

Лабораторная работа №10

Дисциплина: операционные системы

Пронякова Ольга Максимовна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	13
	Список литературы	14

Список иллюстраций

4.1	Выполнение команд	8
4.2	Выполнение команд	9
4.3	Результат выполнения команд	9
4.4	Результат выполнения команд	9
4.5	Код исполнения	10
4.6	Код исполнения	10
4.7	Результат выполнения команд	11
4.8	Код исполнения	11
4.9	Результат выполнения команд	11
4.10	Результат выполнения команд	12

Список таблиц

1 Цель работы

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

2 Задание

1. Написать 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. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.
3. Написать 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.
4. Написать командный файл, который получает в качестве аргумента командной 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.

3 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 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). 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.

4 Выполнение лабораторной работы

Способ использования команд архивации узнаю, изучив справку. Пишу скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в моем домашнем каталоге. При этом файл архивируется фрхиватором tar(рис. 4.1) (рис. 4.2).

```
olga@ompronyakova:~$ man tar
olga@ompronyakova:~$ mkdir backup
olga@ompronyakova:~$ ls
abc1          ll          ski.plases
assets        may         snap
australia     monthly    text.txt
backup        my_os      text.txt~
bin           newdir     work
blog          Olga1325.github.io  Видео
c.cpp         pandoc-2.19.2  Документы
c.cpp.save    pandoc-2.19.2-linux-amd64.tar.gz  Загрузки
conf.txt      pandoc-crossref  Изображения
feathers      pandoc-crossref.1  Музыка
file.txt      pandoc-crossref-Linux.tar.xz  Общедоступные
lab07.sh     play       'Рабочий стол'
lab07.sh~    reports    Шаблоны
olga@ompronyakova:~$ vi lab101
```

Рис. 4.1: Выполнение команд


```
tar -cf lab101.tar lab101
mv lab101.tar ~/backup
```

Рис. 4.2: Выполнение команд

Результат выполнения команд(рис. 4.3).

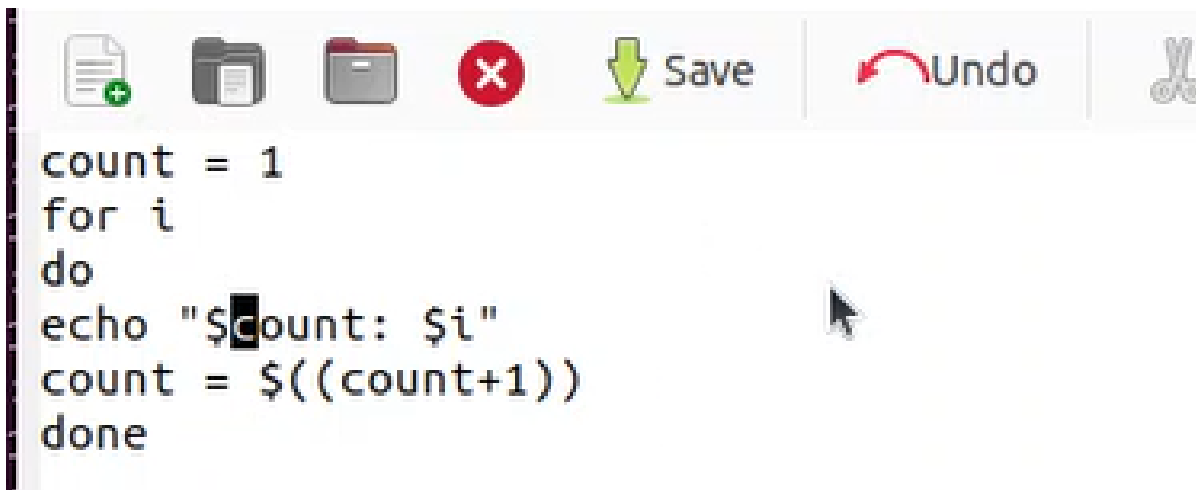
```
olga@ompronyakova:~$ chmod +x lab101
olga@ompronyakova:~$ ./lab101
olga@ompronyakova:~$ ls backup
lab101.tar
olga@ompronyakova:~$
```

Рис. 4.3: Результат выполнения команд

Пишу пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять(рис. 4.4) (рис. 4.5).

```
olga@ompronyakova:~$ emacs lab102
olga@ompronyakova:~$ chmod +x lab102
olga@ompronyakova:~$ ./lab102 21 34 54 23 3 5 2 45 36 43 9
1: 21
2: 34
3: 54
4: 23
5: 3
6: 5
7: 2
8: 45
9: 36
10: 43
11: 9
olga@ompronyakova:~$ emacs
```

