

# **Отчёт по лабораторной работе №7**

**Дисциплина: Операционные системы**

Аветисян Давид Артурович

# 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>8</b>  |
| <b>4</b> | <b>Контрольные вопросы</b>            | <b>19</b> |
| <b>5</b> | <b>Выводы</b>                         | <b>24</b> |

## **Список таблиц**

## Список иллюстраций

|      |   |    |
|------|---|----|
| 3.1  | Записываем названия файлов, содержащихся в каталоге /etc . . .                          | 8  |
| 3.2  | Записываем названия файлов, содержащихся в домашнем каталоге                            | 9  |
| 3.3  | Просматриваем файл . . . . .  | 9  |
| 3.4  | Вывожу имена файлов, имеющих расширение .conf . . . . .                                 | 10 |
| 3.5  | Определим, какие файлы начинаются с символа с . . . . .                                 | 10 |
| 3.6  | Вывод на экран (постранично) файлы, начинающиеся с символа h                            | 11 |
| 3.7  | Запускаем в фоновом режиме процесс, который запишет файлы, начинающиеся с log . . . . . | 11 |
| 3.8  | Проверяем выполненные действия . . . . .  | 12 |
| 3.9  | Запускаю редактор gedit в фоновом режиме . . . . .                                      | 12 |
| 3.10 | Определяем идентификатор процесса gedit . . . . .                                       | 12 |
| 3.11 | Используем kill для завершения процесса gedit . . . . .                                 | 13 |
| 3.12 | Информация о команде kill . . . . .   | 13 |
| 3.13 | Информация о команде df . . . . .   | 14 |
| 3.14 | Информация о команде du . . . . .   | 15 |
| 3.15 | Используем df и du . . . . .  | 16 |
| 3.16 | Информация о команде find . . . . .   | 17 |
| 3.17 | Вывод имен всех директорий, имеющихся в домашнем каталоге .                             | 18 |

# 1 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.

## 2 Задание

1. Осуществите вход в систему, используя соответствующее имя пользователя.
2. Запишите в файл `file.txt` названия файлов, содержащихся в каталоге `/etc`. Допишите в этот же файл названия файлов, содержащихся в вашем домашнем каталоге.
3. Выведите имена всех файлов из `file.txt`, имеющих расширение `.conf`, после чего запишите их в новый текстовый файл `conf.txt`.
4. Определите, какие файлы в вашем домашнем каталоге имеют имена, начинающиеся с символа `s`? Предложите несколько вариантов, как это сделать.
5. Выведите на экран (постранично) имена файлов из каталога `/etc`, начинающиеся с символа `h`.
6. Запустите в фоновом режиме процесс, который будет записывать в файл `~/logfile` файлы, имена которых начинаются с `log`.
7. Удалите файл `~/logfile`.
8. Запустите из консоли в фоновом режиме редактор `gedit`.
9. Определите идентификатор процесса `gedit`, используя команду `ps`, конвейер и фильтр `grep`. Можно ли определить этот идентификатор более простым способом?
10. Прочтите справку (`man`) команды `kill`, после чего используйте её для завершения процесса `gedit`.
11. Выполните команды `df` и `du`, предварительно получив более подробную информацию об этих командах, с помощью команды `man`.
12. Воспользовавшись справкой команды `find`, выведите имена всех директо-

рий, имеющихся в вашем домашнем каталоге.

## 3 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» (рис. -fig. 3.1). Далее с помощью команды «ls -a ~ > file.txt» дописываю в этот же файл названия файлов, содержащихся в моем домашнем каталоге (рис. -fig. 3.2). Командой «cat file.txt» просматриваю файл, чтобы убедиться в правильности действий (рис. -fig. 3.3).

