

# Отчет по лабораторной работе №7

## Дисциплина: Операционные системы

Калистратова Ксения Евгеньевна

### Содержание

Цель работы .....	1
Задачи.....	1
Выполнение лабораторной работы .....	1
Контрольные вопросы .....	9
Выводы .....	13

### Цель работы

Ознакомление с инструментами поиска файлов и фильтрации текстовых данных. Приобретение практических навыков: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

### Задачи

1. Изучить потоки ввода и вывода.
2. Изучить конвейер.
3. Изучить команду поиска файлов.
4. Изучить команду, позволяющую найти указанную строку символов.
5. Изучить команды по проверке использования диска.
6. В ходе работы использовать эти команды и интерпретировать их вывод.
7. Выполнить отчет.

### Выполнение лабораторной работы

- 1) Осуществляю вход в систему, используя свои логин и пароль.
- 2) Для того, чтобы записать в файл file.txt названия файлов, содержащихся в каталоге /etc, использую команду «ls-a/etc> file.txt». Далее с помощью команды «ls-a~ >> file.txt» дописываю в этот же файл названия файлов, содержащихся в моем домашнем каталоге. Командой «catfile.txt» просматриваю файл, чтобы убедиться в правильности действий. (рис. 1) (рис. 2)

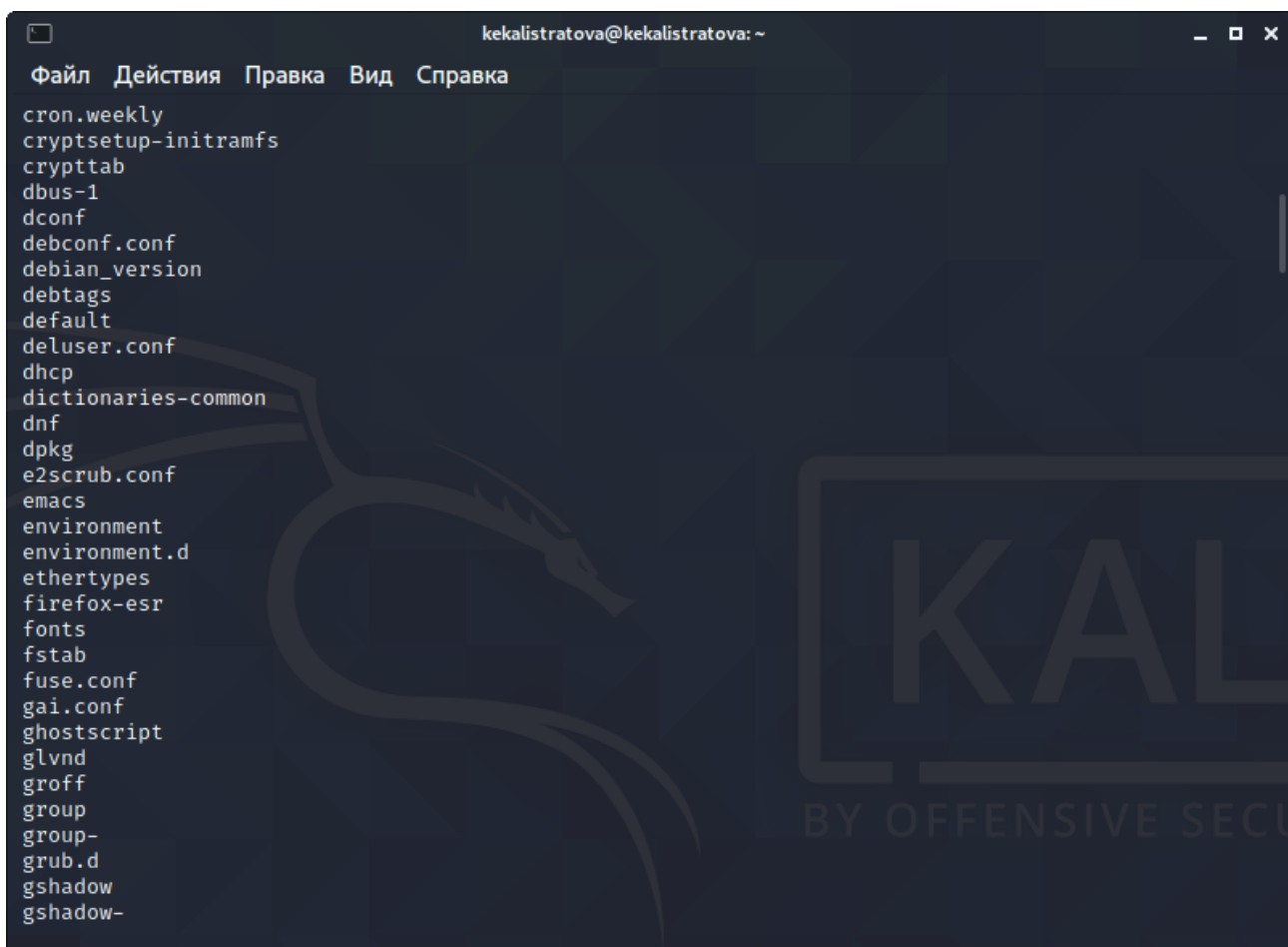


Figure 1: Файл file.txt

```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~/Рабочий стол]  
$ grep -e '\.conf$' file.txt > conf.txt  
  
(kekalistratova@kekalistratova)-[~/Рабочий стол]  
$ cat conf.txt  
