**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 |  | Ягудин Д.Р. |
| Преподаватель |  | Глазунов С.А. |

Санкт-Петербург

2024

## Цель работы

Цель работы является изучение и использование регулярных выражений для обработки текстовых данных. Для этого необходимо изучить синтаксис и возможности регулярных выражений, а после применить полученные навыки на практике в ходе решения задачи.

## Задание

1 вариант.

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.

## Основные теоретические положения

Регулярные выражения (Regular Expressions) – это мощный инструмент для работы с текстовыми данными, который позволяет задать шаблон поиска текста. Эти шаблоны могут содержать символы и специальные конструкции, которые определяют правила поиска соответствий. В языке программирования C для работы с регулярными выражениями используется библиотека regex.h.

Библиотека regex.h содержит функции для работы с регулярными выражениями, такие как компиляция, сопоставление и освобождение регулярного выражения. Для использования функций библиотеки regex.h необходимо включить заголовочный файл <regex.h> и скомпилировать программу с флагом -lregex.

С помощью функций из библиотеки regex.h можно осуществлять поиск, замену, разбиение и извлечение данных из текстовых строк в соответствии с заданным шаблоном. В языке C для описания регулярных выражений используется специальный синтаксис, который позволяет указывать символы и конструкции для определения шаблона поиска.

Использование регулярных выражений и библиотеки regex.h позволяет обрабатывать текстовые данные эффективно, удобно и с минимальными усилиями.

## Выполнение работы

Подключаются необходимые библиотеки: *stdlib.h*, *stdio.h*, *string.h* и *regex.h*.

В переменную *PATTERN* записывается необходимое регулярное выражение.

Функция *solution* считывает текст из ввода пользователя, и сравнивеает подстроки на совпадение с регулярный выржением, после чего вызывает функцию output.

Функция *output* принимает строку, которую разделяет на сайт — файл и выводит ее.

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 |
|  | 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 | Проверка на наличие www перед доменным именем |
|  | 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 | Проверка на валидность выражений с доменами более высокого уровня и на наличие пути до файла |

## Выводы

Цель данной работы заключалась в изучении и практическом применении регулярных выражений для обработки текстовых данных. Были изучены основные синтаксические конструкции и возможности регулярных выражений. Полученные знания были успешно применены для решения практической задачи, демонстрирующей использование регулярных выражений в реальной ситуации. Таким образом, цель данной работы была успешно достигнута.

# Приложение А Исходный код программы

Название файла: main.c

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<regex.h>

const char\* PATTERN = "([a-z]+\\:\\/\\/)?(www\\.)?(([a-z]+\\.)+[a-z]+)\\/([a-z]+\\/)\*([a-z]+\\.[a-z0-9]+)";

const char\* END\_STRING = "Fin.";

const int MAX\_MATCHES = 10;

const int BUFFER\_SIZE = 2000;

void solution();

void output(regmatch\_t\* matching\_str, char\* buffer);

int main(){

solution();

}

void solution(){

regex\_t regular;

regmatch\_t matching\_str[MAX\_MATCHES];

regcomp(&regular, PATTERN, REG\_EXTENDED);

char buffer[BUFFER\_SIZE];

fgets(buffer, BUFFER\_SIZE, stdin);

while(strncmp(buffer, END\_STRING, strlen(END\_STRING)) != 0){

if(regexec(&regular, buffer, MAX\_MATCHES, matching\_str, 0) == 0){

output(matching\_str, buffer);

}

fgets(buffer, BUFFER\_SIZE, stdin);

}

}

void output(regmatch\_t\* matching\_str, char\* buffer){

for(int i = matching\_str[3].rm\_so; i < matching\_str[3].rm\_eo; i++){

printf("%c", buffer[i]);

}

printf(" - ");

for(int i = matching\_str[6].rm\_so; i < matching\_str[6].rm\_eo; i++){

printf("%c", buffer[i]);

}

printf("\n");

}