

# **Лабораторная работа №11**

**Программирование в командном процессоре ОС UNIX. Ветвления и  
циклы**

Перевощиков Данил Алексеевич

# Содержание

1	Цель работы	5
2	Ход работы	6
3	Вывод	11
4	Контрольные вопросы	12

## Список иллюстраций

2.1	Первый скрипт. . . . .	6
2.2	Выполнение первого скрипта. . . . .	7
2.3	Программа на языке c++. . . . .	7
2.4	Второй скрипт. . . . .	8
2.5	Выполнение второго скрипта. . . . .	8
2.6	Третий скрипт. . . . .	8
2.7	Выполнение третьего скрипта. . . . .	9
2.8	Третий скрипт. . . . .	9
2.9	Выполнение четвертого скрипта. . . . .	9
2.10	Результат выполнения четвертого скрипта. . . . .	10


## Список таблиц

# 1 Цель работы

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

## 2 Ход работы

1. Используя команды `getopts` `grep`, написали командный файл, который анализирует командную строку с ключами: `-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, определяемые ключом `-p`.(рис. 2.1)



```
1 #!/bin/bash
2
3 while getopts "i:o:p:cn" opt
4 do
5     case $opt in
6         i)inputfile="$OPTARG";;
7         o)outputfile="$OPTARG";;
8         p)sample="$OPTARG";;
9         c)reg="";;
10        n)line="";;
11    esac
12 done
13
14 grep -n "$sample" "$inputfile" > "$outputfile"
```

Рис. 2.1: Первый скрипт.

Мы использовали файл `conf.txt` из одной из прошлых работ.(рис. 2.2)

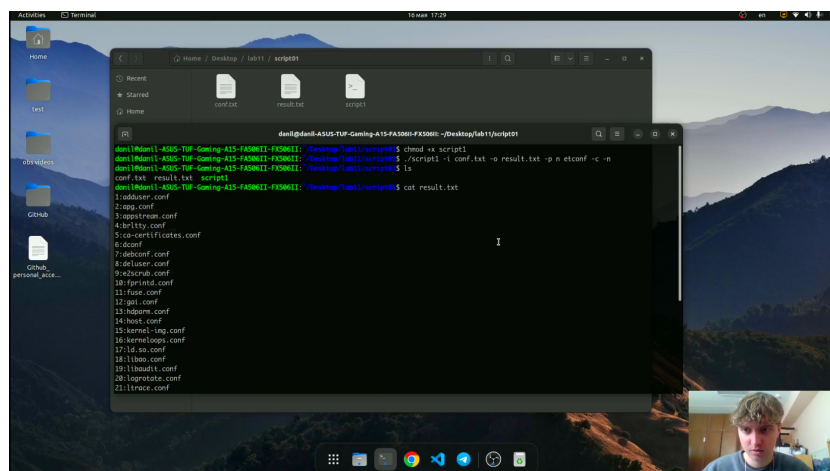


Рис. 2.2: Выполнение первого скрипта.

2. Написали на языке c++ 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 вызывает эту программу и, проанализировав с помощью команды `$?`, выдает сообщение о том, какое число было введено.(рис. 2.3, 2.4, 2.5)

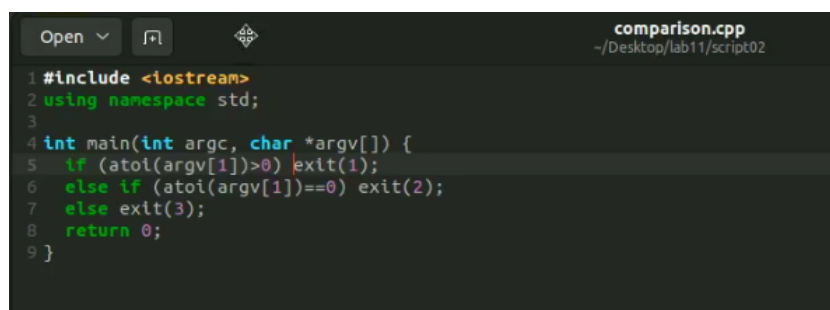


Рис. 2.3: Программа на языке c++.



```
1 #!/bin/bash
2
3 RES=result
4 SRC=comparison.cpp
5
6 if [ "$SRC" -nt "$RES" ]
7 then
8     echo "Creating $RES ..."
9     g++ -o $RES $SRC
10 fi
11
12 ./ $RES $1
13
14 ec=?
15
16 if [ "$ec" == "1" ]
17 then
18     echo "input > 0"
19 fi
20
21 if [ "$ec" == "2" ]
22 then
23     echo "input = 0"
24 fi
25
26 if [ "$ec" == "3" ]
27 then
28     echo "input < 0"
29 fi
```

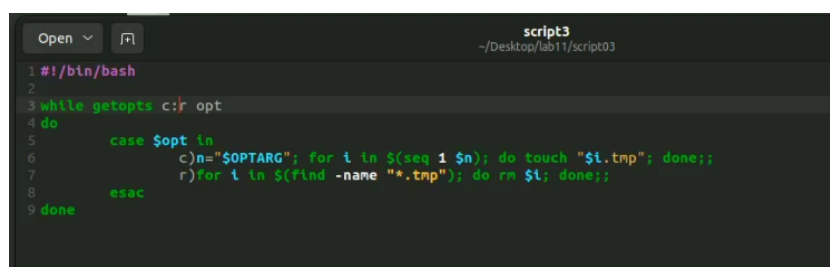
Рис. 2.4: Второй скрипт.



```
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script02
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$ chmod +x script2
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$ ./script2 -3
Creating result ...
input < 0
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$ ./script2 0
input = 0
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$ 6
6: command not found
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$ ./script2 6
input > 0
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II:~/Desktop/lab11/script02$
```

Рис. 2.5: Выполнение второго скрипта.

