

# Лабораторная работа №11

---

Шубнякова Дарья, НКАбд-03-22

1. Цель
2. Теоретическое введение
3. Основные задачи
4. Процесс выполнения
5. Вывод
6. Список литературы

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

Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек: - оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций; - C-оболочка (или csh) — надстройка на оболочке Борна, использующая C-подобный синтаксис команд с возможностью сохранения истории выполнения команд; - оболочка Корна (или ksh) — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки Борна; - BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек C и Корна (разработка компании Free Software Foundation).

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; `-rшаблон` — 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`.

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.

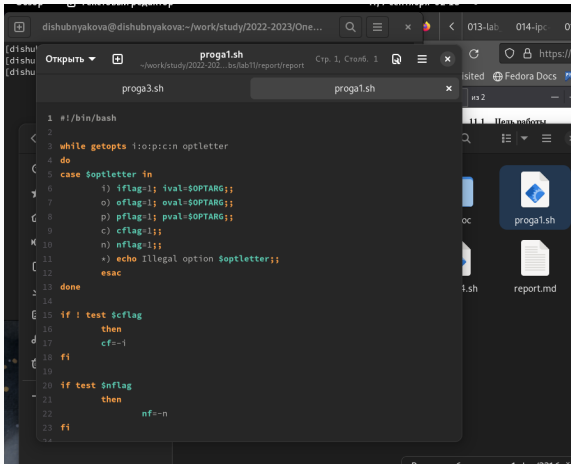
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до [?] (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы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



# Процесс выполнения

1. Командный файл, который анализирует командную строку с ключами.



```
#!/bin/bash
while getopts i:op:c:n 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
```

Рис. 1: Программа

# Процесс выполнения

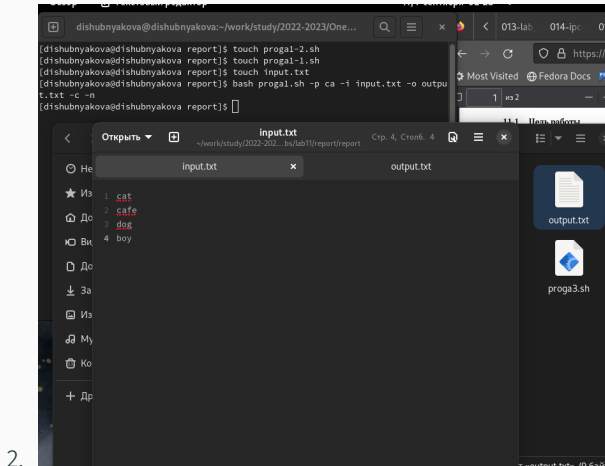


Рис. 2: Выполнение программы

# Процесс выполнения

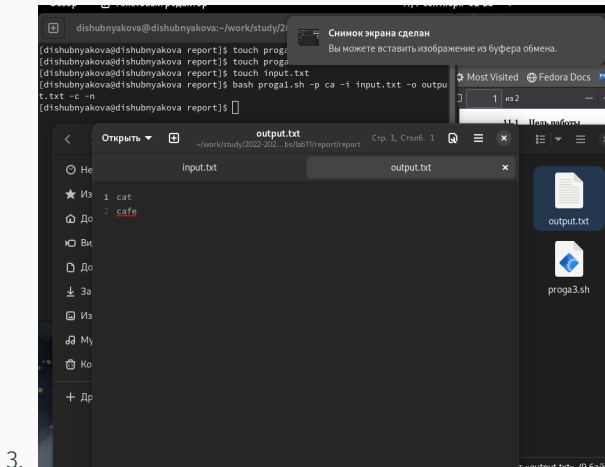
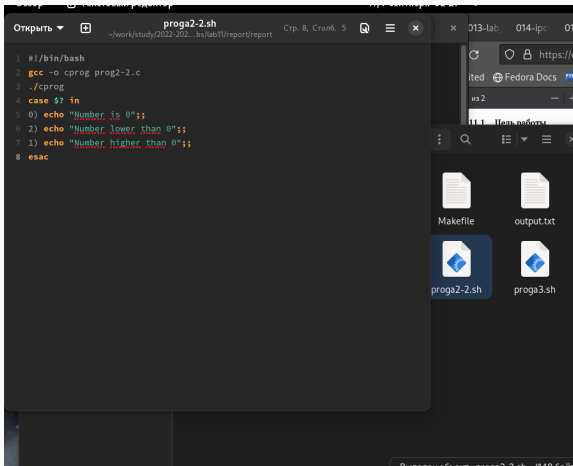