adduser.conf  
ca-certificates.conf  
debconf.conf  
deluser.conf  
e2scrub.conf  
fuse.conf  
gai.conf  
host.conf  
ipsec.conf  
kernel-img.conf  
ld.so.conf  
libaudit.conf  
logrotate.conf  
mke2fs.conf  
nftables.conf  
nsswitch.conf  
pam.conf  
resolv.conf  
rsyslog.conf  
sensors3.conf  
strongswan.conf  
sudo.conf  
sudo_logsrvd.conf  
sysctl.conf  
ucf.conf  
updatedb.conf
```

Figure 2: Файл file.txt

- 3) Вывожу имена всех файлов из file.txt, имеющих расширение .conf и записываю их в новый текстовый файл conf.txt с помощью команды «grep -e '\.conf\$' file.txt > conf.txt». Командой «cat conf.txt» проверяю правильность выполненных действий. (рис. 3)

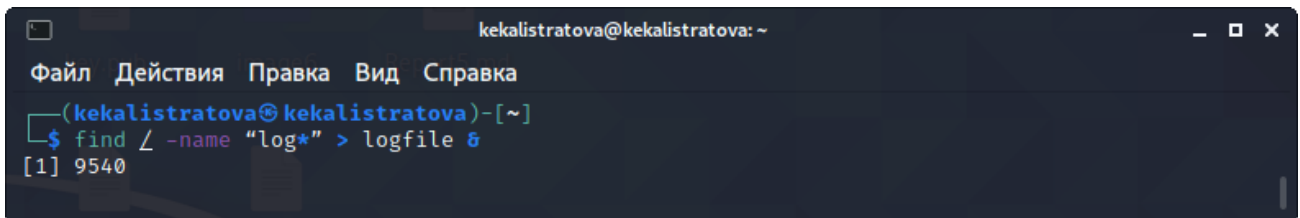
```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ ls ~ /c*  
ls: невозможно получить доступ к '/c*': Нет такого файла или каталога  
/home/kekalistratova:  
abc1  australia  may      my_os      ski.plases  Документы  Музыка  Шаблоны  
abc11 conf.txt   monthly  play       sudo_plugin.h  Загрузки  Общедоступные  
abcl  feather   morefun1 reports    Видео        Изображения  'Рабочий стол'
```

Figure 3: Расширение .conf

- 4) Определить, какие файлы в моем домашнем каталоге имеют имена, начинающиеся с символа c, можно несколькими командами: «find ~ -maxdepth 1 -name "c" -print» (опция maxdepth 1 необходима для того, чтобы файлы находились только в домашнем каталоге (не в его подкаталогах)), «ls ~/c» и «ls -a | grep c\*». (рис. 4)



