

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Хассан Факи Абакар¹

18 апреля, 2023, Москва, Россия

¹Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

Изучить основы программирования в оболочке ОС UNIX.
Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Задачи лабораторной работы

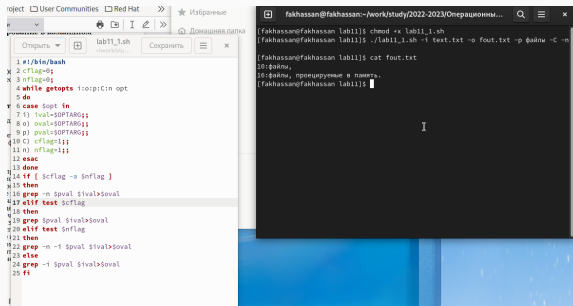
1 Выполнить 4 задания

Процесс выполнения лабораторной работы

1. Используя команды `getopts` `grep` напишем командный файл, который анализирует командную строку с ключами и выполним его: `-i inputfile` — прочитать данные из указанного файла; `-o outputfile` — вывести данные в указанный файл; `-r шаблон` — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк;

а затем ищет в указанном файле нужные строки

Выполнение работы



The image shows a Linux desktop environment. On the left, a text editor window titled 'lab11_1.sh' is open, displaying a shell script. The script starts with a shebang and a loop that reads command-line options. It then uses a case statement to handle different options, setting variables like 'ival', 'oval', 'pval', 'cflag', and 'nflag'. Finally, it uses grep to search for patterns in a file. On the right, a terminal window is open, showing the execution of the script. The user has run 'chmod +x lab11_1.sh' and then './lab11_1.sh -i text.txt -o fout.txt -p fvalnu -C -n'. The terminal output shows the script's execution, including the cat command being used to display the contents of 'fout.txt'.

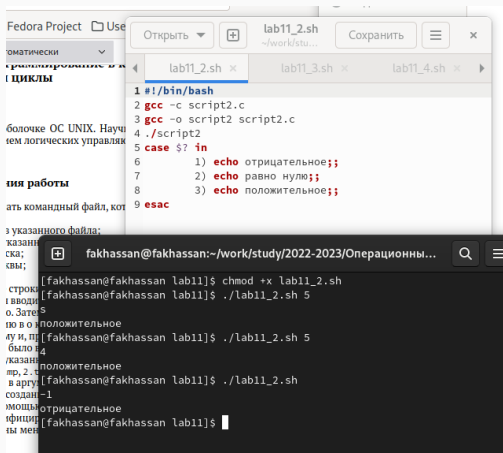
```
1 #!/bin/bash
2 cflag=0;
3 nflag=0;
4 while getopts i:orpC:n opt
5 do
6 case $opt in
7 i) ival=$OPTARG;;
8 o) oval=$OPTARG;;
9 p) pval=$OPTARG;;
10 C) cflag=1;;
11 n) nflag=1;;
12 esac
13 done
14 if [ $cflag -a $nflag ]
15 then
16 grep -n $pval $ival>$oval
17 elif test $cflag
18 then
19 grep $pval $ival>$oval
20 elif test $nflag
21 then
22 grep -n -i $pval $ival>$oval
23 else
24 grep -i $pval $ival>$oval
25 fi
```

```
[fakhassan@fakhassan lab11]$ chmod +x lab11_1.sh
[fakhassan@fakhassan lab11]$ ./lab11_1.sh -i text.txt -o fout.txt -p fvalnu -C -n
[fakhassan@fakhassan lab11]$ cat fout.txt
10:fvalnu,
16:fvalnu, проецируемые в память.
[fakhassan@fakhassan lab11]$
```

Рис. 1: Задание 1

2. Напишем сначала на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем завершим программу при помощи функции `exit(n)`, передавая информацию о коде завершения в оболочку. Командный файл вызовет эту программу и, проанализировав с помощью команды `$?`, выдаст сообщение о том, какое число было введено

Выполнение работы



```
Fedora Project Use
Открыть lab11_2.sh ~/work/stu... Сохранить
lab11_2.sh x lab11_3.sh x lab11_4.sh x
1 #!/bin/bash
2 gcc -c script2.c
3 gcc -o script2 script2.c
4 ./script2
5 case $? in
6     1) echo отрицательное;;
7     2) echo равно нулю;;
8     3) echo положительное;;
9 esac
[fakhassan@fakhassan lab11]$ chmod +x lab11_2.sh
[fakhassan@fakhassan lab11]$ ./lab11_2.sh 5
положительное
[fakhassan@fakhassan lab11]$ ./lab11_2.sh 4
положительное
[fakhassan@fakhassan lab11]$ ./lab11_2.sh -1
отрицательное
[fakhassan@fakhassan lab11]$
```

Рис. 2: Задание 2

3. Напишем командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N

Выполнение работы

The image shows a terminal window with two panes. The top pane displays the source code of a shell script named `lab11_3.sh`. The script uses `while` loops to create and manage temporary files. The bottom pane shows the execution of the script and the resulting file system state.

```
1 #!/bin/bash
2 let i=$1+1
3 while (( i--=1 ))
4 do touch $i.tmp
5 done
6 let j=$2+1;
7 while (( j--=1 ))
8 do rm $j.tmp
9 done
```

Below the script, the terminal shows the execution of the script and the output of the `ls` command:

```
[fakhassan@fakhassan lab11]$ chmod +x lab11_3.sh
[fakhassan@fakhassan lab11]$ ./lab11_3.sh 5
[fakhassan@fakhassan lab11]$ ls
```

File	Permissions	Owner	Group	File	Permissions	Owner	Group
lab11_1.sh	-rwxr-xr-x	fakhassan	fakhassan	lab11_4.sh	-rwxr-xr-x	fakhassan	fakhassan
1.tmp	-rw-r--r--	fakhassan	fakhassan	script2	-rwxr-xr-x	fakhassan	fakhassan
2.tmp	-rw-r--r--	fakhassan	fakhassan	presentation	-rwxr-xr-x	fakhassan	fakhassan
3.tmp	-rw-r--r--	fakhassan	fakhassan	script2.c	-rwxr-xr-x	fakhassan	fakhassan
4.tmp	-rw-r--r--	fakhassan	fakhassan	report	-rwxr-xr-x	fakhassan	fakhassan
5.tmp	-rw-r--r--	fakhassan	fakhassan	script2.o	-rwxr-xr-x	fakhassan	fakhassan
fout.txt	-rw-r--r--	fakhassan	fakhassan				

Рис. 3: Задание 3

4. Напишем командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад.

Выполнение работы

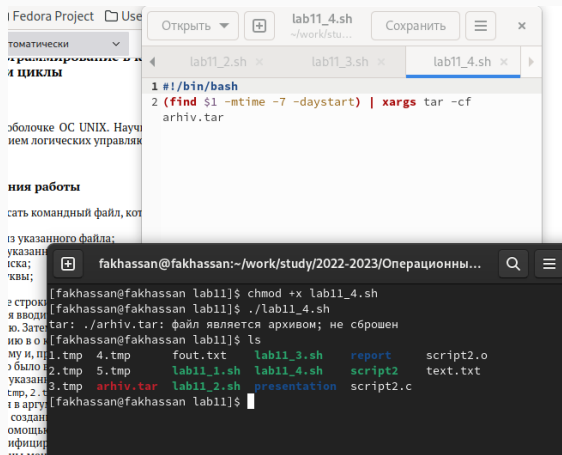


Рис. 4: Задание 4

Выводы по проделанной работе

В данной работе мы изучили основы программирования в оболочке ОС UNIX и писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.