

# **Отчёт по лабораторной работе 6**

**Дисциплина: Операционные системы**

Волчок Кристина Александровна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Контрольные вопросы</b>	<b>16</b>
<b>5</b>	<b>Выводы</b>	<b>21</b>
	<b>Список литературы</b>	<b>22</b>

## Список иллюстраций

3.1	Запись в файл file.txt названия файлов, содержащихся в каталоге /etc	7
3.2	содержание . . . . .	8
3.3	Выводим имена всех файлов из file.txt, имеющих расширение .conf	8
3.4	Сами файлы . . . . .	9
3.5	Имена файлов начинающихся с символа с . . . . .	9
3.6	Имена файлов начинающихся с символа h в каталоге etc . . . . .	10
3.7	Имена файлов начинающихся с символа h . . . . .	10
3.8	Запуск в фоновом режиме . . . . .	11
3.9	Удаление . . . . .	11
3.10	Запуск в фоновом режиме . . . . .	11
3.11	Идентификатор процесса gedit . . . . .	12
3.12	Идентификатор процесса gedit . . . . .	12
3.13	Справка man kill . . . . .	13
3.14	kill . . . . .	13
3.15	Команды df,du . . . . .	13
3.16	Команда df . . . . .	14
3.17	Команда du . . . . .	15
3.18	Справка комнды find . . . . .	15

## Список таблиц

# 1 Цель работы

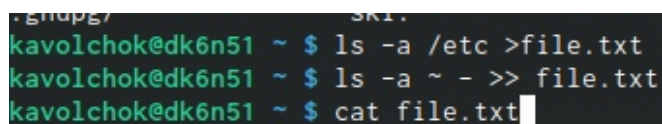
В ходе выполнения лабораторной работы я должна ознакомиться с инструментами поиска файлов и фильтрации текстовых данных. Приобрести практические навыки по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

## 2 Задание

1. Запишем в файл `file.txt` названия файлов, содержащихся в каталоге `/etc`.  
Допишем в этот же файл названия файлов, содержащихся в домашнем каталоге.
2. Выведем имена всех файлов из `file.txt`, имеющих расширение `.conf`, после чего запишем их в новый текстовый файл `conf.txt`.
3. Определим, какие файлы в вашем домашнем каталоге имеют имена, начинавшиеся с символа `s`?
4. Выведем на экран (по странично) имена файлов из каталога `/etc`, начинающиеся с символа `h`.
5. Запустим в фоновом режиме процесс, который будет записывать в файл `~/logfile` файлы, имена которых начинаются с `log`.
6. Удалим файл `~/logfile`.
7. Запустим из консоли в фоновом режиме редактор `gedit`.
8. Определим идентификатор процесса `gedit`, используя команду `ps`, конвейер и фильтр `grep`.
9. Прочтем справку (`man`) команды `kill`, после чего используйте её для завершения процесса `gedit`.
10. Выполним команды `df` и `du`, предварительно получив более подробную информацию об этих командах, с помощью команды `man`.
11. Воспользовавшись справкой команды `find`, выведем имена всех директорий, имеющихся в домашнем каталоге.

### 3 Выполнение лабораторной работы

1. Запишем в файл file.txt названия файлов, содержащихся в каталоге /etc. Допишем в этот же файл названия файлов, содержащихся в домашнем каталоге.(рис. 3.1; 3.2)



```
kavolchok@dk6n51 ~ $ ls -a /etc >file.txt  
kavolchok@dk6n51 ~ $ ls -a ~ - >> file.txt  
kavolchok@dk6n51 ~ $ cat file.txt
```

Рис. 3.1: Запись в файл file.txt названия файлов, содержащихся в каталоге /etc

```
ntpd.conf
nvme
OGRE
omniorb
openafs
OpenGL
OpenGLid.ini
openldap
openmpi
opt
os-release
PackageKit
paludis
pam.d
pango
papersize
passwd
passwd-
pear.conf
pe-format2.conf
php
pipewire
pkcs11
pki
plymouth
pmount.allow
pmount.conf
polkit-1
portage
postgresql-10
postgresql-11
postgresql-12
postgresql-13
postgresql-9.4
povray
ppp
prelink.conf
prelink.conf.d
printcap
```

