**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**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3344 |  | Якимова Ю.А. |
| 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

Перед началом работы были подключены следующие библиотеки:

* *<stdio.h>* (для ввода и вывода)
* *<string.h>* (для работы со строками)
* *<regex.h>* (для работы с регулярными выражениями)

Далее в функции *main()* была проинициализирована строка *pattern*, содержащая регулярное выражение для поиска предложений, содержащих примеры команд в оболочке суперпользователя. Регулярное выражение было скомпилировано с использованием функции *regcomp()* и помещено в структуру *reg\_str*.

Были объявлены массивы *matches* и *sent* для хранения информации о совпадениях групп в регулярном выражении и предложения, поданного на вход, соответственно.

Был запущен цикл *while*, который считывает входные данные, пока строка не будет равнозначна “Fin.”. Каждая строка считывается с помощью функции *fgets* и помещается в массив *sent*. Далее строка *sent* проверяется на соответствие скомпилированному регулярному выражению. Если соответствие найдено, функция *regexec* возвращает 0, иначе - ненулевое значение. В случае соответствия с помощью полей структур из массива *matches* выводятся символы из первой и второй групп регулярного выражения.

После выводавысвобождается память под структуру для скомпилированного регулярного выражения. После чего программа завершается.

Разработанный программный код см. в приложении А.

## Тестирование.

Результаты тестирования представлены в табл. 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 | - |
|  | 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. | roofsafast - d dadad ad a  root - exit  root - su bsax | - |

## Выводы

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

# Приложение А Исходный код программы

Название файла: Yakimova\_Yuliya\_lb1.c

#include <stdio.h>  
#include <regex.h>  
#include <string.h>  
  
int main()  
{  
 char\* pattern = "([a-zA-Z0-9\_]+)@[0-9a-zA-Z\_-]+: ?~ ?# (.+\n)";  
   
 regex\_t reg\_str;  
 regcomp(&reg\_str, pattern, REG\_EXTENDED);  
   
 regmatch\_t matches[3];  
 char sent[100];  
   
 while(strcmp(sent, "Fin.")) {  
 fgets(sent, 99, stdin);  
   
 if (!(regexec(&reg\_str, sent, 3, matches, 0))) {  
 for (int i = matches[1].rm\_so; i < matches[1].rm\_eo; i++) {  
 printf("%c", sent[i]);  
 }  
   
 printf(" - ");  
   
 for (int j = matches[2].rm\_so; j < matches[2].rm\_eo; j++) {  
 printf("%c", sent[j]);  
 }  
 }  
 }  
   
 regfree(&reg\_str);  
   
 return 0;  
}