

# **Отчет по лабораторной работе №13**

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

Разанацуа Сара Естэлл

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>4</b>	<b>Выводы</b>	<b>14</b>

## Список иллюстраций

3.1	Создание файла . . . . .	8
3.2	Код программы . . . . .	9
3.3	Результат работы программы . . . . .	9
3.4	Результат работы программы . . . . .	10
3.5	Код программы на Си . . . . .	10
3.6	Код программы . . . . .	11
3.7	Результат работы программы . . . . .	11
3.8	Код программы . . . . .	12
3.9	Результат работы программы . . . . .	12
3.10	Код программы . . . . .	13
3.11	Результат работы программы . . . . .	13

## Список таблиц

# 1 Цель работы

Цель данной лабораторной работы - изучить основы программирования в оболочке ОС UNIX, научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

## 2 Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:
  - `-iinputfile` — прочитать данные из указанного файла;
  - `-ooutputfile` — вывести данные в указанный файл;
  - `-р`шаблон — указать шаблон для поиска;
  - `-С` — различать большие и малые буквы;
  - `-п` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-р`.
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.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до  $\infty$  (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).
4. Написать 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).

### 3 Выполнение лабораторной работы

- Создаю файл с разрешением на исполнение (рис.1).

```
serazanacua@serazanacua:~$ chmod +x l1l.sh
serazanacua@serazanacua:~$ bash l1l.sh -p улит -i input.txt -o output.txt -c -n
grep: input.txt: Нет такого файла или каталога
serazanacua@serazanacua:~$ touch input.txt
serazanacua@serazanacua:~$ touch output.txt
serazanacua@serazanacua:~$ bash l1l.sh -p улит -i input.txt -o output.txt -c -n
serazanacua@serazanacua:~$
```

Рис. 3.1: Создание файла

- а затем ищет в указанном файле нужные строки, определяемые ключом -p (рис.2).



```
#!/bin/bash

while getopts i:o:p:cn optletter
do
case $optletter in
  i) iflag=1; ival=$OPTARG;;
  o) oflag=1; oval=$OPTARG;;
  p) pflag=1; pval=$OPTARG;;
  c) cflag=1;;
  n) nflag=1;;
  *) echo Illegal option $optletter;;
  esac
done

if ! test $cflag
then
  cf=-i
fi

if test $nflag
then
  nf=-n
fi

grep $cf $nf $pval $ival >> $oval
```

Рис. 3.2: Код программы

- Результат работы программы в файле output.txt (рис.3).

```
заходит в бар улитка
наливает шампанское
подходит дональд дак
просит налить ему вино
|
```

Рис. 3.3: Результат работы программы



Рис. 3.4: Результат работы программы

- Создаю исполняемый файл для второй программы, также создаю файл 12.c для программы на Си. Пишу программу на языке Си, 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 (рис.5).

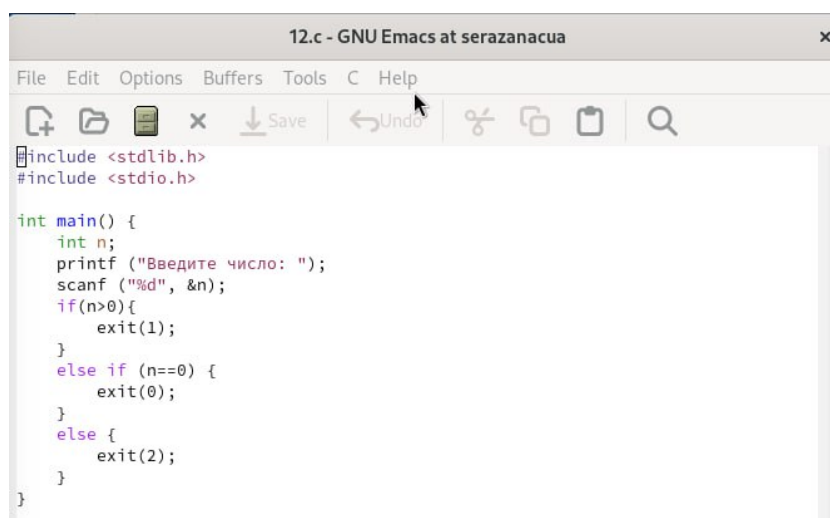



Рис. 3.5: Код программы на Си

- Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено (рис.6).

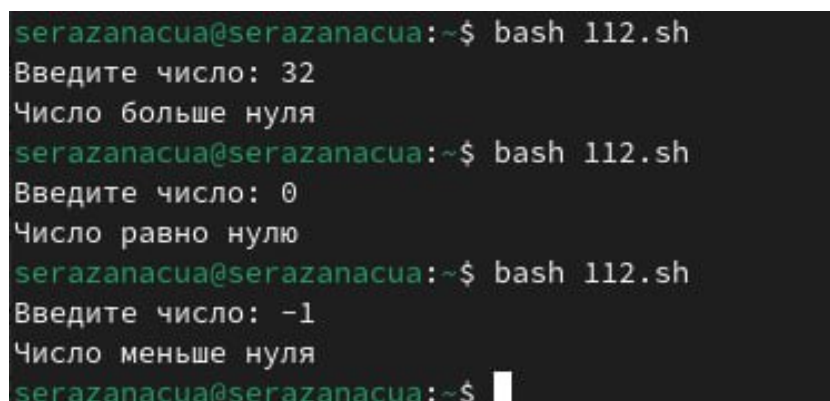


```
#!/bin/bash

gcc -o cprog 12.c
./cprog
case $? in
0) echo "Число равно нулю";;
1) echo "Число больше нуля";;
2) echo "Число меньше нуля";;
esac
```

