

Лабораторная работа №7

Операционные системы

Башиянц Александра Кареновна

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Выводы	16

Список иллюстраций

3.1	Копирование файла в текущем каталоге	6
3.2	Копирование нескольких файлов в каталог	6
3.3	Копирование файлов в произвольном каталоге	6
3.4	Копирование каталогов в текущем каталоге	7
3.5	Копирование каталогов в произвольном каталоге	7
3.6	Переименование файлов в текущем каталоге	7
3.7	Перемещение файлов в другой каталог	7
3.8	Переименование каталогов в текущем каталоге	8
3.9	Перемещение каталога в другой каталог	8
3.10	Переименование каталога, не являющегося текущим	8
3.11	Право выполнения для владельца	8
3.12	Лишение права	9
3.13	Запрет на чтение	9
3.14	Право записи для группы	9
3.15	equipment	9
3.16	mv equipment	10
3.17	mv equipment	10
3.18	equipment	10
3.19	mkdir equipment	10
3.20	mv equipment	11
3.21	mv equipment	11
3.22	chmod для директорий	11
3.23	chmod для файлов	11
3.24	cp feathers	12
3.25	mv files.old	12
3.26	cp play	12
3.27	mv fun	13
3.28	chmod feathers	13
3.29	chmod play	13
3.30	man mount	14
3.31	man fsck	14
3.32	man mkfs	15
3.33	man kill	15

1 Цель работы

Цель данной работы — приобретение практических навыков работы с файловой системой.

2 Задание

В этой лабораторной работе необходимо изучить работу с файловой системой.

Необходимо научиться:

- Создавать файлы;
- Копировать файлы и директории;
- Перемещать файлы и директории;
- Настраивать права доступа к файлам и директориям.

3 Выполнение лабораторной работы

Скопируем файл ~/abc1 в файл april и в файл may (рис. 3.1).

```
akbashiyanc@akbashiyanc:~$ cd
akbashiyanc@akbashiyanc:~$ touch abc1
akbashiyanc@akbashiyanc:~$ cp abc1 april
akbashiyanc@akbashiyanc:~$ cp abc1 may
akbashiyanc@akbashiyanc:~$ ls
abc1      Downloads  package.json  sqlite
april     LICENSE   Pictures      work
bin       may       README.md     Видеос
Desktop  newdir    sqlite-autoconf-3250200  Докум
akbashiyanc@akbashiyanc:~$
```

Рис. 3.1: Копирование файла в текущем каталоге

Скопируем файлы april и may в каталог monthly (рис. 3.2).

```
akbashiyanc@akbashiyanc:~$ cp april may monthly
akbashiyanc@akbashiyanc:~$ ls
abc1      Desktop    may        package.json  sqlite
april     Downloads  monthly    Pictures      sqlite
bin       LICENSE    newdir     README.md     work
akbashiyanc@akbashiyanc:~$ ls monthly
april may
akbashiyanc@akbashiyanc:~$
```

Рис. 3.2: Копирование нескольких файлов в каталог

Скопируем файл monthly/may в файл с именем june (рис. 3.3).

```
akbashiyanc@akbashiyanc:~$ cp monthly/may monthly/june
akbashiyanc@akbashiyanc:~$ ls monthly
april june may
akbashiyanc@akbashiyanc:~$
```

Рис. 3.3: Копирование файлов в произвольном каталоге

Скопируем каталог monthly в каталог monthly.00 (рис. 3.4).

```
akbashiyanc@akbashiyanc:~$ mkdir monthly.00
akbashiyanc@akbashiyanc:~$ cp -r monthly monthly.00
akbashiyanc@akbashiyanc:~$ ls
abc1      Desktop  may      newdir    README
april     Downloads monthly  package.json sqlite
bin       LICENSE  monthly.00 Pictures  sqlite
akbashiyanc@akbashiyanc:~$ ls monthly.00
monthly
```

Рис. 3.4: Копирование каталогов в текущем каталоге

Скопируем каталог monthly.00 в каталог /tmp (рис. 3.5).

```
akbashiyanc@akbashiyanc:~$ cp -r monthly.00 /tmp
akbashiyanc@akbashiyanc:~$ ls /tmp
monthly.00
systemd-private-cf4a53cba8594c90b2c1a5573270d871-abrt
systemd-private-cf4a53cba8594c90b2c1a5573270d871-chrom
systemd-private-cf4a53cba8594c90b2c1a5573270d871-color
```

