САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Программирование»

Тема: Лабораторная работа № 1. Регулярные выражения

Студент гр. 3343 _____ Пивоев Н. М.

Преподаватель _____ Государкин Я. С.

Санкт-Петербург

2024

Цель работы

Научиться использовать регулярные выражения, заложенные в библиотеке regex.h, написав программу 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

Программа получает на вход текст, заканчивающийся строчкой ‘Fin.’. Из текста она выводит только команды в оболочке суперпользователя.

В *regexString* хранится регулярное выражение, по которому отбираются строчки. С помощью regcomp в *compiledString* сохраняется обработанное регулярное выражение, которое можно использовать впоследствии.

Далее идёт построчное считывание текста. В случае, если строка не является конечной, идёт её отбор по регулярному выражению через regexec. Далее идёт обход по группам (где 0 группа – всё выражение, 1 группа – первые скобки в выражении, а 2 группа – вторые скобки) и строчка выводится в формате ‘имя’ – ‘команда’. Затем освобождается память

Разработанный программный код см. в приложении А.

Тестирование

Таблица 1.

№	Входные данные	Выходные данные	Комментарии
1.	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	Ожидаемый вывод.

Выводы

В ходе выполнения работы, был освоен синтаксис, необходимый для написания регулярных выражений на языке Си, а также написана программа с их использованием.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <stdio.h>
#include <regex.h>
#include <string.h>
/*
gcc ./main.c -lregex
./a.exe
*/
int main (){
    char * regexString = "([0-9a-zA-Z_-]+)@[0-9a-zA-Z_-]+: *~ *#
(.+)";

    size_t groups = 3;
    regex_t compiledString;
    regmatch_t groupArray[groups];

    regcomp(&compiledString, regexString, REG_EXTENDED);

    char s[1000];
    while (fgets(s,1000,stdin)){
        if (strstr(s, "Fin.") != NULL)
            break;

        if (regexec(&compiledString, s, groups, groupArray, 0) == 0){
            for (int i = 1; i < groups; ++i){
                for (int j = groupArray[i].rm_so; j <
groupArray[i].rm_eo; ++j)
                    printf("%c", s[j]);
                if (i == 1)
                    printf(" - ");
            }
        }
    }
    regfree(&compiledString);
    return 0;
}
```