

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Карапетян Завен Арамович НБИбд-01-21¹

26 мая, 2022, 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

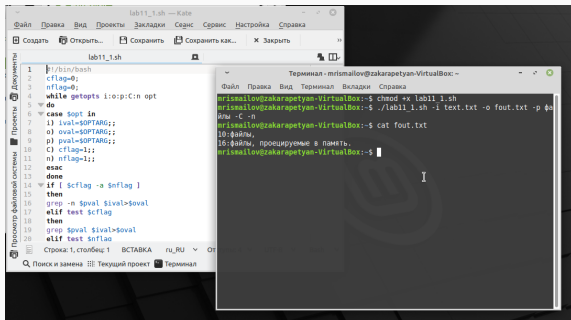
1 Выполнить 4 задания

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 screenshot of a development environment. On the left, a code editor window titled 'lab11_1.sh — Kate' displays a shell script. The script starts with a shebang, sets variables for flags and file names, and uses a while loop to process input. It then uses a case statement to handle different options like -i, -o, -p, -C, and -n. Finally, it uses grep to search for specific patterns in the input files. On the right, a terminal window titled 'Терминал - mrismailov@zakarapetyan-VirtualBox -' shows the execution of the script. The user runs 'chmod +x lab11_1.sh' and then './lab11_1.sh -i text.txt -o fout.txt -p #b'. The output shows the script processing the input files and displaying the results.

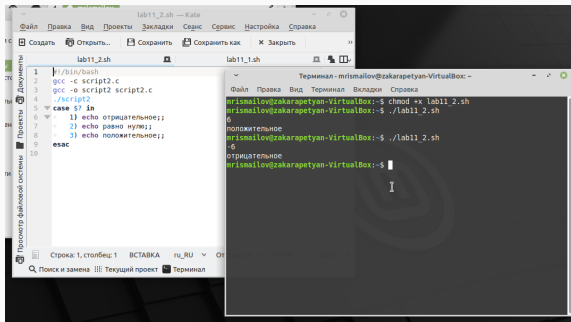
```
1 #!/bin/bash
2 cflag=0;
3 nflag=0;
4 while getopts i:o:p:C:n opt
5 do
6     case $opt in
7         i) ival=$OPTARG;;
8         o) oval=$OPTARG;;
9         p) pval=$OPTARG;;
10        C) cflag=1;;
11        n) nflag=1;;
12    esac
13 done
14 if [ $cflag -a $nflag ]
15 then
16     grep -n $pval $ival>$oval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

```
Терминал - mrismailov@zakarapetyan-VirtualBox -
Файл Правка Вид Терминал Вкладки Справка
mrismailov@zakarapetyan-VirtualBox:~$ chmod +x lab11_1.sh
mrismailov@zakarapetyan-VirtualBox:~$ ./lab11_1.sh -i text.txt -o fout.txt -p #b
Ялы -C -n
mrismailov@zakarapetyan-VirtualBox:~$ cat fout.txt
10:файлы,
10:файлы, проецируемые в память,
mrismailov@zakarapetyan-VirtualBox:~$
```

Figure 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



The screenshot displays a Linux desktop environment with two windows. The left window, titled 'lab11_2.sh — Kate', is a text editor showing a C program named 'script2.c'. The code defines a case statement for '1' that prints 'отрицательное;', '2' that prints 'равно нулю;', and '3' that prints 'положительное;', followed by an 'esac' statement and a 'read' command. The right window, titled 'Терминал - mrismailov@zakarapetyan-VirtualBox -', shows the execution of the script. The user has made the script executable with 'chmod +x lab11_2.sh' and then run it with './lab11_2.sh'. The terminal output shows the prompt '6', followed by 'положительное' and 'отрицательное' on separate lines, and then a new prompt '6'.

```
1 #!/bin/bash
2 gcc -c script2.c
3 gcc -o script2 script2.c
4 ./script2
5 case $1 in
6     1) echo отрицательное;;
7     2) echo равно нулю;;
8     3) echo положительное;;
9 esac
10 read
```

```
mrismailov@zakarapetyan-VirtualBox:~$ chmod +x lab11_2.sh
mrismailov@zakarapetyan-VirtualBox:~$ ./lab11_2.sh
6
положительное
отрицательное
mrismailov@zakarapetyan-VirtualBox:~$
```

Figure 2: Задание 2

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

Выполнение работы

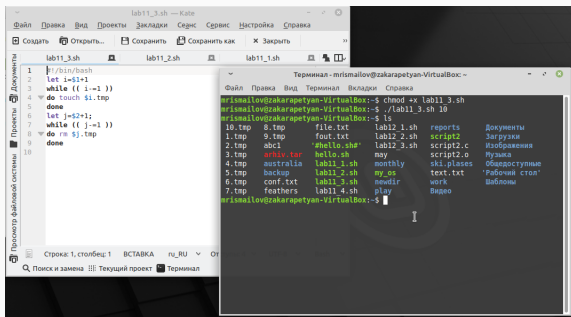


Figure 3: Задание 3

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

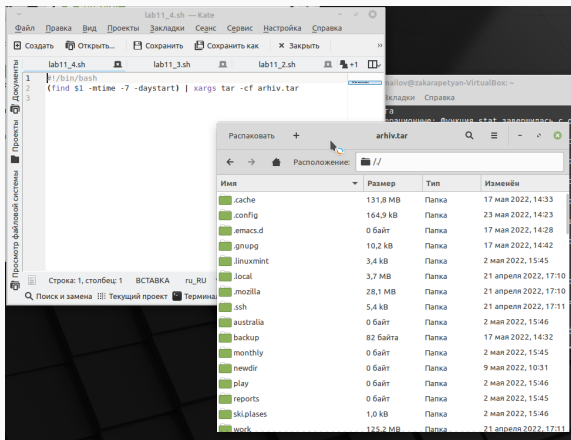


Figure 4: Задание 4

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.