

# **Отчёт по лабораторной работе №5**

**Поиск файлов. Перенаправление ввода-вывода. Просмотр запущенных процессов**

Дарья Сергеевна Кочина

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>5</b>	<b>Выводы</b>	<b>24</b>

## Список иллюстраций

4.1	Запись файлов . . . . .	8
4.2	Файлы в file.txt . . . . .	9
4.3	Вывод имён файлов . . . . .	9
4.4	Файлы с расширением .conf . . . . .	10
4.5	Создание файлов . . . . .	10
4.6	Команда «find /etc -maxdepth1 -name “h*”   less» . . . . .	10
4.7	Список файлов . . . . .	11
4.8	Команда «find/ -name“log*” > logfile&» . . . . .	12
4.9	Фоновый режим процесса . . . . .	12
4.10	Команда «cat logfile» . . . . .	12
4.11	Удаление файла с помощью команды rm . . . . .	13
4.12	Редактор gedit . . . . .	13
4.13	Команда «pgrep gedit» . . . . .	13
4.14	Команда «ps  grep-i“gedit”» . . . . .	13
4.15	Команда man kill . . . . .	14
4.16	Информация по команде kill . . . . .	14
4.17	Завершение процесса gedit . . . . .	14
4.18	Команды df, du . . . . .	15
4.19	Информация по команде df . . . . .	15
4.20	Информация по команде du . . . . .	16
4.21	Команда df в терминале . . . . .	16
4.22	Команда du в терминале . . . . .	16
4.23	Команда man find . . . . .	17
4.24	Информация по команде find . . . . .	17
4.25	Команда «find~ -typed» . . . . .	17
4.26	Имена всех директорий . . . . .	18

# 1 Цель работы

Целью данной лабораторной работы является ознакомление с инструментами поиска файлов и фильтрации текстовых данных. А также приобретение практических навыков по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

## 2 Задание

Ознакомиться с инструментами поиска файлов и фильтрации текстовых данных.

## 3 Теоретическое введение

### Перенаправление ввода-вывода

В системе по умолчанию открыто три специальных потока:

- `stdin` — стандартный поток ввода (по умолчанию: клавиатура), файловый дескриптор 0;
- `stdout` — стандартный поток вывода (по умолчанию: консоль), файловый дескриптор 1;
- `stderr` — стандартный поток вывод сообщений об ошибках (по умолчанию: консоль), файловый дескриптор 2.

Большинство используемых в консоли команд и программ записывают результаты своей работы в стандартный поток вывода `stdout`. Например, команда `ls` выводит в стандартный поток вывода (консоль) список файлов в текущей директории. Потоки вывода и ввода можно перенаправлять на другие файлы или устройства. Проще всего это делается с помощью символов `>`, `»`, `<`, `«`.

### Поиск файла

Команда `find` используется для поиска и отображения на экран имён файлов, соответствующих заданной строке символов. Путь определяет каталог, начиная с которого по всем подкаталогам будет вестись поиск.

## 4 Выполнение лабораторной работы

1. Осуществила вход в систему, используя соответствующее имя пользователя.
2. Записала, в файл file.txt название файлов, содержащийся в определённом каталоге. Для того, чтобы записать в файл file.txt названия файлов, содержащихся в каталоге /etc, использовала команду «ls-a/etc> file.txt». Далее с помощью команды «ls-a~ » file.txt» дописываю в этот же файл названия файлов, содержащихся в домашнем каталоге. Командой «catfile.txt» просматриваю файл, чтобы убедиться в правильности действий. (рис. [4.12], [4.2])

```
dskochina@dk8n80 ~ $ ls -a /etc >file.txt
dskochina@dk8n80 ~ $ ls -a ~ - >> file.txt
ls: невозможно получить доступ к '-': Нет такого файла или каталога
dskochina@dk8n80 ~ $ cat file.txt
.
..
a2ps
acpi
adjtime
afs.keytab
alsa
apache2
apparmor.d
appstream.conf
ati
audit
autofs
avahi
bash
bash_completion.d
bindresvport.blacklist
binfmt.d
blkid.tab.old
bluetooth
brltty
brltty.conf
ca-certificates
ca-certificates.conf
cachefilesd.conf
cfg-update.conf
cfg-update.hosts
cgroup
```

