Лабораторная работа №8

**Тема:** Создание и применение регулярных выражений при разработке программ.

**Цель:** Научится создавать регулярные выражения и применять их при разработке программ.

**Техническое оснащение:** MS Windows 10, MS Word 2019, MS VS 2019 CE.

Выполнение работы

Вариант 18

**Пример 1.** Использование метода IsMatch(), который возвращает логическое значение.

Проверка работы первого примера (рисунок 1).

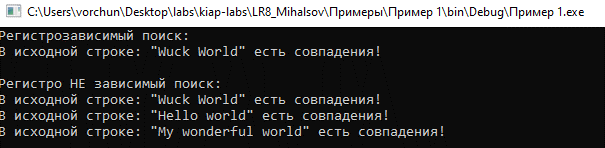


Рисунок 1 – Проверка работы примера 1

**Пример 2.** Использование свойства Match.Success

Проверка работы второго примера (рисунок 2).

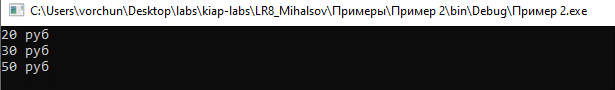


Рисунок 2 – Проверка работы примера 2

**Пример 3.** Использование метода Matches

Проверка работы третьего примера (рисунок 3).

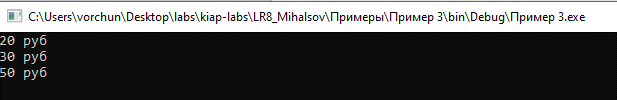


Рисунок 3 – Проверка работы примера 3

**Пример 4.** Использование метода Regex.Replace():

Проверка работы четвертого примера (рисунок 4).

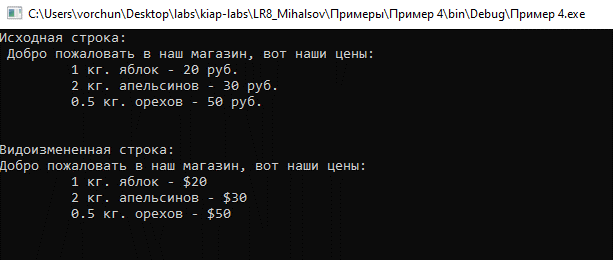


Рисунок 4 – Проверка работы примера 4

**Пример 5.** Использование методов для нахождения однокоренных слов.

Проверка работы пятого примера (рисунок 5).

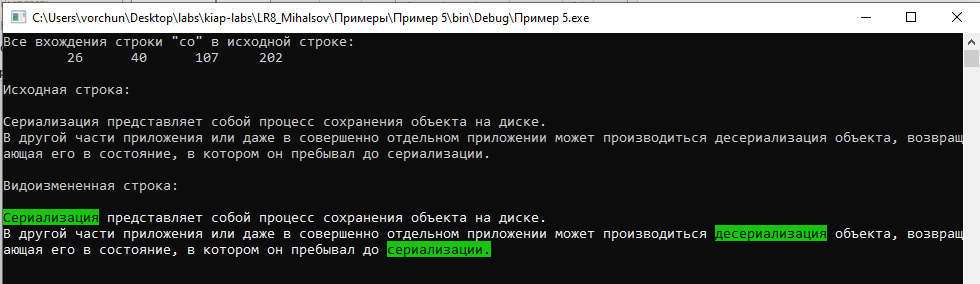


Рисунок 5 – Проверка работы примера 5

**Задание 1**. Написать программу, которая выводит на дисплей введённые текст, по нажатию клавиши поочередно выделяет в тексте лишние пробелы между словами; выводит текст на экран дисплея ещё раз, удаляя лишние пробелы между словами и начиная каждое предложение с новой строки.

Форма программы (рисунок 6).