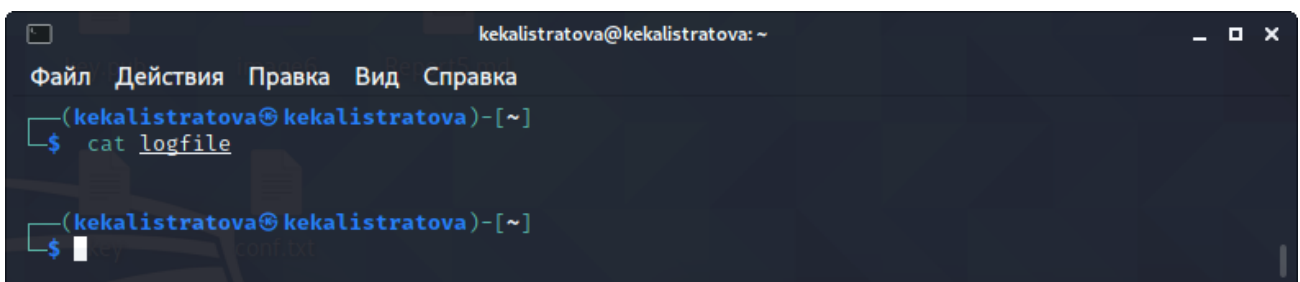
- 6) Запускаю в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log, используя команду «find/ -name "log\*" > logfile &». (рис. 7)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ find / -name "log*" > logfile &  
[1] 9540
```

Figure 7: Файлы, имена которых начинаются с log.

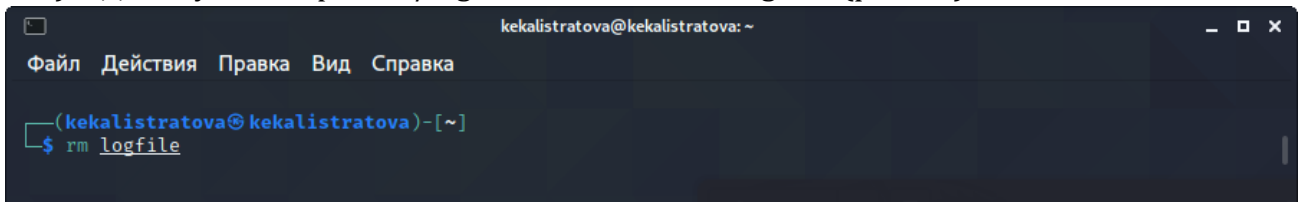
Командой «cat logfile» проверяю выполненные действия. (рис. 8) (рис. 9)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ cat logfile  
  
(kekalistratova@kekalistratova)-[~]  
$
```

Figure 8: cat logfile

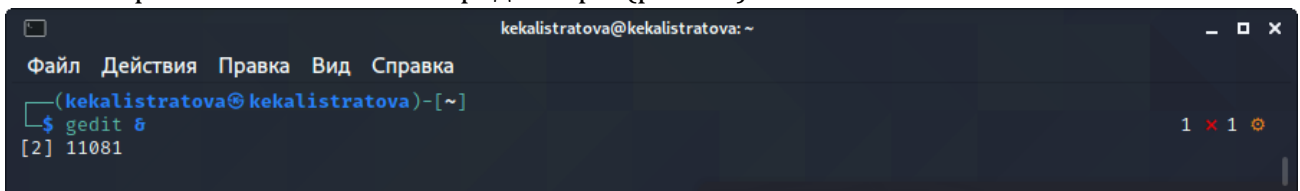
- 7) Далее удаляю файл ~/logfile командой «rm logfile». (рис. 10)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ rm logfile
```

Figure 10: Удаление файла

- 8) Запускаю редактор gedit в фоновом режиме командой «gedit &». После этого на экране появляется окно редактора. (рис. 11)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ gedit &  
[2] 11081
```

