

# **Отчет по лабораторной работе 11**

**Программирование в командном процессоре ОС UNIX. Командные  
файлы**

Шалыгин Георгий Эдуардович, НФИбд-02-20

# Содержание

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Цель работы</b>                     | <b>4</b>  |
| <b>2</b> | <b>Техническое обеспечение:</b>        | <b>5</b>  |
| <b>3</b> | <b>Условные обозначения и термины:</b> | <b>6</b>  |
| <b>4</b> | <b>Теоретическое введение:</b>         | <b>7</b>  |
| <b>5</b> | <b>Выполнение лабораторной работы</b>  | <b>10</b> |
| <b>6</b> | <b>Выводы</b>                          | <b>14</b> |
| <b>7</b> | <b>Библиография</b>                    | <b>15</b> |

# List of Figures

|     |  |    |
|-----|--|----|
| 5.1 | Создание нового файла . . . . .                      | 10 |
| 5.2 | Текст 1 скрипта . . . . .                            | 10 |
| 5.3 | Результат создание backup-архива . . . . .           | 11 |
| 5.4 | Текст 2 скрипта . . . . .                            | 11 |
| 5.5 | Вывод аргументов в консоль, результат . . . . .      | 11 |
| 5.6 | Текст 3 скрипта . . . . .                            | 12 |
| 5.7 | Описание содержимого директории, результат . . . . . | 12 |
| 5.8 | Текст 4 скрипта . . . . .                            | 13 |
| 5.9 | Количество файлов опр. расширения . . . . .          | 13 |

# 1 Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

Объект исследования: система UNIX.

Предмет исследования: программирование в UNIX.

## **2 Техническое обеспечение:**

- Характеристики техники: AMD Ryzen 5 3500U 2.1 GHz, 8 GB оперативной памяти, 50 GB свободного места на жёстком диске;
- ОС Windows 10 Home
- Git 2.31.1
- Google Chrome 91.0.4472.19
- VirtualBox 2.0
- CentOS 7

### 3 Условные обозначения и термины:

**Текстовым редактором**(text editor) называют программу, которая предназначена для редактирования (составления и изменения) файлов, содержащих только текст. [1]

**Командный язык** - это язык, на котором пользователь взаимодействует с системой в интерактивном режиме.

**Командный интерпретатор**, интерпретатор командной строки - компьютерная программа, часть операционной системы, обеспечивающая базовые возможности управления компьютером посредством интерактивного ввода команд через интерфейс командной строки или последовательного исполнения пакетных командных файлов.[3]

Подробнее в [2] и [3].

## 4 Теоретическое введение:

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 оболочкой Борна, использующая 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).

POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.

Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна. Рассмотрим основные элементы программирования в оболочке bash. В других оболочках большинство команд будет совпадать с описанными ниже. [3]

## **Переменные в языке программирования bash**

Командный процессор bash обеспечивает возможность использования переменных типа строка символов. Имена переменных могут быть выбраны пользователем. Пользователь имеет возможность присвоить переменной значение некоторой строки символов. Например, команда `mark=/usr/andy/bin` присваивает значение строки символов `/usr/andy/bin` переменной `mark` типа строка символов.

Использование значения, присвоенного некоторой переменной, называется подстановкой. Для того чтобы имя переменной не сливалось с символами, которые могут следовать за ним в командной строке, при подстановке в общем случае используется следующая форма записи: `${имя переменной}`. [3]

### **Использование арифметических вычислений. Операторы `let` и `read`**

Оболочка bash поддерживает встроенные арифметические функции. Команда `let` является показателем того, что последующие аргументы представляют собой выражение, подлежащее вычислению. Простейшее выражение — это единичный терм (`term`), обычно целочисленный.

Команда `let` берет два операнда и присваивает их переменной. Положительным моментом команды `let` можно считать то, что для идентификации переменной ей не нужен знак доллара; вы можете писать команды типа `let sum=x+7`, и `let` будет искать переменную `x` и добавлять к ней 7.

