ГУАП

КАФЕДРА № 33

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | А.И. Дубинин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5 |
| РАЗРАБОТКА БАЗЫ ДАННЫХ И ПОДКЛЮЧЕНИЕ DESKTOP-КЛИЕНТА ЛОГГИРОВАНИЯ ИСКЛЮЧЕНИЙ В РАСПРЕДЕЛЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЕ |
| по курсу: Технологии и методы программирования |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3136 |  |  |  | В.А. Панчук |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023 г.

Оглавление

[1. Цель лабораторной работы 3](#_Toc136125749)

[2. Постановка задачи 3](#_Toc136125750)

[3. Основное задание 3](#_Toc136125751)

[3.1. Листинг программы 5](#_Toc136125752)

[3.2. Пример работы программы 9](#_Toc136125753)

[4. Дополнительное задание 10](#_Toc136125754)

[4.1. Листинг программы 10](#_Toc136125755)

[4.2. Пример работы программы 11](#_Toc136125756)

[5. Вывод 11](#_Toc136125757)

# Цель лабораторной работы

Получить навыки работы с регулярными выражениями, поиска сигнатур в логе обработки исключений, за мены и извлечения «подозрительных» словоформ. Разработка динамических библиотек.

# Постановка задачи

В лабораторной работе на языке C# необходимо разработать форму с функционалом парссинга текста по материалам лекции.

Необходимо выполнить в лабораторной работе задание под вариантом 14 (с использованием языка C#).

Следующая часть лабораторной работы включает в себя разработку программы на языке C#, в которой должен быть реализован парссинг сохраненных лог-журналов в DLL. Определить время парссинга каждого журнала и всех выбранных журналов. Программа должна была быть выполнена с использованием интерфейса Windows Forms. Обработанные исключения фиксировать на главной форме проекта и в текстовом файле с указанием даты и времени проявления исключений, а также свойств класса Exception.

# Основное задание

В проекте с лабораторными работами добавить форму с функционалом парссинга текста по материалам лекции. Доработать обработчик кнопки «Выделить включения» так, чтобы при НОВОМ выборе сигнатуры в comdoBox ВЫБРАННАЯ сигнатура отображалась в richBox иным цветом.

Добавить функционал, когда при добавлении в richBox текста не из файла (ввод текста пользователем) осуществлялся поиск последовательно выбираемых сигнатур.

Для сформированного лог-журнала на ПЕРВОЙ форме проекта определить, в каких формах появлялись исключения и по сколько раз они появились в каждой форме (включения наименований этих форм в лог-журнале выделить цветом)

В результате парссинга выделить в лог-журнале все включения со словоформами form без учета регистра, добавив перед ними иным цветом свой IP-адрес, если эти вхождения, попали в интервал дат, введенных пользователем (от <дд.мм.гггг> до <дд.мм.гггг>).

Реализовать парссинг сохраненных лог-журналов в DLL (для каждого jрегулярного выражения отдельный класс в DLL). Определить время парссинга каждого журнала и всех выбранных журналов..

Алгоритм парссинга программы приведен на рисунке 1.

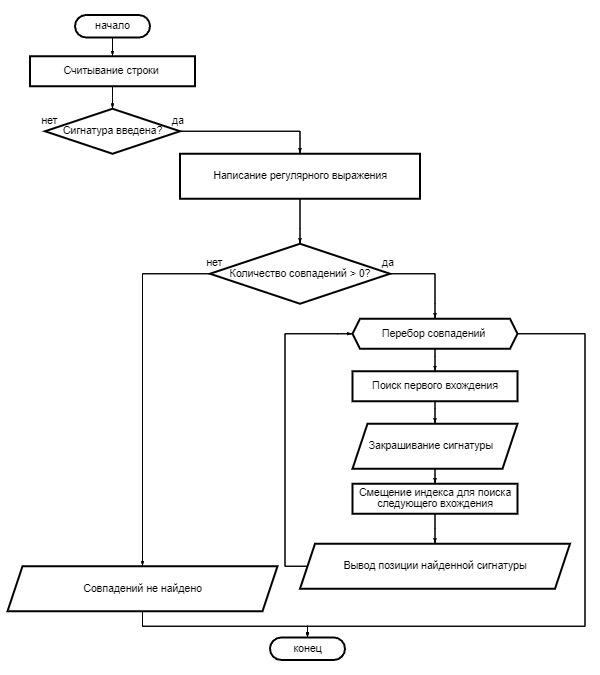


Рисунок − Блок-схема парссинга

## Листинг программы

Программа парссинга текста:

private void SetSelectionStyle(int startIndex, int len, FontStyle style)

