**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. 3343 |  | Пухов А.Д. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

## Цель работы.

Изучение и применение на практике регулярных выражений в языке Си.

## Задание.

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, 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.

Написанная программа (см. приложение А) выполняет поиск ссылок на различные файлы в сети интернет. Она проверяет введённые с клавиатуры строки на соответствие регулярному выражению, если строка соответствует то происходит вывод в формате <название\_сайта> - <имя\_файла>.

**Переменные:**

* char \*regexString – регулярная строка
* LEN\_STRING – длина строки
* char str[LEN\_STRING] – строка введённая с клавиатуры
* regex\_t regexCompiled – структура в которой хранится скомпилированное регулярное выражение
* regmatch\_t groupArray[] – структура в которой хранятся адреса групп

## Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | 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 | ОК |

## Выводы.

В данной лабораторной работе было изучено и применено на практике написание регулярных выражений в языке Си. И были изучены функции из библиотеки regex.h требуемые для выполнения задания.

# Приложение А Исходный код программы

Название файла: app-001.c

#include <stdio.h>

#include <regex.h>

#include <string.h>

#include <stdlib.h>

#define LEN\_STRING 1000

int main()

{

char \*regexString = "([A-Za-z]+:\\/\\/)?(www\\.)?(([A-Za-z0-9\\-]+\\.)+[A-Za-z0-9\\-]+)\\/([A-Za-z0-9\\/\\-\_]+\\/)\*([A-Za-z0-9\_\\-]+\\.[A-Za-z0-9]+)";

char str[LEN\_STRING];

regex\_t regexCompiled;

regmatch\_t groupArray[7];

if (regcomp(&regexCompiled, regexString, REG\_EXTENDED))

{

printf("Error in regexString\n");

return 0;

}

while (101)

{

fgets(str, LEN\_STRING, stdin);

if (strstr(str, "Fin.") != 0)

{

break;

}

else if (regexec(&regexCompiled, str, 7, groupArray, 0) == 0)

{

for (int i = groupArray[3].rm\_so; i < groupArray[3].rm\_eo; i++)

{

printf("%c", str[i]);

}

printf(" - ");

for (int i = groupArray[6].rm\_so; i < groupArray[6].rm\_eo; i++)

{

printf("%c", str[i]);

}

printf("\n");

}

}

regfree(&regexCompiled);

return 0;

}