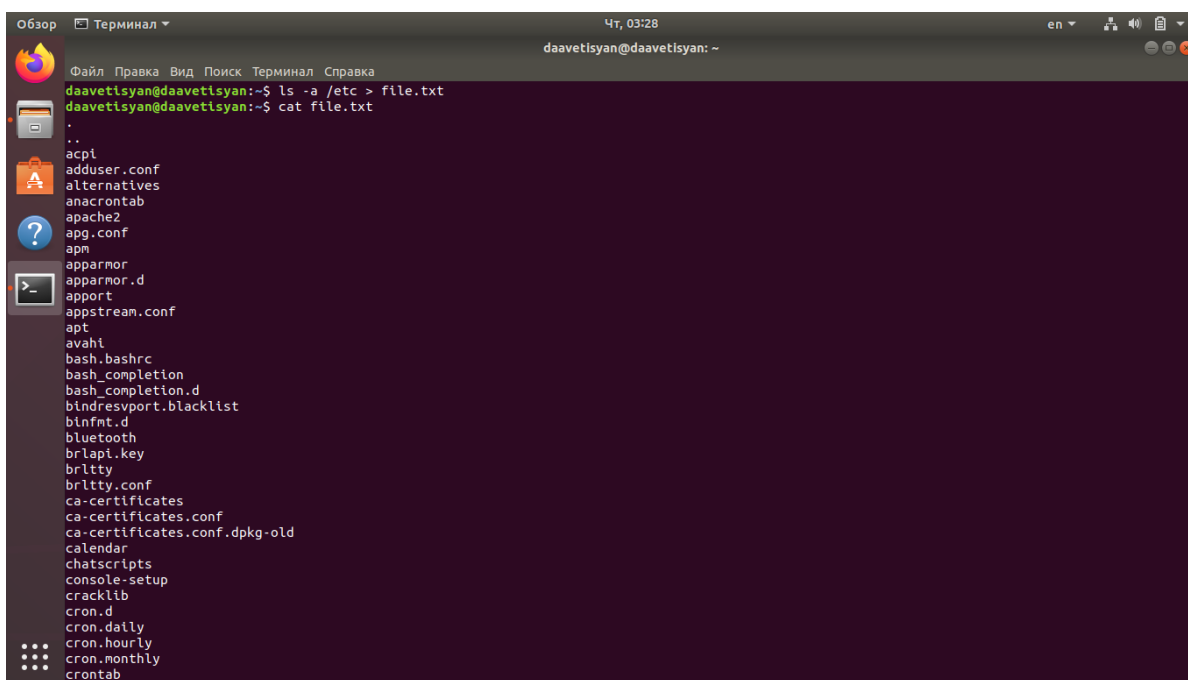


Рис. 3.1: Записываем названия файлов, содержащихся в каталоге /etc



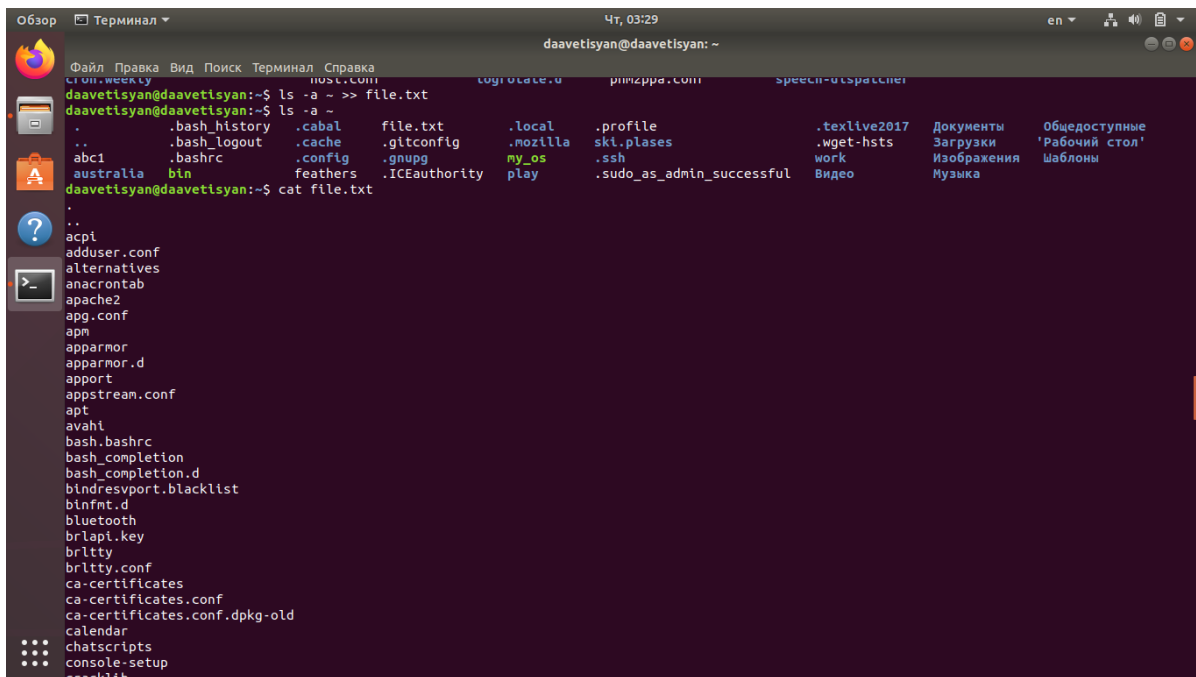