Figure 11: Редактор gedit

- 9) Чтобы определить идентификатор процесса gedit, использую команду «ps | grep -i "gedit"». Из рисунка видно, что наш процесс имеет PID 7018. Узнать идентификатор процесса можно также, используя команду «pgrep gedit» или «pidof gedit». (рис. 12)

```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ ps | grep -i "gedit" 1  
(kekalistratova@kekalistratova)-[~]  
$ 1 x 1
```

Figure 12: Идентификатор процесса gedit

- 10) Прочитав информацию о команде kill с помощью команды «man kill», используя её для завершения процесса gedit(команда «kill 11457»). (рис. 13) (рис. 14) (рис. 15)

Figure 13: man kill

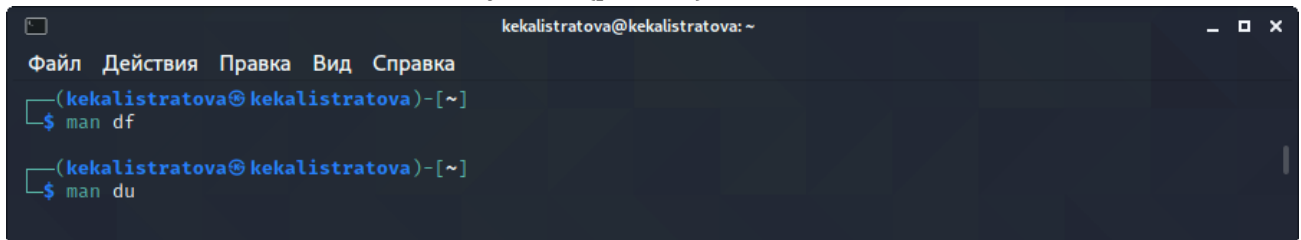
```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
KILL(1) User Commands KILL(1)  
NAME  
kill - send a signal to a process  
SYNOPSIS  
kill [options] <pid> [ ... ]  
DESCRIPTION  
The default signal for kill is TERM. Use -l or -L to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -9, -SIGKILL 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.  
-l, --list [signal]  
List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.  
Manual page kill(1) line 1 (press h for help or q to quit)
```

Figure 14: Справка команды kill

```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)-[~]  
$ kill 11457 1  
(kekalistratova@kekalistratova)-[~]  
$ 1  
[1] + terminated gedit  
(kekalistratova@kekalistratova)-[~]  
$
```

Figure 15: Завершение процесса gedit

- 11) С помощью команд «man df»и «man du» узнаю информацию по необходимым командам и далее использую их. (рис. 16)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)~  
$ man df  
(kekalistratova@kekalistratova)~  
$ man du
```

Figure 16: Команда man df

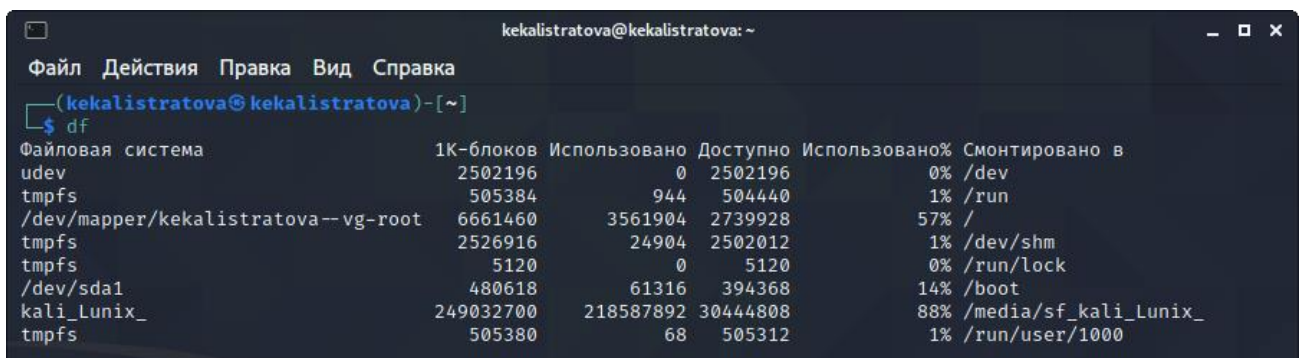
df – утилита,показывающаясписок всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования. (рис. 17)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
DF(1) User Commands DF(1)  
NAME  
df - report file system disk space usage  
SYNOPSIS  
df [OPTION] ... [FILE] ...  
DESCRIPTION  
This manual page documents the GNU version of df. df displays the amount of disk 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. Disk 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 disk 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.  
Manual page df(1) line 1 (press h for help or q to quit)
```