Рис. 4.1: Запись файлов



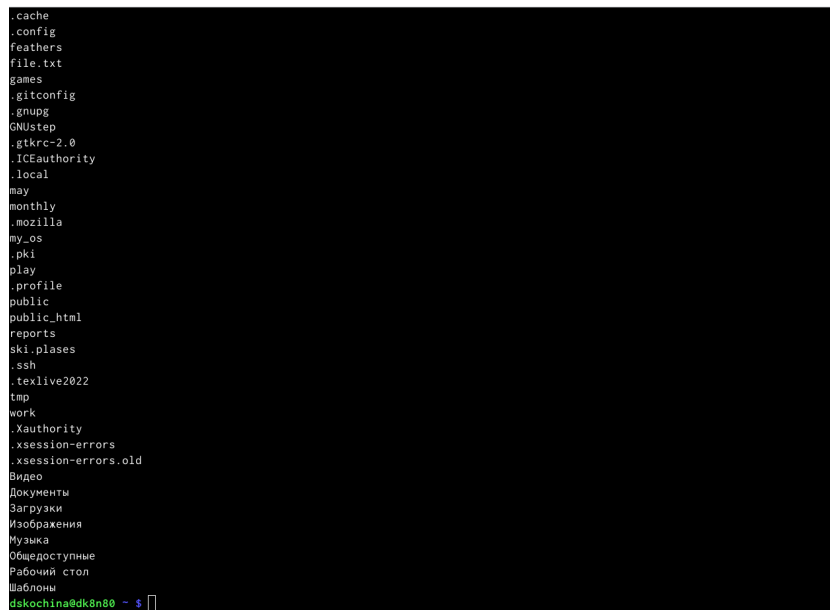


Рис. 4.2: Файлы в file.txt

3. Вывела имена всех файлов из file.txt, имеющих расширение .conf и записала их в новый текстовый файл conf.txt, используя команду «grep -e '.conf\$' file.txt > conf.txt». С помощью команды «cat conf.txt» проверяю правильность выполненных действий. (рис. [4.3], [4.4])



Рис. 4.3: Вывод имён файлов

```
request-key.conf
resolv.conf
roff-pass.conf
rsyncd.conf
rsyslog.conf
samba.conf
sddm.conf
sensors3.conf
signond.conf
smartd.conf
sudo.conf
sudo_logsrvd.conf
udhcpd.conf
updatedb.conf
vconsole.conf
whois.conf
xattr.conf
xinetd.conf
xtables.conf
dskochina@dk8n80 ~ $
```

Рис. 4.4: Файлы с расширением .conf

4. Определить, какие файлы в домашнем каталоге имеют имена, начинающиеся с символа с, можно несколькими командами:

- 1) «find ~ -maxdepth 1 -name "с\*" -print» (опция maxdepth 1 необходима для того, чтобы файлы находились строго только в домашнем каталоге);
- 2) «ls ~/с\*»;
- 3) «ls -a ~ | grep с\*». (рис. [4.5])

```
dskochina@dk8n80 ~ $ find ~ -maxdepth 1 -name "с*" -print
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/conf.txt
dskochina@dk8n80 ~ $ ls ~/с*
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/conf.txt
dskochina@dk8n80 ~ $ ls -a ~ | grep с*
conf.txt
dskochina@dk8n80 ~ $
```

Рис. 4.5: Создание файлов

5. Чтобы вывести на экран (постранично) имена файлов из каталога /etc, начинающиеся с символа h, я использовала команду «find /etc -maxdepth 1 -name "h\*" | less». (рис. [4.6], [4.7])

```
dskochina@dk8n80 ~ $ find /etc -maxdepth 1 -name "h*" | less
```