Рис. 3: Выполнение программы

4. Программа на Си, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.



```
Открыть ▾ + prog2-2.sh ~/work/study/2022-2023...bs/lab/1/report/report Сrp. 8, Cronb. 5 x x 013-lab 014-lpc 012
```

```
1 #!/bin/bash
2 gcc -o cprog prog2-2.c
3 ./cprog
4 case $? in
5 0) echo "Number is 0";;
6 2) echo "Number lower than 0";;
7 1) echo "Number higher than 0";;
8 esac
```

Makefile output.txt prog2-2.sh prog3.sh

Рис. 4: Программа

# Процесс выполнения

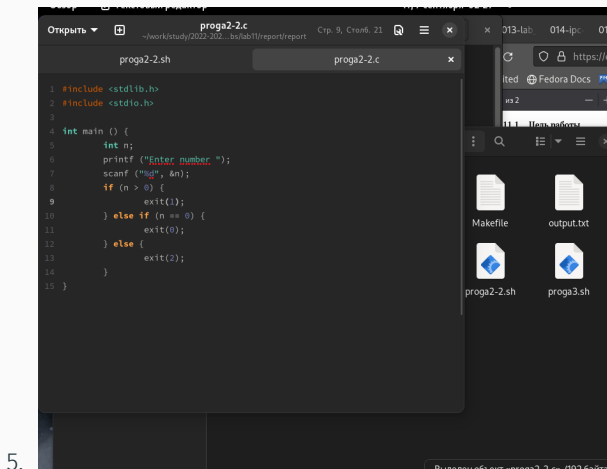
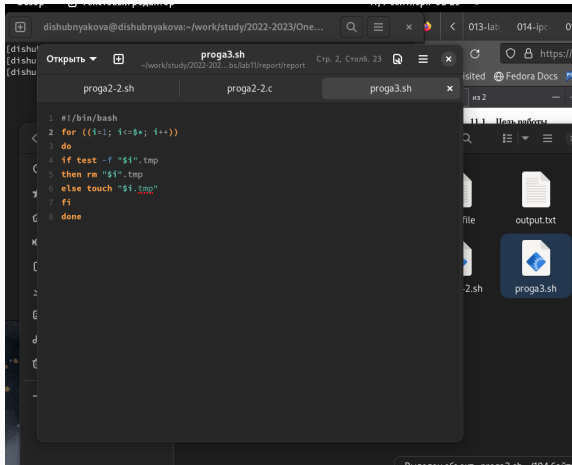


Рис. 5: Программа

6. Командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до [?] (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.).



```
#!/bin/bash
1 for ((i=1; i<=10; i++))
2 do
3     if test -f "$i".tmp
4     then rm "$i".tmp
5     else touch "$i".tmp
6     fi
7 done
```

The screenshot shows a terminal window with a shell script named `proga3.sh`. The script is designed to create a series of files numbered 1 to 10. It uses a `for` loop to iterate through the numbers. For each number `i`, it checks if a file named `$i.tmp` already exists using the `test` command. If the file exists, it is removed using the `rm` command. If the file does not exist, it is created using the `touch` command. The terminal output shows the execution of the script, with files 1 through 10 being created. In the background, a file explorer window is visible, showing the directory containing the files `1.tmp` through `10.tmp`.

# Процесс выполнения

7.

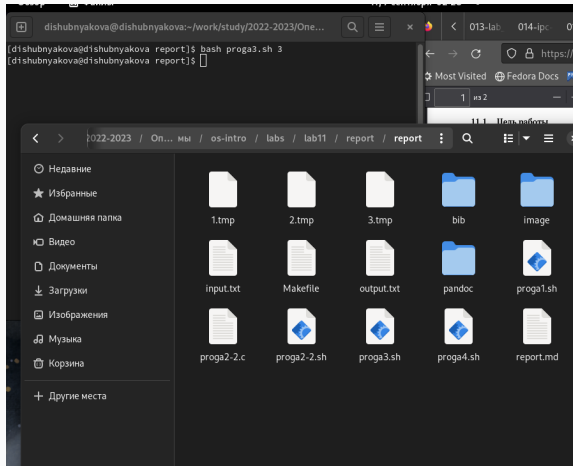


Рис. 7: Выполнение программы

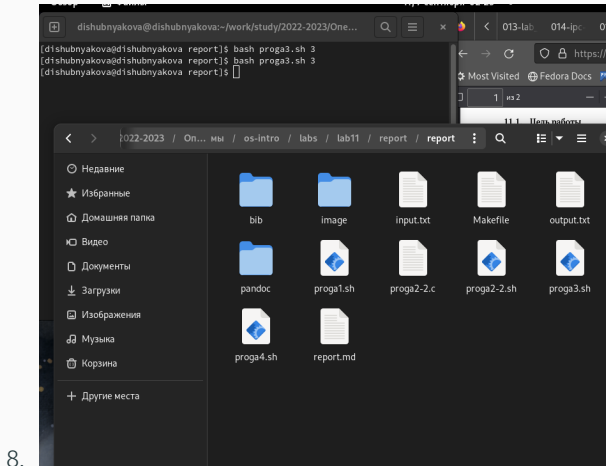


Рис. 8: Выполнение программы



9. Командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории.

