

ОС

Lab05

Группа: НБИбд02-21

Студент: Межидов Хамзат

1. Выполните все примеры, приведённые в первой части описания лабораторной работы

```
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ mkdir april
[liveuser@localhost-live ~]$ mv april july
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads july Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ mv july april
[liveuser@localhost-live ~]$ ls
april bin Desktop Documents Downloads Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ mv april july
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads july Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ S█

[liveuser@localhost-live ~]$ mkdir monthly.00
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads july monthly.00 Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ mv july monthly.00
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads monthly.00 Music Pictures Public Templates tmp Videos
[liveuser@localhost-live ~]$ cd monthly.00
[liveuser@localhost-live monthly.00]$ ls
july
[liveuser@localhost-live monthly.00]$ cd ..
[liveuser@localhost-live ~]$ mv monthly.00 monthly.01
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads monthly.01 Music Pictures Public Templates tmp Videos

[liveuser@localhost-live ~]$ mkdir reports
[liveuser@localhost-live ~]$ mv monthly.01 reports
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads Music Pictures Public reports Templates tmp Videos
[liveuser@localhost-live ~]$ cd reports
[liveuser@localhost-live reports]$ ls
monthly.01
[liveuser@localhost-live reports]$ cd monthly.01/
[liveuser@localhost-live monthly.01]$ ls
july reports
[liveuser@localhost-live reports]$ cd ..
[liveuser@localhost-live ~]$ mv reports/monthly.01 reports/monthly
[liveuser@localhost-live ~]$ cd reports/
[liveuser@localhost-live reports]$ ls
monthly
[liveuser@localhost-live reports]$ █
```

2. Выполните следующие действия, зафиксировав в отчёте по лабораторной работе используемые при этом команды и результаты их выполнения:

2.1. Скопируйте файл /usr/include/sys/io.h в домашний каталог и назовите его equipment. Если файла io.h нет, то используйте любой другой файл в каталоге /usr/include/sys/ вместо него.

```

[liveuser@localhost-live ~]$ mkdir usr
[liveuser@localhost-live ~]$ cd usr
[liveuser@localhost-live usr]$ mkdir include
[liveuser@localhost-live usr]$ cd include/
[liveuser@localhost-live include]$ mkdir sys
[liveuser@localhost-live include]$ cd sys
[liveuser@localhost-live sys]$ touch io.h
[liveuser@localhost-live sys]$ ls
io.h
[liveuser@localhost-live sys]$ cd ..
[liveuser@localhost-live include]$ cd ..
[liveuser@localhost-live usr]$ cd ..
[liveuser@localhost-live ~]$ cd usr/include/sys/
[liveuser@localhost-live sys]$ cp io.h equipment
[liveuser@localhost-live sys]$ ls
equipment io.h

```

2.2. В домашнем каталоге создайте директорию ~/ski.places.

2.3. Переместите файл equipment в каталог ~/ski.places.

```

[liveuser@localhost-live ~]$ mv equipment ski.places
[liveuser@localhost-live ~]$ ls
bin Desktop Documents Downloads Music Pictures Public reports ski.places Templates tmp usr Videos
[liveuser@localhost-live ~]$ cd ski.places/
[liveuser@localhost-live ski.places]$ ls
equipment
[liveuser@localhost-live ski.places]$

```

2.4. Переименуйте файл ~/ski.places/equipment в ~/ski.places/equiplist.

```

[liveuser@localhost-live ~]$ mv ski.places/equipment ski.places/equiplist
[liveuser@localhost-live ~]$ cd ski.places/
[liveuser@localhost-live ski.places]$ ls
equiplist
[liveuser@localhost-live ski.places]$

```

2.5. Создайте в домашнем каталоге файл abc1 и скопируйте его в каталог ~/ski.places, назовите его equiplist2.

```

[liveuser@localhost-live ~]$ touch abc1
[liveuser@localhost-live ~]$ ls
abc1 bin Desktop Documents Downloads Music Pictures Public reports ski.places Templates tmp usr Videos
[liveuser@localhost-live ~]$ mv abc1 ski.places/
[liveuser@localhost-live ~]$ mv ski.places/abc1 ski.places/equiplist2
[liveuser@localhost-live ~]$ cd ski.places/
[liveuser@localhost-live ski.places]$ ls
equiplist equiplist2
[liveuser@localhost-live ski.places]$

```

2.6. Создайте каталог с именем equipment в каталоге ~/ski.places.