Рис. 3.5: Копирование каталогов в произвольном каталоге

Изменим название файла april на july в домашнем каталоге (рис. 3.6).

```
akbashiyanc@akbashiyanc:~$ cd
akbashiyanc@akbashiyanc:~$ mv april july
akbashiyanc@akbashiyanc:~$ ls
abc1      Downloads  may      newdir    README
bin       july      monthly  package.json sqlite
Desktop  LICENSE   monthly.00 Pictures  sqlite
akbashiyanc@akbashiyanc:~$
```

Рис. 3.6: Переименование файлов в текущем каталоге

Переместим файл july в каталог monthly.00 (рис. 3.7).

```
akbashiyanc@akbashiyanc:~$ mv july monthly.00
akbashiyanc@akbashiyanc:~$ ls
abc1      LICENSE  newdir    sqlite-autoconf-3250200
bin       may      package.json sqlite-autoconf-3250200.tar
Desktop  monthly  Pictures   work
Downloads monthly.00 README.md  Видео
akbashiyanc@akbashiyanc:~$ ls monthly.00
july monthly
akbashiyanc@akbashiyanc:~$
```

Рис. 3.7: Перемещение файлов в другой каталог

Переименовать каталог monthly.00 в monthly.01 (рис. 3.8).

```
akbashiyanc@akbashiyanc:~$ mv monthly.00 monthly.01
akbashiyanc@akbashiyanc:~$ ls
abcl1      LICENSE    newdir     sqlite-autoconf-3250200
bin        may        package.json  sqlite-autoconf-3250200.tar
Desktop    monthly    Pictures    work
Downloads  monthly.01 README.md   Видео
akbashiyanc@akbashiyanc:~$
```

Рис. 3.8: Переименование каталогов в текущем каталоге

Переместим каталог monthly.01 в каталог reports (рис. 3.9).

```
akbashiyanc@akbashiyanc:~$ mkdir reports
akbashiyanc@akbashiyanc:~$ ^[[200~mv monthly.01 reports
bash: mv: команда не найдена...
akbashiyanc@akbashiyanc:~$ ^C
akbashiyanc@akbashiyanc:~$ ^C
akbashiyanc@akbashiyanc:~$ mv monthly.01 reports
akbashiyanc@akbashiyanc:~$ ls
abcl1    Downloads  monthly    Pictures    sqlite-autoconf-3250200
bin      LICENSE    newdir     README.md   sqlite-autoconf-3250200.tar
Desktop  may        package.json  reports     work
akbashiyanc@akbashiyanc:~$ ;s reports
bash: синтаксическая ошибка рядом с неожиданным маркером «;»
akbashiyanc@akbashiyanc:~$ ls reports
monthly.01
akbashiyanc@akbashiyanc:~$
```

Рис. 3.9: Перемещение каталога в другой каталог

Переименовать каталог reports/monthly.01 в reports/monthly (рис. 3.10).

```
akbashiyanc@akbashiyanc:~$ mv reports/monthly.01 reports/monthly
akbashiyanc@akbashiyanc:~$ ls reports
monthly
akbashiyanc@akbashiyanc:~$
```

Рис. 3.10: Переименование каталога, не являющегося текущим

Создадим файл ~/may с правом выполнения для владельца (рис. 3.11).

```
akbashiyanc@akbashiyanc:~$ cd
akbashiyanc@akbashiyanc:~$ touch may
akbashiyanc@akbashiyanc:~$ ls -l may
-rw-r--r--. 1 akbashiyanc akbashiyanc 0 map 24 09:18 may
akbashiyanc@akbashiyanc:~$ chmod u+x may
akbashiyanc@akbashiyanc:~$ ls -l may
-rwxr--r--. 1 akbashiyanc akbashiyanc 0 map 24 09:18 may
akbashiyanc@akbashiyanc:~$
```

Рис. 3.11: Право выполнения для владельца

Лишим владельца файла ~/may права на выполнение (рис. 3.12).

```
akbashiyc@akbashiyc:~$ chmod u-x may
akbashiyc@akbashiyc:~$ ls -l may
-rw-r--r--. 1 akbashiyc akbashiyc 0 мар 24 09:18 may
akbashiyc@akbashiyc:~$
```