Рис. 4.6: Команда «find /etc -maxdepth 1 -name "h\*" | less»

```
/etc/hosts.allow
/etc/hsqldb
/etc/httpd
/etc/harbour
/etc/hostname
/etc/harbour.cfg
/etc/hal
/etc/hotplug.d
/etc/highlight
/etc/host.conf
/etc/htdig
/etc/hosts
/etc/hotplug
lines 1-13/13 (END)
```

Рис. 4.7: Список файлов

6. Запускаю в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log, используя команду «find/ -name“log\*” > logfile&». Так как в фоновом режиме у меня запустился беспрерывный процесс записывания файла, я сделала скриншоты некоторых частей работы процесса. Командой «cat logfile» проверяю выполненные действия. (рис. [4.8], [4.9], [4.10])

```
dskochina@dk8n80 ~ $ find / -name "log*" >logfile&
```

Рис. 4.8: Команда «find/ -name“log\*” > logfile&»

```
find: '/etc/cups/certs': Отказано в доступе
find: '/etc/skey': Отказано в доступе
find: '/etc/mail/spamassassin/sa-update-keys': Отказано в доступе
find: '/etc/cron.weekly': Отказано в доступе
find: '/etc/lvm/cache': Отказано в доступе
find: '/etc/sudoers.d': Отказано в доступе
find: '/etc/cron.daily': Отказано в доступе
find: '/etc/cron.hourly': Отказано в доступе
find: '/etc/fcron': Отказано в доступе
find: '/etc/polkit-1/rules.d': Отказано в доступе
find: '/etc/cron.monthly': Отказано в доступе
find: '/etc/audit/plugins.d': Отказано в доступе
find: '/etc/multipath': Отказано в доступе
```

Рис. 4.9: Фоновый режим процесса

```
dskochina@dk8n80 ~ $ cat logfile
/etc/logrotate.d
/etc/pam.d/login
/etc/firejail/login.users
/etc/libmbios/logging.conf
/etc/logrotate.conf
/etc/ImageMagick-7/log.xml
/etc/systemd/logind.conf
/etc/systemd/system/timers.target.wants/logrotate.timer
/etc/login.access
/etc/login.defs
/com/lib/portage/distfiles/log-0.4.16.crate
/com/lib/portage/distfiles/logrotate-3.20.1.tar.xz.asc
/com/lib/portage/distfiles/logrotate-3.20.1.tar.xz
/com/lib/portage/distfiles/log-0.4.8.crate
/com/lib/portage/distfiles/log-0.4.17.crate
/com/lib/portage/distfiles/log-0.4.14.crate
/com/lib/portage/distfiles/log4cpp-1.1.3.tar.gz
/com/lib/portage/distfiles/log-0.4.13.crate
/com/lib/portage/distfiles/logrotate-3.21.0.tar.xz.asc
/com/lib/portage/distfiles/logrotate-3.21.0.tar.xz
/com/lib/portage/distfiles/log4cplus-2.0.7.tar.bz2
/com/lib/portage/distfiles/logmein-hamachi-2.1.0.198-x64.tgz
/com/lib/portage/distfiles/logical-unification-0.4.5.gh.tar.gz
/com/lib/portage/distfiles/log4r-1.1.10.gem
/com/lib/portage/distfiles/logmein-hamachi-2.1.0.203-x64.tgz
/com/lib/portage/distfiles/log-0.4.11.crate
/com/lib/portage/extras/texlive/archive/logix.tar.xz
/com/lib/portage/extras/texlive/archive/logbox.doc.tar.xz
/com/lib/portage/extras/texlive/archive/logbox.source.tar.xz
/com/lib/portage/extras/texlive/archive/logbox.tar.xz
/com/lib/portage/extras/texlive/archive/logical-markup-utils.doc.tar.xz
/com/lib/portage/extras/texlive/archive/logical-markup-utils.tar.xz
/com/lib/portage/extras/texlive/archive/logicproof.doc.tar.xz
/com/lib/portage/extras/texlive/archive/logicproof.source.tar.xz
/com/lib/portage/extras/texlive/archive/logicproof.tar.xz
/com/lib/portage/extras/texlive/archive/logicpuzzle.doc.tar.xz
/com/lib/portage/extras/texlive/archive/logicpuzzle.tar.xz
```

