

Лабораторная работа №10.

Программирование в командном процессоре ОС UNIX. Командные файлы.

Александр Андреевич Шуплецов

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение работы	7
4	Выводы	12
	Список литературы	13

Список иллюстраций

3.1	изучение справки tar	7
3.2	текст скрипта, делающий резервную копию	8
3.3	проверка скрипта, делающего резервную копию	8
3.4	текст скрипта, обрабатывающего любое произвольное число аргументов	8
3.5	проверка скрипта, обрабатывающего любое произвольное число аргументов	9
3.6	текст аналога команды ls	9
3.7	проверка аналога команды ls	10
3.8	текст командного файла, вычисляющего кол-во файлов в директории	10
3.9	проверка командного файла, вычисляющего кол-во файлов в директории	11

Список таблиц

1 Цель работы

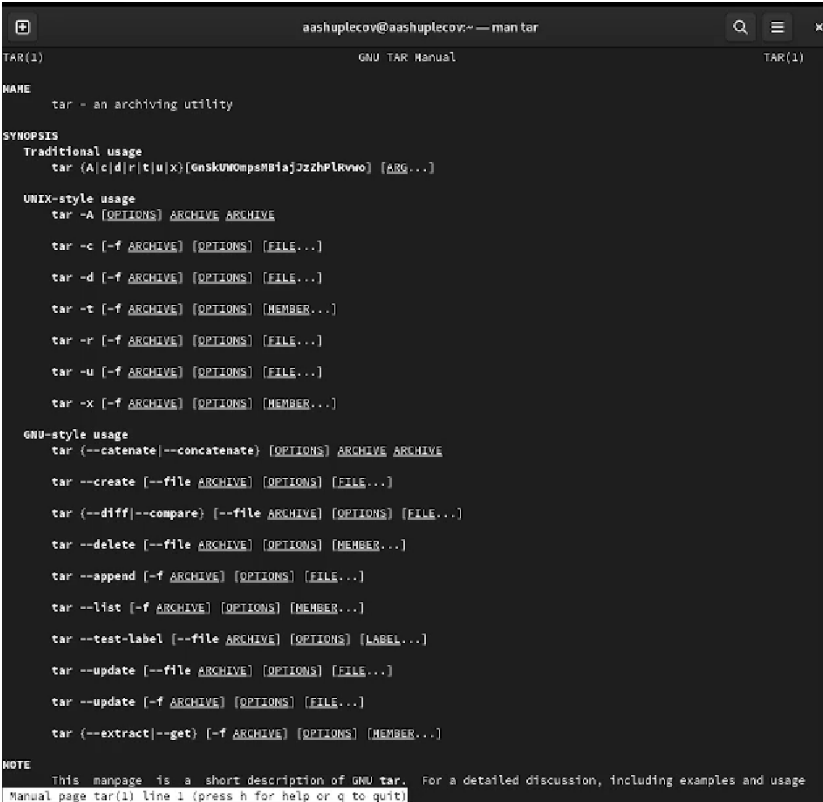
Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

2 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.

3 Выполнение работы

1. Изучим справку tar.

A screenshot of a terminal window titled "aashuplecov@aashuplecov:~ — man tar". The window displays the GNU tar manual page. The content is as follows:

```
TAR(1)                                GNU TAR Manual                                TAR(1)

NAME
  tar - an archiving utility

SYNOPSIS
  Traditional usage
  tar (A|c|d|r|t|u|x|G|nskUwompamBiaJzzhPlkwmo) [ARG...]

  UNIX-style usage
  tar -A [OPTIONS] ARCHIVE ARCHIVE

  tar -c [-f ARCHIVE] [OPTIONS] [FILE...]
  tar -d [-f ARCHIVE] [OPTIONS] [FILE...]
  tar -t [-f ARCHIVE] [OPTIONS] [MEMBER...]
  tar -r [-f ARCHIVE] [OPTIONS] [FILE...]
  tar -u [-f ARCHIVE] [OPTIONS] [FILE...]
  tar -x [-f ARCHIVE] [OPTIONS] [MEMBER...]

  GNU-style usage
  tar (--catenate|--concatenate) [OPTIONS] ARCHIVE ARCHIVE

  tar --create [--file ARCHIVE] [OPTIONS] [FILE...]
  tar (--diff|--compare) [--file ARCHIVE] [OPTIONS] [FILE...]
  tar --delete [--file ARCHIVE] [OPTIONS] [MEMBER...]
  tar --append [-f ARCHIVE] [OPTIONS] [FILE...]
  tar --list [-f ARCHIVE] [OPTIONS] [MEMBER...]
  tar --test-label [--file ARCHIVE] [OPTIONS] [LABEL...]
  tar --update [--file ARCHIVE] [OPTIONS] [FILE...]
  tar --update [-f ARCHIVE] [OPTIONS] [FILE...]
  tar (--extract|--get) [-f ARCHIVE] [OPTIONS] [MEMBER...]

NOTE
  This manpage is a short description of GNU tar. For a detailed discussion, including examples and usage
  Manual page tar(1) line 1 (press h for help or q to quit)
```

Рис. 3.1: изучение справки tar

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 нашем домашнем каталоге.

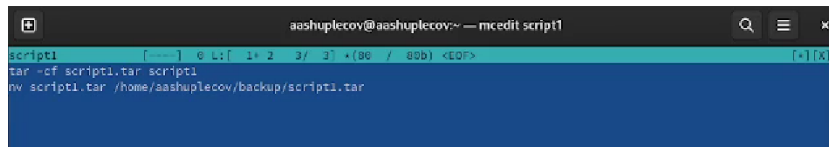


Рис. 3.2: текст скрипта, делающий резервную копию

3. Убедимся, что скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в нашем домашнем каталоге, работает.