3. Написали командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N. Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл умеет удалять все созданные им файлы (если они существуют).(рис. 2.6, 2.7)



```
1 #!/bin/bash
2
3 while getopts c:r opt
4 do
5     case $opt in
6         c)n="$OPTARG"; for i in $(seq 1 $n); do touch "$i.tmp"; done;;
7         r)for i in $(find -name "*.tmp"); do rm $i; done;;
8     esac
9 done
```

Рис. 2.6: Третий скрипт.



```
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$ chmod +x script3
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$ ./script3 -c 8
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$ ls
1.tmp 2.tmp 3.tmp 4.tmp 5.tmp 6.tmp 7.tmp 8.tmp script3
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$ ./script3 -r
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$ ls
script3
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script03$
```

Рис. 2.7: Выполнение третьего скрипта.

4. Написали командный файл, который с помощью команды tar запаковывает в архив файлы, которые были изменены менее недели тому назад.(рис. 2.8, 2.9)

```
script4
~/Desktop/lab11/script04

1 #!/bin/bash
2
3 while getopts :p: opt
4 do
5     case $opt in
6         p)dir="$OPTARG";;
7     esac
8 done
9
10 find $dir -mtime -7 -mtime +0 -type f > arch.txt
11
12 tar -cf result.tar -T arch.txt
```

Рис. 2.8: Третий скрипт.

```
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script04
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script04$ chmod +x script4
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script04$ ./script4 -p /home/danil/
tar: Removing leading '/' from member names
tar: Removing leading '/' from hard link targets
danil@danil-ASUS-TUF-Gaming-A15-FA506II-FX506II: ~/Desktop/lab11/script04$
```

Рис. 2.9: Выполнение четвертого скрипта.

После этого создался архив result.tar.(рис. 2.10)

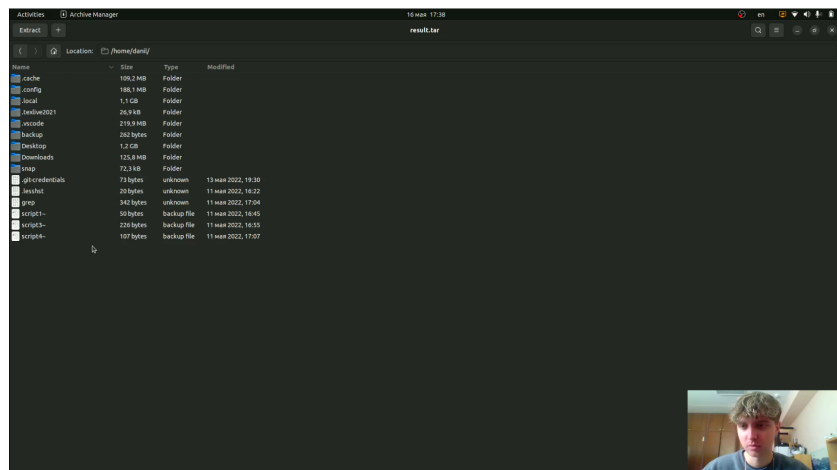


Рис. 2.10: Результат выполнения четвертого скрипта.

## 3 Вывод

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

## 4 Контрольные вопросы

### 1. Каково предназначение команды *getopts*?

Осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.

### 2. Какое отношение метасимволы имеют к генерации имён файлов?

При перечислении имен файлов текущего каталога можно использовать следующие символы: - \* — соответствует произвольной, в том числе и пустой строке; - ? — соответствует любому одному символу; - [c1-c1] — соответствует любому символу, лексикографически находящемуся между символами c1 и c2. - echo \* — выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды ls; - ls \*.c — выведет все файлы с последними двумя символами, равными .c. - echo prog.? — выдаст все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются prog.. - [a-z]\* — соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита.

### 3. Какие операторы управления действиями вы знаете?

Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости от результатов проверки некоторого условия. Для решения подобных задач язык программирования bash предоставляет Вам возможность использовать такие управляющие конструкции, как for, case, if и while. С точки зрения командного процессора эти

управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути дела являются операторами языка программирования `bash`. Поэтому при описании языка программирования `bash` термин оператор будет использоваться наравне с термином команда.

#### 4. *Какие операторы используются для прерывания цикла?*

Два несложных способа позволяют вам прерывать циклы в оболочке `bash`. Команда `break` завершает выполнение цикла, а команда `continue` завершает данную итерацию блока операторов. Команда `break` полезна для завершения цикла `while` в ситуациях, когда условие перестает быть правильным. Пример бесконечного цикла `while`, с прерыванием в момент, когда файл перестает существовать:

- `while true do`
- `if [! -f $file] then break`
- `fi`
- `sleep 10 done`

#### 5. *Для чего нужны команды `false` и `true`?*

Команды ОС UNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях. Команда `test`, например, создана специально для использования в командных файлах. Единственная функция этой команды заключается в выработке кода завершения.

#### 6. *Что означает строка `if test -f mans/i.$s`, встреченная в командном файле?*

Введенная строка означает условие существования файла `mans/i.$s`

#### 7. *Объясните различия между конструкциями `while` и `until`.*

Если речь идет о 2-х параллельных действиях, то это while. когда мы показываем, что сначала делается 1-е действие. потом оно заканчивается при наступлении 2-го действия, применяем until.