Рис. 4.10: Команда «cat logfile»

7. Удалила файл ~/logfile с помощью команды «rm logfile».

```
dskochina@dk8n80 ~ $ rm logfile
dskochina@dk8n80 ~ $
```

Рис. 4.11: Удаление файла с помощью команды rm

8. Запускаю редактор gedit в фоновом режиме командой «gedit&». После этого на экране появляется окно редактора. (рис. [??])

```
dskochina@dk8n80 ~ $ gedit &
[1] 5557
dskochina@dk8n80 ~ $
```

Рис. 4.12: Редактор gedit

9. Чтобы определить идентификатор процесса gedit, использую команду «ps | grep -i "gedit"» (Скриншот -4.13). Из рисунка видно, что наш процесс имеет PID 5557. Узнать идентификатор процесса можно также, используя команду «pgrep gedit» или «pidof gedit». (рис. [4.13], [4.14])

```
dskochina@dk8n80 ~ $ pgrep gedit
5557
dskochina@dk8n80 ~ $
```

Рис. 4.13: Команда «pgrep gedit»

```
dskochina@dk8n80 ~ $ ps | grep -i "gedit"
```

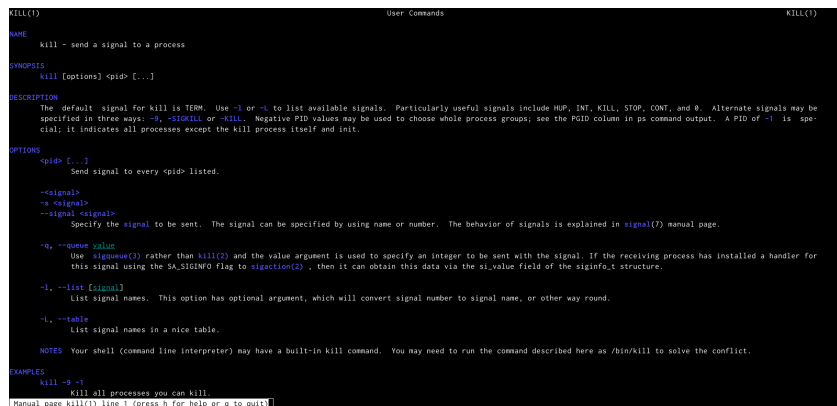
Рис. 4.14: Команда «ps | grep -i "gedit"»

10. Прочитав информацию о команде kill с помощью команды «man kill», используя её для завершения процесса gedit (команда «kill 5557»). (рис. [4.15], [4.16], [4.17])



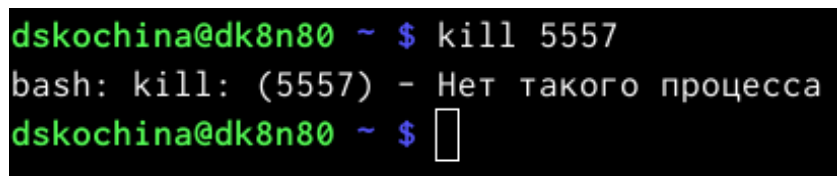
```
dskochina@dk8n80 ~ $ man kill
```

Рис. 4.15: Команда man kill



```
KILL(1)                                User Commands                                KILL(1)
NAME
  kill - send a signal to a process.
SYNOPSIS
  kill [options] <pid> [...]
DESCRIPTION
  The default signal for kill is TERM. Use -l or -t to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -s, -SIGINT or -KILL. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of -1 is special; it indicates all processes except the kill process itself and init.
OPTIONS
  <pid> [...]
    Send signal to every <pid> listed.
  --signal=
  -s <signal>
  --signal <signal>
    Specify the signal to be sent. The signal can be specified by using name or number. The behavior of signals is explained in signal(7) manual page.
  -q, --queue <value>
    Use <signal(2)> rather than kill(2) and the value argument is used to specify an integer to be sent with the signal. If the receiving process has installed a handler for this signal using the SA_SIGINFO flag to sigaction(2), then it can obtain this data via the si_value field of the siginfo_t structure.
  -l, --list [<signal>]
    List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.
  -L, --table
    List signal names in a nice table.
NOTES
  Your shell (command line interpreter) may have a built-in kill command. You may need to run the command described here as /bin/kill to solve the conflict.
EXAMPLES
  kill -9 -1
    Kill all processes you can kill.
Manual page kill(1) line 1 (press h for help or q to quit)
```