Рис. 3.2: содержание

2. Выведем имена всех файлов из file.txt, имеющих расширение .conf, после чего запишем их в новый текстовый файл conf.txt. Рис.3.3;3.4

```
Шаблоны
kavolchok@dk6n51 ~ $ grep -e '\.conf$' file.txt > conf.txt
kavolchok@dk6n51 ~ $ cat conf
```

Рис. 3.3: Выводим имена всех файлов из file.txt, имеющих расширение .conf



```
ldap.conf
ld.so.conf
libaudit.conf
lightdm.conf
locale.conf
logrotate.conf
mailutils.conf
make.conf
man.conf
man_db.conf
mdadm.conf
metalog.conf
mke2fs.conf
mlocate-cron.conf
modules.conf
mplayer.conf
nscd.conf
nslcd.conf
nss-ldapd.conf
nsswitch.conf
nsswitch-sss.conf
ntp.conf
ntpd.conf
pear.conf
pe-format2.conf
pmount.conf
prelink.conf
pump.conf
pwdb.conf
rc.conf
request-key.conf
resolv.conf
rsyncd.conf
rsyslog.conf
sandbox.conf
sddm.conf
sensors3.conf
signond.conf
smartd.conf
```

Рис. 3.4: Сами файлы

3. Определим, какие файлы в вашем домашнем каталоге имеют имена, начинавшиеся с символа с? Рис.3.5

```
kavolchok@dk6n51 ~ $ find -name "c*"
kavolchok@dk6n51 ~ $ find ~ -name "abc1" -print
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.local/share/Trash/files/abc1
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/abc1
kavolchok@dk6n51 ~ $ find ~ -name "c*" -print
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-active.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-backdrop-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-backdrop-active.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/assets/close-backdrop-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/gtk-3.0/colors.css
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/kdeconnect/certificate.pem
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/kdeconnect/config
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/pulse/cookie
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/libreoffice/4/user/config
/afs/.dk.sci.pfu.edu.ru/home/k/a/kavolchok/.config/libreoffice/4/user/extensions/share
```

Рис. 3.5: Имена файлов начинающихся с символа с

4. Выведем на экран (по странично) имена файлов из каталога /etc, начинаю-

щиеся с символа h.Рис.3.6;3.7

```
etc/logrotate.d/chrony  
avolchok@dk6n51 ~ $ find /etc -name "h*" | less
```

Рис. 3.6: Имена файлов начинающихся с символа h в каталоге etc

```
файл  права  вид  закладка  настройка  справка  
/etc/mercurial/hgrc.d  
/etc/httpd  
/etc/init.d/hdparm  
/etc/init.d/hotplug  
/etc/init.d/hsqldb  
/etc/init.d/hddtemp  
/etc/brltty/Text/he.ttb  
/etc/brltty/Text/hy.ttb  
/etc/brltty/Text/hr.ttb  
/etc/brltty/Text/hi.ttb  
/etc/brltty/Text/hu.ttb  
/etc/brltty/Contraction/ha.ctb  
/etc/brltty/Input/hd  
/etc/brltty/Input/hw  
/etc/brltty/Input/ht  
/etc/brltty/Input/hm  
/etc/brltty/Input/bm/horizontal.kti  
/etc/hotplug  
/etc/hsqldb  
/etc/runlevels/default/hdparm  
/etc/runlevels/boot/hostname  
/etc/runlevels/boot/hwclock  
/etc/harbour.cfg  
/etc/hotplug.d  
/etc/distcc/hosts  
/etc/host.conf  
/etc/avahi/hosts  
/etc/highlight  
find: '/etc/cron.hourly': Отказано в доступе  
find: '/etc/cups/certs': Отказано в доступе  
find: '/etc/cups/ssl': Отказано в доступе  
find: '/etc/munge': Отказано в доступе  
find: '/etc/fcron': Отказано в доступе  
find: '/etc/audit/plugins.d': Отказано в доступе  
find: '/etc/sudoers.d': Отказано в доступе  
find: '/etc/audit/plugins.d': Отказано в доступе  
find: '/etc/cron.weekly': Отказано в доступе  
find: '/etc/cron.monthly': Отказано в доступе  
lines 1-38/66 44%
```