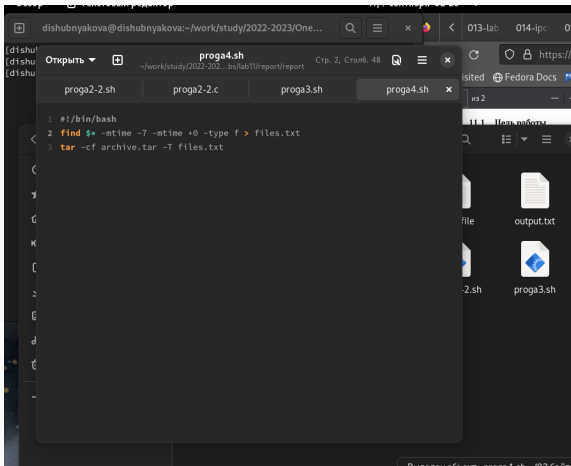


Рис. 9: Программа

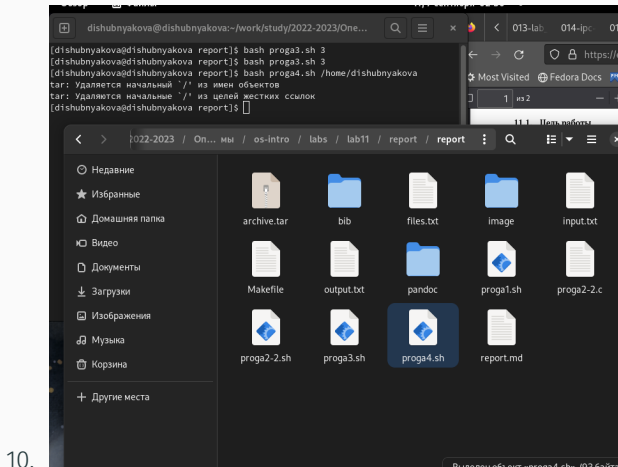


Рис. 10: Выполнение программы

# Процесс выполнения

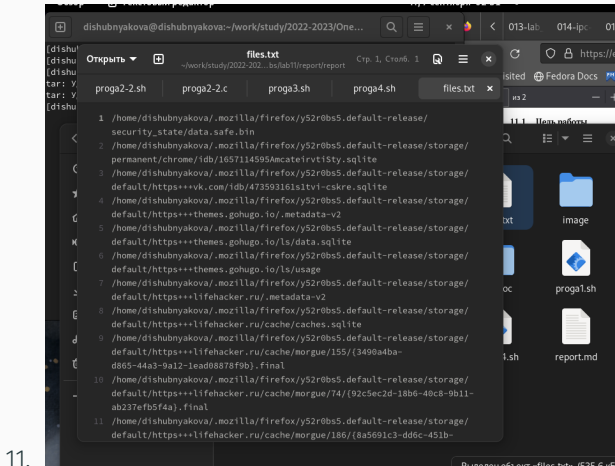


Рис. 11: Выполнение программы

Научились писать более сложные командные файлы с логическими управляющими и циклами.

1. Dash P. Getting started with oracle vm virtualbox. Packt Publishing Ltd, 2013. 86 p.
2. Colvin H. Virtualbox: An ultimate guide book on virtualization with virtualbox. CreateSpace Independent Publishing Platform, 2015. 70 p.
3. van Vugt S. Red hat rhcsa/rhce 7 cert guide : Red hat enterprise linux 7 (ex200 and ex300). Pearson IT Certification, 2016. 1008 p.
4. Робачевский А., Немнюгин С., Стесик О. Операционная система unix. 2-е изд. Санкт-Петербург: БХВ-Петербург, 2010. 656 p.
5. Немет Э. et al. Unix и Linux: руководство системного администратора. 4-е изд. Вильямс, 2014. 1312 p.
6. Колисниченко Д.Н. Самоучитель системного администратора Linux. СПб.: БХВ-Петербург, 2011. 544 p.
7. Robbins A. Bash pocket reference. O'Reilly Media, 2016. 156 p.

Спасибо за внимание!