Рис. 3.12: Лишение права

Создадим каталог monthly с запретом на чтение для членов группы и всех остальных пользователей (рис. 3.13).

```
akbashiyc@akbashiyc:~$ mkdir monthly
mkdir: невозможно создать каталог «monthly»: Файл существует
akbashiyc@akbashiyc:~$ chmod g-r, o-r monthly
chmod: неверный режим: «g-r,»
По команде «chmod --help» можно получить дополнительную информацию.
akbashiyc@akbashiyc:~$ chmod g-r o-r monthly
chmod: невозможно получить доступ к 'o-r': Нет такого файла или каталога
akbashiyc@akbashiyc:~$ chmod g-r monthly
akbashiyc@akbashiyc:~$ chmod o-r monthly
```

Рис. 3.13: Запрет на чтение

Создадим файл ~/abc1 с правом записи для членов группы (рис. 3.14).

```
akbashiyc@akbashiyc:~$ touch abc1
akbashiyc@akbashiyc:~$ chmod g+w abc1
akbashiyc@akbashiyc:~$ ls -l abc1
-rw-rw-r--. 1 akbashiyc akbashiyc 0 мар 24 09:22 abc1
akbashiyc@akbashiyc:~$
```

Рис. 3.14: Право записи для группы

Скопируем файл /usr/include/sys/io.h в домашний каталог и назовём его equipment. Для этого используем команду cp (рис. 3.15).

```
akbashiyc@akbashiyc:~$ cp /usr/include/sys/io.h ~/equipment
akbashiyc@akbashiyc:~$ ls
abc1      Downloads  may        package.json  reports
bin       equipment  monthly    Pictures       sqlite-autoconf-3250
Desktop  LICENSE   newdir     README.md     sqlite-autoconf-3250
akbashiyc@akbashiyc:~$
```

Рис. 3.15: equipment

Переместим файл equipment в каталог ~/ski.plases. Для этого используем команду mv (рис. 3.16).

```

akbashiyanc@akbashiyanc:~$ mv equipment ski.plases
akbashiyanc@akbashiyanc:~$ ls
abc1      Downloads  monthly    Pictures    ski.plases
bin       LICENSE   newdir     README.md  sqlite-autoconf-32
Desktop   may       package.json  reports    sqlite-autoconf-32
akbashiyanc@akbashiyanc:~$ ls ski.plases
equipment
akbashiyanc@akbashiyanc:~$

```

Рис. 3.16: mv equipment

Переименуем файл ~/ski.plases/equipment в ~/ski.plases/equiplist. Для этого используем команду mv (рис. 3.17).

```

akbashiyanc@akbashiyanc:~$ mv ~/ski.plases/equipment ~/ski.plases/equiplist
akbashiyanc@akbashiyanc:~$ ls ski.plases
equiplist
akbashiyanc@akbashiyanc:~$

```

Рис. 3.17: mv equipment

Создадим в домашнем каталоге файл abc1 и скопируем его в каталог ~/ski.plases, назовём его equiplist2. Для этого используем команду cp (рис. 3.18).

```

akbashiyanc@akbashiyanc:~$ touch abc1
akbashiyanc@akbashiyanc:~$ cp abc1 ski.plases/equiplist2
akbashiyanc@akbashiyanc:~$ ls ski.plases
equiplist equiplist2
akbashiyanc@akbashiyanc:~$

```

Рис. 3.18: equipment

Создадим каталог с именем equipment в каталоге ~/ski.plases. Для этого используем команду mkdir (рис. 3.19).

```

akbashiyanc@akbashiyanc:~$ mkdir ski.plases/equipment
akbashiyanc@akbashiyanc:~$ ls ski.plases
equiplist equiplist2 equipment
akbashiyanc@akbashiyanc:~$

```

Рис. 3.19: mkdir equipment

Переместим файлы ~/ski.plases/equiplist и equiplist2 в каталог ~/ski.plases/equipment. Для этого используем команду mv (рис. 3.20).

```

akbashiyanc@akbashiyanc:~$ mv ski.places/equiplist ski.places/equipment
akbashiyanc@akbashiyanc:~$ ls ski.places
equiplist2  equipment
akbashiyanc@akbashiyanc:~$ mv ski.places/equiplist2 ski.places/equipment
akbashiyanc@akbashiyanc:~$ ls ski.places
equipment
akbashiyanc@akbashiyanc:~$ ls ski.places/equipment
equiplist  equiplist2

```

Рис. 3.20: mv equipment

Создадим и переместим каталог ~/newdir в каталог ~/ski.places и назовём его plans. Для этого используем команду mv (рис. 3.21).

```

akbashiyanc@akbashiyanc:~$ ls
abcl  Downloads  monthly  Pictures  ski.places
bin   LICENSE    newdir   README.md sqlite-autoconf-3250200
Desktop  may       package.json  reports  sqlite-autoconf-3250200.tar
akbashiyanc@akbashiyanc:~$ mv newdir ski.places/plans
akbashiyanc@akbashiyanc:~$ ls ski.places
equipment  plans
akbashiyanc@akbashiyanc:~$

