**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. 3343 |  | Силяев Р.А. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

## Цель работы

Научиться использовать регулярные выражения на языке C, используя библиотеку regex

## 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 следующий 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.

## Выполнение работы

Программа получает на вход текст до строки, содержащей слово 'Fin.'. Извлекаются только примеры команд в оболочке суперпользователя, используя регулярное выражение, хранящееся в *regexString*. Обработанное регулярное выражение сохраняется в *regexCompiled* с использованием *regcomp*. Затем происходит построчное считывание текста. Если строка не является последней, она проверяется на соответствие регулярному выражению с использованием *regexec*. После этого проверяется по группам, и выводится строка в формате <имя пользователя> - <имя\_команды>.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования содержатся в таблице 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | Run docker container:  kot@kot-ThinkPad:~$ docker run -d --name  stepik stepik/challenge-avr:latest  You can get into running /bin/bash  command in interactive mode:  kot@kot-ThinkPad:~$ docker  exec -it stepik "/bin/bash"  Switch user: su :  root@84628200cd19: ~ # su box  box@84628200cd19: ~ $ ^C  Exit from box: box@5718c87efaa7:  ~ $ exit  exit from container:  root@5718c87efaa7: ~ # exit  kot@kot-ThinkPad:~$ ^C  Fin. | root - su box  root - exit | 🙂 |

## Выводы

В результате выполнения лабораторной работы был изучен способ работы с регулярными выражениями на языке C, кроме того была написана программа, в которой используются регулярные выражения.

# Приложение А Исходный код программы

Название файла: main.py

#include <stdio.h>

#include <regex.h>

#include <string.h>

int main ()

{

char \* regexString = "([0-9a-zA-Z\_-]+)@[0-9a-zA-Z\_-]+: \*~ \*# (.+)";

size\_t maxGroups = 3;

regex\_t regexCompiled;

regmatch\_t groupArray[maxGroups];

regcomp(&regexCompiled, regexString, REG\_EXTENDED);

char s[1000];

while (fgets(s,1000,stdin)){

if(strstr(s, "Fin.") != NULL){

break;

}

if (regexec(&regexCompiled, s, maxGroups, groupArray, 0) == 0)

{

for (int i = 1; i < maxGroups; i++)

{

for(int j=groupArray[i].rm\_so;j<groupArray[i].rm\_eo;j++)

printf("%c",s[j]);

if (i == 1){

printf(" - ");

}

}

}

}

regfree(&regexCompiled);

return 0;

}