Рис. 4.16: Информация по команде kill



```
dskochina@dk8n80 ~ $ kill 5557
bash: kill: (5557) - Нет такого процесса
dskochina@dk8n80 ~ $
```

Рис. 4.17: Завершение процесса gedit

11. С помощью команд «man df» и «man du» узнаю информацию по необходимым командам и далее использую их. df – утилита, показывающая список всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования. Синтаксис: df [опции] устройств. du – утилита, предназначенная для вывода информации об

объеме дискового пространства, занятого файлами и директориями. Она принимает путь к элементу файловой системы и выводит информацию о количестве байт дискового пространства или блоков диска, задействованных для его хранения. Синтаксис: `du [опции] каталог или файл`. (рис. [4.18], [4.19], [4.20], [4.21], [4.22])

```
dskochina@dk8n80 ~ $ man df
[4]+  Остановлен      man df
dskochina@dk8n80 ~ $ man du
[5]+  Остановлен      man du
dskochina@dk8n80 ~ $
```

Рис. 4.18: Команды `df`, `du`

```
df(1)                                User Commands                                df(1)
NAME
  df - report file system space usage
SYNOPSIS
  df [OPTION]... [FILE]...
DESCRIPTION
  This manual page documents the GNU version of df. df displays the amount of space available on the file system containing each file name argument. If no file name is given, the space available on all currently mounted file systems is shown. Space is shown in 1K blocks by default, unless the environment variable POSIXLY_CORRECT is set, in which case 512-byte blocks are used.
  If an argument is the absolute file name of a device node containing a mounted file system, df shows the space available on that file system rather than on the file system containing the device node. This version of df cannot show the space available on unmounted file systems, because on most kinds of systems doing so requires very nonportable intimate knowledge of file system structures.
OPTIONS
  Show information about the file system on which each FILE resides, or all file systems by default.
  Mandatory arguments to long options are mandatory for short options too.
  -a, --all
      include pseudo, duplicate, inaccessible file systems
  -B, --block-size=SIZE
      scale sizes by SIZE before printing them; e.g., "-BM" prints sizes in units of 1,048,576 bytes; see SIZE format below
  -h, --human-readable
      print sizes in powers of 1024 (e.g., 1023M)
  -H, --si
      print sizes in powers of 1000 (e.g., 1.1G)
  -i, --inodes
      list inode information instead of block usage
Manual page df(1) line 1 (press h for help or q to quit)
```

Рис. 4.19: Информация по команде `df`

```
du(1)                                User Commands                                du(1)

NAME
    du - estimate file space usage

SYNOPSIS
    du [OPTION]... [FILE]...
    du [OPTION]... --files-from FILE

DESCRIPTION
    Summarize device usage of the set of FILES, recursively for directories.
    Mandatory arguments to long options are mandatory for short options too.

    -0, --null
        end each output line with NUL, not newline

    -a, --all
        write counts for all files, not just directories

    --apparent-size
        print apparent sizes rather than device usage; although the apparent size is usually smaller, it may be larger due to holes in ("sparse" files, internal fragmentation, indirect blocks, and the like)

    -B, --block-size=SIZE
        scale sizes by SIZE before printing them; e.g., "-BM" prints sizes in units of 1,048,576 bytes; see SIZE format below

    -b, --bytes
        equivalent to "--apparent-size --block-size=1"

    -c, --total
        produce a grand total

    -D, --dereference-args
        dereference only symlinks that are listed on the command line

    -d, --max-depth=NUM
        Manual page du(1) line 1 (press h for help or q to quit)
```

