**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(ка) гр. 3343 |  | Гельман П.Е. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

## Цель работы

Цель лабораторной работы заключается в изучении регулярных выражений и их применении в программах на языке Си.

## 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

Для работы с регулярными выражениями используется библиотека regex.h

Сначала зададим четыре переменных: size\_t groups, максимальное количество групп в регулярном выражении, regex\_t regex, переменная для хранения скомпилированного регулярного выражения, regmatch\_t matches[groups], массив, в котором хранятся индексы начала и конца групп, char\* pattern, регулярное выражение.

Далее компиляция регулярного выражения осуществляется благодаря функции int regcomp(regex\_t \*preg, const char \*pattern, int cflags). Если его нельзя скомпилировать, программа завершается ошибкой.

Затем происходит считывание текста из буфера построчно до строки «Fin.».

Проверка строки на соответствие заданному регулярному выражению происходит благодаря функции int regexec(const regex\_t \*preg, const char \*String, size\_t nmatch, regmatch\_t \*pmatch, int eflags). Если проверка прошла успешно, на экран выводится первая и вторая группы — имя пользователя и команда. Далее очищается память, выделенная под работу функции regcomp().

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| № п/п | Входные данные | Выходные данные |
|  | Welcome to virtual machine management system:  me@my-laptop:~# vm start ubuntu-vm  Starting virtual machine ubuntu-vm...  Virtual machine ubuntu-vm is now running.  You can connect to it using SSH:  me@my-laptop:~$ ssh ubuntu@192.168.1.100  Password:  Welcome to Ubuntu 20.04 LTS!  [ubuntu@ubuntu-vm](mailto:ubuntu@ubuntu-vm):~# ^C  Exiting SSH session.  Shutting down virtual machine:  ubuntu@ubuntu-vm:~$ exit  Virtual machine ubuntu-vm is shutting down...  me@my-laptop:~$ ^C  Fin. | me - vm start ubuntu-vm  ubuntu - ^C |
|  | kot@kot-ThinkPad:~$ docker run -d --name my\_container busybox  kot@kot-ThinkPad:~$ docker exec -it my\_container /bin/sh  root@container\_id:~ # ls  root@container\_id:~ # touch new\_file.txt  root@container\_id:~ # exit  kot@kot-ThinkPad:~$ docker stop my\_container  kot@kot-ThinkPad:~$ docker rm my\_container  Fin. | root — ls  root - touch new\_file.txt  root - exit |

## Выводы

Были изучены регулярные выражения и работа с ними на языке Си. Реализована программа, которая находит все строки в тексте, где команды вызваны от имени суперпользователя, и выводит их на экран. Для этого использовалась библиотека regex.h и ее функции.

# Приложение А Исходный код программы

Название файла: main.c

#include <stdio.h>

#include <regex.h>

#include <string.h>

int main() {

char s[101];

size\_t groups = 3;

regex\_t regex;

regmatch\_t matches[groups];

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

if (regcomp(&regex, pattern, REG\_EXTENDED)) {

return 1;

}

while (fgets(s, 100, stdin)) {

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

break;

}

if (regexec(&regex, s, groups, matches, 0) == 0) {

if (matches[1].rm\_so != -1 && matches[2].rm\_so != -1) {

s[matches[1].rm\_eo] = '\0';

s[matches[2].rm\_eo] = '\0';

printf("%s - %s", &s[matches[1].rm\_so], &s[matches[2].rm\_so]);

}

}

}

regfree(&regex);

return 0;

}