Рис. 3.2: Записываем названия файлов, содержащихся в домашнем каталоге

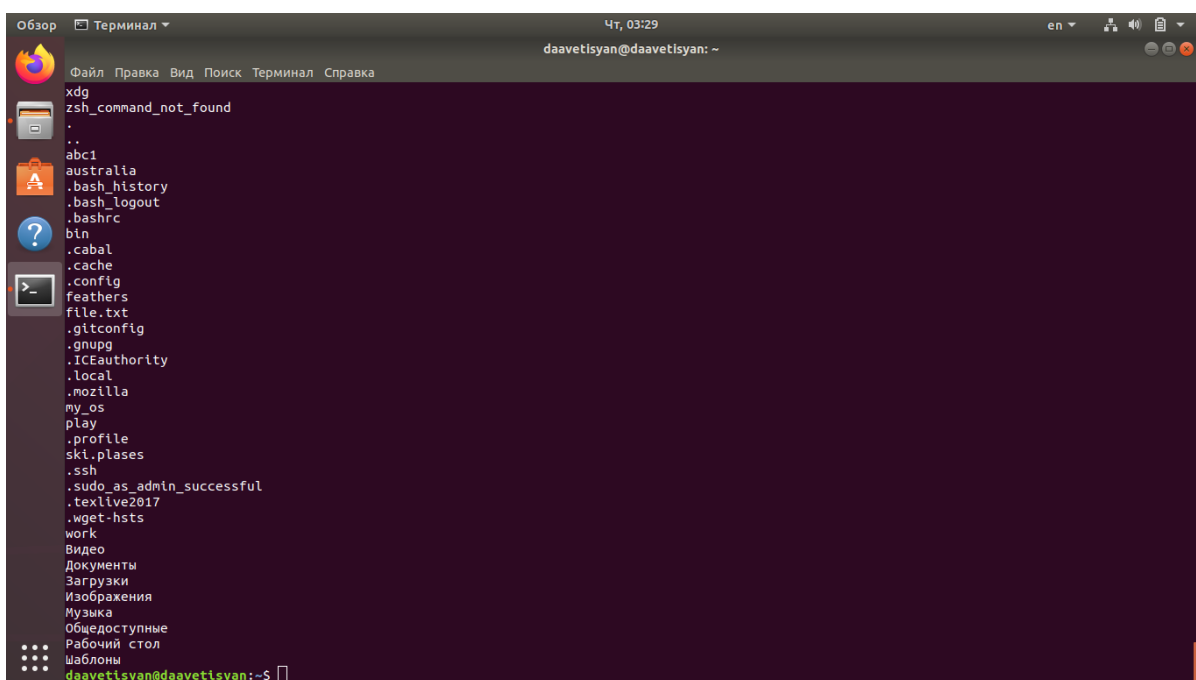
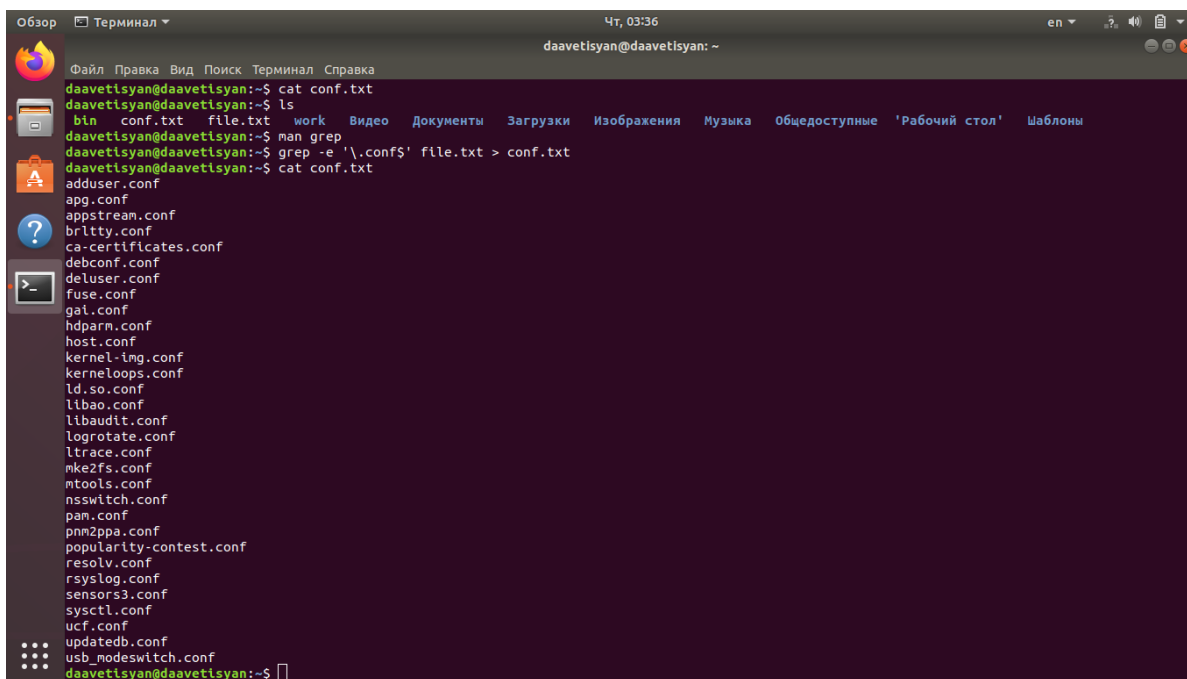


