

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Элсаиед Адел Мансоур Абделхалим Мохамед¹

26 апреля, 2024, 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

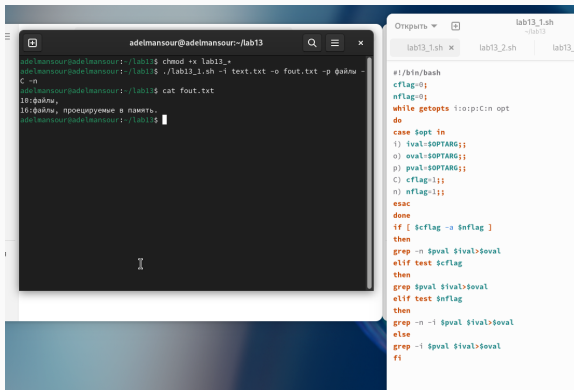
1 Выполнить 4 задания

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 terminal window and a code editor side-by-side. The terminal window, titled 'adelmansour@adelmansour:~/lab13', shows the execution of a shell script 'lab13.sh' with arguments '-i text.txt -o fout.txt -p @allm -C -n'. The output of the script is displayed in the terminal, showing the contents of 'fout.txt' which include the command 'cat fout.txt' and the output '10: @allm, процессоры в памяти.'.

```
adelmansour@adelmansour:~/lab13$ chmod +x lab13.sh
adelmansour@adelmansour:~/lab13$ ./lab13.sh -i text.txt -o fout.txt -p @allm -C -n
adelmansour@adelmansour:~/lab13$ cat fout.txt
10: @allm,
10: @allm, процессоры в памяти.
adelmansour@adelmansour:~/lab13$
```

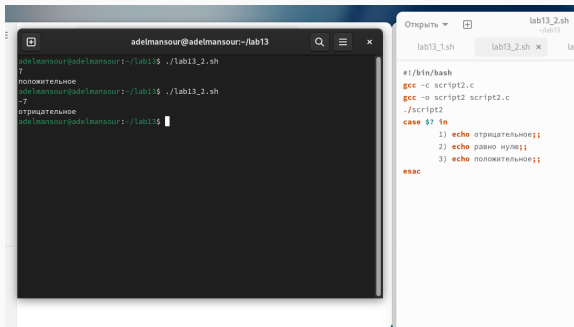
The code editor, titled 'lab13_1.sh', shows the source code of the script. The code is a shell script that takes arguments and processes them. It includes a 'while' loop to process options and a 'case' statement to handle different options. The script uses 'grep' to search for patterns in the input file.

```
#!/bin/bash
cflag=0
nflag=0
while getopts i:op:C:n opt
do
case $opt in
i) ival=$OPTARG;;
o) oval=$OPTARG;;
p) pval=$OPTARG;;
C) cflag=1;;
n) nflag=1;;
esac
done
if [ $cflag -a $nflag ]
then
grep -n $pval $ival>$oval
elif test $cflag
then
grep $pval $ival>$oval
elif test $nflag
then
grep -n -i $pval $ival>$oval
else
grep -i $pval $ival>$oval
fi
```

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



The image shows a terminal window on the left and a code editor on the right. The terminal window, titled 'adelmansour@adelmansour:~/lab13', shows the execution of a script 'lab13_2.sh'. The script takes an argument '7' and prints 'положительное' (positive). The code editor, titled 'lab13_2.sh', shows the script's source code, which uses a case statement to check if the argument is negative, zero, or positive.

```
adelmansour@adelmansour:~/lab13$ ./lab13_2.sh
7
положительное
adelmansour@adelmansour:~/lab13$ ./lab13_2.sh
-7
отрицательное
adelmansour@adelmansour:~/lab13$
```

```
#!/bin/bash
gcc -c script2.c
gcc -o script2 script2.c
./script2
case $? in
  1) echo отрицательное;;
  2) echo равно нулю;;
  3) echo положительное;;
esac
```

Рис. 2: Задание 2

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

Выполнение работы

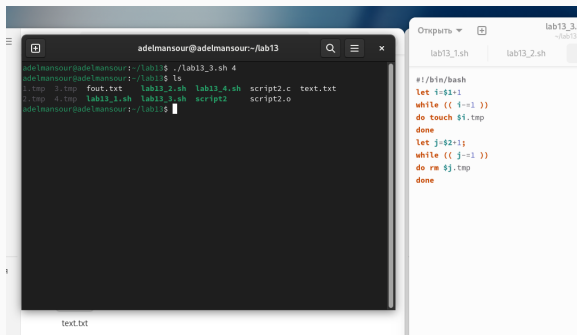
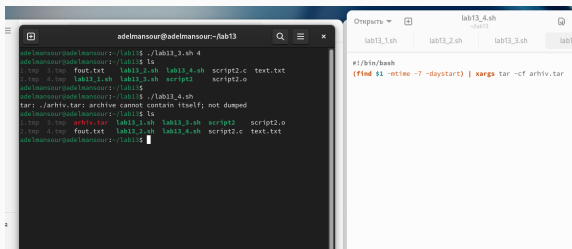


Рис. 3: Задание 3

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



The image shows two overlapping windows from a Linux desktop environment. The background window is a terminal titled 'adelmansour@adelmansour:~/lab13'. It displays the execution of a script 'lab13_3.sh' with four arguments: 'fout.txt', 'lab13_2.sh', 'lab13_4.sh', and 'script2.c'. The script's output shows a loop of four steps, each running a 'tar' command to create an archive. The first 'tar' command fails with the error 'tar: ./arhiv.tar: archive cannot contain itself; not dumped'. The subsequent steps succeed. The foreground window is a file manager titled 'lab13_4.sh' showing the contents of the 'lab13' directory. It lists files 'lab13_1.sh', 'lab13_2.sh', 'lab13_3.sh', and 'lab13_4.sh'. The 'lab13_4.sh' file is selected, and its contents are displayed in the right pane. The script content includes a shebang, a sleep command, and a 'find' command that searches for files named 'arhiv.tar' and runs 'xargs tar -cf arhiv.tar' on them.

```
adelmansour@adelmansour:~/lab13$ ./lab13_3.sh 4
adelmansour@adelmansour:~/lab13$ ls
1: step 3: step fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
2: step 4: step lab13_1.sh lab13_2.sh script2 script2.o
adelmansour@adelmansour:~/lab13$ ./lab13_4.sh
tar: ./arhiv.tar: archive cannot contain itself; not dumped
adelmansour@adelmansour:~/lab13$ ls
1: step 3: step arhiv.tar lab13_1.sh lab13_2.sh script2 script2.o
2: step 4: step fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
adelmansour@adelmansour:~/lab13$
```

```
#!/bin/bash
sleep 1
(find $1 -mtime -7 -daystart) | xargs tar -cf arhiv.tar
```

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.