Рис. 3.7: Имена файлов начинающихся с символа h

5. Запустим в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log.Рис.3.8

```

kavolchok@dk6n51 ~ $ find /var/log -name "log*" -print > logfile &
[1] 38310
kavolchok@dk6n51 ~ $ find: '/var/log/everything': Отказано в доступе
find: '/var/log/sandbox': Отказано в доступе
find: '/var/log/sshhd': Отказано в доступе
find: '/var/log/telnet': Отказано в доступе
find: '/var/log/kernel': Отказано в доступе
find: '/var/log/pwdfail': Отказано в доступе
find: '/var/log/mail': Отказано в доступе
find: '/var/log/portage': Отказано в доступе
find: '/var/log/mysql': Отказано в доступе
find: '/var/log/critical': Отказано в доступе
find: '/var/log/crond': Отказано в доступе
find: '/var/log/munge': Отказано в доступе
find: '/var/log/audit': Отказано в доступе
find: '/var/log/private': Отказано в доступе
find: '/var/log/hal': Отказано в доступе
find: '/var/log/cron': Отказано в доступе
find: '/var/log/apache2': Отказано в доступе
find: '/var/log/gdm': Отказано в доступе

[1]+  Выход 1          find /var/log -name "log*" -print > logfile
kavolchok@dk6n51 ~ $

```

Рис. 3.8: Запуск в фоновом режиме

6. Удалим файл ~/logfile.Рис.3.9

```

kavolchok@dk6n51 ~ $ rm logfile
kavolchok@dk6n51 ~ $ gedit &

```

Рис. 3.9: Удаление

7. Запустим из консоли в фоновом режиме редактор geditРис.3.10.

```

kavolchok@dk6n51 ~ $ gedit &
[1] 39494

```

Рис. 3.10: Запуск в фоновом режиме

8. Определим идентификатор процесса gedit, используя команду ps, конвейер и фильтр.Рис.3.11;3.12 grep.

```

kavolchok@dk6n51 ~ $ ps grep
error: list of process IDs must follow p

Usage:
ps [options]

Try 'ps --help <simple|list|output|threads|misc|all>'
or 'ps --help <s|l|o|t|m|a>'
for additional help text.

For more details see ps(1).
[1]+  Завершён          gedit
kavolchok@dk6n51 ~ $ ps | grep- i| "gredit"
bash: grep-: команда не найдена
bash: gredit: команда не найдена
kavolchok@dk6n51 ~ $ ps | grep gedit
kavolchok@dk6n51 ~ $ ps
      PID TTY          TIME CMD
    29716 pts/5        00:00:00 bash
    41277 pts/5        00:00:00 ps
kavolchok@dk6n51 ~ $ ps aux
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT ST
root           1  0.0  0.1 183616 11752 ?        Ss   13
root           2  0.0  0.0      0     0 ?        S    13
root           3  0.0  0.0      0     0 ?        I<   13
root           4  0.0  0.0      0     0 ?        I<   13
root           6  0.0  0.0      0     0 ?        I<   13
root           8  0.0  0.0      0     0 ?        I<   13
root           9  0.0  0.0      0     0 ?        S    13
root          10  0.0  0.0      0     0 ?        S    13
root          11  0.0  0.0      0     0 ?        S    13
root          12  0.1  0.0      0     0 ?        I    13
root          13  0.0  0.0      0     0 ?        S    13
root          14  0.0  0.0      0     0 ?        S    13
root          15  0.0  0.0      0     0 ?        S    13
root          16  0.0  0.0      0     0 ?        S    13
root          17  0.0  0.0      0     0 ?        S    13
root          19  0.0  0.0      0     0 ?        I<   13
root          20  0.0  0.0      0     0 ?        S    13

```

Рис. 3.11: Идентификатор процесса gedit

```

kavolchok@dk6n51 ~ $ ps aux | grep gedit
kavolch+  34991  0.4  1.7 847388 138800 ?        Sl   17:30   0:09 /usr/bin/gedit --ga
plication-service
kavolch+  41347  0.0  0.0 10156   916 pts/5    S+   18:07   0:00 grep --colour=auto
edit