Рис. 3.3: Просматриваем файл

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

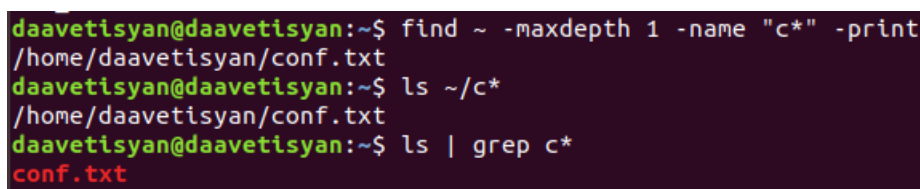
'conf\$' file.txt > conf.txt». Командой «cat conf.txt» проверяю правильность выполненных действий (рис. -fig. 3.4).



```
Обзор Терминал Чт, 03:36 en
daavetisyan@daavetisyan: ~
Файл Правка Вид Поиск Терминал Справка
daavetisyan@daavetisyan:~$ cat conf.txt
daavetisyan@daavetisyan:~$ ls
bin conf.txt file.txt work Видео Документы Загрузки Изображения Музыка Общедоступные 'Рабочий стол' Шаблоны
daavetisyan@daavetisyan:~$ man grep
daavetisyan@daavetisyan:~$ grep -e '\.conf$' file.txt > conf.txt
daavetisyan@daavetisyan:~$ cat conf.txt
adduser.conf
apg.conf
appstream.conf
brltty.conf
ca-certificates.conf
debconf.conf
deluser.conf
fuse.conf
gal.conf
hdparm.conf
host.conf
kernel-img.conf
kerneloops.conf
ld.so.conf
libao.conf
libaudit.conf
logrotate.conf
ltrace.conf
mke2fs.conf
mtools.conf
nsswitch.conf
pam.conf
pnm2ppa.conf
popularity-contest.conf
rsync.conf
rsyslog.conf
sensors3.conf
sysctl.conf
ucf.conf
updatedb.conf
usb_modeswitch.conf
daavetisyan@daavetisyan:~$
```