Figure 17: Команда man

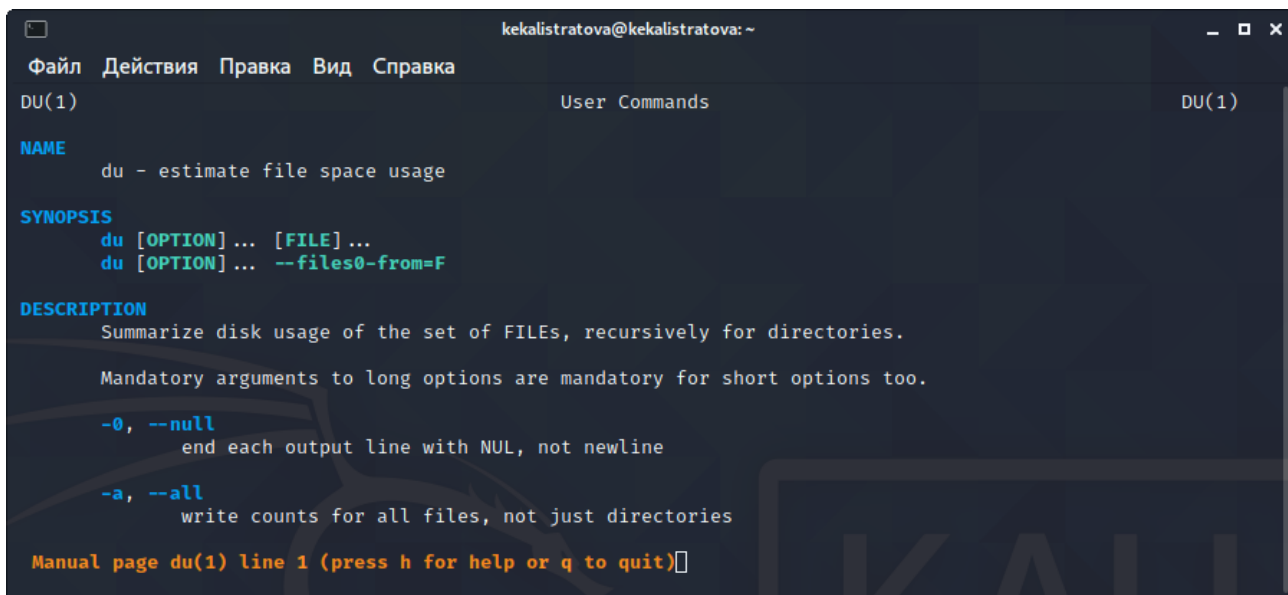
Синтаксис: df[опции]устройство. (рис. 18)



```
kekalistratova@kekalistratova: ~  
Файл Действия Правка Вид Справка  
(kekalistratova@kekalistratova)~  
$ df  
Файловая система 1K-блоков Использовано Доступно Использовано% Смонтировано в  
udev 2502196 0 2502196 0% /dev  
tmpfs 505384 944 504440 1% /run  
/dev/mapper/kekalistratova--vg-root 6661460 3561904 2739928 57% /  
tmpfs 2526916 24904 2502012 1% /dev/shm  
tmpfs 5120 0 5120 0% /run/lock  
/dev/sda1 480618 61316 394368 14% /boot  
kali_Linux_ 249032700 218587892 30444808 88% /media/sf_kali_Linux_  
tmpfs 505380 68 505312 1% /run/user/1000
```