{

richTextBox1.Select(startIndex, len);

Font temp\_font = richTextBox1.SelectionFont;

richTextBox1.SelectionFont = new Font(richTextBox1.SelectionFont, richTextBox1.SelectionFont.Style | style);

switch ((int)comboBox1.SelectedIndex)

{

case 0:

richTextBox1.SelectionColor = System.Drawing.Color.Red;

break;

case 1:

richTextBox1.SelectionColor = System.Drawing.Color.Green;

break;

case 2:

richTextBox1.SelectionColor = System.Drawing.Color.Purple;

break;

default:

richTextBox1.SelectionColor = System.Drawing.Color.Blue;

break;

}

}

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

{

int save\_cur = richTextBox1.SelectionStart;

string text\_info = richTextBox1.Text;

string reg = comboBox1.Text;

Reg1 Regular = new Reg1(reg);

MatchCollection matches = Regular.parse\_main(text\_info);

if (matches.Count > 0)

{

richTextBox1.Select(0, richTextBox1.TextLength);

richTextBox1.SelectionColor = System.Drawing.Color.Black;

richTextBox1.SelectionFont = new Font(richTextBox1.SelectionFont, richTextBox1.SelectionFont.Style | FontStyle.Regular);

textBox1.Text = "Совпадения: \r\n";

foreach (Match match in matches)

{

textBox1.Text += match.Index + " позиция: \t" + match.Value + "\r\n";

SetSelectionStyle(match.Index, match.Length, FontStyle.Underline);

}

richTextBox1.SelectionStart = save\_cur;

richTextBox1.Select(save\_cur, 0);

}

else

{

textBox1.Text = "Совпадений не найдено \r\n";

}

}

DLL библиотека:

namespace ClassLibrary1

{

public class Reg1

{

string reg;

public Reg1(string regex)

{

reg = regex;

}

public MatchCollection parse\_main(string text)

{

Regex regex = new Regex(@reg);

MatchCollection matches = regex.Matches(text);

return matches;

}

}

}

Программа парссинга лог-журналов:

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

{

try

{

int save\_cur = mainForm.debugBox.SelectionStart;

Reg1 Regular = new Reg1("[0-3][0-9].[0-1][0-9].[1-2][0-9][0-9][0-9]");

Reg1 RegularForm = new Reg1("[f,F]orm");

string IP = " (192.168.77.129)";

mainForm.debugBox.Select(0, mainForm.debugBox.TextLength);

mainForm.debugBox.SelectionColor = System.Drawing.Color.Black;

mainForm.debugBox.SelectionFont = new Font(mainForm.debugBox.SelectionFont, mainForm.debugBox.SelectionFont.Style | FontStyle.Regular);

MatchCollection matches = Regular.parse\_main(mainForm.debugBox.Text);

MatchCollection forms = RegularForm.parse\_main(mainForm.debugBox.Text);

int shift = 0;

Match rebot = matches[0];

foreach (Match match in matches)

{

if (SetSelectStyle(rebot.Index + shift, rebot.Length, FontStyle.Underline) == 1)

{

foreach (Match form in forms)

{

if (form.Index < match.Index && form.Index > rebot.Index)

{

mainForm.debugBox.Text = mainForm.debugBox.Text.Substring(0, form.Index + shift) + IP + mainForm.debugBox.Text.Substring(form.Index + shift);

shift += IP.Length;

}

}

}

rebot = match;

}

if (SetSelectStyle(rebot.Index + shift, rebot.Length, FontStyle.Underline) == 1)

{

foreach (Match form in forms)

{

if (form.Index > rebot.Index)

{

mainForm.debugBox.Text = mainForm.debugBox.Text.Substring(0, form.Index + shift) + IP + mainForm.debugBox.Text.Substring(form.Index + shift);

shift += IP.Length;

}

}

}

shift = 0;

rebot = matches[0];

foreach (Match match in matches)

{

if (SetSelectStyle(rebot.Index + shift, rebot.Length, FontStyle.Underline) == 1)

{

foreach (Match form in forms)

{

if (form.Index < match.Index && form.Index > rebot.Index)

{

mainForm.debugBox.Select(form.Index + 2 + shift, 16);

shift += IP.Length;

mainForm.debugBox.SelectionColor = System.Drawing.Color.Green;

