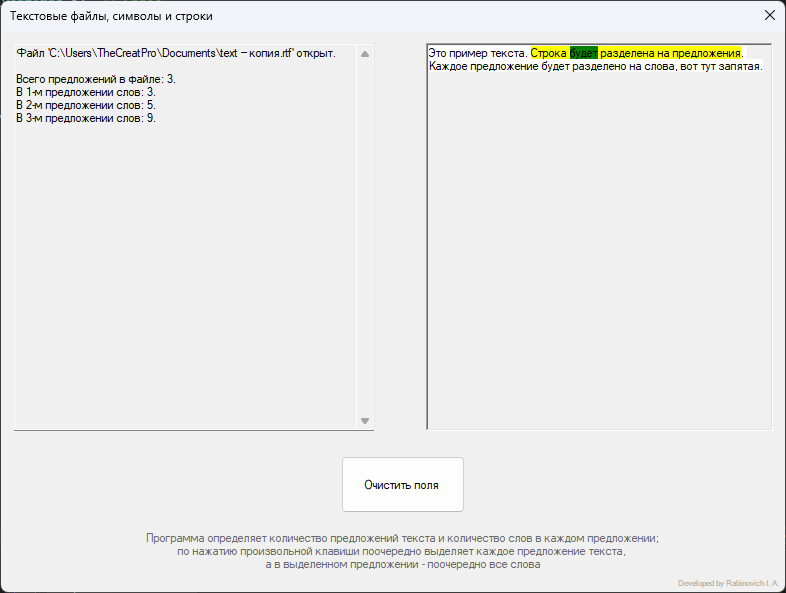


Рисунок 4 – Экранная форма

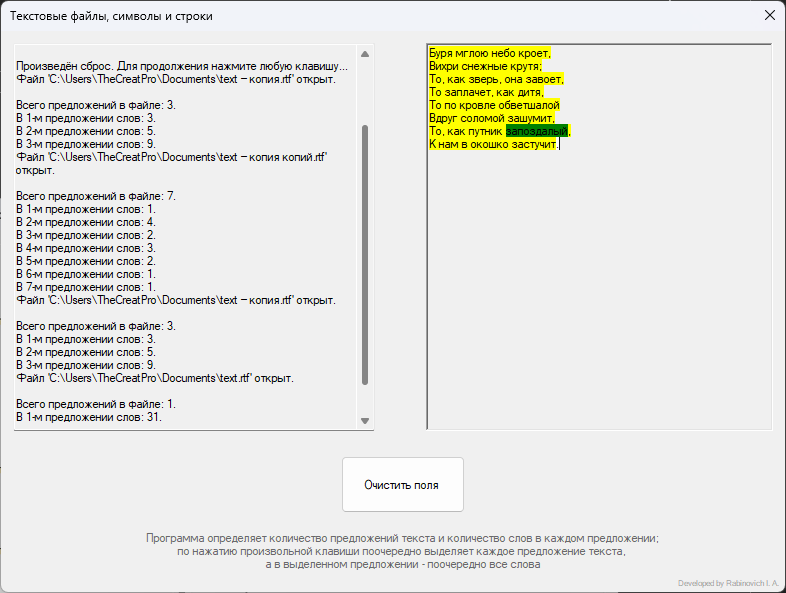


Рисунок 5 – Экранная форма

**Вывод**

После выполнения лабораторной работы было создано программное решение, которое позволяет пользователю взаимодействовать с строковыми данными через графический интерфейс и производить операции согласно индивидуальному заданию.