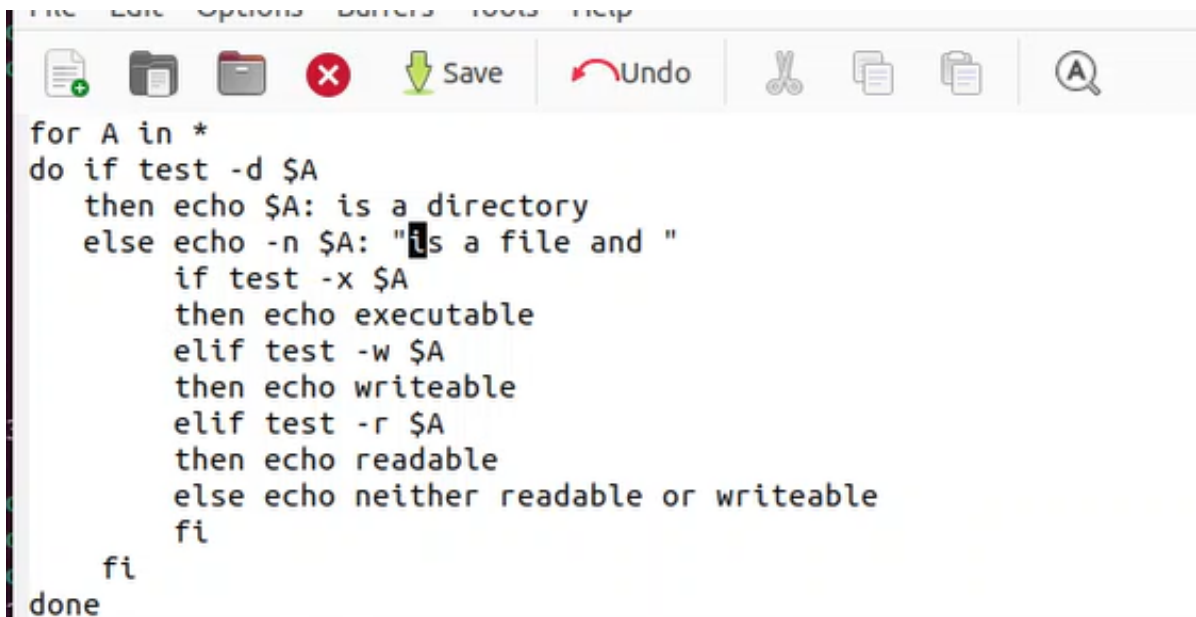
Рис. 4.4: Результат выполнения команд

A screenshot of a code editor window. The toolbar at the top includes icons for a new file, a folder, a save icon, a close icon, a green arrow pointing down labeled 'Save', a red curved arrow labeled 'Undo', and a scissors icon. The code in the editor is a shell script:

```
count = 1
for i
do
echo "$count: $i"
count = $((count+1))
done
```

Рис. 4.5: Код исполнения

Пишу 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. 4.6) (рис. 4.7).

A screenshot of a code editor window. The toolbar at the top includes icons for a new file, a folder, a save icon, a close icon, a green arrow pointing down labeled 'Save', a red curved arrow labeled 'Undo', a scissors icon, a copy icon, a paste icon, and a magnifying glass icon. The code in the editor is a shell script:

```
for A in *
do if test -d $A
then echo $A: is a directory
else echo -n $A: "is a file and "
if test -x $A
then echo executable
elif test -w $A
then echo writeable
elif test -r $A
then echo readable
else echo neither readable or writeable
fi
fi
done
```

Рис. 4.6: Код исполнения

```
olga@ompronyakova:~$ emacs lab103
olga@ompronyakova:~$ chmod +x lab103
olga@ompronyakova:~$ ./lab103
abc1: is a file and writeable
assets: is a file and writeable
australia: is a directory
backup: is a directory
bin: is a directory
blog: is a directory
c.cpp: is a file and writeable
c.cpp.save: is a file and writeable
conf.txt: is a file and writeable
feathers: is a file and writeable
file.txt: is a file and writeable
lab07.sh: is a file and writeable
lab07.sh~: is a file and writeable
lab101: is a file and executable
```

Рис. 4.7: Результат выполнения команд

Пишу командный файл, который получает в качестве аргумента командной 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. 4.8) (рис. 4.9) (рис. 4.10).

```
echo Input a directiry
read dir
echo Input a file format
read format
find $dir -maxdepth 1 -name "$format" -type f| wc -l
```

Рис. 4.8: Код исполнения

```
olga@ompronyakova:~$ emacs lab104
olga@ompronyakova:~$ chmod +x lab104
```

Рис. 4.9: Результат выполнения команд

```
olga@ompronyakova:~$ ./lab104
Input a directory
/home/olga
Input a file format
.txt
3
olga@ompronyakova:~$
```

Рис. 4.10: Результат выполнения команд

5 Выводы

Изучила основы программирования в оболочке ОС UNIX/Linux. Научилась писать небольшие командные файлы.

Список литературы

1. Лабораторная работа №10