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		**Сьомак Д.А.**
Преподаватель		**Глазунов С.А.**

Санкт-Петербург

2023

Цель работы

Освоение работы с регулярными выражениями, получение навыков их составления. Получение практического опыта по применению регулярных выражений на языке программирования C.

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, 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

Были подключены стандартные библиотеки C вместе с библиотекой regex.h, которая нужна для работы с регулярными выражениями. Было задано регулярное выражение regstr и максимальное количество его групп maxGroups. Далее была задана переменная reg, которая хранит выражение после компиляции и массив groupArray, в котором хранятся индексы начала и конца строки, при проверке на совместимость с регулярным выражением. Регулярное выражение было скомпилировано с помощью функции regcomp, при этом была реализована проверка на компиляцию. Далее был создан динамический массив str для хранения входных данных. Был реализован цикл while, который заканчивает свою работу при обнаружении строки Fin., на каждой его итерации строка, считываемая с помощью fgets, проверяется на соответствие регулярному выражению. Если строка подходит, то циклы for выводят на экран 3 и 6 группу выражения (доменное имя сайта и домены верхнего уровня, имя файла с расширением), между ними выводиться -, а в конце \n с помощь printf. Когда цикл заканчивает свою работу, освобождается память для регулярного выражения, а также для динамического массива входных данных. Программа завершается.

Исходный 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 Fin.	google.com - track.mp3 google.com.edu - hello.avi	-
2.	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.	qwe.edu.etu.yahooo.org.net.ru - qwe.q skype.com - qwe.avi	-

Выводы

Были освоены правила написания регулярных выражений, получены навыки их составления. Был получен практический опыт использования регулярных выражений посредством написания программы на языке программирования C.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Somak_Demid_lb1.c

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

int main(){
    char * regstr = "([A-Za-z]+:\\/\\/)?(www\\.)?([a-zA-Z0-9]+(\\.[a-
zA-Z0-9]+)+)(\\/[a-zA-Z\\/]+)?\\/([A-Za-z]+\\.[a-zA-Z0-9]+)";
    size_t maxGroups = 7;

    regex_t reg;
    regmatch_t groupArray[maxGroups];

    if (regcomp(&reg, regstr, REG_EXTENDED))
    {
        printf("Can't compile regular expression\n");
        return 0;
    };

    char* str = malloc(sizeof(char) * 10000);

     while(1) {
      fgets(str, 10000, stdin);

      if (strncmp(str, "Fin.",4) == 0) break;

      if(regexec(&reg, str, maxGroups, groupArray, 0) == 0){
           for (size_t i = groupArray[3].rm_so; i < groupArray[3].rm_eo;
i++) {
                printf("%c", str[i]);
            }

            printf(" - ");

            for    (size_t    j    =    groupArray[6].rm_so;    j    <
groupArray[6].rm_eo; j++) {
                printf("%c", str[j]);
            }
            printf("\n");
     }
}
    regfree(&reg);
    free(str);
    return 0;
}
```