2.7. Переместите файлы ~/ski.places/equiplist и equiplist2 в каталог ~/ski.places/equipment.

```

[liveuser@localhost-live ~]$ cd ski.plases/
[liveuser@localhost-live ski.plases]$ mkdir equipment
[liveuser@localhost-live ski.plases]$ mv equipment equiplist1 equiplist2
mv: указанная цель 'equiplist2' не является каталогом
[liveuser@localhost-live ski.plases]$ mv equiplist1 equiplist2 equipment
mv: не удалось выполнить stat для 'equiplist1': Нет такого файла или каталога
[liveuser@localhost-live ski.plases]$ mv equiplist2 equipment
mv: не удалось выполнить stat для 'equiplist2': Нет такого файла или каталога
[liveuser@localhost-live ski.plases]$ mv equipment equiplist1
[liveuser@localhost-live ski.plases]$ ls
equiplist  equiplist1
[liveuser@localhost-live ski.plases]$ mv equiplist equiplist2
[liveuser@localhost-live ski.plases]$ mv equiplist equiplist1
mv: не удалось выполнить stat для 'equiplist': Нет такого файла или каталога
[liveuser@localhost-live ski.plases]$ mv equiplist2 equiplist1
[liveuser@localhost-live ski.plases]$ ls
equiplist1
[liveuser@localhost-live ski.plases]$ mv equiplist1 equipment
[liveuser@localhost-live ski.plases]$ cd
[liveuser@localhost-live ~]$ cd ski.plases/
[liveuser@localhost-live ski.plases]$ ls
equipment
[liveuser@localhost-live ski.plases]$ ls equipment/
equiplist1  equiplist2
[liveuser@localhost-live ski.plases]$

```

2.8. Создайте и переместите каталог ~/newdir в каталог ~/ski.plases и назовите его plans

```

[liveuser@localhost-live ~]$ mkdir newdir
[liveuser@localhost-live ~]$ mv newdir ski.plases
[liveuser@localhost-live ~]$ ls ski.plases/
equipment  newdir
[liveuser@localhost-live ~]$ mv ski.plases/
equipment/ newdir/
[liveuser@localhost-live ~]$ mv ski.plases/newdir ski.plases/plans
[liveuser@localhost-live ~]$ ls ski.plases/
equipment  plans
[liveuser@localhost-live ~]$

```

3. Определите опции команды chmod, необходимые для того, чтобы присвоить перечисленным ниже файлам выделенные права доступа, считая, что в начале таких прав нет:

