**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**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Какира .У.Н. |
| Преподаватель |  | Государкин Я. С. |

Санкт-Петербург

2024

**Цель работы**

изучите, как использовать регулярные выражения на языке Си для поиска определенных шаблонов в строках. Попрактикуйтесь в написании простых регулярных выражений, соответствующих желаемому текстовому шаблону.

**Задание**

На вход программе подается текст, представляющий собой набор предложений с новой строки. Текст заканчивается предложением "**Fin.**" В тексте могут встречаться примеры запуска программ в командной строке Linux. Требуется, используя регулярные выражения, найти только примеры команд в оболочке суперпользователя и вывести на экран пары <имя пользователя> - <имя\_команды>. Если предложение содержит какой-то пример команды, то гарантируется, что после нее будет символ переноса строки.

Примеры имеют слеующий вид:

* Сначала идет имя пользователя, состоящее из букв, цифр и символа **\_**
* Символ @
* Имя компьютера, состоящее из букв, цифр, символов **\_** и **-**
* Символ **:** и **~**
* Символ **$**, если команда запущена в оболочке пользователя и **#**, если в оболочке суперпользователя. При этом между двоеточием, тильдой и **$** или **#** могут быть пробелы.
* Пробел
* Сама команда и символ переноса строки.

Описание функций

* Функция printGroup отвечает за вывод группы символов из строки в соответствии с заданным диапазоном.

**Выполнение работы**

Описание функций:

Описание функций int main(): главная функция программы, в ней компилируется regex\_

и проводится сравнение со строкой пришедшей из функции char \*fetchSentence, при положительном результате печатает ответ

Функция fetch Sentence считывает предложение от пользователя до тех пор, пока не будет введено "Fine.", и возвращает предложение в виде динамически выделяемой строки.

функция печати сопоставленных групп:

Эта функция извлекает сопоставленные группы из соответствия регулярному выражению и печатает их. Он использует поля rm\_so и rm\_eo в структуре rematch\_t для извлечения подстрок из входной строки.

В целом, эти функции работают сообща, считывая предложения пользователя, применяя шаблон регулярного выражения для извлечения конкретной информации и распечатывая извлеченные данные.

**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 <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <regex.h>

#define GROUP\_SIZE 3

char \*fetchSentence() {

char ch;

int len = 0;

char \*sentence = calloc(len+1, sizeof(char));

while ((ch = getchar()) != '\n') {

sentence = realloc(sentence, sizeof(char) \* len+2);

sentence[len++] = ch;

if(strcmp(sentence, "Fin.") == 0) return sentence;

}

return sentence;

}

int main() {

regex\_t regex;

const char \*pattern = "([a-zA-Z0-9\_]+)@[a-zA-Z0-9\_-]+: \*~ \*# (.+)";

regcomp(&regex, pattern, REG\_EXTENDED);

char \*sentence;

while (strcmp((sentence = fetchSentence()), "Fin.") != 0) {

regmatch\_t matches[GROUP\_SIZE];

if (regexec(&regex, sentence, GROUP\_SIZE, matches, 0) == 0) {

printf("%.\*s - %.\*s\n", matches[1].rm\_eo - matches[1].rm\_so, &(sentence[matches[1].rm\_so]),

matches[2].rm\_eo - matches[2].rm\_so, &(sentence[matches[2].rm\_so]));

}

free(sentence);

}

return 0;

}