Команда `read` позволяет читать значения переменных со стандартного ввода:

```
echo "Please enter Month and Day of Birth ?"
```

```
read mon day trash [3]
```

### **Командные файлы и функции**

Последовательность команд может быть помещена в текстовый файл. Такой файл называется командным. Далее этот файл можно выполнить по команде: `bash командный_файл [аргументы]`

Чтобы не вводить каждый раз последовательности символов bash, необходимо изменить код защиты этого командного файла, обеспечив доступ к этому файлу по выполнению. Это может быть сделано с помощью команды `chmod +x имя_файла`



### **Передача параметров в командные файлы и специальные переменные**

Пусть к командному файлу `where` имеется доступ по выполнению и этот командный файл содержит следующий конвейер: `who | grep $1`.

В ходе интерпретации файла командным процессором вместо комбинации символов `$1` осуществляется подстановка значения первого и единственного параметра `andy`.

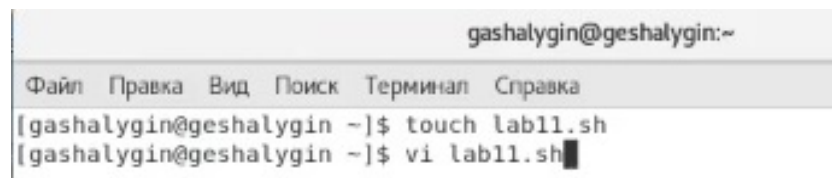
Команда `shift` позволяет удалять первый параметр и сдвигает все остальные на места предыдущих. При использовании в командном файле комбинации символов `$#` вместо неё будет осуществлена подстановка числа параметров, указанных в командной строке при вызове данного командного файла на выполнение.

### **Оператор цикла `for`**

В обобщённой форме оператор цикла `for` выглядит следующим образом: `for имя [in список-значений] do список-команд done`. [3]

## 5 Выполнение лабораторной работы

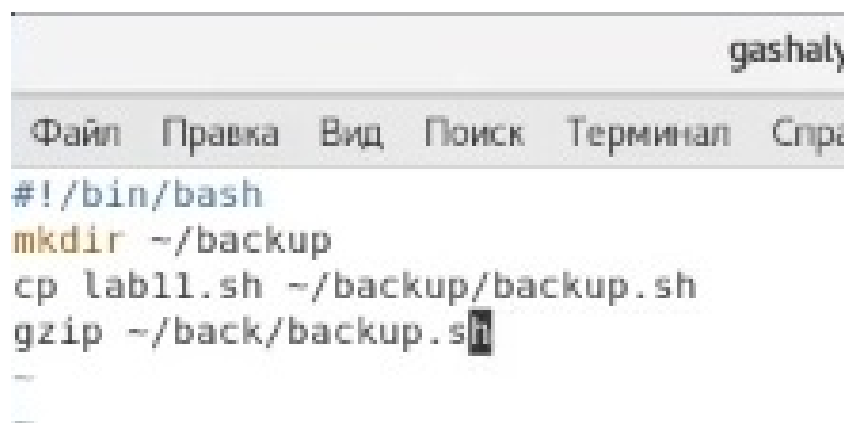
1. Создадим новый файл lab11.sh. (рис. 5.1)



```
gashalygin@geshalygin:~  
Файл Правка Вид Поиск Терминал Справка  
[gashalygin@geshalygin ~]$ touch lab11.sh  
[gashalygin@geshalygin ~]$ vi lab11.sh
```

Figure 5.1: Создание нового файла

2. Напишем скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге (рис. 5.2).



```
gashalygin@geshalygin:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
mkdir ~/backup  
cp lab11.sh ~/backup/backup.sh  
gzip ~/back/backup.sh
```