Рис. 3.6: Код программы

- Программа работает корректно (рис. fig:007).



```
serazanacua@serazanacua:~$ bash 112.sh
Введите число: 32
Число больше нуля
serazanacua@serazanacua:~$ bash 112.sh
Введите число: 0
Число равно нулю
serazanacua@serazanacua:~$ bash 112.sh
Введите число: -1
Число меньше нуля
serazanacua@serazanacua:~$
```

Рис. 3.7: Результат работы программы

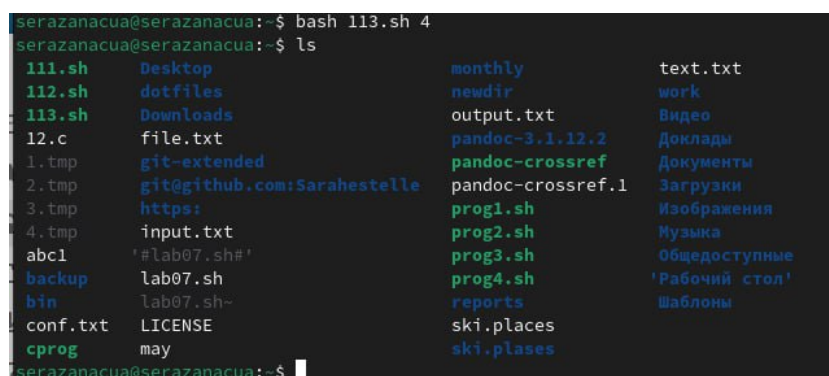
- Создаю исполняемый файл для третьей программы. Командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют) (рис.fig:008).



```
#!/bin/bash
for((i=1; i<=4; i++))
do
if test -f "$i".tmp
then rm "$i".tmp
else touch "$i.tmp"
fi
done
```

Рис. 3.8: Код программы

- Проверяю, что программа создала файлы и удалила их при соответствующих запросах (рис. fig:009).



```
serazanacua@serazanacua:~$ bash 113.sh 4
serazanacua@serazanacua:~$ ls
111.sh      Desktop      monthly      text.txt
112.sh      dotfiles     newdir       work
113.sh      Downloads    output.txt   Видео
12.c        file.txt     pandoc-3.1.12.2 Доклады
1.tmp       git-extended pandoc-crossref Документы
2.tmp       git@github.com:Sarahestelle pandoc-crossref.1 Загрузки
3.tmp       https:       prog1.sh     Изображения
4.tmp       input.txt   prog2.sh     Музыка
abcl        '#lab07.sh#' prog3.sh     Общедоступные
backup      lab07.sh    prog4.sh     'Рабочий стол'
bin         lab07.sh~   reports      Шаблоны
conf.txt    LICENSE     ski.places
cprog       may         ski.places
serazanacua@serazanacua:~$
```

Рис. 3.9: Результат работы программы

- Создаю исполняемый файл для четвертой программы. Это 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. fig:010).

```
114.sh - GNU Emacs at serazanacua
File Edit Options Buffers Tools Sh-Script Help
[Icons: Save, Undo, Cut, Copy, Paste, Find]
#! /bin/bash
find $* -mtime -7 -mtime +0 -type f > FILES.txt
tar -cf archive.tar -T FILES.txt
```

Рис. 3.10: Код программы

- Проверяю работу программы (рис. fig:011).

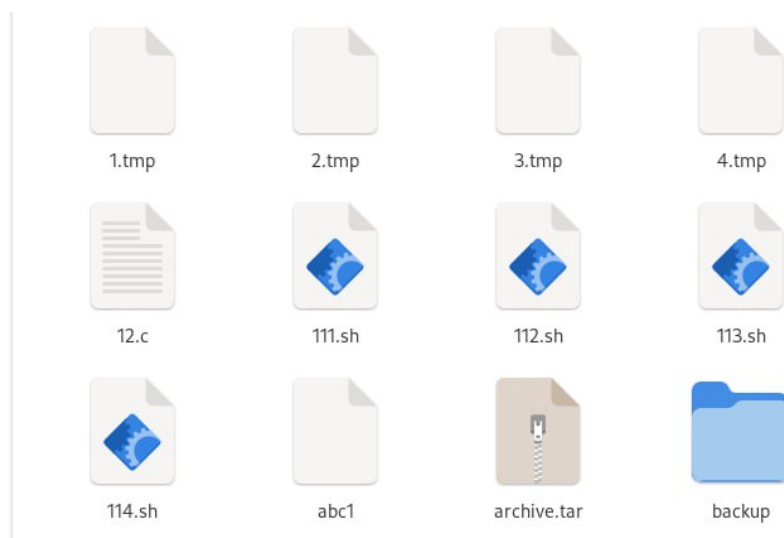


Рис. 3.11: Результат работы программы

## 4 Выводы

- При выполнении данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX, научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.