**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 |  | Волох И.О. |
| 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 она сохраняет данные в структуру Bigger\_Info, после чего происходит вывод в формате <название\_сайта> - <имя\_файла>.

**Переменные:**

* MAX\_RAZ – длина строки
* char gotten[MAX\_RAZ] – строка введённая с клавиатуры
* regex\_t regor – структура в которой хранится скомпилированное регулярное выражение
* regmatch\_t groups[] – 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 |  |

## 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: main.c

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <regex.h>

#define MAX\_RAZ 250

struct Bigger\_Info {

char \*adress;

char \*file;

};

int main() {

regex\_t regor;

int rep;

char gotten[MAX\_RAZ];

struct Bigger\_Info \*whole\_data = NULL;

int count = 0;

regmatch\_t groups[7];

rep = regcomp(&regor, "(http://)?(www\\.)?(([A-Za-z0-9\\-]+\\.)+[A-Za-z0-9\\-]+)\\/([A-Za-z0-9\\/\\-\_]+\\/)\*([A-Za-z0-9\_\\-]+\\.[A-Za-z0-9]+)", REG\_EXTENDED);

while (1) {

fgets(gotten, MAX\_RAZ - 1, stdin);

if (strstr(gotten, "Fin.") != NULL) {

break;

}

rep = regexec(&regor, gotten, 7, groups, 0);

if (rep == 0) {

whole\_data = realloc(whole\_data, (count + 1) \* sizeof(struct Bigger\_Info));

whole\_data[count].adress = strndup(gotten + groups[3].rm\_so, groups[3].rm\_eo - groups[3].rm\_so);

whole\_data[count].file = strndup(gotten + groups[6].rm\_so, groups[6].rm\_eo - groups[6].rm\_so);

count++;

}

}

for (int i = 0; i < count; i++) {

printf("%s - %s\n", whole\_data[i].adress, whole\_data[i].file);

free(whole\_data[i].adress);

free(whole\_data[i].file);

}

free(whole\_data);

regfree(&regor);

return 0;

}