

# Операционные системы

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

---

Яковлева Дарья Сергеевна

19 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

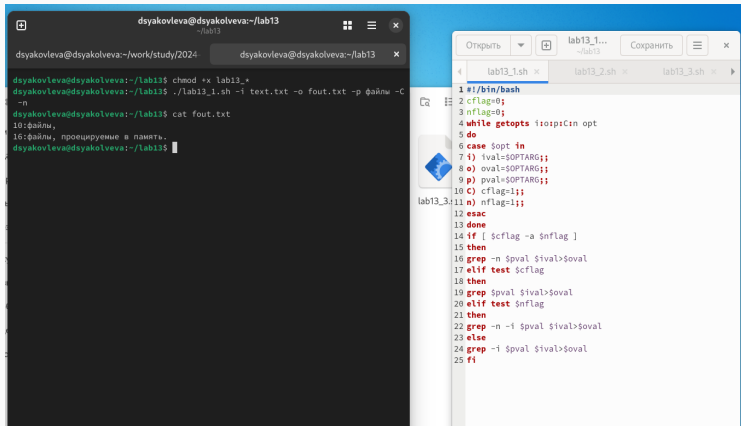
## Процесс выполнения лабораторной работы

---

1. Используя команды `getopts` `grep` 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; `-p 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, titled 'dsyakovleva@dsyakovleva:~/lab13', shows the execution of a shell script 'lab13\_1.sh' with arguments 'text.txt', 'fout.txt', and '-p'. The script's output is displayed in the terminal. The code editor, titled 'lab13\_1...', shows the source code of the script 'lab13\_1.sh'.

```
dsyakovleva@dsyakovleva:~/lab13$ chmod +x lab13_*
dsyakovleva@dsyakovleva:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C
-n
dsyakovleva@dsyakovleva:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
dsyakovleva@dsyakovleva:~/lab13$
```

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

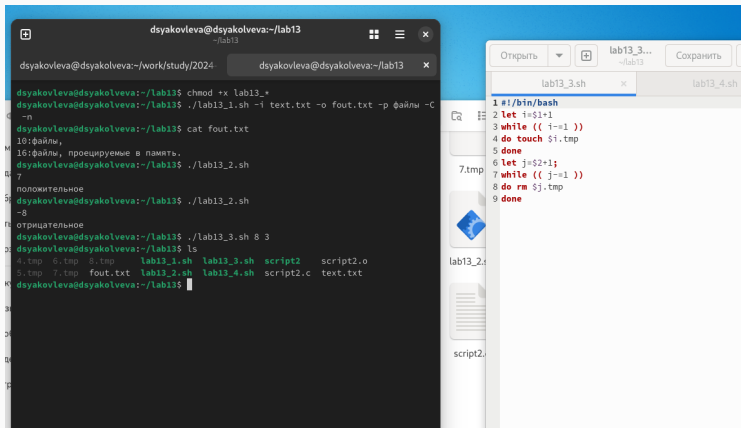
The image shows a terminal window on the left and a code editor on the right. The terminal window, titled 'dsyakolveva@dsyakolveva:~/lab13', shows the execution of a shell script 'lab13\_2.sh'. The script's output is displayed in the terminal, showing the contents of 'fout.txt' and the results of the script's execution. The code editor on the right, titled 'lab13\_2...', shows the source code of 'lab13\_2.sh'. The code is a shell script that compiles 'script2.c' and runs it, then uses a case statement to echo different messages based on the script's output.

```
dsyakolveva@dsyakolveva:~/lab13$ chmod +x lab13_*
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C
-n
dsyakolveva@dsyakolveva:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_2.sh
7
положительное
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_2.sh
-8
отрицательное
dsyakolveva@dsyakolveva:~/lab13$
```

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

Рис. 2: Задание 2

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



The image shows a terminal window on the left and a file editor on the right. The terminal window displays the execution of several shell scripts in a directory named /lab13. The file editor shows the content of lab13\_3.sh, which is a shell script that uses loops and touch commands to create and manage temporary files.

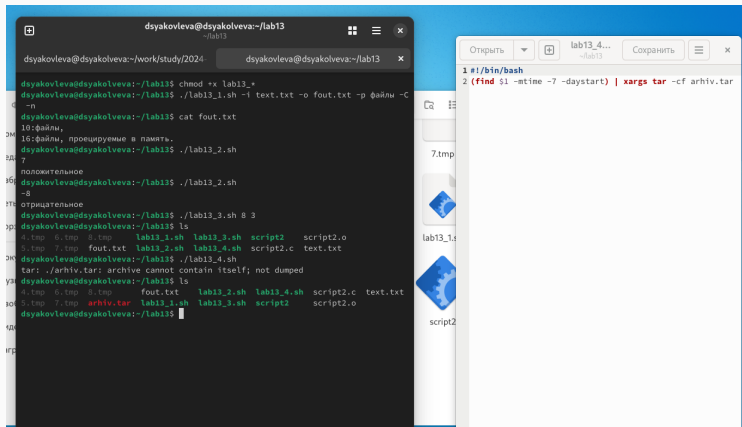
```
dsyakovleva@dsyakolveva:~/lab13
dsyakovleva@dsyakolveva:~/work/study/2024
dsyakovleva@dsyakolveva:~/lab13$ chmod +x lab13_*
dsyakovleva@dsyakolveva:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C
-n
dsyakovleva@dsyakolveva:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
dsyakovleva@dsyakolveva:~/lab13$ ./lab13_2.sh
7
положительное
dsyakovleva@dsyakolveva:~/lab13$ ./lab13_2.sh
-8
отрицательное
dsyakovleva@dsyakolveva:~/lab13$ ./lab13_3.sh 8 3
dsyakovleva@dsyakolveva:~/lab13$ ls
4.tmp 6.tmp 8.tmp lab13_1.sh lab13_3.sh script2 script2.o
5.tmp 7.tmp fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
dsyakovleva@dsyakolveva:~/lab13$
```

```
lab13_3.sh
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 a terminal window and a file manager. The terminal window, titled 'dsyakolveva@dsyakolveva:~/lab13', displays the following commands and output:

```
dsyakolveva@dsyakolveva:~/lab13$ chmod +x lab13_*
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_1.sh -f text.txt -o fout.txt -p файлы -C
-n
dsyakolveva@dsyakolveva:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_2.sh
7
положительное
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_2.sh
-8
отрицательное
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_3.sh 8 3
dsyakolveva@dsyakolveva:~/lab13$ ls
4.tmp 6.tmp 8.tmp lab13_1.sh lab13_3.sh script2 script2.o
5.tmp 7.tmp fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
dsyakolveva@dsyakolveva:~/lab13$ ./lab13_4.sh
tar: ./arhiv.tar: archive cannot contain itself; not dumped
dsyakolveva@dsyakolveva:~/lab13$ ls
4.tmp 6.tmp 8.tmp fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
5.tmp 7.tmp arhiv.tar lab13_1.sh lab13_3.sh script2 script2.o
dsyakolveva@dsyakolveva:~/lab13$
```

The file manager window, titled 'lab13\_4...', shows the following commands and output:

```
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.