**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Программирование»**

Тема: **Регулярные выражения**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3344 |  | Хангулян С. К. |
| Преподаватель |  | Глазунов С. А. |

Санкт-Петербург

2024

## Цель работы

Изучение основ работы с регулярными выражениями, написание небольшой программы, способной находить в тексте определенные выражения по некоторым шаблонам и работать с ними.

## Задание

Вариант 1

На вход программе подается текст, представляющий собой набор предложений с новой строки. Текст заканчивается предложением "Fin." В тексте могут встречаться ссылки на различные файлы в сети интернет. Требуется, используя [регулярные выражения](https://e.moevm.info/mod/lesson/view.php?id=509), найти все эти ссылки в тексте и вывести на экран пары «название сайта» - «имя файла». Гарантируется, что если предложение содержит какой-то пример ссылки, то после ссылки будет символ переноса строки.

Ссылки могут иметь следующий вид:

* Могут начинаться с названия протокола, состоящего из букв и **://** после;
* Перед доменным именем сайта может быть www;
* Далее доменное имя сайта и один или несколько доменов более верхнего уровня;
* Далее возможно путь к файлу на сервере;
* И, наконец, имя файла с расширением.

## Выполнение работы

Вначале был инициализирован текст input\_text и указатель на регулярное выражение pattern. Регулярное выражение включает в себя 7 групп. Первая группа – шаблон одного из трех возможных протоколов, вторая и третья – шаблон «://» и «www.» соответственно. Все они могут встречаться либо по одному разу, либо не встречаться вовсе. Дальше идет четвертая и пятая группы - шаблон доменного имени сайта, который состоит из букв, цифр и минимум одной точки. Шестая группа – необязательный путь к файлу, обязательно содержит «/» и любые другие символы. Седьмая группа – имя файла и его расширение. Объявлено максимальное количество групп max\_groups, специальные переменные compiled\_pattern и group\_array, куда будут записаны скомпилированное регулярное выражение и группы регулярного выражения по отдельности соответственно. Далее с помощью команды regcomp происходит компиляция регулярного выражения, после чего идет бесконечный цикл. Прерывается он, когда на вход подается предложение «Fin.», сравнение происходит с помощью функции strstr. В теле цикла с помощью функции fgets считываются предложения с новой строки, затем, если с помощью функции regexec находится соответствие с шаблоном, на экран выводится четвертая группа (с помощью итерации от начала до конца группы с помощью rm\_so и rm\_eo), тире, седьмая группа (аналогичным образом) и символ переноса строки. По выходе из цикла с помощью функции regfree освобождается скомпилированное ранее регулярное выражение.

## Тестирование

Результаты тестирования представлены в таблице 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № | Входные данные | Выходные данные | Комментарии |
| 1 | This is simple url:  http://www.google.com/track.mp3  May be more than one upper level  domain http://www.google.com.edu/hello.avi  Many of them.  Rly. Look at this!  http://www.qwe.edu.etu.yahooo.org.net.ru/qwe.q  Some other protocols  ftp://skype.com/qqwe/qweqw/qwe.avi  Fin. | google.com - track.mp3  google.com.edu - hello.avi  qwe.edu.etu.yahooo.org.net.ru - qwe.q  skype.com - qwe.avi | Корректно |

## Выводы

Были изучены основы работы регулярных выражений, а также написана небольшая программа, способная находить в тексте с помощью регулярного выражения ссылки и выводить их в виде «название сайта» - «имя файла».

# ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Khangulyan\_Sargis\_lb1

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <regex.h>

int main(){

char input\_text[1000];

char\* pattern = "(http|https|ftp)?(:\\/\\/)?(www\\.)?([A-Za-z0-9]+(\\.[a-zA-Z0-9]+)+)(\\/.+)\*\\/([A-Za-z0-9]+\\.[A-Za-z0-9]+)";

size\_t max\_groups = 8;

regex\_t compiled\_pattern;

regmatch\_t group\_array[max\_groups];

regcomp(&compiled\_pattern, pattern, REG\_EXTENDED);

while (1){

fgets(input\_text, 1000, stdin);

if (regexec(&compiled\_pattern, input\_text, max\_groups, group\_array, 0) == 0){

for (int i = group\_array[4].rm\_so; i < group\_array[4].rm\_eo; i++)

printf("%c", input\_text[i]);

printf(" - ");

for (int j = group\_array[7].rm\_so; j < group\_array[7].rm\_eo; j++)

printf("%c", input\_text[j]);

printf("\n");

}

if (strstr(input\_text, "Fin."))

break;

}

regfree(&compiled\_pattern);

return 0;

}