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

Студентка гр. 3343 _____ Лобова Е. И.

Преподаватель _____ Государкин Я.С.

Санкт-Петербург
2024

Цель работы

Целью работы является освоение работы с регулярными выражениями в языке Си на примере использующей их программы.

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 используемых переменных:

- Максимальное количество групп в регулярном выражении – *size_t maxGroups*.
- Строка, содержащая регулярное выражение - *char * regexString*.
- Переменная, используемая для хранения скомпилированного регулярного выражения - *regex_t regexCompiled*.
- Массив, в котором хранятся индексы начала и конца групп - *regmatch_t groupArray[maxGroups]*.

Для решения задания лабораторной работы были выполнены следующие задачи:

- Компиляция регулярного выражения с помощью функции *int regcomp(regex_t * restrict preg, const char * restrict pattern, int cflags)*. Если его невозможно скомпилировать, то выводится сообщение об ошибке.
- Считывание в цикле строк, до строки «Fin.».
- Проверка строки на соответствие регулярному выражению с помощью функции *int regexec(const regex_t * restrict preg, const char * restrict string, size_t nmatch, regmatch_t pmatch[restrict], int eflags)*. В случае соответствия вывод четвертой и седьмой группы, котором соответствуют название сайта и имя сайта.
- Освобождение памяти, выделенной функцией *regcomp()* с помощью функции *void regfree(regex_t *preg)*.

Функции, реализованные в программе:

- Функция *void printGroup(char* s, regmatch_t group)* принимает на вход строку и массив групп. Так как в массиве хранятся индексы начала и конца группы, то мы с помощью цикла можем посимвольно вывести группу.

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	Для различных ссылок программа работает корректно.

Выводы

Были изучены регулярные выражения и их реализация в языке Си.

Разработана программа, которая, используя регулярные выражения, находит все ссылки в тексте и вывод на экран. Для работы с регулярными выражениями были использованы функции из библиотеки <regex.h>.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

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

int main ()
{
    char * regexString = "([A-Za-z]+:\\/{2})?(w{3}\\.)?(([A-Za-z0-
9\\-]+\\.)+[A-Za-z0-9\\-]+)\\/([A-Za-z0-9_\\-]+\\/)*([A-Za-z0-9_\\-
]+\\.[a-z0-9]+)";

    size_t maxGroups = 7;
    regex_t regexCompiled;
    regmatch_t groupArray[maxGroups];

    if (regcomp(&regexCompiled, regexString, REG_EXTENDED))
    {
        printf("Wowm no - can't compile regular expression\n");
        return 0;
    };

    char s[100];
    while(1){
        fgets(s,100,stdin);
         if (strstr(s,"Fin.") != NULL){
            break;
        }
        if (regexec(&regexCompiled, s, maxGroups, groupArray, 0) ==
0){
            printGroup(s,groupArray[3]);
            printf(" - ");
            printGroup(s,groupArray[6]);
            printf("\n");
        }
    }
    regfree(&regexCompiled);
    return 0;
}
```