

# Программирование в командном процессоре ОС UNIX. Ветвления и циклы

---

Сайфидинов Фируз Фаросатшоевич НБИбд-02-22<sup>1</sup>

17 апреля, 2023, Москва, Россия

<sup>1</sup>Российский 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

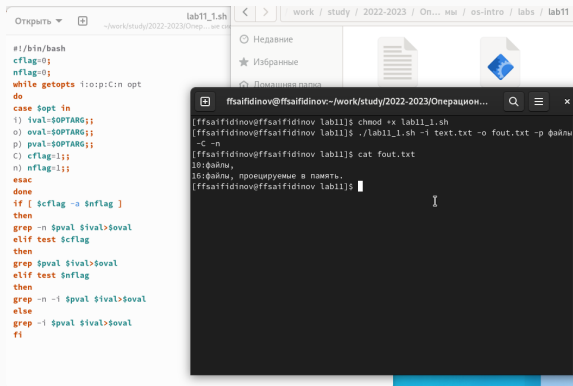
# **Процесс выполнения лабораторной работы**

---

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 Linux desktop environment. In the background, a file manager window displays the contents of a directory named 'lab11'. It contains a file 'lab11\_1.sh' and a folder 'Недавние' (Recent). In the foreground, a terminal window is open, showing the execution of the script 'lab11\_1.sh'. The script is a shell script that takes two arguments, 'text.txt' and 'fout.txt', and performs a series of operations including file creation, permission setting, and a loop of file operations.

```
#!/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
```

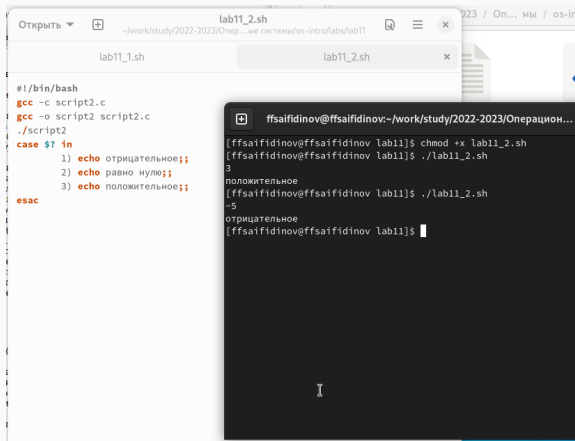
```
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...$ chmod +x lab11_1.sh
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...$ ./lab11_1.sh -i text.txt -o fout.txt -p файлы
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...$
```

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 two overlapping terminal windows. The background window, titled 'lab11\_2.sh', displays the source code of a shell script. The script starts with a shebang, compiles a C program 'script2.c' using 'gcc', and then uses a 'case' statement to check the exit status of './script2'. The 'case' statement has three branches: '1) echo отрицательное;;' for a non-zero exit status, '2) echo равно нулю;;' for a zero exit status, and '3) echo положительное;;' for a non-zero exit status. The foreground window, titled 'ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...', shows the execution of the script. It starts with 'chmod +x lab11\_2.sh', followed by './lab11\_2.sh' which outputs '3' and 'положительное'. A second './lab11\_2.sh' execution outputs '-5' and 'отрицательное'.

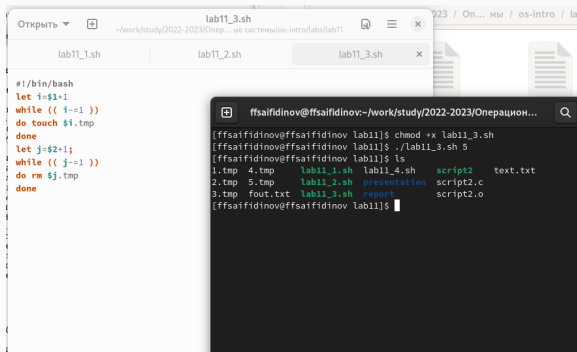
```
#!/bin/bash
gcc -c script2.c
gcc -o script2 script2.c
./script2
case $? in
    1) echo отрицательное;;
    2) echo равно нулю;;
    3) echo положительное;;
esac
```

```
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...
[ffsaifidinov@ffsaifidinov lab11]$ chmod +x lab11_2.sh
[ffsaifidinov@ffsaifidinov lab11]$ ./lab11_2.sh
3
положительное
[ffsaifidinov@ffsaifidinov lab11]$ ./lab11_2.sh
-5
отрицательное
[ffsaifidinov@ffsaifidinov lab11]$
```

Рис. 2: Задание 2

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

# Выполнение работы



The image shows a web browser window with a tab titled 'lab11\_3.sh' and a terminal window. The browser displays the content of the 'lab11\_3.sh' script, which is a bash script that creates a loop to touch files and then removes them. The terminal window shows the execution of the script, with the user running 'chmod +x lab11\_3.sh' and then './lab11\_3.sh 5'. The terminal output shows the script creating files '1.tmp', '2.tmp', and '3.tmp' and then removing them.

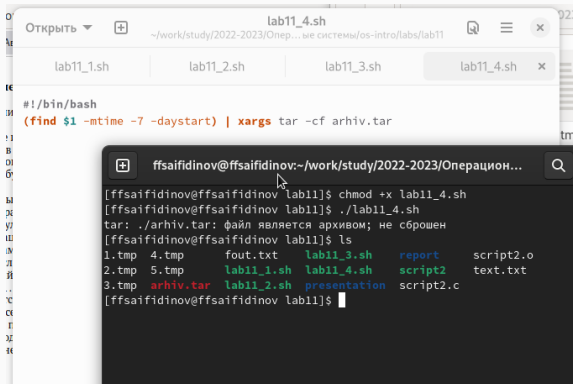
```
#!/bin/bash
let i=$1+1
while (( i-->0 ))
do touch $i.tmp
done
let j=$2+1;
while (( j-->0 ))
do rm $j.tmp
done
```

```
ffsaifidinov@ffsaifidinov:~/work/study/2022-2023/Операцион...
[ffsaifidinov@ffsaifidinov lab11]$ chmod +x lab11_3.sh
[ffsaifidinov@ffsaifidinov lab11]$ ./lab11_3.sh 5
[ffsaifidinov@ffsaifidinov lab11]$ ls
1.tmp  4.tmp  lab11_1.sh  lab11_4.sh  script2  text.txt
2.tmp  5.tmp  lab11_2.sh  presentation  script2.c
3.tmp  fout.txt  lab11_3.sh  report      script2.o
[ffsaifidinov@ffsaifidinov lab11]$
```

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 screenshot shows a terminal window with a tab labeled 'lab11\_4.sh'. The terminal content is as follows:

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

Below this, a smaller terminal window is overlaid, showing the execution of the script:

```
[ffsaifidinov@ffsaifidinov lab11]$ chmod +x lab11_4.sh
[ffsaifidinov@ffsaifidinov lab11]$ ./lab11_4.sh
tar: ./arhiv.tar: файл является архивом; не сброшен
[ffsaifidinov@ffsaifidinov lab11]$ ls
```

1.tmp	4.tmp	fout.txt	lab11_3.sh	report	script2.o
2.tmp	5.tmp	lab11_1.sh	lab11_4.sh	script2	text.txt
3.tmp		arhiv.tar	lab11_2.sh	presentation	script2.c

The final prompt is `[ffsaifidinov@ffsaifidinov lab11]$`.

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.