Рис. 4.20: Информация по команде du

```
dskochina@dk8n80 ~ $ df
Файловая система  1K-блоков  Использовано  Доступно  Использовано%  Смонтировано в
none              3999704      16796      3982908           1% /run
udev              10240         0         10240           0% /dev
tmpfs             3999704         0      3999704           0% /dev/shm
/dev/sda8         484939832    74261532    385971244        17% /
tmpfs             3999708      79696      3920012           2% /tmp
/dev/sda6         50090536     13788     47499852          1% /var/cache/openafs
AFS               2147483647    0  2147483647           0% /afs
tmpfs             799940       200       799740           1% /run/user/4662
dskochina@dk8n80 ~ $
```

Рис. 4.21: Команда df в терминале

```
4      ./blog/.git/modules/public/objects/20
3      ./blog/.git/modules/public/objects/e3
3      ./blog/.git/modules/public/objects/78
3      ./blog/.git/modules/public/objects/70
3      ./blog/.git/modules/public/objects/fb
3      ./blog/.git/modules/public/objects/ae
3      ./blog/.git/modules/public/objects/d7
3      ./blog/.git/modules/public/objects/25
3      ./blog/.git/modules/public/objects/30
3      ./blog/.git/modules/public/objects/bf
3      ./blog/.git/modules/public/objects/a8
3      ./blog/.git/modules/public/objects/6c
3      ./blog/.git/modules/public/objects/db
7714   ./blog/.git/modules/public/objects
4      ./blog/.git/modules/public/logs/refs/remotes/origin
6      ./blog/.git/modules/public/logs/refs/remotes
3      ./blog/.git/modules/public/logs/refs/heads
11     ./blog/.git/modules/public/logs/refs
14     ./blog/.git/modules/public/logs
7802   ./blog/.git/modules/public
7804   ./blog/.git/modules
```

Рис. 4.22: Команда du в терминале

12. Вывела имена всех директорий, имеющихсх в моем домашнем каталоге с



помощью команды «find~ -typed», предварительно получив информацию с помощью команды «man find». (рис. [4.23], [4.24], [4.25], [4.26])

```
dskochina@dk8n80 ~ $ man find

[6]+  Остановлен      man find
dskochina@dk8n80 ~ $ █
```

Рис. 4.23: Команда man find

```
FIND(1)                                General Commands Manual                                FIND(1)

NAME
  find - search for files in a directory hierarchy

SYNOPSIS
  find [-H] [-L] [-P] [-D debugopts] [-Olevel] [starting-point...] [expression]

DESCRIPTION
  This manual page documents the GNU version of find. GNU find searches the directory tree rooted at each given starting-point by evaluating the given expression from left to right, according to the rules of precedence (see section OPERATORS), until the outcome is known (the left hand side is false for and operations, true for or), at which point find moves on to the next file name. If no starting-point is specified, '.' is assumed.

  If you are using find in an environment where security is important (for example if you are using it to search directories that are writable by other users), you should read the 'Security Considerations' chapter of the findutils documentation, which is called finding files and comes with findutils. That document also includes a lot more detail and discussion than this manual page, so you may find it a more useful source of information.

OPTIONS
  The -H, -L, and -P options control the treatment of symbolic links. Command-line arguments following these are taken to be names of files or directories to be examined, up to the first argument that begins with '-', or the argument '-' or '!'. That argument and any following arguments are taken to be the expression describing what is to be searched for. If no paths are given, the current directory is used. If no expression is given, the expression -print is used (but you should probably consider using -prints instead, anyway).

  This manual page talks about 'options' within the expression list. These options control the behaviour of find but are specified immediately after the last path name. The five 'real' options -H, -L, -P, -D and -O must appear before the first path name, if at all. A double dash -- could theoretically be used to signal that any remaining arguments are not options, but this does not really work due to the way find determines the end of the following path arguments: it does that by reading until an expression argument comes (which also starts with a '-'). Now, if a path argument would start with a '-', then find would treat it as expression argument instead. Thus, to ensure that all start points are taken as such, and especially to prevent that wildcard patterns expanded by the calling shell are not mistakenly treated as expression arguments, it is generally safer to prefix wildcards or dubious path names with either './' or to use absolute path names starting with '/'. Alternatively, it is generally safe though non-portable to use the GNU option -filesep-from to pass arbitrary starting points to find.

  -P      Never follow symbolic links. This is the default behaviour. When find examines or prints information about files, and the file is a symbolic link, the information used shall be taken from the properties of the symbolic link itself.

  -L      Follow symbolic links. When find examines or prints information about files, the information used shall be taken from the properties of the file to which the link points, not from the link itself (unless it is a broken symbolic link or find is unable to examine the file to which the link points). Use of this option implies -noleaf. If you later use the -P option, -noleaf will still be in effect. If -L is in effect and find discovers a symbolic link to a subdirectory during its search, the

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

Рис. 4.24: Информация по команде find