Рис. 3.4: Вывожу имена файлов, имеющих расширение .conf

4. Определить, какие файлы в моем домашнем каталоге имеют имена, начинающиеся с символа с, можно несколькими командами: «find ~ -maxdepth 1 -name "с" -print» (опция *maxdepth 1* необходима для того, чтобы файлы находились только в домашнем каталоге (не в его подкаталогах)), «ls ~/с» и «ls -a ~ | grep с\*» (рис. -fig. 3.5).



```
daavetisyan@daavetisyan:~$ find ~ -maxdepth 1 -name "с*" -print
/home/daavetisyan/conf.txt
daavetisyan@daavetisyan:~$ ls ~/с*
/home/daavetisyan/conf.txt
daavetisyan@daavetisyan:~$ ls | grep с*
conf.txt
```

Рис. 3.5: Определим, какие файлы начинаются с символа с

5. Чтобы вывести на экран (постранично) имена файлов из каталога /etc, начинающиеся с символа h, воспользуемся командой «find /etc -maxdepth 1

–name “h\*” | less» (рис. -fig. 3.6).

```
/etc/host.conf
/etc/hostname
/etc/hosts.deny
/etc/hp
/etc/hosts.allow
/etc/hosts
/etc/hdparm.conf
~
~
~
```

Рис. 3.6: Вывод на экран (постранично) файлы, начинающиеся с символа h

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. -fig. 3.7). Командой «cat logfile» проверяю выполненные действия (рис. -fig. 3.8).
7. Удаляю файл ~/logfile командой «rm logfile».

