

Лабораторная работа №10

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

Салькова Кристина Михайловна

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	создание файла1	8
4.2	скрипт1	9
4.3	проверка	9
4.4	создание файла2	9
4.5	скрипт2	10
4.6	проверка	10
4.7	создание файла3	10
4.8	скрипт3	11
4.9	проерка	11
4.10	создание файла4	11
4.11	скрипт4	12
4.12	проверка	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

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.



```
ksa@ksa:~$ man tar
ksa@ksa:~$ touch script.sh
bash: touch: команда не найдена
ksa@ksa:~$ touch script.sh
ksa@ksa:~$ chmod +x script.sh
ksa@ksa:~$
```

Рис. 4.1: создание файла1



Рис. 4.2: скрипт1



Рис. 4.3: проверка

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.

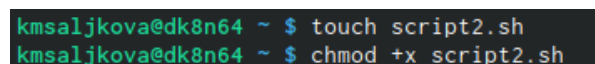


Рис. 4.4: создание файла2



Рис. 4.5: скрипт2



Рис. 4.6: проверка

3. Написать командный файл — аналог команды ls (без использования самой этой ко- манды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.

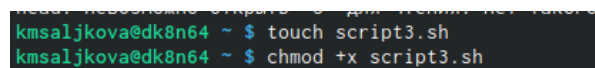


Рис. 4.7: создание файла3

```

1 #!/bin/bash
2 for A in *
3 do if test -d $A
4     then echo $A: is a directory
5     else echo -n $A: is a file and
6         if test -w $A
7         then echo writeable
8         elif test -r $A
9         then echo readable
10        else echo ntither readable nor writeable
11        fi
12    fi
13 done

```

Рис. 4.8: скрипт3

```

kmsaljkova@dk8n64 ~ $ ./script3.sh
2: is a file andwriteable
2.cdp: is a file andwriteable
2.o: is a file andwriteable
a: is a directory
a.txt: is a file andwriteable
australia: is a directory
backup: is a directory
bin: is a directory
conf.txt: is a file andwriteable
feathers: is a file andwriteable
file.old: is a file andwriteable
file.txt: is a file andwriteable
GNUstep: is a directory
may: is a file andwriteable
my_os: is a file and./script3.sh: строка 8: test: команда не найдена
ntither readable nor writeable
nc: is a directory
public: is a directory
public.html: is a directory
script2.sh: is a file andwriteable
script2.sh: is a file andwriteable
script3.sh: is a file andwriteable
script.sh: is a file andwriteable
t: is a file andwriteable
t.cdp: is a file andwriteable
t.cdp.save: is a file andwriteable
text.txt: is a file andwriteable
tmp: is a directory
t.o: is a file andwriteable
work: is a directory
Видео: is a directory
Документы: is a directory
Загрузки: is a directory
Изображения: is a directory
./script3.sh: строка 3: test: слишком много аргументов
./script3.sh: строка 6: test: слишком много аргументов
./script3.sh: строка 8: test: команда не найдена
ntither readable nor writeable
Музыка: is a directory
Общедоступные: is a directory
./script3.sh: строка 3: test: Рабочий: ожидается бинарный оператор
Рабочий стон: is a file and./script3.sh: строка 6: test: Рабочий: ожидается бинарный оператор
./script3.sh: строка 8: test: команда не найдена
ntither readable nor writeable
Шаблоны: is a directory

```

Рис. 4.9: проерка

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

```

kmsaljkova@dk8n64 ~ $ touch script4.sh
kmsaljkova@dk8n64 ~ $ chmod +x script4.sh

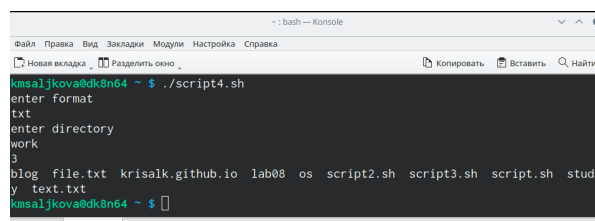
```

Рис. 4.10: создание файла4

A screenshot of a GNU Emacs editor window titled 'script4.sh - GNU Emacs at dk8n64'. The window shows a shell script with the following content:

```
#!/bin/bash
direct=''
form=''
echo 'enter format'
read form
echo 'enter directory'
read direct
find "$direct" -name "*, $form" -type f | wc -l
ls $direct
```

Рис. 4.11: скрипт4

A screenshot of a terminal window titled '~: bash — Konsole'. The terminal shows the execution of the script4.sh script. The user enters 'format' for the format and 'work' for the directory. The script then lists files in the 'work' directory, showing 'blog file.txt krisalk.github.io lab08 os script2.sh script3.sh script.sh study text.txt'.

```
kmsaljkova@dk8n64 ~ $ ./script4.sh
enter format
format
enter directory
work
3
blog file.txt krisalk.github.io lab08 os script2.sh script3.sh script.sh stud
y text.txt
kmsaljkova@dk8n64 ~ $
```

Рис. 4.12: проверка

5 Выводы

Мы изучили основы программирования в оболочке ОС UNIX/Linux и научились писать небольшие командные файлы

Список литературы