```

Рис. 3.21: mv equipment

Создадим директории australia и play, и с помощью chmod изменим права (рис. 3.22).

```

akbashiyanc@akbashiyanc:~$ mkdir australia
akbashiyanc@akbashiyanc:~$ chmod -R u=rwx,g=r,o=r australia
akbashiyanc@akbashiyanc:~$ ls -l australia
итого 0
akbashiyanc@akbashiyanc:~$ ls -l australia
итого 0
akbashiyanc@akbashiyanc:~$ getfalc australia
bash: getfalc: команда не найдена...
Аналогичная команда: 'getfacl'
akbashiyanc@akbashiyanc:~$ mkdir play
akbashiyanc@akbashiyanc:~$ chmod -R u=rwx,g=x,o=x play
akbashiyanc@akbashiyanc:~$ touch mv os

```

Рис. 3.22: chmod для директорий

Создадим файлы my_os и feathers, и с помощью chmod изменим права (рис. 3.23).

```

akbashiyanc@akbashiyanc:~$ chmod u=rwx,g=r,o=r my_os
akbashiyanc@akbashiyanc:~$ chmod u=rw,g=rw,o=r feathers

```

Рис. 3.23: chmod для файлов

Скопируем файл ~/features в файл ~/files.old. Для этого используем команду cp (рис. 3.24).

```
akbashiyanc@akbashiyanc:~$ cp features files.old
akbashiyanc@akbashiyanc:~$ ls
abc1      Downloads  may        Pictures
australia features   monthly    play
bin       files.old  my_os      README.md
Desktop   LICENSE   package.json reports
akbashiyanc@akbashiyanc:~$
```

Рис. 3.24: cp features

Переместим файл ~/files.old в каталог ~/play. Для этого используем команду mv (рис. 3.25).

```
akbashiyanc@akbashiyanc:~$ mv files.old play
akbashiyanc@akbashiyanc:~$ ls play
files.old
akbashiyanc@akbashiyanc:~$
```

Рис. 3.25: mv files.old

Скопируем каталог ~/play в каталог ~/fun. Для этого используем команду cp (рис. 3.26).

```
akbashiyanc@akbashiyanc:~$ cp -r play fun
akbashiyanc@akbashiyanc:~$ ls
abc1      Downloads  may        Pictures
australia features   monthly    play
bin       fun        my_os      README.md
Desktop   LICENSE   package.json reports
akbashiyanc@akbashiyanc:~$
```

Рис. 3.26: cp play

Переместим каталог ~/fun в каталог ~/play и назовём его games. Для этого используем команду mv (рис. 3.27).

```
akbashiyanc@akbashiyanc:~$ mv fun play
akbashiyanc@akbashiyanc:~$ ls play
files.old  fun
akbashiyanc@akbashiyanc:~$
```

Рис. 3.27: mv fun

Лишим владельца файла ~/features права на чтение. Заметим, что теперь мы не можем читать файл. Вернем все обратно (рис. 3.28).

```
^[[LA^C
akbashiyanc@akbashiyanc:~$ chmod u-r features
akbashiyanc@akbashiyanc:~$ cat features
cat: features: Отказано в доступе
akbashiyanc@akbashiyanc:~$ chmod u+r features
```

Рис. 3.28: chmod features

Лишим владельца каталога ~/play права на выполнение. Заметим, что теперь мы не можем выполнять команды с каталогом и файлами внутри. Вернем все обратно (рис. 3.29).

```
akbashiyanc@akbashiyanc:~$ chmod u-x play
akbashiyanc@akbashiyanc:~$ cd play
bash: cd: play: Отказано в доступе
akbashiyanc@akbashiyanc:~$ chmod u+x play
akbashiyanc@akbashiyanc:~$
```