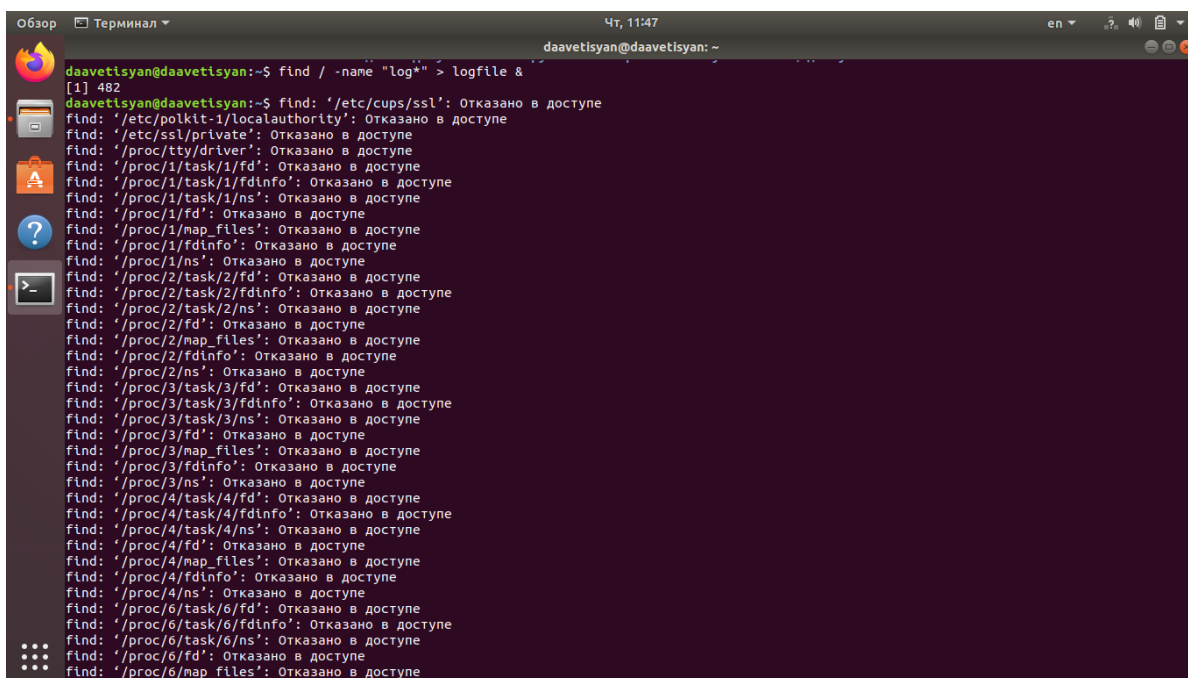


Рис. 3.7: Запускаем в фоновом режиме процесс, который запишет файлы, начинающиеся с log

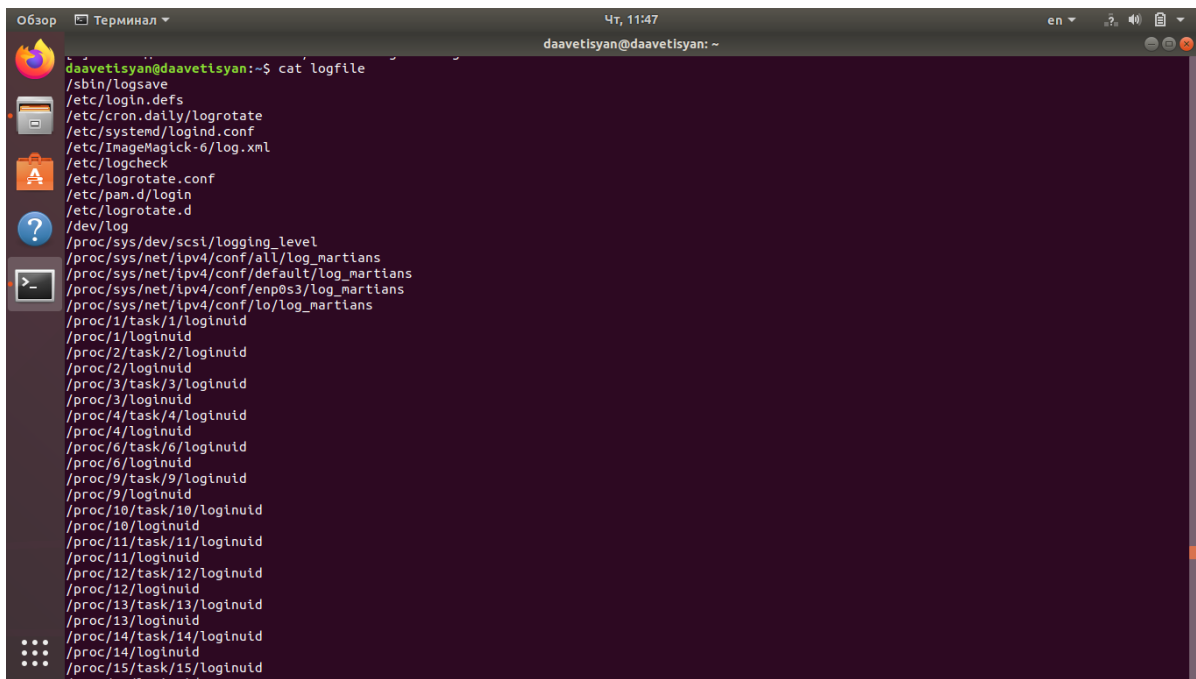


Рис. 3.8: Проверяем выполненные действия

8. Запускаю редактор gedit в фоновом режиме командой «gedit &» (рис. -fig. 3.9). После этого на экране появляется окно редактора.

```
daavetisyan@daavetisyan:~$ gedit &
[1] 518
```

