

# Операционные системы

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

---

Александра Адмиральская

20 апреля 2025

Российский университет дружбы народов, Москва, Россия

## Цели и задачи работы

---

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

1 Выполнить 4 задания

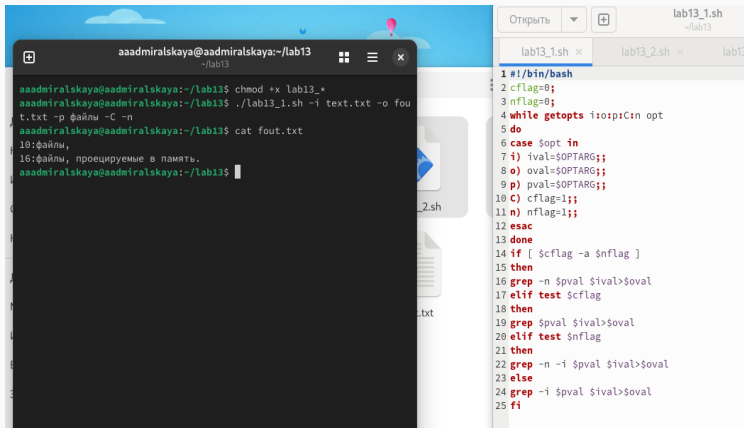
## Процесс выполнения лабораторной работы

---

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 script file on the right. The terminal window has a title bar 'aaadmirlskaya@aaadmirlskaya:~/lab13' and shows the following commands and output:

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

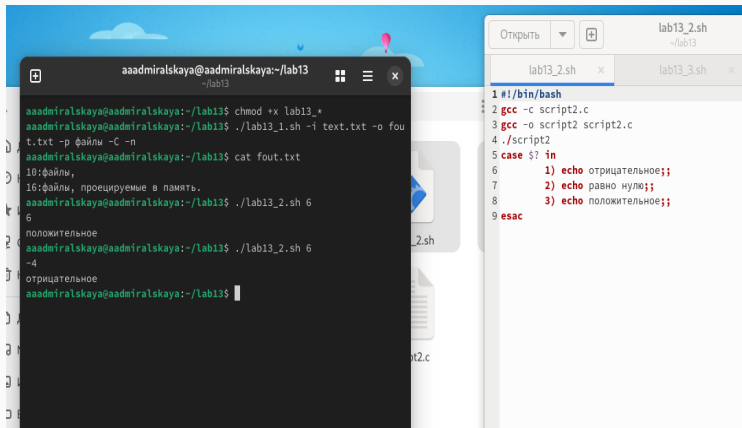
The script file on the right is titled 'lab13\_1.sh' and contains the following code:

```
1#!/bin/bash
2cflag=0;
3nflag=0;
4while getopts i:o:p: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





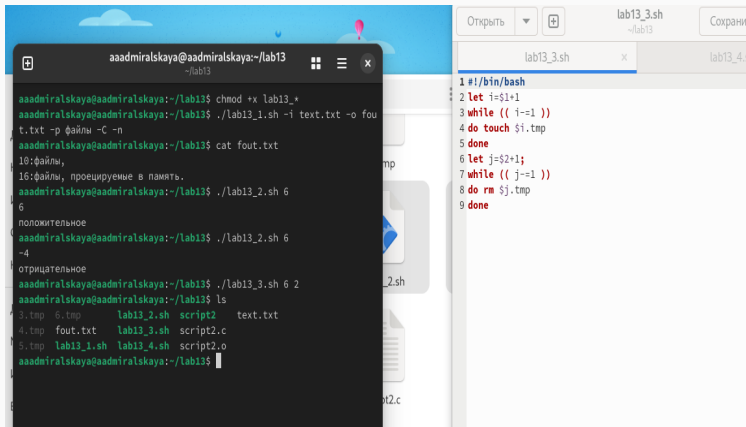
The image shows a terminal window and a code editor. The terminal window, titled 'aaadmirlskaya@aadmiralskaya:~/lab13', shows the execution of a shell script 'lab13\_2.sh' with arguments '6' and '4'. The script outputs 'положительное' (positive) for '6' and 'отрицательное' (negative) for '4'. The code editor, titled 'lab13\_2.sh', shows the script code:

```
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 has a title bar 'aaadmiralskaya@aaadmiralskaya:~/lab13' and shows the execution of several shell scripts. The file editor on the right shows the content of 'lab13\_3.sh'.