Figure 5.2: Текст 1 скрипта

3. Результат работы скрипта, видим архив в нужной директории. (рис. 5.3).

```
gashalygin@geshalygin:~/backup
Файл Правка Вид Поиск Терминал Справка
[gashalygin@geshalygin ~]$ chmod +x lab11.sh
[gashalygin@geshalygin ~]$ ./lab11.sh
gzip: /home/gashalygin/back/backup.sh: No such file or directory
[gashalygin@geshalygin ~]$ vi lab11.sh
[gashalygin@geshalygin ~]$ ./lab11.sh
mkdir: невозможно создать каталог «/home/gashalygin/backup»: Файл существует
[gashalygin@geshalygin ~]$ rmdir backup -r
rmdir: неверный ключ - «r»
По команде «rmdir --help» можно получить дополнительную информацию.
[gashalygin@geshalygin ~]$ rmdir backup
rmdir: не удалось удалить «backup»: Каталог не пуст
[gashalygin@geshalygin ~]$ rm backup -r
[gashalygin@geshalygin ~]$ ./lab11.sh
[gashalygin@geshalygin ~]$ ls
abcl1      feathers  my_os      reports    Загрузки
australia  file      my_os,     ski.places Изображения
australia, file.txt  pandoc-crossref test.txt  Музыка
backup     lab11.sh  pandoc-crossref.1 works      Общедоступные
conf.txt   may       play       Видео     Рабочий стол
equiplist  monthly  r          Документы Шаблоны
[gashalygin@geshalygin ~]$ cd backup/
[gashalygin@geshalygin backup]$ ls
backup.sh.gz
[gashalygin@geshalygin backup]$
```

Figure 5.3: Результат создание backup-архива

4. Напишем командный файл, который будет последовательно распечатывать значения всех переданных аргументов. (рис. 5.4)

```
Файл Правка Вид Поиск
#!/bin/bash
head -1
```


Figure 5.4: Текст 2 скрипта

5. Результат работы скрипта. (рис. 5.5)

```
[gashalygin@geshalygin backup]$ vi lab11_2.sh
[gashalygin@geshalygin backup]$ ./lab11_2.sh
1 2 3 4 5 6 7
1 2 3 4 5 6 7
[gashalygin@geshalygin backup]$
```

Figure 5.5: Вывод аргументов в консоль, результат

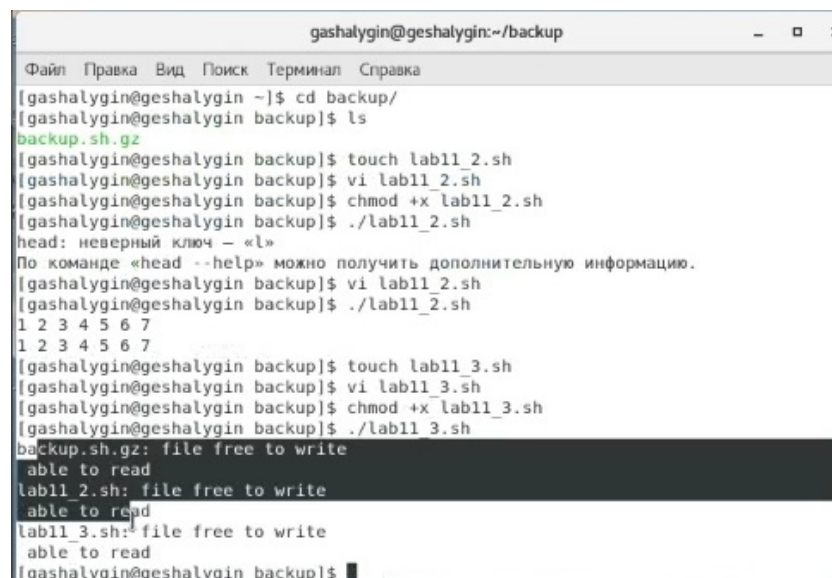
6. Напишем командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (рис. 5.6)



```
gashalygin@geshalygin:~/backup
Файл Правка Вид Поиск Терминал Справка
#!/bin/bash
for smth in *
do if test -d $smth
then echo $smth: "directory"
else echo -n $smth: "file"
if test -w $smth
then echo " free to write"
if test -r $smth
then echo " able to read"
else echo " not able for reading"
fi
fi
done
```

Figure 5.6: Текст 3 скрипта

7. Результат работы скрипта. (рис. 5.7)