}

}

}

rebot = match;

}

if (SetSelectStyle(rebot.Index + shift, rebot.Length, FontStyle.Underline) == 1)

{

foreach (Match form in forms)

{

if (form.Index > rebot.Index)

{

mainForm.debugBox.Select(form.Index + 2 + shift, 16);

shift += IP.Length;

mainForm.debugBox.SelectionColor = System.Drawing.Color.Green;

}

}

}

matches = Regular.parse\_main(mainForm.debugBox.Text);

foreach (Match match in matches)

{

SetSelectStyle(match.Index, match.Length, FontStyle.Underline);

}

mainForm.debugBox.SelectionStart = save\_cur;

mainForm.debugBox.Select(save\_cur, 0);

mainForm.debugBox.SelectionColor = System.Drawing.Color.Black;

mainForm.debugBox.SelectionFont = new Font(mainForm.debugBox.SelectionFont, mainForm.debugBox.SelectionFont.Style | FontStyle.Regular);

}

catch (Exception error)

{

mainForm.debugBox.Text += mainForm.dateLabel + "" + mainForm.timeLabel + "\r\n";

mainForm.debugBox.Text += "Метод: " + error.TargetSite + "\r\n";

mainForm.debugBox.Text += error.Message + "\r\n";

mainForm.debugBox.Text += error.Source + "\r\n";

mainForm.debugBox.Text += "Трассировка стека:\r\n" + error.StackTrace + "\r\n";

mainForm.debugBox.Text += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\r\n";

}

}

private int SetSelectStyle(int startIndex, int len, FontStyle style)

{

mainForm.debugBox.Select(startIndex, len);

Font temp\_font = mainForm.debugBox.SelectionFont;

int a = 0;

string temp = mainForm.debugBox.Text.Substring(startIndex, 10);

string[] my\_data = temp.Split('.');

string[] start = beginD.Text.Split('.');

string[] end = endD.Text.Split('.');

int my\_y = int.Parse(my\_data[2]);

int s\_y = int.Parse(start[2]);

int e\_y = int.Parse(end[2]);

int my\_m = int.Parse(my\_data[1]);

int s\_m = int.Parse(start[1]);

int e\_m = int.Parse(end[1]);

int my\_d = int.Parse(my\_data[0]);

int s\_d = int.Parse(start[0]);

int e\_d = int.Parse(end[0]);

if (my\_y > s\_y && my\_y < e\_y)

a = 1;

if (my\_y == s\_y || my\_y == e\_y)

{

if (my\_m > s\_m && my\_m < e\_m)

a = 1;

if (my\_m == s\_m || my\_m == e\_m)

{

if (my\_d > s\_d && my\_d < e\_d)

a = 1;

}

}

switch (a)

{

case 0:

mainForm.debugBox.SelectionColor = System.Drawing.Color.Red;

break;

case 1:

mainForm.debugBox.SelectionColor = System.Drawing.Color.Green;

break;

}

return a;