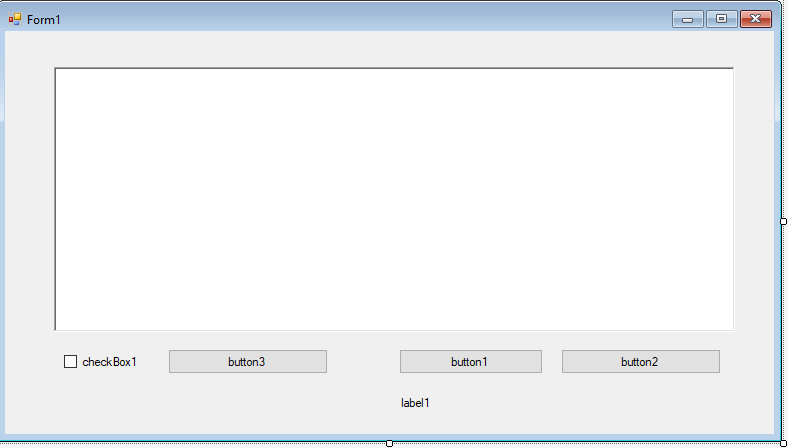


Рисунок 6 – Форма программы

На форме программы расположены следующие компоненты: richTextBox1 – который необходим для ввода пользователем текста для обработки, а также для отображения процесса обработки текста; checkBox1 – необходимый для переключения режимов обработки текста (мгновенный и последовательный); button1 – необходимый для вызова функции удаления лишних пробелов; button2 – необходимый для вызова функции разделения предложений; button3 – необходимый для итерации действий в последовательном режиме; label1 – необходимый для вывода информационных сообщений.

Преобразуем форму программы (рисунок 7).

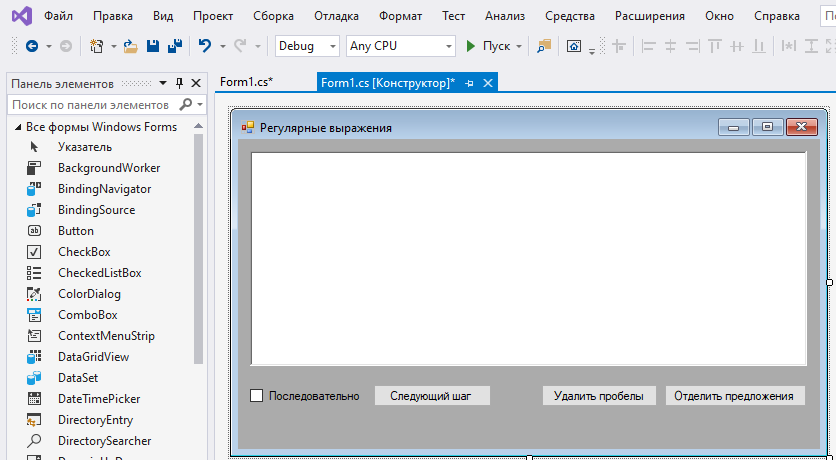


Рисунок 7 – Преобразованная форма

Листинг программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Text.RegularExpressions;

namespace Задание\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox1.Checked)

button3.Visible = true;

else

button3.Visible = false;

}

MatchCollection matchs;

Regex regex;

string spaseexp;

string spasepatert;

string finaltext;

static bool initsteping = false;

static List<int> greenList;

static int greenLenth;

string msg;

private void button3\_Click(object sender, EventArgs e)

{

if(matchs.Count == 0)

{

richTextBox1.Select(0, richTextBox1.Text.Length);

richTextBox1.SelectionBackColor = Color.White;

richTextBox1.Select(0,0);

label1.Text = finaltext;

button1.Enabled = true;

button2.Enabled = true;

button3.Enabled = false;

return;

}

if(initsteping)

{

msg = msg.Remove(matchs[0].Index, matchs[0].Length);

msg = msg.Insert(matchs[0].Index, spasepatert);

richTextBox1.Text = msg;

greenList.Add(matchs[0].Index);

}

else

{

initsteping = true;

}

matchs = regex.Matches(msg);

SelectMath(matchs);

SelectGreen(greenList, greenLenth);

}

void SelectMath(MatchCollection mc)

{

foreach (Match i in mc)

{

SelectOneColor(i.Index, i.Length, Color.Yellow);

}

}

void SelectGreen(List<int> gl, int length)

{

foreach(var i in gl)

{

SelectOneColor(i, length, Color.Green);

}

}

void SelectOneColor(int index, int length, Color c)

{

richTextBox1.Select(index, length);

richTextBox1.SelectionBackColor = c;

}

private void button1\_Click(object sender, EventArgs e)

{

if(!checkBox1.Checked)

{

fastSpace();

return;

}

spaseexp = @"\s{2,}";

spasepatert = " ";

finaltext = "Все лишние пробыле убраны";

initSteping();

}

private void button2\_Click(object sender, EventArgs e)

{

if (!checkBox1.Checked)

{

fastNewLine();

return;

}

spaseexp = @"[.][ ]";

spasepatert = ".\n";

finaltext = "Все предложения разделены";

initSteping();

}

private void initSteping()

{

button1.Enabled = false;

button2.Enabled = false;

button3.Enabled = true;

greenLenth = spasepatert.Length;

greenList = new List<int>();

regex = new Regex(spaseexp);

msg = richTextBox1.Text;

matchs = regex.Matches(msg);

}

private void fastSpace()