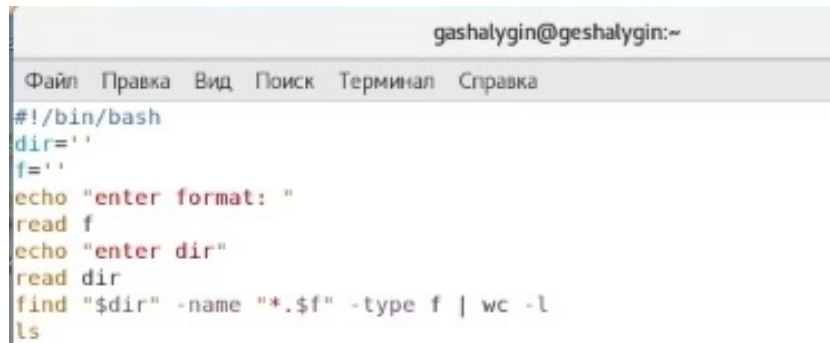


```
gashalygin@geshalygin:~/backup
Файл Правка Вид Поиск Терминал Справка
[gashalygin@geshalygin ~]$ cd backup/
[gashalygin@geshalygin backup]$ ls
backup.sh.gz
[gashalygin@geshalygin backup]$ touch lab11_2.sh
[gashalygin@geshalygin backup]$ vi lab11_2.sh
[gashalygin@geshalygin backup]$ chmod +x lab11_2.sh
[gashalygin@geshalygin backup]$ ./lab11_2.sh
head: неверный ключ - «l»
По команде «head --help» можно получить дополнительную информацию.
[gashalygin@geshalygin backup]$ vi lab11_2.sh
[gashalygin@geshalygin backup]$ ./lab11_2.sh
1 2 3 4 5 6 7
1 2 3 4 5 6 7
[gashalygin@geshalygin backup]$ touch lab11_3.sh
[gashalygin@geshalygin backup]$ vi lab11_3.sh
[gashalygin@geshalygin backup]$ chmod +x lab11_3.sh
[gashalygin@geshalygin backup]$ ./lab11_3.sh
backup.sh.gz: file free to write
able to read
lab11_2.sh: file free to write
able to read
lab11_3.sh: file free to write
able to read
[gashalygin@geshalygin backup]$
```

Figure 5.7: Описание содержимого директории, результат

8. Напишем командный файл, который получает в качестве аргумента команд-

ной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки. (рис. 5.8)

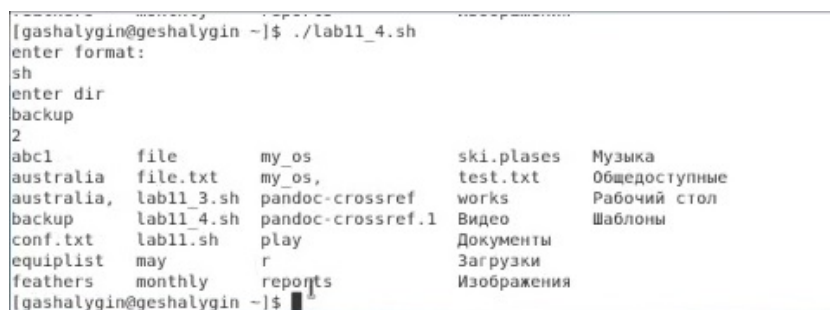


```

gashalygin@geshalygin:~
Файл Правка Вид Поиск Терминал Справка
#!/bin/bash
dir=''
f=''
echo "enter format: "
read f
echo "enter dir"
read dir
find "$dir" -name ".*$f" -type f | wc -l
ls
  
```

Figure 5.8: Текст 4 скрипта

9. Результат работы скрипта, видим два файла - как и должно быть. (рис. 5.9)



```

[gashalygin@geshalygin ~]$ ./lab11_4.sh
enter format:
sh
enter dir
backup
2
abcl      file      my_os      ski.plases Музыка
australia file.txt  my_os,     test.txt   Общедоступные
australia, lab11_3.sh pandoc-crossref works      Рабочий стол
backup     lab11_4.sh pandoc-crossref.1 Видео      Шаблоны
conf.txt   lab11.sh   play       Документы
equiplist  may       r           Загрузки
feathers   monthly   reports    Изображения
[gashalygin@geshalygin ~]$
  
```

Figure 5.9: Количество файлов опр. расширения

## **6 Выводы**

В процессе работы над лабораторной работы были изучены основы программирования в оболочке ОС UNIX/Linux, получены навыки написания небольших командных файлов.

## 7 Библиография

1. <https://docs.altlinux.org/ru-RU/archive/2.3/html-single/junior/alt-docs-extras-linuxnovice/ch02s10.html>
2. <http://bourabai.kz/os/shells.htm>
3. Д.С. Кулябов, А.В. Королькова / Администрирование локальных систем. Лабораторные работы. — М.: Российский университет дружбы народов, 2017. — 119 с.