

# Лабораторная работа №11

Программирование в командном процессоре ОС UNIX. Ветвления и  
циклы

Гнатюк Анастасия Станиславовна

# Содержание


Цель работы	3
Выполнение лабораторной работы	4
Вывод	8

## Цель работы

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

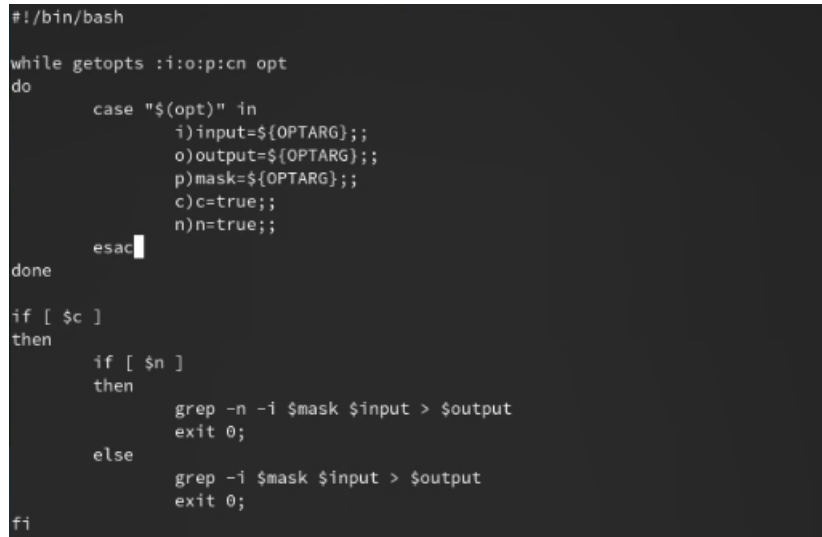
# Выполнение лабораторной работы

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-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, определяемые ключом `-r`.



```
[asgnatyuk@fedora lab11]$ vi script
```

Рис. 1: Рис.1



```
#!/bin/bash
while getopts :i:o:p:cn opt
do
    case "${opt}" in
        i)input=${OPTARG};;
        o)output=${OPTARG};;
        p)mask=${OPTARG};;
        c)c=true;;
        n)n=true;;
    esac
done
if [ $c ]
then
    if [ $n ]
    then
        grep -n -i $mask $input > $output
        exit 0;
    else
        grep -i $mask $input > $output
        exit 0;
    fi
fi
```

Рис. 2: Рис.2

2. Написать на языке Си 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 в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.

```
[asgnatyuk@fedora lab11]$ vi script2.cpp
```

Рис. 3: Рис.3

```
#include <iostream>
using namespace std;

int main(int argc, char *argv[]){
    if (atoi(argv[1]))>0) {
        exit(1);
    } else if (atoi(argv[1])==0){
        exit(2);
    } else {
        exit(3);
    }
    return 0;
}
```

Рис. 4: Рис.4

```
#!/bin/bash

CC=g++
EXEC=compare
SRC=compare.cpp

if [ "$SRC" != "$EXEC" ]
then
    echo "REBUILDING $EXEC ....."
    $CC -o $EXEC $SRC
fi

./$EXEC $1

ec=$?
if [ "$ec" == "1" ]
then
    echo "input > 0"
```

Рис. 5: Рис.5

```

fi

./$EXEC $1

ec=$?
if [ "$ec" == "1" ]
then
    echo "input > 0"
fi

if [ "$ec" == "2" ]
then
    echo "input > 0"
fi

if [ "$ec" == "3" ]
then
    echo "input < 0"
fi
-- INSERT --

```

Рис. 6: Рис.6

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (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).

```

#!/bin/bash

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

```

Рис. 7: Рис.7

```

[asgnatyuk@fedora lab11]$ ./script3 -c 6
[asgnatyuk@fedora lab11]$ ls
1.tmp  3.tmp  5.tmp  compare.cpp  script1      script2_1.cpp  script2.cpp
2.tmp  4.tmp  6.tmp  conf.txt     script2_1    script2.c      script3

```

Рис. 8: Рис.8

```
[asgnatyuk@fedora lab11]$ ./script3 -r  
[asgnatyuk@fedora lab11]$ ls  
compare.cpp  script1      script2_1.cpp  script2.cpp  
conf.txt     script2_1    script2.c      script3
```

Рис. 9: Рис.9

## Вывод

Я изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.