3.1. drwxr--r-- ... australia

3.2. drwx--x--x ... play

3.3. -r-xr--r-- ... my\_os

3.4. -rw-rw-r-- ... feathers При необходимости создайте нужные файлы

```

[liveuser@localhost-live ~]$ mkdir 123
[liveuser@localhost-live ~]$ cd 123
[liveuser@localhost-live 123]$ touch australia
[liveuser@localhost-live 123]$ touch play
[liveuser@localhost-live 123]$ touch my_os
[liveuser@localhost-live 123]$ touch feathers
[liveuser@localhost-live 123]$ ls
australia feathers my_os play
[liveuser@localhost-live 123]$ ls -l australia
-rw-rw-r--. 1 liveuser liveuser 0 мая 7 15:17 australia
[liveuser@localhost-live 123]$ ls -l feathers
-rw-rw-r--. 1 liveuser liveuser 0 мая 7 15:18 feathers
[liveuser@localhost-live 123]$ ls -l play
-rw-rw-r--. 1 liveuser liveuser 0 мая 7 15:17 play
[liveuser@localhost-live 123]$ ls -l my_os
-rw-rw-r--. 1 liveuser liveuser 0 мая 7 15:17 my_os
[liveuser@localhost-live 123]$ chmod u+x australia
[liveuser@localhost-live 123]$ ls -l australia
-rwxrw-r--. 1 liveuser liveuser 0 мая 7 15:17 australia
[liveuser@localhost-live 123]$ chmod u-x play
[liveuser@localhost-live 123]$ ls -l play
-rw-rw-r--. 1 liveuser liveuser 0 мая 7 15:17 play
[liveuser@localhost-live 123]$ chmod g-r, r-o my_os
chmod: неверный режим: «g-r,»
По команде «chmod --help» можно получить дополнительную информацию.
[liveuser@localhost-live 123]$ chmod g-r, o-r my_os
chmod: неверный режим: «g-r,»
По команде «chmod --help» можно получить дополнительную информацию.
[liveuser@localhost-live 123]$ chmod u-r, o-r my_os
chmod: неверный режим: «u-r,»
По команде «chmod --help» можно получить дополнительную информацию.
[liveuser@localhost-live 123]$ chmod u+r, o-r my_os
chmod: неверный режим: «u+r,»
По команде «chmod --help» можно получить дополнительную информацию.
[liveuser@localhost-live 123]$

```

4. Прodelайте приведённые ниже упражнения, записывая в отчёт по лабораторной работе используемые при этом команды:

#### 4.1. Просмотрите содержимое файла /etc/password.

```

[liveuser@localhost-live ~]$ mkdir play
[liveuser@localhost-live ~]$ mkdir fun
[liveuser@localhost-live ~]$ mv play fun
[liveuser@localhost-live ~]$ mv fun games
[liveuser@localhost-live ~]$ ls
.  ..  .ssh  Desktop  Documents  Downloads  etc  games  Music  Pictures  Public  reports  ski-places  Templates  tmp  usr  Videos
[liveuser@localhost-live ~]$ cd games
[liveuser@localhost-live games]$ ls
play
[liveuser@localhost-live games]$ cd
[liveuser@localhost-live ~]$

```

#### 4.2. Скопируйте файл ~/feathers в файл ~/file.old.

```

[liveuser@localhost-live ~]$ cd 123
[liveuser@localhost-live 123]$ cp feathers file.old
cp: невозможно открыть 'feathers' для чтения: Отказано в доступе
[liveuser@localhost-live 123]$ chmod u+r feathers
[liveuser@localhost-live 123]$ cp feathers file.old
[liveuser@localhost-live 123]$ ls
australia feathers file.old my_os play
[liveuser@localhost-live 123]$

```

#### 4.3. Переместите файл ~/file.old в каталог ~/play.

```
[liveuser@localhost-live ~]$ mkdir play
[liveuser@localhost-live ~]$ mkdir fun
[liveuser@localhost-live ~]$ mv play fun
[liveuser@localhost-live ~]$ mv fun games
[liveuser@localhost-live ~]$ ls
.  ..  123  bin  Desktop  Documents  Downloads  etc  games  Music  Pictures  Public  reports  ski.places  Templates  tmp  usr  Videos
[liveuser@localhost-live ~]$ cd games
[liveuser@localhost-live games]$ ls
play
[liveuser@localhost-live games]$ cd
[liveuser@localhost-live ~]$
```

4.4. Скопируйте каталог ~/play в каталог ~/fun.

4.5. Переместите каталог ~/fun в каталог ~/play и назовите его games.

```
[liveuser@localhost-live ~]$ mkdir play
[liveuser@localhost-live ~]$ mkdir fun
[liveuser@localhost-live ~]$ mv play fun
[liveuser@localhost-live ~]$ mv fun games
[liveuser@localhost-live ~]$ ls
.  ..  123  bin  Desktop  Documents  Downloads  etc  games  Music  Pictures  Public  reports  ski.places  Templates  tmp  usr  Videos
[liveuser@localhost-live ~]$ cd games
[liveuser@localhost-live games]$ ls
play
[liveuser@localhost-live games]$ cd
[liveuser@localhost-live ~]$
```

4.6. Лишите владельца файла ~/feathers права на чтение.

