

# Программирование в командном процессоре ОС UNIX.

## Ветвления и циклы

---

Лебедева Ольга Андреевна

<sup>1</sup>RUDN University, Moscow, Russian Federation

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

Цикл `for` - это оператор языка программирования `bash`, который позволяет многократно выполнять код. Цикл `for` классифицируется как оператор итерации, то есть это повторение процесса в сценарии `bash`. Например, вы можете запустить команду или задачу UNIX 5 раз или прочитать и обработать список файлов с помощью цикла `for`.

Цикл `while` используется для выполнения заданный набор команд неизвестное число раз до тех пор , как данное условие принимает значение истинно.

1. Используя команды `getopts` `grep`, написали командный файл, который анализирует командную строку с ключами: (рис. 1)

- `-iinputfile` — прочитать данные из указанного файла;
- `-ooutputfile` — вывести данные в указанный файл;
- `-rшаблон` — указать шаблон для поиска;
- `-C` — различать большие и малые буквы;
- `-n` — выдавать номера строк.

# Задание 1

```
while getopts "i:o:p:C:n" opt
do
case $opt in
i)inputfile="$OPTARG";;
o)outputfile="$OPTARG";;
p)shablon="$OPTARG";;
C)registr="";;
n)number="";;
esac
done
grep -n "$shablon" "$inputfile" > "$outputfile"
```

Figure 1: Код скрипта1

Сделаем файл исполняемым. (рис. 2)

```
salebedeva@dkn52 ~$ chmod +x lab11.1
salebedeva@dkn52 ~$ ./lab11.1 -i conf.txt -o output.txt -p n etconf -C -n
salebedeva@dkn52 ~$ enacs output.txt
```

Figure 2: Исполняем файл

Проверим результат работы программы. (рис. 3)

```
1:appstream.conf
2:brltty.conf
3:ca-certificates.conf
4:cachedfilesd.conf
5:cfg-update.conf
6:cpufreq-bench.conf
```

## Задание 2

2. Написали на языке программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено. (рис. 4) (рис. 5)

```
echo "Insert number"
read n
if [ $n -gt 0 ]
then echo ">0"
elif [ $n -eq 0 ]
then echo "=0"
else echo "<0"
fi
```

Figure 4: Код скрипта2

```
oalebedeva@dk8n52 ~ $ chmod +x lab11_2
oalebedeva@dk8n52 ~ $ ./lab11_2
Insert number
7
>0
oalebedeva@dk8n52 ~ $ ./lab11_2
Insert number
0
=0
oalebedeva@dk8n52 ~ $ ./lab11_2
Insert number
-12
<0
```

Figure 5: Результат работы

## Задание 3

3. Написали командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до [?] (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют). (рис. 6)

```
while getopts "c:r" opt
do
case $opt in
c)n="$OPTARG"; for i in $(seq 1 $n); do touch "$i.tmp"; done;;
r)for i in $(find -name "*.tmp"); do rm $i; done;;
esac
done
```

Figure 6: Код скрипта3

Задали программе создать 4 файла tmp. (рис. 7)

```
oalebedeva@dk8n52 ~ $ ./lab11_3 -c 4
oalebedeva@dk8n52 ~ $ ls
1.tmp      bin          lab07.sh
2.tmp      conf.txt     lab07.sh~
3.tmp      feathers     '#lab11_1#'
4.tmp      file.txt     lab11_1
```



## Задание 3

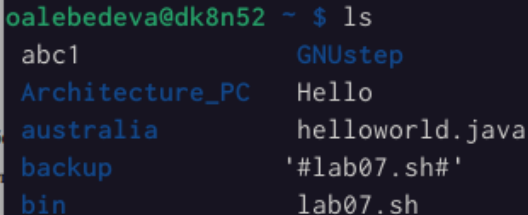
Теперь одной командой удалим созданные файлы. (рис. 8)



```
oalebedeva@dk8n52 ~ $ ./lab11_3 -r
```

Figure 8: Команда удаления

Проверяем содержание домашнего каталога, файлы были удалены. (рис. 9)



```
oalebedeva@dk8n52 ~ $ ls
abc1          GNUstep
Architecture_PC Hello
australia     helloworld.java
backup        '#lab07.sh#'
bin           lab07.sh
```

Figure 9: Результат работы

## Задание 4

4. Написали командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find). (рис. 10)

```
while getopts ":p:" opt; do
case $opt in
p)dir="$OPTARG";;
esac
done

find $dir -mtime +0 -mtime -7 -print0 | xargs -0 tar -cf archive.tar
```

Figure 10: Код скрипта4

Файлы, которые были изменены менее недели тому назад: (рис. 11)

backup	lab07.sh	my_os	script3~
bin	lab07.sh~	my_os1	script4
conf.txt	'#lab11_1#'	output.txt	ski.plases

Figure 11: Результат работы

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