

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Айдарбекова Алия Робертовна¹

22 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

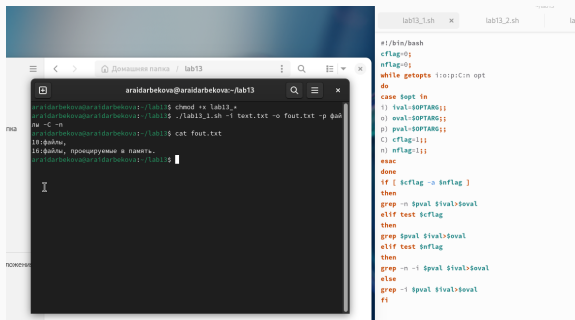
1 Выполнить 4 задания

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. The terminal window, titled 'araidarbekova@araidarbekova:~/lab13', displays the following commands and output:

```
araidarbekova@araidarbekova:~/lab13$ chmod +x lab13.sh
araidarbekova@araidarbekova:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p 400
pi -C -n
araidarbekova@araidarbekova:~/lab13$ cat fout.txt
10:файлы,
16:файлы, прокинутые в память.
araidarbekova@araidarbekova:~/lab13$
```

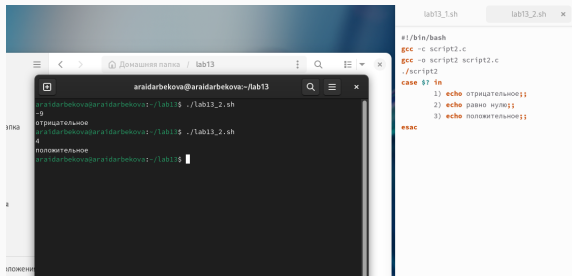
The code editor, titled 'lab13_1.sh', shows the following script:

```
#!/bin/bash
cflag=0
nflag=0
while getopts :ioip:Cin 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



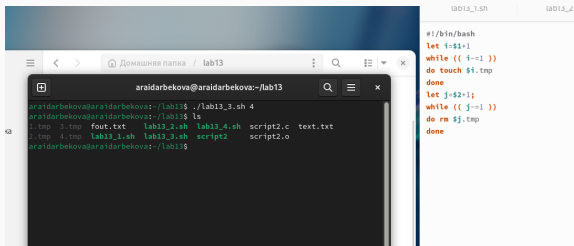
The image shows a terminal window and a code editor. The terminal window, titled 'araidarbekova@araidarbekova:~/lab13', displays the execution of a script named 'lab13_2.sh'. The script takes two arguments: '-f' and '4'. It prints 'отрицательное' (negative) for the first argument and 'положительное' (positive) for the second argument. The code editor, titled 'lab13_2.sh', shows the source code of the script. It starts with a shebang line '#!/bin/bash', followed by compilation commands 'gcc -c script2.c' and 'gcc -o script2 script2.c', and then the execution command './script2'. A case statement follows, handling three arguments: '1)' for 'отрицательное', '2)' for 'равно нулю', and '3)' for 'положительное'.

```
#!/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



The image shows a terminal window and a code editor. The terminal window, titled 'araidarbekova@araidarbekova:~/lab13', displays the execution of a script 'lab13_3.sh' with argument '4'. The output shows a directory listing of files created by the script. The code editor, titled 'lab13_3.sh', shows the script's content, which is a bash script that iterates from i=1 to 4, creating files 'f{i}.tmp' and 'lab13_{i}.sh' in each iteration.

```
araidarbekova@araidarbekova:~/lab13$ ./lab13_3.sh 4
araidarbekova@araidarbekova:~/lab13$ ls
3.tmp  3.tmp  fout.txt  lab13_2.sh  lab13_4.sh  script2.c  text.txt
2.tmp  4.tmp  lab13_1.sh  lab13_3.sh  script2    script2.o
```

```
#!/bin/bash
let i=$1+1
while (( i-- ))
do touch ${i}.tmp
done
let j=$2+1;
while (( j-- ))
do rm ${j}.tmp
done
```

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

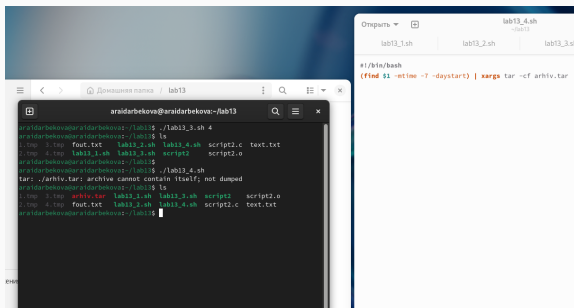


Рис. 4: Задание 4

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.