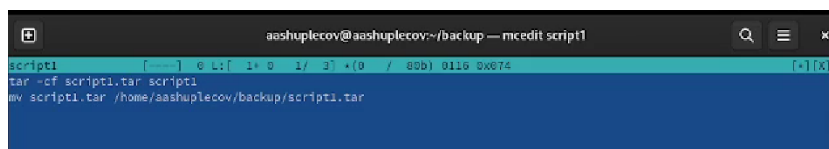


Рис. 3.3: проверка скрипта, делающего резервную копию

4. Напишем пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять.

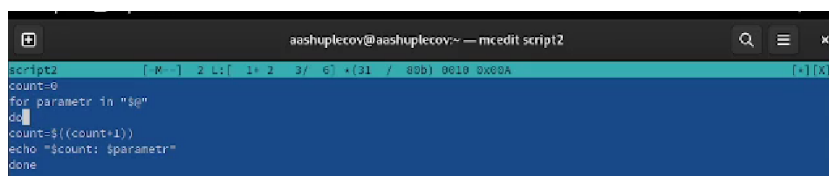


Рис. 3.4: текст скрипта, обрабатывающего любое произвольное число аргументов

5. Убедимся, что скрипт, обрабатывающий любое произвольное число аргументов, работает.


```
[aashuplecov@aashuplecov ~]$ mcedit script2

[aashuplecov@aashuplecov ~]$ chmod +x script2
[aashuplecov@aashuplecov ~]$ ./script2 1 2 3 4 5 6 7 8 9 10 11 12
1: 1
2: 2
3: 3
4: 4
5: 5
6: 6
7: 7
8: 8
9: 9
10: 10
11: 11
12: 12
```

Рис. 3.5: проверка скрипта, обрабатывающего любое произвольное число аргументов

6. Напишем командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`).

```
aashuplecov@aashuplecov:~ — mcedit script3
script3 [-----] 4 L: [ 1+11 12/ 12 ] *(253 / 253b) <EOF> [~][X]
for A in *
do if test -d $A
then echo $A: is a directory
else echo -n $A: is a file and
if test -w $A
then echo writeable
elif test -r $A
then echo readable
else echo neither readable nor writeable
fi
fi
done
```

Рис. 3.6: текст аналога команды `ls`

7. Убедимся, что командный файл - аналог команды `ls`, работает.

```

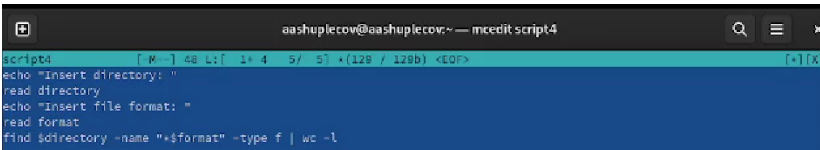
[aashuplecov@aashuplecov ~]$ mcedit scripts3

[aashuplecov@aashuplecov ~]$ chmod +x scripts3
[aashuplecov@aashuplecov ~]$ ./scripts3
abc1: is a file andwriteable
australia: is a directory
backup: is a directory
bin: is a directory
conf.txt: is a file andwriteable
ed: is a file andwriteable
ed.pub: is a file andwriteable
feathers: is a file andwriteable
file.txt: is a file andwriteable
#lab0701#: is a file andwriteable
lab07.sh: is a file andwriteable
lab07.sh-: is a file andwriteable
may: is a file andwriteable
monthly: is a directory
my_os: is a file andreadable
otchet: is a directory
play: is a directory
presentation: is a directory
reports: is a directory
rsa: is a file andwriteable
rsa.pub: is a file andwriteable
script1: is a file andwriteable
script2: is a file andwriteable
scripts3: is a file andwriteable
ski.places: is a directory
work: is a directory
Видео: is a directory
Документы: is a directory
Загрузки: is a directory
Изображения: is a directory
Музыка: is a directory
Общедоступные: is a directory
./scripts3: строка 2: test: Рабочий: ожидается бинарный оператор
Рабочий стол: is a file and./scripts3: строка 5: test: Рабочий: ожидается бинарный оператор
./scripts3: строка 7: test: Рабочий: ожидается бинарный оператор
neither readable nor writeable
шаблоны: is a directory

```

Рис. 3.7: проверка аналога команды ls

8. Напишем 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.



```

script4 [P] 48 L[ 1* 4 5/ 5] *(128 / 128b) <EOF>
echo "insert directory: "
read directory
echo "insert file format: "
read format
find $directory -name "$format" -type f | wc -l

```

Рис. 3.8: текст командного файла, вычисляющего кол-во файлов в директории

9. Убедимся, что 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, работает

```
[aashuplecov@aashuplecov ~]$ mcedit script4  
  
[aashuplecov@aashuplecov ~]$ chmod +x script4  
[aashuplecov@aashuplecov ~]$ ./script4  
Insert directory:  
/home/aashuplecov  
Insert file format:  
.md  
78
```

Рис. 3.9: проверка командного файла, вычисляющего кол-во файлов в директории

4 Выводы

Я изучил основы программирования в оболочке ОС UNIX/Linux, научился писать небольшие командные файлы.

Список литературы

Кулябов Д.С. “Материалы к лабораторным работам”