

**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. Вариант 1**

Студент гр. 3343



Коршков А.А.

Преподаватель

Государкин Я.С.

Санкт-Петербург

2024

## **Цель работы**

Научиться составлять регулярные выражения, работать с группами, писать программы с использованием библиотеки `regex.h` для работы с регулярными выражениями, находить подходящие строки и извлекать из них необходимую информацию.

## **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

Первым делом необходимо привести текст к удобному для нас виду. Для этого создадим динамический массив `text`, пока в тексте не будет найдена строка «Fin.».

После того, как был записан весь текст, необходимо разбить его на отдельные строки. Для этого создадим динамический `sentences`, и с помощью `strtok` разобьем предложение на отдельные строки и запишем их в массив `sentences`.

В `reg_sentence` будет записано регулярное выражение, которое состоит из 6 групп. Первая группа проверяет, есть ли протокол из букв и (пример `https://`), вторая проверяет наличие «`www.`», следующая группа отвечает за полное доменное имя сайта, четвертая сохраняет последний уровень домена (непятая отвечает за наличие или отсутствие пути к файлу на сервере, шестая отвечает за название файл на сервере

Из 6 групп для решения поставленной задачи необходимо использовать информацию из 3 и 6 групп (доменное имя и файл).

Match 1	94-135	<code>http://www.google.com.edu/folder/hello.avi</code>
Group 1	n/a	<code>http://</code>
Group 2	n/a	<code>www.</code>
Group 3	n/a	<code>google.com.edu</code>
Group 4	n/a	<code>.edu</code>
Group 5	n/a	<code>/folder</code>
Group 6	n/a	<code>/hello.avi</code>

Рисунок 1 – Пример разбиения на группы

Следующим шагом необходимо скомпилировать выражение в форму, подходящую для последующего поиска с помощью функции `regexes()`. Для этого необходимо вызвать функцию `regcomp()`, которая скомпилирует выражение в переменную `regexCompiled`. Если по какой-то причине функция не сможет этого

сделать, тогда будет выведено сообщение об ошибке и программа завершит свою работу.

Затем создаём массив `groupArray`, который будет хранить информацию о каждой строке. Далее пробегаемся по циклу и проверяем наличие ссылки в строке с помощью функции `regexes()`. Если ссылка была найдена, то с помощью цикла от начальной до конечной позиции посимвольно выводится информация из 3-ей группы (домен) и 6-ой группы (имя файла).

## **Выводы**

Были изучены принципы создания регулярных выражений, как использовать их в языке С, принципы и методы работы с библиотекой `regex.h`. Написана программа, которая разбивает текст на строки, находит ссылки с помощью регулярного выражения и выводит их в необходимом формате.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <regex.h>

int main(int argc, char const *argv[])
{
    char *text = (char *)malloc(sizeof(char));
    char **sentences = (char **)malloc(sizeof(char *));

    int count = 0;
    char symbol;
    while (strstr(text, "Fin.") == NULL)
    {
        symbol = getchar();
        text[count++] = symbol;
        text = realloc(text, sizeof(char) * (count + 2));
    }
    text[count] = '\0';
    char *sentence = strtok(text, "\n");
    count = 0;
    while (sentence != NULL)
    {
        sentences[count++] = sentence;
        sentences = (char **)realloc(sentences, sizeof(char *) * (count +
1));
        sentence = strtok(NULL, "\n");
    }
    char *reg_sentence = "([a-zA-Z]+:\\/{2})?(w{3}\\.)?([a-zA-Z0-
9\\-]+(\\.[a-zA-Z\\-]+)+)(\\/[A-z]+)*(\\/[A-z0-9]+\\.[A-z0-9]+)";
    size_t maxGroups = 7;
    regex_t regexCompiled;
    regmatch_t groupArray[maxGroups];
    if (regcomp(&regexCompiled, reg_sentence, REG_EXTENDED))
    {
        printf("Wowm no - can't compile regular expression\n");
        return 0;
    };
    for (int i = 0; i < count; i++)
    {
        if (regexexec(&regexCompiled, sentences[i], maxGroups, groupArray,
0) == 0)
        {
            for (int j = groupArray[3].rm_so; j < groupArray[3].rm_eo;
j++)
                printf("%c", sentences[i][j]);
            printf(" - ");
            for (int j = groupArray[6].rm_so + 1; j <
groupArray[6].rm_eo; j++)
                printf("%c", sentences[i][j]);
            printf("\n");
        }
    }
}
```

```
    }  
    free(text);  
    free(sentences);  
    regfree(&regexCompiled);  
    return 0;  
}
```



## **ПРИЛОЖЕНИЕ Б**

### **ТЕСТИРОВАНИЕ**

#### **Тест № 1.**

##### **Входные данные:**

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.

##### **Выходные данные:**

google.com - track.mp3

google.com.edu - hello.avi

qwe.edu.etu.yahooo.org.net.ru - qwe.q

skype.com - qwe.avi

#### **Тест № 2.**

##### **Входные данные:**

Hello

[google.com/track.mp3](http://www.google.com/track.mp3)

domain <http://www.google.com.about.edu/folder/hell3o.mp4>

Many of them.

Fin.

##### **Выходные данные:**

google.com - track.mp3

google.com.about.edu - hell3o.mp4