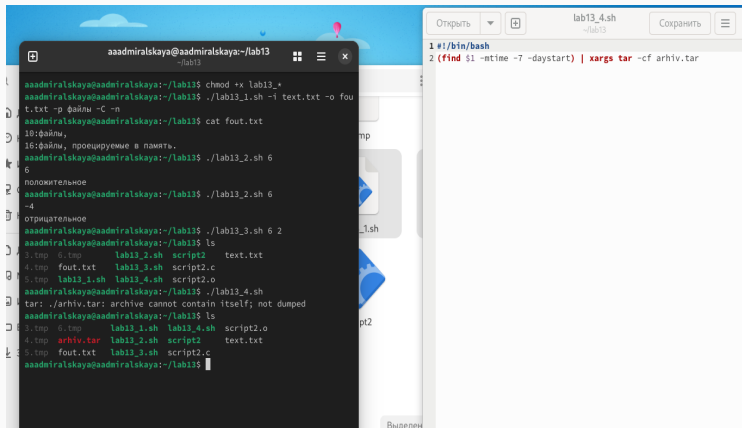
```
aaadmiralskaya@aaadmiralskaya:~/lab13$ chmod +x lab13_*
aaadmiralskaya@aaadmiralskaya:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C -n
aaadmiralskaya@aaadmiralskaya:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
aaadmiralskaya@aaadmiralskaya:~/lab13$ ./lab13_2.sh 6
6
положительное
aaadmiralskaya@aaadmiralskaya:~/lab13$ ./lab13_2.sh 6
-4
отрицательное
aaadmiralskaya@aaadmiralskaya:~/lab13$ ./lab13_3.sh 6 2
aaadmiralskaya@aaadmiralskaya:~/lab13$ ls
3.tmp  6.tmp      lab13_2.sh  script2    text.txt
4.tmp  fout.txt    lab13_3.sh  script2.c
5.tmp  lab13_1.sh lab13_4.sh  script2.o
aaadmiralskaya@aaadmiralskaya:~/lab13$
```

```
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
```

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 'aaadmirlskaya@aaadmirlskaya:~/lab13' with a dark theme. It displays the execution of several shell scripts: 'lab13\_1.sh' (creating text.txt), 'lab13\_2.sh' (creating fout.txt), and 'lab13\_3.sh' (creating script2.o). It also shows the output of 'ls' and an error message from 'tar' when trying to create 'arhiv.tar' because it contains itself. The foreground window is a text editor titled 'lab13\_4.sh' with a light theme, showing a shell script that uses 'find' to locate files and 'xargs tar' to archive them into 'arhiv.tar'.

```
aaadmirlskaya@aaadmirlskaya:~/lab13
aaadmirlskaya@aaadmirlskaya:~/lab13$ chmod +x lab13_*
aaadmirlskaya@aaadmirlskaya:~/lab13$ ./lab13_1.sh -i text.txt -o fout
t.txt -p файлы -C -n
aaadmirlskaya@aaadmirlskaya:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
aaadmirlskaya@aaadmirlskaya:~/lab13$ ./lab13_2.sh 6
6
положительное
aaadmirlskaya@aaadmirlskaya:~/lab13$ ./lab13_2.sh 6
-4
отрицательное
aaadmirlskaya@aaadmirlskaya:~/lab13$ ./lab13_3.sh 6 2
aaadmirlskaya@aaadmirlskaya:~/lab13$ ls
3.tmp  6.tmp      lab13_2.sh  script2    text.txt
4.tmp  fout.txt   lab13_3.sh  script2.c
5.tmp  lab13_1.sh lab13_4.sh  script2.o
aaadmirlskaya@aaadmirlskaya:~/lab13$ ./lab13_4.sh
tar: ./arhiv.tar: archive cannot contain itself; not dumped
aaadmirlskaya@aaadmirlskaya:~/lab13$ ls
3.tmp  6.tmp      lab13_1.sh lab13_4.sh  script2.o
4.tmp  arhiv.tar  lab13_2.sh  script2    text.txt
5.tmp  fout.txt  lab13_3.sh  script2.c
aaadmirlskaya@aaadmirlskaya:~/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.