

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. 3344

Бажуков С.В.

Преподаватель

Глазунов С.А.

Санкт-Петербург

2024

Цель работы

Освоение работы с регулярными выражениями на языке Си на примере использующей их программы.

Задание.

Вариант 2. На вход программе подаётся 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

Была подключена стандартная библиотека для ввода и вывода `<stdio.h>`, библиотека для работы со строками `<string.h>` и библиотека для работы с регулярными выражениями `<regex.h>`. Была проинициализирована строка, отвечающая за регулярное выражение `char *regex_string = "(\\w+)@[0-9a-zA-Z_-]+: ?~ ?# (.+\\n)"`. Была объявлена структура для хранения информации о скомпилированном регулярном выражении `regex_t regex_compiled`. Был объявлен массив `regmatch_t group_array[]` размером 3, который будет использоваться для хранения информации о совпадениях групп в регулярном выражении. Была вызвана функция для компиляции регулярного выражения `regcomp(®ex_compiled, regex_string, REG_EXTENDED)`. Был инициализирован массив символов `char s[100]`. Был запущен цикл, который считывал входные данные, пока строка на вход не равнялась "Fin." `while (strcmp(s, "Fin.")).` В if была вызвана функция `if (regexec(®ex_compiled, s, 3, group_array, 0) == 0)`, которая проверяет, соответствует ли строка s регулярному выражению. Если соответствие найдено, функция возвращает 0, иначе - ненулевое значение. Далее была реализована функция `void print_groups_of_string(regmatch_t *group, char *s)` которая выводит символы из первой и второй группы регулярного выражения. После вывода высвобождалась память под структуру для скомпилированного регулярного выражения. После чего программа завершалась.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	<p>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.</p>	<p>root - su box root - exit</p>	-
2.	<p>Switch user: su : roofsafast@8dasd4628200cd19 : ~ # d dadad ad a box@84628200cd19: ~ \$ ^C Exit from box: box@5718c87efaa7: ~ \$ exit exit from container: root@5718c87efaa7: ~ # exit kot@kot-ThinkPad:~\$ ^C Fin. root@da__da:~# su bsax Fin.</p>	<p>roofsafast - d dadad ad a root - exit root - su bsax</p>	-

Выводы

Была освоена работа с регулярными выражениями на языке Си на примере использующей их программы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <regex.h>
#include <string.h>

void printGroup(regmatch_t *groups, char *buf, int index)
{
    for (size_t i = groups[index].rm_so; i < groups[index].rm_eo; i++)
    {
        printf("%c", buf[i]);
    }
}

int main()
{
    char *regexStr = "(\\w+)@([0-9a-zA-Z_-]+): ?~ ?# (.+\\n)";

    regex_t regexComp;
    regcomp(&regexComp, regexStr, REG_EXTENDED);

    regmatch_t groups[4];

    char buf[1000] = "";
    while (strcmp(buf, "Fin."))
    {
        fgets(buf, 1000, stdin);
        if (regexexec(&regexComp, buf, 4, groups, 0) == 0)
        {
            printGroup(groups, buf, 1);
            printf(" - ");
            printGroup(groups, buf, 3);
        }
    }
    regfree(&regexComp);
    return 0;
}
```