```
dskochina@dk8n80 ~ $ find ~ -type d █
```

Рис. 4.25: Команда «find~ -typed»

```
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/d8
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/94
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/5a
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/ad
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/40
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/53
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/20
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/e3
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/78
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/70
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/fb
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/ae
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/d7
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/25
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/30
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/bf
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/a8
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/6c
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/objects/db
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/logs
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/logs/refs
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/logs/refs/remotes
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/logs/refs/remotes/origin
/afs/.dk.sci.pfu.edu.ru/home/d/s/dskochina/blog/.git/modules/public/logs/refs/heads
```

Рис. 4.26: Имена всех директорий

## Ответы на контрольные вопросы

1. В системе по умолчанию открыто три специальных потока:

–stdin – стандартный поток ввода (по умолчанию: клавиатура), файловый дескриптор 0;

–stdout – стандартный поток вывода (по умолчанию: консоль), файловый дескриптор 1;

–stderr – стандартный поток вывод сообщений об ошибках (по умолчанию: консоль), файловый дескриптор 2.

Большинство используемых в консоли команд и программ записывают результаты своей работы в стандартный поток вывода stdout.

2. ‘>’ Перенаправление вывода в файл

‘>>’ Перенаправление вывода в файл и открытие файла в режиме добавления (данные добавляются в конец файла)/

3. Конвейер (pipe) служит для объединения простых команд или утилит в цепочки, в которых результат работы предыдущей команды передаётся последующей.

Синтаксис следующий:

команда1|команда2 (это означает, что вывод команды 1 передастся на ввод команде 2)

4. Процесс рассматривается операционной системой как заявка на потребление всех видов ресурсов, кроме одного – процессорного времени. Этот последний важнейший ресурс распределяется операционной системой между другими единицами работы – потоками, которые и получили свое название благодаря тому, что они представляют собой последовательности (потоки выполнения) команд.

Процесс – это выполнение программы. Он считается активной сущностью и реализует действия, указанные в программе.

Программа представляет собой статический набор команд, а процесс это набор ресурсов и данных, использующихся при выполнении программы.

5. `pid`: идентификатор процесса (PID) процесса (`processID`), к которому вызывают метод

`gid`: идентификатор группы UNIX, в котором работает программа.

6. Любую выполняющуюся в консоли команду или внешнюю программу можно запустить в фоновом режиме. Для этого следует в конце имени команды указать знак амперсанда `&`.

Запущенные фоном программы называются задачами (`jobs`). Ими можно управлять с помощью команды `jobs`, которая выводит список запущенных в данный момент задач.

7. `top` – это консольная программа, которая показывает список работающих процессов в системе. Программа в реальном времени отсортирует запущенные процессы по их нагрузке на процессор.

htop – это продвинутый консольный мониторинг процессов. Утилита выводит постоянно меняющийся список системных процессов, который сортируется в зависимости от нагрузки на ЦПУ. Если делать сравнение с top, то htop показывает абсолютно все процессы в системе, время их непрерывного использования, загрузку процессоров и расход оперативной памяти.

8. find – это команда для поиска файлов и каталогов на основе специальных условий. Ее можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и другим подобным критериям.

