

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Марьям Ел Вакил¹

17 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

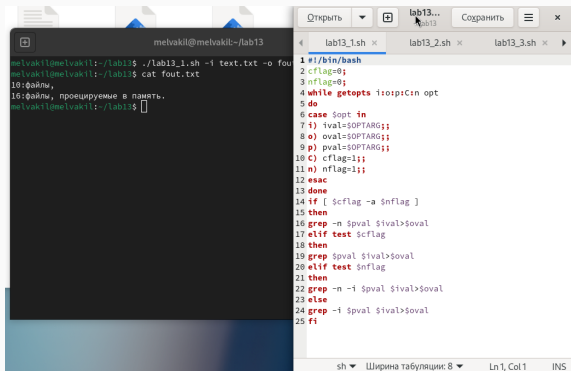
1 Выполнить 4 задания

Процесс выполнения лабораторной работы

1. Используя команды `getopts` `grep` напомним 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 on the left and a code editor on the right. The terminal window displays the execution of a shell script named `lab13_1.sh` with the following output:

```
melvakil@melvakil:~/lab13$ ./lab13_1.sh -i text.txt -o fout
melvakil@melvakil:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
melvakil@melvakil:~/lab13$
```

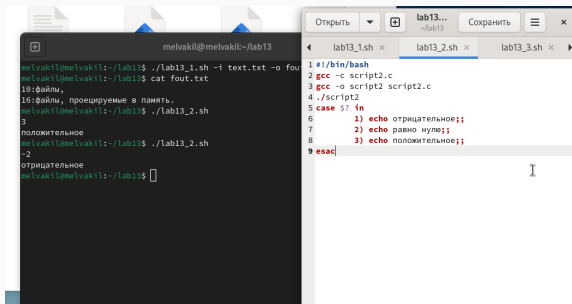
The code editor on the right shows the source code of the script `lab13_1.sh`:

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

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 side-by-side. The terminal window, titled 'melvakil@melvakil:~/lab13', shows the execution of two shell scripts. The first script, 'lab13_1.sh', takes 'text.txt' as input and outputs 'fout.txt'. The second script, 'lab13_2.sh', takes an integer argument and outputs 'положительное' (positive), 'равно нулю' (equal to zero), or 'отрицательное' (negative) based on the value. The code editor, titled 'lab13... -/lab13', shows the source code of 'lab13_2.sh'. It includes a shebang, GCC compilation commands, and a case statement that uses 'echo' to output the appropriate result based on the input value.

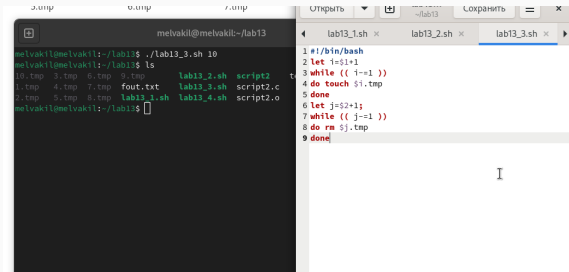
```
melvakil@melvakil:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt
melvakil@melvakil:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
melvakil@melvakil:~/lab13$ ./lab13_2.sh
3
положительное
melvakil@melvakil:~/lab13$ ./lab13_2.sh
-2
отрицательное
melvakil@melvakil:~/lab13$
```

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

Рис. 2: Задание 2

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

Выполнение работы



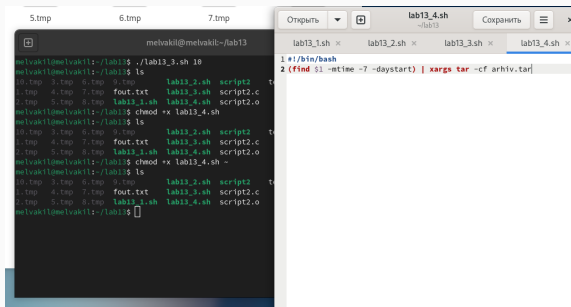
The image shows a terminal window on the left and a file editor on the right. The terminal window displays the execution of a shell script named `lab13_3.sh` with the argument `10`. The script's output lists files in the current directory, including `10.tmp`, `3.tmp`, `6.tmp`, `9.tmp`, `lab13_2.sh`, `script2`, `1.tmp`, `4.tmp`, `7.tmp`, `fout.txt`, `lab13_3.sh`, `script2.c`, `2.tmp`, `5.tmp`, `8.tmp`, `lab13_1.sh`, `lab13_4.sh`, and `script2.o`. The file editor on the right shows the content of `lab13_3.sh`, which is a bash script with the following code:

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

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 a terminal window and a file manager window. The terminal window displays the execution of a script named `lab13_3.sh` with arguments `10` and `10`. The script creates a directory `lab13_4` and a file `script2.c`. The file manager window shows the contents of the `lab13_4` directory, which includes the `script2.c` file and a `tar` archive named `arhiv.tar`.

```
melvaki@melvaki:~/lab13$ ./lab13_3.sh 10
melvaki@melvaki:~/lab13$ ls
10.tmp  3.tmp  6.tmp  9.tmp  lab13_2.sh  script2
1.tmp  4.tmp  7.tmp  fout.txt lab13_3.sh  script2.c
2.tmp  5.tmp  8.tmp  lab13_1.sh lab13_4.sh  script2.o
melvaki@melvaki:~/lab13$ chmod +x lab13_4.sh
melvaki@melvaki:~/lab13$ ls
10.tmp  3.tmp  6.tmp  9.tmp  lab13_2.sh  script2
1.tmp  4.tmp  7.tmp  fout.txt lab13_3.sh  script2.c
2.tmp  5.tmp  8.tmp  lab13_1.sh lab13_4.sh  script2.o
melvaki@melvaki:~/lab13$ chmod +x lab13_4.sh -
melvaki@melvaki:~/lab13$ ls
10.tmp  3.tmp  6.tmp  9.tmp  lab13_2.sh  script2
1.tmp  4.tmp  7.tmp  fout.txt lab13_3.sh  script2.c
2.tmp  5.tmp  8.tmp  lab13_1.sh lab13_4.sh  script2.o
melvaki@melvaki:~/lab13$
```

lab13_4.sh
~lab13

```
1 #!/bin/bash
2 (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.