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. 3341 _____ Бойцов В.А.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург
2024

Цель работы

Целью работы служит освоение работы с регулярными выражениями и их применение на примере программы на языке Си

Для достижения поставленной цели требуется решить следующие задачи:

- Изучить типичные конструкции регулярных выражений;
- Составить регулярное выражение, решающее поставленную задачу;
- Написать программу, использующее написанное регулярное выражение для решения поставленной задачи.

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

Перед написанием кода программы было создано регулярное выражение, которое ищет в тексте все ссылки по шаблону:

([a-z]+\:\/\/)?(www\.)?(([a-z]+\.)+[a-z]+)\/([a-z]+\/)([a-z]+\.[a-z0-9]+)*

В этом регулярном выражении каждая часть ссылки выделена в отдельную группу, а 3-я и 6-я группы содержат в себе название сайта и имя файла соответственно.

Далее была написана программа, использующая представленное выше регулярное выражение.

Для работы с регулярными выражениями подключается библиотека *<regex.h>*.

Инициализируются следующие константы:

Const char pattern* – хранит в себе написанное регулярное выражение (с учётом двойного слеша);

const int maxNumOfGroups – хранит количество групп, использующихся в регулярном выражении;

const int maxBufSize – размер строки-буфера для ввода текста

const char textEnding* – маркер окончания текста.

Была написана функция *void printLinks(char* currentString, regmatch_t* currentGroup)*, которая принимает на вход строку *currentString*, в которой была найдена ссылка, и массив групп *currentGroup*. В функции с помощью двух циклов for, пробегающих значения между границами группы, указанными в *currentGroup*, посимвольно выводятся название сайта и имя файла.

Далее в функции *int main()* создаются переменная *regex_t patternCompiled* для компиляции регулярного выражения, массив *regmatch_t groupsArray[maxNumOfGroups]* для хранения индексов начала и конца групп, строка-буфер *char buf[maxBufSize]*. С помощью функции *regcomp()* регулярное выражение компилируется.

Затем в цикле *do {…} while(),* выполняющемся до тех пор, пока не встретится маркер конца текста, считываются предложения текста с помощью

функции *fgets()*, и, если в предложении будет найдена ссылка с помощью функции *regexec()*, ссылка будет выведена на экран с помощью *printLinks()*.

В конце память от скомпилированного регулярного выражения очищается с помощью функции *regfree()*.

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.	Fin.		Ничего не выведено, как и должно быть.
2.	http://www.google.com/track.mp3 http://www.google.com.edu/hello.avi http://www.qwe.edu.etu.yahooo.org.net.ru/qwe.q Fin.	google.com - track.mp3 google.com.edu - hello.avi qwe.edu.etu.yahooo.org.net.ru - qwe.q	Все ссылки выведены
3.	This is simple url: Fin.		Ничего не выведено, как и должно быть
4.	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	Все ссылки выведены

Выводы

В результате выполнения работы была освоена работа с регулярными выражениями, выполнены поставленные задачи, а именно: были изучены основные конструкции регулярных выражений, написано регулярное выражение, позволяющее искать ссылку в данном тексте, а также написана программа на языке си, использующее данное регулярное выражение для вывода на экран названия сайта и имени файла из ссылок, содержащихся в данном тексте.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

const  char*  pattern="([a-z]+\\:\\/\\/)?(www\\.)?(([a-z]+\\.)+[a-z]
+)\\/([a-z]+\\/)*([a-z]+\\.[a-z0-9]+)";
const int maxNumOfGroups = 7;
const int maxBufSize = 1000;
const char* textEnding = "Fin.";

void printLinks(char* currentString, regmatch_t* currentGroup)
{
    for(int i=currentGroup[3].rm_so;i<currentGroup[3].rm_eo;i++)
        printf("%c", currentString[i]);
    printf(" - ");
    for(int i=currentGroup[6].rm_so;i<currentGroup[6].rm_eo;i++)
            printf("%c", currentString[i]);
    printf("\n");
}

int main()
{
    regex_t patternCompiled;
    regmatch_t groupsArray[maxNumOfGroups];
    regcomp(&patternCompiled, pattern, REG_EXTENDED);
    char buf[maxBufSize];
    do
    {
        fgets(buf, maxBufSize, stdin);
                if(regexec(&patternCompiled,  buf,  maxNumOfGroups,
groupsArray, 0)==0)
        {
            printLinks(buf, groupsArray);
        }
    }
    while(strncmp(buf, textEnding, maxBufSize));
    regfree(&patternCompiled);
    return 0;
}
```