Команда find имеет такой синтаксис:

find[папка][параметры] критерий шаблон [действие]

Папка – каталог в котором будем искать

Параметры – дополнительные параметры, например, глубина поиска, и т.д.

Критерий – по какому критерию будем искать: имя, дата создания, права, владелец и т.д.

Шаблон – непосредственно значение по которому будем отбирать файлы.

Основные параметры:

-P никогда не открывать символические ссылки

-L - получает информацию о файлах по символическим ссылкам. Важно для дальнейшей обработки, чтобы обрабатывалась не ссылка, а сам файл.

-maxdepth - максимальная глубина поиска по подкаталогам, для поиска только в текущем каталоге установите 1.

-depth - искать сначала в текущем каталоге, а потом в подкаталогах

-mount искать файлы только в этой файловой системе.

-version - показать версию утилиты find

-print - выводить полные имена файлов

-typef - искать только файлы

-typed - поиск папки в Linux

Основные критерии:

- name - поиск файлов по имени
- perm - поиск файлов в Linux по режиму доступа
- user - поиск файлов по владельцу
- group - поиск по группе
- mtime - поиск по времени модификации файла
- atime - поиск файлов по дате последнего чтения
- nogroup - поиск файлов, не принадлежащих ни одной группе
- nouser - поиск файлов без владельцев
- newer - найти файлы новее чем указанный
- size - поиск файлов в Linux по их размеру

Примеры:

find~ -type d поиск директорий в домашнем каталоге

find~ -type f -name “.\*” поиск скрытых файлов в домашнем каталоге

9. Файл по его содержимому можно найти с помощью команды grep: «grep -r” слово/выражение, которое нужно найти”».
10. Утилита df, позволяет проанализировать свободное пространство на всех подключенных к системе разделах.
11. При выполнении команды du (без указания папки и опции) можно получить все файлы и папки текущей директории с их размерами. Для домашнего каталога: du ~/
12. Основные сигналы (каждый сигнал имеет свой номер), которые используются для завершения процесса:  
  
SIGINT–самый безобидный сигнал завершения, означает Interrupt. Он отправляется процессу, запущенному из терминала с помощью сочетания клавиш Ctrl+C. Процесс правильно завершает все свои действия и возвращает управление;

SIGQUIT–это еще один сигнал, который отправляется с помощью сочетания клавиш, программе, запущенной в терминале. Он сообщает ей что нужно завершиться и программа может выполнить корректное завершение или проигнорировать сигнал. В отличие от предыдущего, она генерирует дампы памяти. Сочетание клавиш Ctrl+Q;

SIGHUP–сообщает процессу, что соединение с управляющим терминалом разорвано, отправляется, в основном, системой при разрыве соединения с интернетом;

SIGTERM–немедленно завершает процесс, но обрабатывается программой, поэтому позволяет ей завершить дочерние процессы и освободить все ресурсы;

SIGKILL–тоже немедленно завершает процесс, но, в отличие от предыдущего варианта, он не передается самому процессу, а обрабатывается ядром. Поэтому ресурсы и дочерние процессы остаются запущенными.

Также для передачи сигналов процессам в Linux используется утилита kill, её синтаксис: kill [-сигнал] [pid\_процесса] (PID – уникальный идентификатор процесса). Сигнал представляет собой один из выше перечисленных сигналов для завершения процесса.

Перед тем, как выполнить остановку процесса, нужно определить его PID. Для этого используют команды ps и grep. Команда ps предназначена для вывода списка активных процессов в системе и информации о них. Команда grep запускается одновременно с ps (в канале) и будет выполнять поиск по результатам команды ps.

Утилита pkill – это оболочка для kill, она ведет себя точно так же, и имеет тот же синтаксис, только в качестве идентификатора процесса ей нужно передать его имя.

kill all работает аналогично двум предыдущим утилитам. Она тоже принимает имя процесса в качестве параметра и ищет его PID в директории /proc. Но эта

утилита обнаружит все процессы с таким именем и завершит их.

## 5 Выводы

В ходе выполнения данной лабораторной работы я ознакомилась с инструментами поиска файлов и фильтрации текстовых данных. А также приобрела практические навыки по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.