```

Рис. 3.12: Идентификатор процесса gedit

9. Прочтем справку (man) команды kill, после чего используйте её для завершения процесса gedit. Рис.3.13;3.14

```

kavolchok@dk6n51 ~ $ man kill
kavolchok@dk6n51 ~ $ ps aux | grep gedit
kavolch+ 34991  0.5  1.8 923532 146816 ?        Sl   17:30   0:15 /usr/bin/gedit --ga
pplication-service
kavolch+ 43315  0.0  0.0 10156   920 pts/5    S+   18:20   0:00 grep --colour=auto
gedit
kavolchok@dk6n51 ~ $ kill 43315
bash: kill: (43315) - Нет такого процесса
kavolchok@dk6n51 ~ $ kill 34991
kavolchok@dk6n51 ~ $ man df

```

Рис. 3.13: Справка man kill

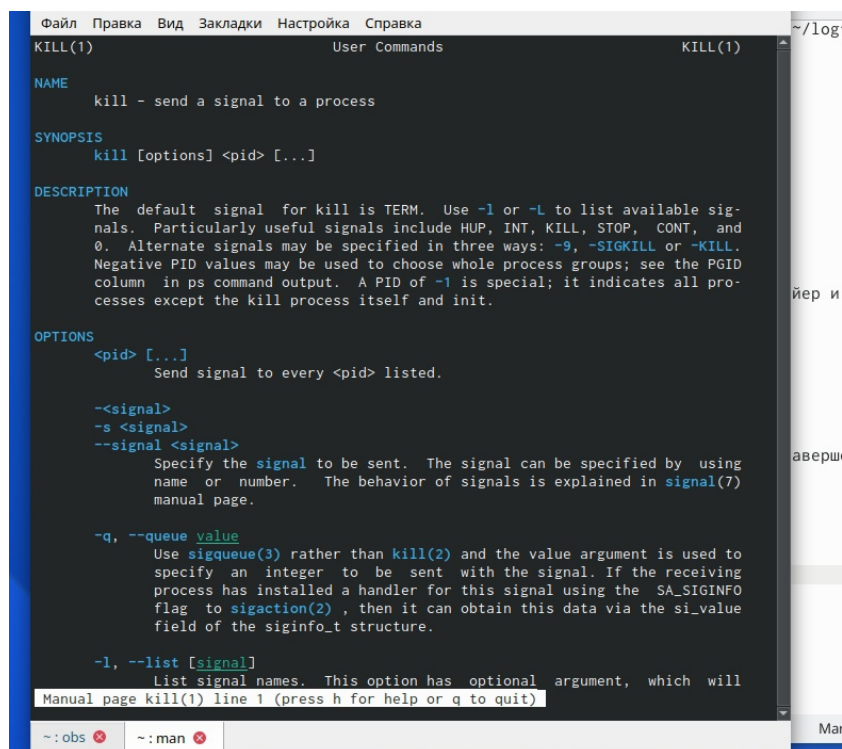


Рис. 3.14: kill

10. Выполним команды df и du, предварительно получив более подробную информацию об этих командах, с помощью команды man. Рис. 3.15; 3.16; 3.17

```

kavolchok@dk6n51 ~ $ man df
kavolchok@dk6n51 ~ $ man du

```

Рис. 3.15: Команды df, du



```
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 can-
not 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)

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

~: bash
```

Рис. 3.16: Команда df

```
Файл  Правка  Вид  Закладки  Настройка  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

    --apparent-size
        print apparent sizes, rather than disk 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

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

Рис. 3.17: Команда du

11. Воспользовавшись справкой команды find, выведем имена всех директорий, имеющих в домашнем каталоге.

```
kavolchok@dk6n51 ~ $ man df
kavolchok@dk6n51 ~ $ man du
```

Рис. 3.18: Справка команды find

## 4 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: команда 1 | команда 2 (это означает, что вывод команды 1 передаётся 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. Про-



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

- 5) `pid`: идентификатор процесса (PID) процесса (process ID), 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
  - -type f - искать только файлы
  - -type d - 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.
- 3.14 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.

## 5 Выводы

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

## **Список литературы**