|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных Технологий

Кафедра Корпоративных Информационных Систем

**ОТЧЕТ**

по Лабораторной Работе №6

на тему

«Регулярные выражения»

по дисциплине

«Разработка программных приложений»

Выполнил студент группы ИКБО-08-18 Валяев Д.А.

Принял старший преподаватель Мирзоян Д.И.

Выполнено «19» февраля 2020 г.

Зачтено «\_\_\_» \_\_\_\_\_\_\_\_\_ 2020 г.

Москва, 2020

**Задание**

Вариант 3.

1. Проверить, что строка является корректным двоичным числом.

2. Проверить, что строка является корректным абсолютным URL.

3. Выделить из строки все вещественные числа.

**Теоретическая часть**

Регулярные выражения являются быстрым, простым и относительно стандартным способом поиска и обработки текстовых данных. Это привело к большой популярности данной технологии, несмотря на ее недостатки. Большинство языков и сред программирования содержат ту или иную библиотеку поддержки регулярных выражений. В C# используется библиотека регулярных выражений, совместимая с Perl 5. Она включена в стандартный набор библиотек и имеет пространство имен System.Text.RegularExpressions.

Для работы с регулярными выражениями служит класс Regex. Один экземпляр класса соответствует одному шаблону регулярных выражений. Это связано с необходимостью поддержания сложных структур данных для работы конечного автомата или формальной грамматики, соответствующих данному выражению. Эти структуры обеспечивают высокую производительность.

Для примера, проверим, есть ли в введенной строке русские буквы:

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

{

Regex r\_test;

r\_test = new Regex("[а-яА-ЯёЁ]");

if (r\_test.IsMatch(textBox1.Text))

textBox2.AppendText("есть\r\n");

else

textBox2.AppendText("нету\r\n");

}

Метод IsMatch определяет, есть ли в строке совпадения с данным шаблоном (регулярным выражением). В случае, если нам необходимо найти конкретные соответствующие элементы, используется метод Matches:

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

{

Regex match\_test = new Regex("[0-9]+ \*\\+ \*[0-9]+");

MatchCollection result = match\_test.Matches(textBox1.Text);

foreach (Match m in result)

textBox2.AppendText(m.Value + "\r\n");

}

В этом примере осуществляется поиск операций сложения целых чисел в строке. Также на этом примере вы можете убедиться, что в обычных регулярных выражения не осуществляется поиск назад (совпадения ищутся только впереди), введя строку типа «5 + 6 + 2».

В случаях, если вам необходимо получить строки, являющиеся лишь частью поискового шаблона, используются поисковые группы (именованные или индексированные). Например, выделим имена пользователей из набора строк:

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

{

Regex pass\_test = new Regex("^Username\\: (?<username>[a-zA-Z0-9]+)", RegexOptions.Multiline);

foreach (Match m in pass\_test.Matches(textBox1.Text))

{

textBox2.AppendText(m.Groups["username"] + "\r\n");

}

}

**Алгоритм выполнения задания**

1. Создать форму
2. Разместить на форме 3 кнопки
3. Разместить на форме 3 поля для ввода
4. Протестировать программное приложение

**Тестирование**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Исходные данные | Ожидаемый результат | Фактический результат | Комментарий |
| 1. | 100101 | Строка является двоичным числом | Строка является двоичным числом | Верно |
| 2. | text 3.45 6.7 text 7.8 | Вещественные числа 3.45, 6.7, 7.8 | Вещественные числа 3.45, 6.7, 7.8 | Верно |
| 3. | https://vk.com | Строка является корректным абсолютным URL | Строка является корректным абсолютным URL | Верно |

**Заключение**

В данной лабораторной работе мы познакомились с регуляными выражениями, а ткже их использованием и продемонстрировали свои результаты на примере созданного программного приложения.

**Исходный код**



Рисунок 1 — форма с кнопками

**import** re  
**import** tkinter **as** tk  
**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
  
root = tk.Tk()  
num\_pattern = **r'^1[0-1]+$'**double\_pattern = **r'[0-9]+.[0-9]+[+e[0-9]\*]?'**URL\_pattern = **r'^https?:\/{2}[\w-]+(\.\w+)+((\/\w+)|(\.\w+))\*\/?$'**

**def** num(text):  
 result = re.match(num\_pattern, text)  
 **if** result:  
 messagebox.showinfo(**"Успех"**, **"Строка является двоичным числом"**)  
 **else**:  
 messagebox.showerror(**"Провал"**, **"Строка не является двоичным числом"**)  
  
  
**def** double\_num(text):  
 **print**(text)  
 result = re.findall(double\_pattern, text)  
 **if** result:  
 messagebox.showinfo(**"Успех"**, **"Вещественные числа: "** + str(result))  
 **else**:  
 messagebox.showerror(**"Провал"**, **"В строке нет вещественных чисел"**)  
  
  
**def** URL(text):  
 **print**(text)  
 result = re.match(URL\_pattern, text)  
 **if** result:  
 messagebox.showinfo(**"Успех"**, **"Строка является корректным абсолютным URL"**)  
 **else**:  
 messagebox.showerror(**"Провал"**, **"Строка не является корректным абсолютным URL"**)  
  
  
label\_num = tk.Label(root, font=10, text=**"Двоичное число"**, fg=**'black'**)  
label\_num.place(relx=0.011, rely=0.20, relwidth=0.20, relheight=0.15)  
  
button\_num = tk.Button(root, text=**"выбрать"**, bg=**'#2E8B57'**, command=**lambda**: num(entry\_num.get()))  
button\_num.place(relx=0.8, rely=0.20, relwidth=0.15, relheight=0.15)  
  
entry\_num = tk.Entry(root, font=12)  
entry\_num.place(relx=0.19, rely=0.20, relwidth=0.60, relheight=0.15)  
  
label\_URL = tk.Label(root, font=10, text=**"URL"**, fg=**'black'**)  
label\_URL.place(relx=0.014, rely=0.6, relwidth=0.20, relheight=0.15)  
  
button\_URL = tk.Button(root, text=**"выбрать"**, bg=**'#2E8B57'**, command=**lambda**: URL(entry\_URL.get()))  
button\_URL.place(relx=0.8, rely=0.6, relwidth=0.15, relheight=0.15)  
  
entry\_URL = tk.Entry(root, font=12)  
entry\_URL.place(relx=0.19, rely=0.6, relwidth=0.60, relheight=0.15)  
  
label\_double = tk.Label(root, font=10, text=**"Вещественное число"**, fg=**'black'**)  
label\_double.place(relx=0, rely=0.4, relwidth=0.20, relheight=0.15)  
  
button\_double = tk.Button(root, text=**"выбрать"**, bg=**'#2E8B57'**, command=**lambda**: double\_num(entry\_double.get()))  
button\_double.place(relx=0.8, rely=0.4, relwidth=0.15, relheight=0.15)  
  
entry\_double = tk.Entry(root, font=12)  
entry\_double.place(relx=0.19, rely=0.4, relwidth=0.60, relheight=0.15)  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 root.title(**"regular expressions"**)  
 root.geometry(**"1100x200"**)  
 root.resizable(False, False)  
 root.mainloop()