Figure 18: Команда df

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. (рис. 19)



```
kekalistratova@kekalistratova: ~
Файл Действия Правка Вид Справка
DU(1) User Commands DU(1)
NAME
    du - estimate file space usage
SYNOPSIS
    du [OPTION]... [FILE]...
    du [OPTION]... --files0-from=F
DESCRIPTION
    Summarize disk 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
Manual page du(1) line 1 (press h for help or q to quit)
```

Figure 19: Команда `man du`

Синтаксис: `du [опции] каталог_или_файл`. (рис. 20)

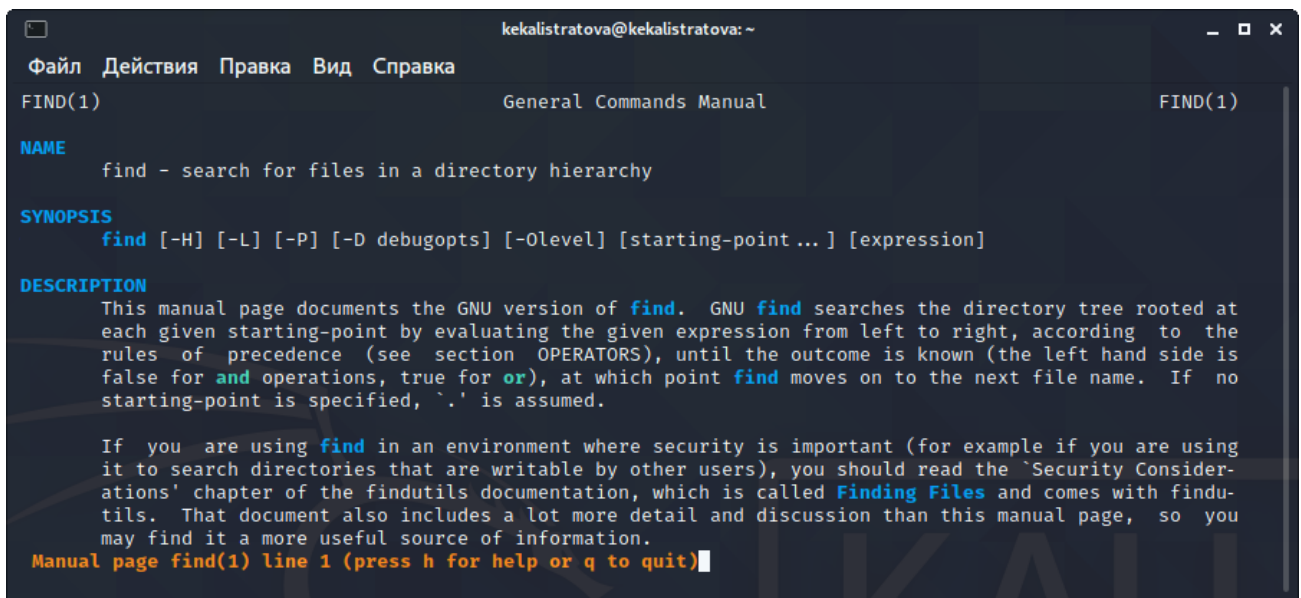


```
(kekalistratova@kekalistratova)-[~]
$ du
4    ./Документы
4    ./monthly
4    ./play/games/fun
8    ./play/games
12   ./play
500  ./rpmdb
4    ./gnupg/private-keys-v1.d
12   ./gnupg
4    ./australia
4    ./Видео
4    ./reports/monthly/monthly
8    ./reports/monthly
12   ./reports
16   ./ssh
4    ./Шаблоны
4    ./ski.places/equipment
4    ./ski.places/plans
12   ./ski.places
60   ./cache/fontconfig
4    ./cache/sessions
796  ./cache/gstreamer-1.0
```