{

string msg = richTextBox1.Text;

string pattern = @"\s+";

string target = " ";

Regex regex = new Regex(pattern);

string result = regex.Replace(msg, target);

richTextBox1.Text = result;

label1.Text = "Все лишние пробелы были удалены";

}

private void fastNewLine()

{

string msg = richTextBox1.Text;

string pattern = @"[.]\s{0,}";

string target = ".\n";

Regex regex = new Regex(pattern);

string result = regex.Replace(msg, target);

richTextBox1.Text = result;

label1.Text = "Все предложения начинаються с новой строки";

}

}

}

Проверка функциональной реализации программы. С начала проверим первый режим. Данный режим мгновенно удаляет все пробелы в введенном тексте.

Ввод текста в форму (рисунок 8).

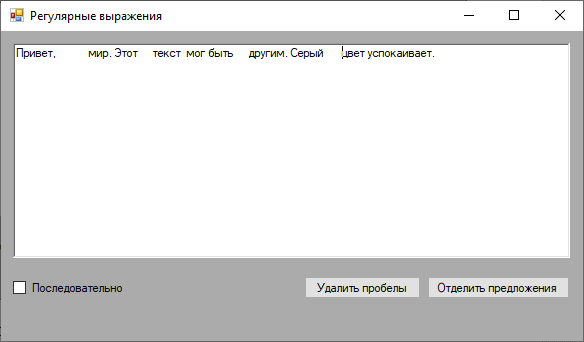


Рисунок 8 – Ввод текста в форму для первой проверки

Нажимаем на кнопку 1, для мгновенного удаления пробелов (рисунок 9).

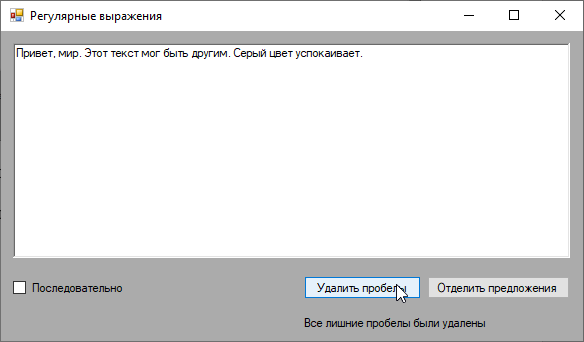


Рисунок 9 – Проверка мгновенного удаления пробелов

Теперь проверим функцию мгновенного отделения пробелов (рисунок 10).

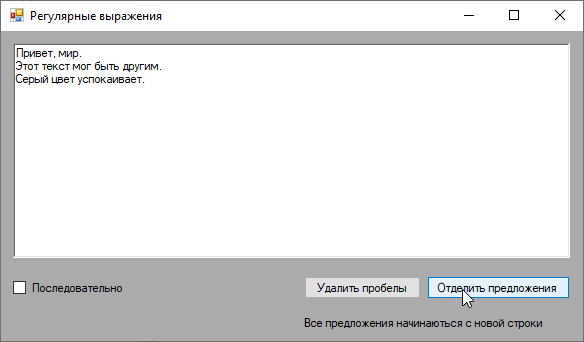


Рисунок 10 – Проверка мгновенного отделение предложений.

Проверим последовательное удаления пробелов.

Вводим текст для обработки (рисунок 11).

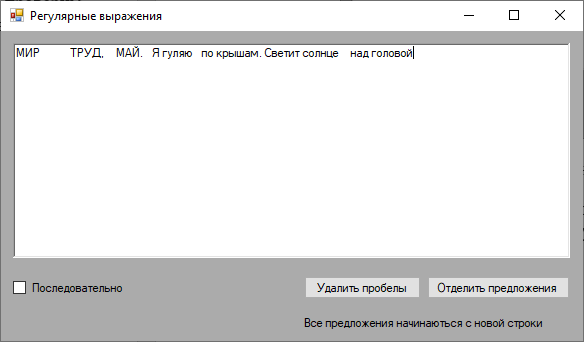


Рисунок 11 – Ввод текста для обработки

Нажимаем на checkBox1, для включения второго режима и отображения кнопки для выполнения итераций(рисунок 12).

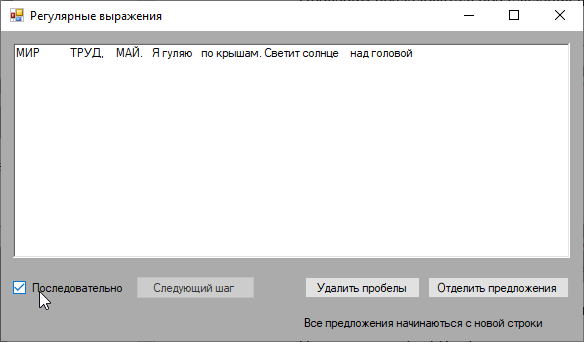


Рисунок 12 – Включения последовательного режима

Запускам последовательное удаление пробелов нажатием на первую кнопку (рисунок 13).