4.7. Что произойдёт, если вы попытаетесь просмотреть файл ~/feathers командой cat?

```
[liveuser@localhost-live ~]$ cd 123
[liveuser@localhost-live 123]$ chmod u-r feathers
[liveuser@localhost-live 123]$ cat feathers
cat: feathers: Отказано в доступе
[liveuser@localhost-live 123]$
```

4.8. Что произойдёт, если вы попытаетесь скопировать файл ~/feathers?

4.9. Дайте владельцу файла ~/feathers право на чтение.

```
[liveuser@localhost-live 123]$ chmod u+r feathers
[liveuser@localhost-live 123]$ cp feathers file.old
[liveuser@localhost-live 123]$ ls
australia  feathers  file.old  my_os  play
[liveuser@localhost-live 123]$
```

4.10. Лишите владельца каталога ~/play права на выполнение.

4.11. Перейдите в каталог ~/play. Что произошло?

4.12. Дайте владельцу каталога ~/play право на выполнение.

```
[liveuser@localhost-live ~]$ mkdir play
[liveuser@localhost-live ~]$ chmod u-xp play
chmod: неверный режим: «u-xp»
По команде «chmod --help» можно получить дополнительную информацию.
[liveuser@localhost-live ~]$ chmod u-x play
[liveuser@localhost-live ~]$ cd play
bash: cd: play: Отказано в доступе
[liveuser@localhost-live ~]$ chmod u+x play
[liveuser@localhost-live ~]$
```

```
[liveuser@localhost-live 123]$ chmod u-x play
[liveuser@localhost-live 123]$ cat play
[liveuser@localhost-live 123]$ cat play
dsalklksalkdsalkd
[liveuser@localhost-live 123]$ chmod u+x play
[liveuser@localhost-live 123]$
```