Рис. 3.29: chmod play

Прочитаем man по командам mount, fsck, mkfs, kill (рис. 3.30-3.33).

```
akbashiyanc@akbashiyanc:/home/akbashiyanc — man mount
akbashiyanc@akbashiyanc:/home/akbashiyanc — man mount x root@akbashiyanc:~$

MOUNT(8) System Administration

NAME
    mount - mount a filesystem

SYNOPSIS
    mount [-h|-V]

    mount [-l] [-t fstype]

    mount -a [-fFnrsvw] [-t fstype] [-O optlist]

    mount [-fnrsvw] [-o options] device mountpoint

    mount [-fnrsvw] [-t fstype] [-o options] device mountpoint

    mount --bind|--rbind|--move olddir newdir

    mount --make-[shared|slave|private|unbindable|rshared|rslave|rprivate|runbindable] mountpoint

DESCRIPTION
    All files accessible in a Unix system are arranged in one big tree, the file hierarchy, which is spread out over several devices. The mount command serves to attach the filesystem to the tree.
```

Рис. 3.30: man mount

```
akbashiyanc@akbashiyanc:/home/akbashiyanc — man fsck
akbashiyanc@akbashiyanc:/home/akbashiyanc — man fsck x root@akbashiyanc:~$

FSCK(8) System Administration

NAME
    fsck - check and repair a Linux filesystem

SYNOPSIS
    fsck [-lsAVRTMNP] [-r [fd]] [-C [fd]] [-t fstype] [filesystem...] [--] [fs-specific-options]

DESCRIPTION
    fsck is used to check and optionally repair one or more Linux filesystems. filesystem can be a device (e.g., /dev/hdc1, /dev/sdb2), a mount point (e.g., /, /usr, /home), or a filesystem label or UUID (e.g., UUID=8868abf6-88c5-4a83-98b8-bfc24057f7bd or LABEL=root). Normally, the fsck program will check all filesystems on different physical disk drives in parallel to reduce the total amount of time needed.

    If no filesystems are specified on the command line, and the -A option is not specified, fsck will check all filesystems in /etc/fstab serially. This is equivalent to the -As options.

    The exit status returned by fsck is the sum of the following conditions:

    0      No errors
    1      Filesystem errors corrected
    2      System should be rebooted
```

Рис. 3.31: man fsck

```
akbashiyanc@akbashiyanc:/home/akbashiyanc — man mkfs
akbashiyanc@akbashiyanc:/home/akbashiyanc — man mkfs x root@akbashiyanc:~$

MKFS(8) System Administration

NAME
  mkfs - build a Linux filesystem

SYNOPSIS
  mkfs [options] [-t type] [fs-options] device [size]

DESCRIPTION
  This mkfs frontend is deprecated in favour of filesystem specific mkfs.<type> utils.

  mkfs is used to build a Linux filesystem on a device, usually a hard disk partition. The
  the device name (e.g., /dev/hda1, /dev/sdb2), or a regular file that shall contain the
  is the number of blocks to be used for the filesystem.

  The exit status returned by mkfs is 0 on success and 1 on failure.

  In actuality, mkfs is simply a front-end for the various filesystem builders (mkfs.fstype).
  filesystem-specific builder is searched for via your PATH environment setting only. Please
  filesystem-specific builder manual pages for further details.

OPTIONS
  -t, --type type
    Specify the type of filesystem to be built. If not specified, the default filesystem
    used.

  fs-options
    Filesystem-specific options to be passed to the real filesystem builder.
```

Рис. 3.32: man mkfs

```
akbashiyanc@akbashiyanc:/home/akbashiyanc — man kill
akbashiyanc@akbashiyanc:/home/akbashiyanc — man kill x root@akbashiyanc:~$

KILL(1) User Commands

NAME
  kill - terminate a process

SYNOPSIS
  kill [-signal|-s signal|-p] [-q value] [-a] [--timeout milliseconds signal] [--] pid|p
  kill -l [number] | -L

DESCRIPTION
  The command kill sends the specified signal to the specified processes or process groups.

  If no signal is specified, the TERM signal is sent. The default action for this signal is
  This signal should be used in preference to the KILL signal (number 9), since a process
  TERM signal in order to perform clean-up steps before terminating in an orderly fashion.
  terminate after a TERM signal has been sent, then the KILL signal may be used; be aware
  be caught, and so does not give the target process the opportunity to perform any clean

  Most modern shells have a builtin kill command, with a usage rather similar to that of
  The --all, --pid, and --queue options, and the possibility to specify processes by comm
  extensions.

  If signal is 0, then no actual signal is sent, but error checking is still performed.

ARGUMENTS
  The list of processes to be signaled can be a mixture of names and PIDs.

  pid
    Each pid can be expressed in one of the following ways:

    n
      where n is larger than 0. The process with PID n is signaled.

    0
      All processes in the current process group are signaled.
```

Рис. 3.33: man kill

4 Выводы

В этой лабораторной работе мы изучили работу с файловой системой.