Figure 20: Команда `du`

- 12) Вывожу имена всех директорий, имеющихсх в моем домашнем каталоге с помощью команды «find ~ -type d», предварительно получив информацию с помощью команды «man find». (рис. 21) (рис. 22)





```
kekalistratova@kekalistratova: ~
Файл Действия Правка Вид Справка
FIND(1) General Commands Manual FIND(1)

NAME
  find - search for files in a directory hierarchy

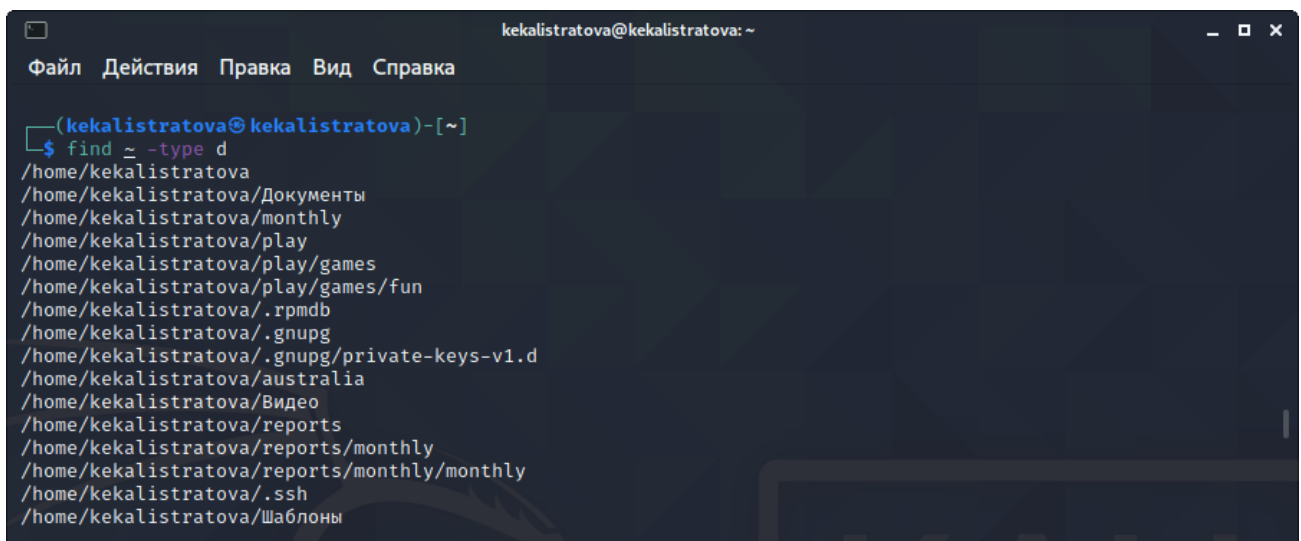
SYNOPSIS
  find [-H] [-L] [-P] [-D debugopts] [-Olevel] [starting-point ...] [expression]

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.

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

Figure 21: Справка команды *find*



```
kekalistratova@kekalistratova: ~
Файл Действия Правка Вид Справка

(kekalistratova@kekalistratova)-[~]
$ find ~ -type d
/home/kekalistratova
/home/kekalistratova/Документы
/home/kekalistratova/monthly
/home/kekalistratova/play
/home/kekalistratova/play/games
/home/kekalistratova/play/games/fun
/home/kekalistratova/.rpmdb
/home/kekalistratova/.gnupg
/home/kekalistratova/.gnupg/private-keys-v1.d
/home/kekalistratova/australia
/home/kekalistratova/Видео
/home/kekalistratova/reports
/home/kekalistratova/reports/monthly
/home/kekalistratova/reports/monthly/monthly
/home/kekalistratova/.ssh
/home/kekalistratova/Шаблоны
```

Figure 22: Команда *find*

## 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 передается последующей.

Синтаксис следующий:

команда1|команда2 (это означает, что вывод команды 1 передается 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+;/
  - 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 `kill` – 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.

`killall` 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.

## Выводы

В ходе выполнения данной лабораторной работы я изучила инструменты поиска файлов и фильтрации текстовых данных, а также приобрела практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.