}

## Пример работы программы

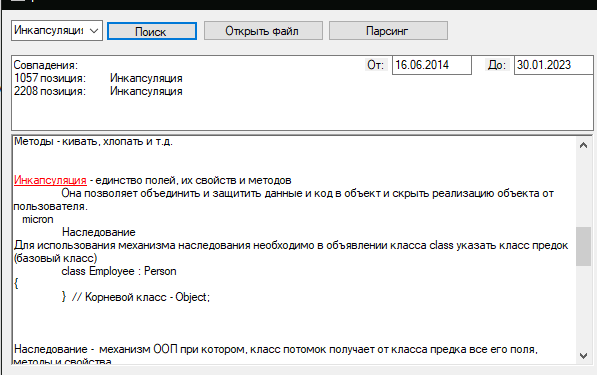


Рисунок 2 – Пример работы парссинга текста

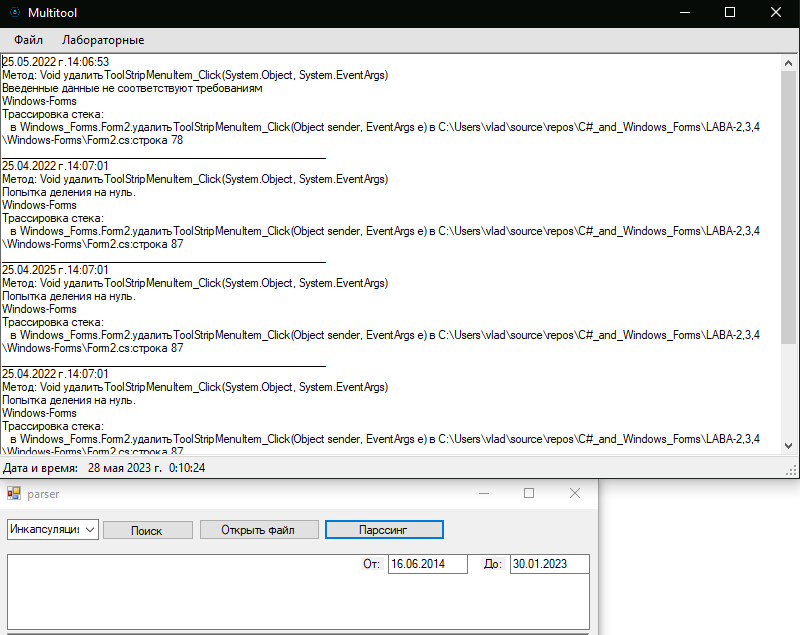


Рисунок − До нажатия кнопки Парссинг

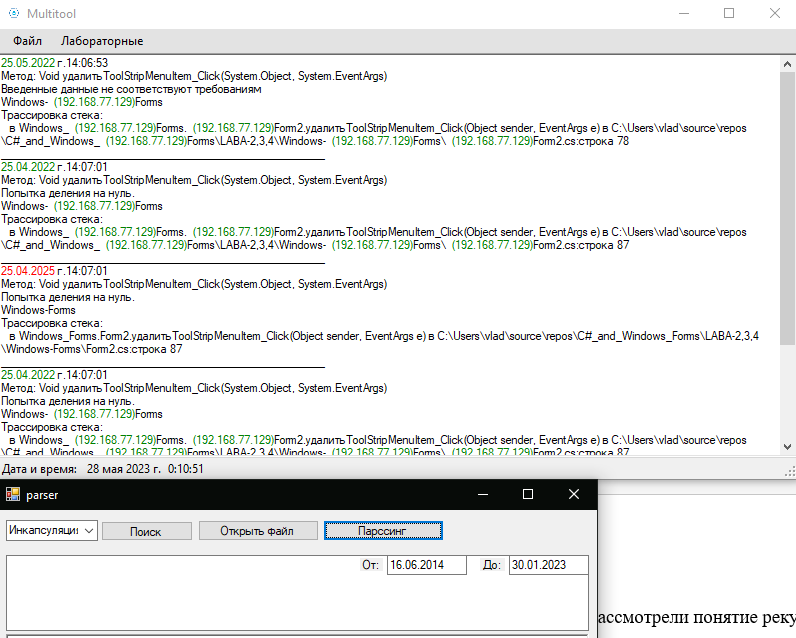


Рисунок − После нажатия кнопки Парссинг

# Дополнительное задание

Создать функцию выделения слов, заканчивающихся на букву "n" и начинающихся на "m".

## Листинг программы

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

{

int save\_cur = richTextBox1.SelectionStart;

string text\_info = richTextBox1.Text;

string reg = comboBox1.Text;

if (comboBox1.SelectedIndex == 2)

reg = "(?<=[ ,\\n])m\\w+?n[^a-zA-Z]";

Reg1 Regular = new Reg1(reg);

MatchCollection matches = Regular.parse\_main(text\_info);

## Пример работы программы

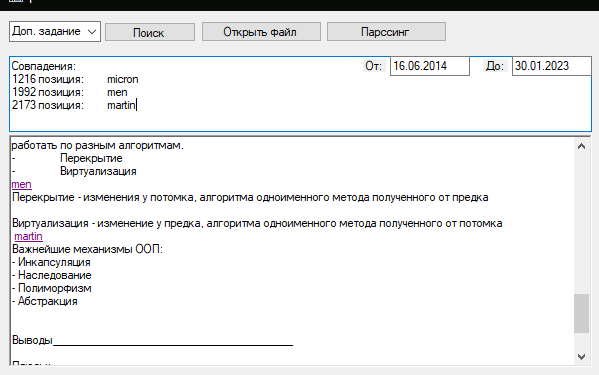


Рисунок − Пример работы программы дополнительного задания

# Вывод

Получили навыки работы с регулярными выражениями, поиска сигнатур в логе обработки исключений, замены и извлечения «подозрительных» словоформ. Разобрались с разработкой динамических библиотек.