5. Прочитайте ман по командам mount, fsck, mkfs, kill и кратко их охарактеризуйте, приведя примеры

```
kill(1)                                User Commands                                kill(1)
NAME
    kill - terminate a process

SYNOPSIS
    kill [-signal|-s signal|-p] [-q value] [-w] [--timeout milliseconds]
    signal [-i] pidname...
    kill -l {number} | -t

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 to terminate the process. This signal should be used
    in preference to the KILL signal (number 9), since a process may
    install a handler for the TERM signal in order to perform clean-up
    steps before terminating in an orderly fashion. If a process does not
    terminate after a TERM signal has been sent, then the KILL signal may
    be used; be aware that the latter signal cannot be caught, and so does
    not give the target process the opportunity to perform any clean-up
    before terminating.

    Most modern shells have a builtin kill command, with a usage rather
    similar to that of the command described here. The --all, --pid, and
    --name options, and the possibility to specify processes by command
    name, are local 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.

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

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

    *
        all processes in the current process group are signaled.

    Manual page kill(1) line 1 (press h for help or q to quit)
```

```
mount(8)                                System Administration                                mount(8)
NAME
    mount - mount a filesystem

SYNOPSIS
    mount [-h] [-V]
    mount [-l] [-t fstype]
    mount -a [-ffstype] [-t fstype] [-o optlist]
    mount [-fstype] [-o options] device[mountpoint]
    mount [-fstype] [-t fstype] [-o options] device mountpoint
    mount --bind|--bind|--move olddir newdir
    mount --make {shared|slave|private|unbindable|shared|slave|private|unbindable} mountpoint

DESCRIPTION
    All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at /. These files can be spread out over several devices. The mount command serves to attach the filesystem found on some device
    to the big file tree. Conversely, the umount(8) command will detach it again. The filesystem is used to control how data is stored on the device or provided in a virtual way by network or other services.

    The standard form of the mount command is:

    mount -t type device dir

    This tells the kernel to attach the filesystem found on device (which is of type type) at the directory dir. The option -t type is optional. The mount command is usually able to detect a filesystem. The root permissions are
    necessary to mount a filesystem by default; see section "Non-superuser mounts" below for more details. The previous contents (if any) and owner and mode of dir become invisible, as long as this filesystem remains mounted,
    the pathname dir refers to the root of the filesystem on device.

    If only the directory or the device is given, for example:

    mount /dir

    then mount looks for a mountpoint (and if not found then for a device) in the /etc/fstab file. It's possible to use the --target or --source options to avoid ambiguous interpretation of the given argument. For example:

    mount --target /mountpoint

    The same filesystem may be mounted more than once, and in some cases (e.g., network filesystems) the same filesystem may be mounted on the same mountpoint multiple times. The mount command does not implement any policy to
    control this behavior. All behavior is controlled by the kernel and it is usually specific to the filesystem driver. The exception is --all, in this case already mounted filesystems are ignored (see --all below for more
    details).

    Listing the mounts
    Manual page mount(8) line 1 (press h for help or q to quit)
```

```
fsck(8)                                System Administration                                fsck(8)
NAME
    fsck - check and repair a Linux filesystem

SYNOPSIS
    fsck [-laRWtWMP] [-t {fd}] [-t {fd}] [-t fstype] [filesystem...] [--] [--specific-options]

DESCRIPTION
    fsck is used to check and optionally repair one or more Linux filesystems. filesystem can be a device name (e.g., /dev/sda1, /dev/sdb2), a mount point (e.g., /, /usr, /home), or an filesystem label or UUID specifier (e.g.,
    UUID=6a89a1f6-84c2-4a43-80e6-5fca057f7bdc or LABEL=root). Normally, the fsck program will try to handle filesystems on different physical disk drives in parallel to reduce the total amount of time needed to check all of them.

    If no filesystems are specified on the command line, and the -A option is not specified, fsck will default to checking filesystems in /etc/fstab serially. This is equivalent to the -Aa 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
    4
        Filesystem errors left uncorrected
    8
        Operational error
    16
        Usage or syntax error
    32
        Checking canceled by user request
    128
        Shared-library error

    The exit status returned when multiple filesystems are checked is the bit-wise OR of the exit statuses for each filesystem that is checked.

    In actuality, fsck is simply a front-end for the various filesystem checkers (fsck_*[type]) available under Linux. The filesystem-specific checker is searched for in the PATH environment variable. If the PATH is undefined then
    fallback to /sbin.

    Manual page fsck(8) line 1 (press h for help or q to quit)
```



```
06:00p  Tegenman  C6,7 max 15:34  en  [System Administration]  mfs(8)

NAME
    mfs - build a linux filesystem

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

DESCRIPTION
    This mfs frontend is deprecated in favour of filesystem specific mfs-type utils.

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

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

    In actuality, mfs is simply a front-end for the various filesystem builders (mfs-type) available under linux. The filesystem-specific builder is searched for via your PATH environment setting only. Please see the 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 type (currently ext2) is used.

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

    -V, --verbose
        Produce verbose output, including all filesystem-specific commands that are executed. Specifying this option more than once inhibits execution of any filesystem-specific commands. This is really only useful for testing.

    -V, --version
        Display version information and exit. (Option -V will display version information only when it is the only parameter, otherwise it will work as --verbose.)

    -h, --help
        Display help text and exit.

BUGS
    All generic options must precede and not be combined with filesystem-specific options. Some filesystem-specific programs do not automatically detect the device size and require the size parameter to be specified.

AUTHORS
    David Engel <dauid@dds.com>, from A. van Rossum <waltje@u.walt.nl.mugnet.org>, from <kernel@cs.cmu.edu> <kernel@cs.cmu.edu>.

    The manual page was shamelessly adapted from Remy Card's version for the ext2 filesystem.

SEE ALSO
    fs(1), bootfs(8), fsck(1), mkfs(8), mkfs.bfs(8), mkfs.bfs(8), mkfs.ext2(8), mkfs.ext3(8), mkfs.ext4(8), mkfs.minix(8), mkfs.minix(8), mkfs.vfat(8), mkfs.vfat(8)

Manual page mfs(8) line 4 (press h for help or q to exit)
```