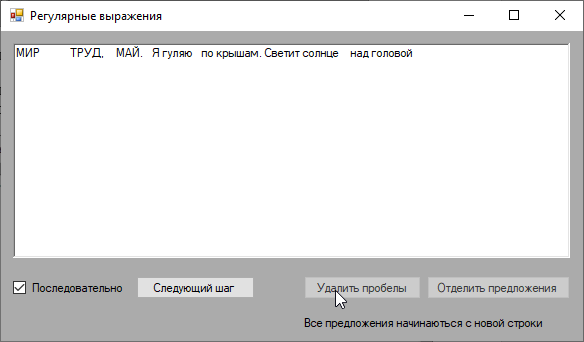


Рисунок 14 – Включение последовательного удаления пробелов

Производим первую итерацию, которая найдёт все ненужные пробелы, выделив их желтым цветом (рисунок 15).

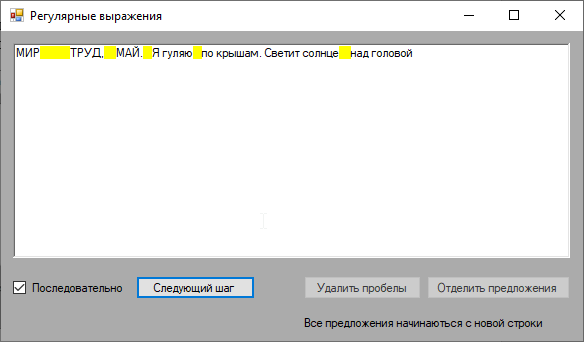


Рисунок 15 – Первая итерация по поиску всех лишних пробелов

Выполняем следующую итерацию, которая удалит первые ненужные пробелы, а заместо этого выделит это место зеленным цветом (рисунок 16).

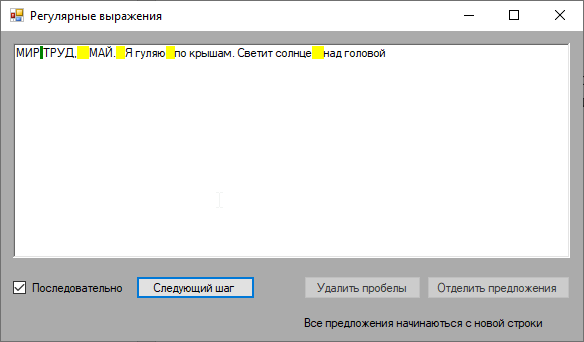


Рисунок 16 – Выполнение итерации, удаляющие первые ненужные пробелы

Следующие итерации происходят аналогичным способом (рисунок 17)

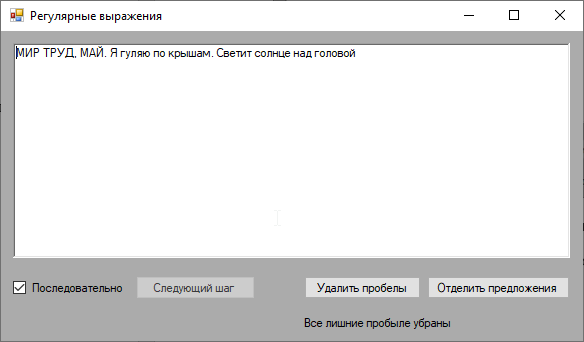


Рисунок 17 – Конец последовательного удаления

Аналогичным способом происходит разделение предложений (рисунок 18).

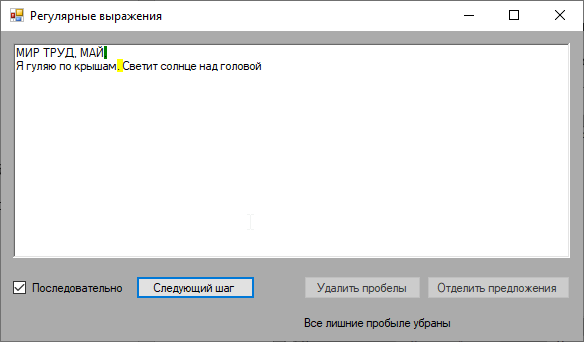


Рисунок 18 – Последовательное разделение предложений

Вывод: В результате выполнение лабораторной работы были разработаны навыки по реализацию регулярных выражений в языке C#.