САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Программирование»

Тема: Лабораторная работа № 1. Регулярные выражения. Вариант 1.

Студентка гр. 3343 _____ Калиберов Н.И

Преподаватель _____ Государкин Я. С.

Санкт-Петербург

2024

Цель работы

Изучить и научиться применять функции библиотеки regex.h языка Си для поиска совпадений в строках при помощи регулярных выражений. Освоить навыки для написания регулярных выражений на 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

Описание функций:

- int main(): главная функция программы, в случае, если регулярное выражение нельзя скомпилировать, завершает выполнение программы, иначе выводит на экран все совпадения согласно заданию и возвращает 0.
- void printMatch (char* s, regmatch_t groupArray): выводит в консоль посимвольно группу совпадения регулярного выражения из строки; принимает на вход исходную строку и совпавшее регулярное выражение из диапазона.

Разработанный программный код см. в приложении А.

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	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	Выходные данные соответствуют ожиданиям.
2.	This is simple url: http://www.google.-aaaaaa.com/track.mp3 May be more than one upper level domain http://www.google.com.edu/hello.avi Many of them. youtube.en/file.f Rly. Look at this! Fin.	google.-aaaaaa.com - track.mp3 google.com.edu - hello.avi youtube.en - file.f	Выходные данные соответствуют ожиданиям.
3.	This is simple url: http://www.google.aaaaaa.com//track.mp3 May be more than one upper level	google.com.edu - hello.avi youtube.en - file.f google_google.com.edu	Выходные данные соответствуют

	domain http://www.google.com.edu/hello.avi Many of them. youtube.en/file.f Rly. Look at this! This is simple url: aaa://googleaaaaaacom/a.a May be more than one upper level domain http://www.google_google.com.edu/hello.avi Fin.	- hello.avi	ожиданиям.

Выводы

В ходе выполнения лабораторной работы были освоены необходимые навыки для использования регулярных выражений на языке Си с помощью библиотеки regex.h, а также для составления регулярных выражений согласно требованиям. Были изучены необходимые языковые конструкции и особенности записи регулярных выражений на языке Си.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

#define MAX_LINE_LENGTH 1024
#define MAX_GROUPS 10

void extractMatch(char *source, regmatch_t match) {
    for (int i = match.rm_so; i < match.rm_eo; i++) {
        printf("%c", source[i]);
    }
}

int main() {
    char *pattern = "([A-Za-z]+://)?(www\\.)?([A-Za-z0-9.-]+\\.[A-
Za-z]+)((/[A-Za-z0-9_-]+)*)/([A-Za-z0-9_-]+\\.[A-Za-z0-9]+)\n";
    regex_t regexCompiled;
    regmatch_t groupArray[MAX_GROUPS];

    if (regcomp(&regexCompiled, pattern, REG_EXTENDED)) {
        fprintf(stderr, "Could not compile regex\n");
        return 1;
    }

    char line[MAX_LINE_LENGTH];

    while (fgets(line, sizeof(line), stdin)) {
        if (strncmp(line, "Fin.", 4) == 0) {
            break;
        }
        if (regexec(&regexCompiled, line, MAX_GROUPS, groupArray, 0)
== 0) {
            extractMatch(line, groupArray[3]);
            printf(" - ");
```

```
            extractMatch(line, groupArray[6]);
            printf("\n");
        }
    }

    regfree(&regexCompiled);
    return 0;
}
```