Рис. 3.9: Запускаю редактор gedit в фоновом режиме

9. Чтобы определить идентификатор процесса gedit, использую команду «ps | grep -i "gedit"». Наш процесс имеет PID 518. Узнать идентификатор процесса можно также, используя команду «pgrep gedit» или «pidof gedit» (рис. -fig. 3.10).

```
daavetisyan@daavetisyan:~$ ps | grep -i "gedit"
518 pts/0    00:00:00 gedit
daavetisyan@daavetisyan:~$ pgrep gedit
518
daavetisyan@daavetisyan:~$ pidof gedit
518
```

Рис. 3.10: Определяем идентификатор процесса gedit

10. Прочитав информацию о команде kill с помощью команды «man kill», используя её для завершения процесса gedit (команда «kill 518») (рис. -fig. 3.11) (рис. -fig. 3.12).

```
daavetisyan@daavetisyan:~$ man kill
daavetisyan@daavetisyan:~$ kill 518
daavetisyan@daavetisyan:~$ man df
[1]+  Завершено      gedit
daavetisyan@daavetisyan:~$ man du
```

Рис. 3.11: Используем kill для завершения процесса gedit

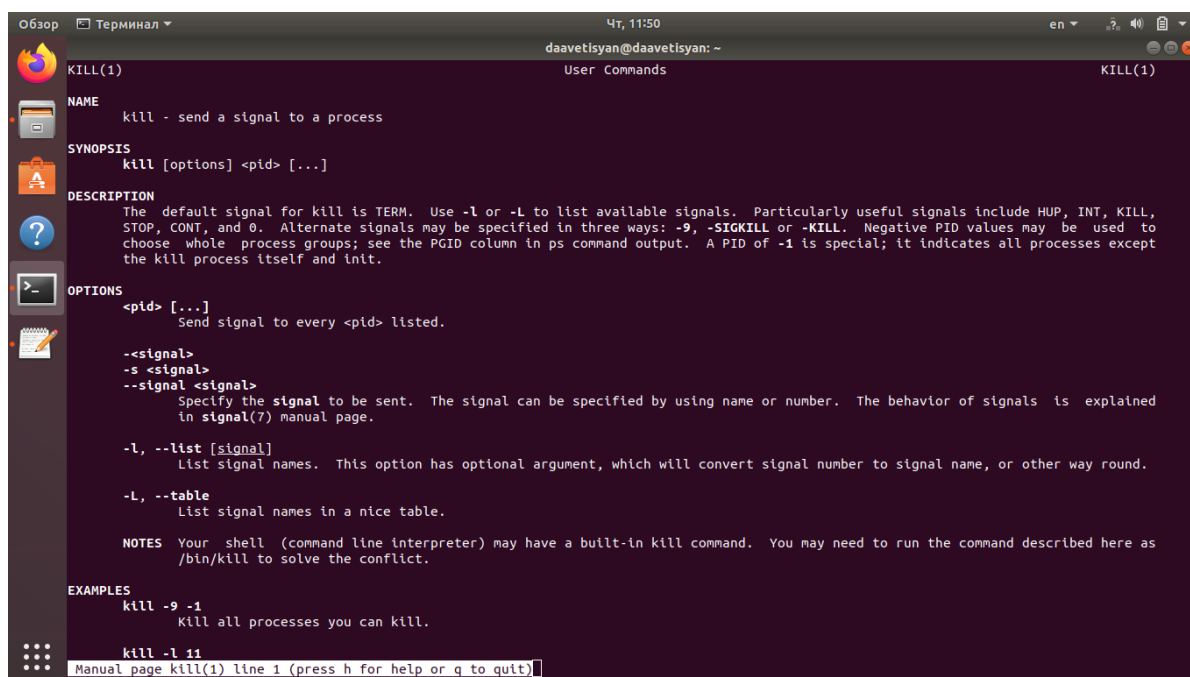


Рис. 3.12: Информация о команде kill

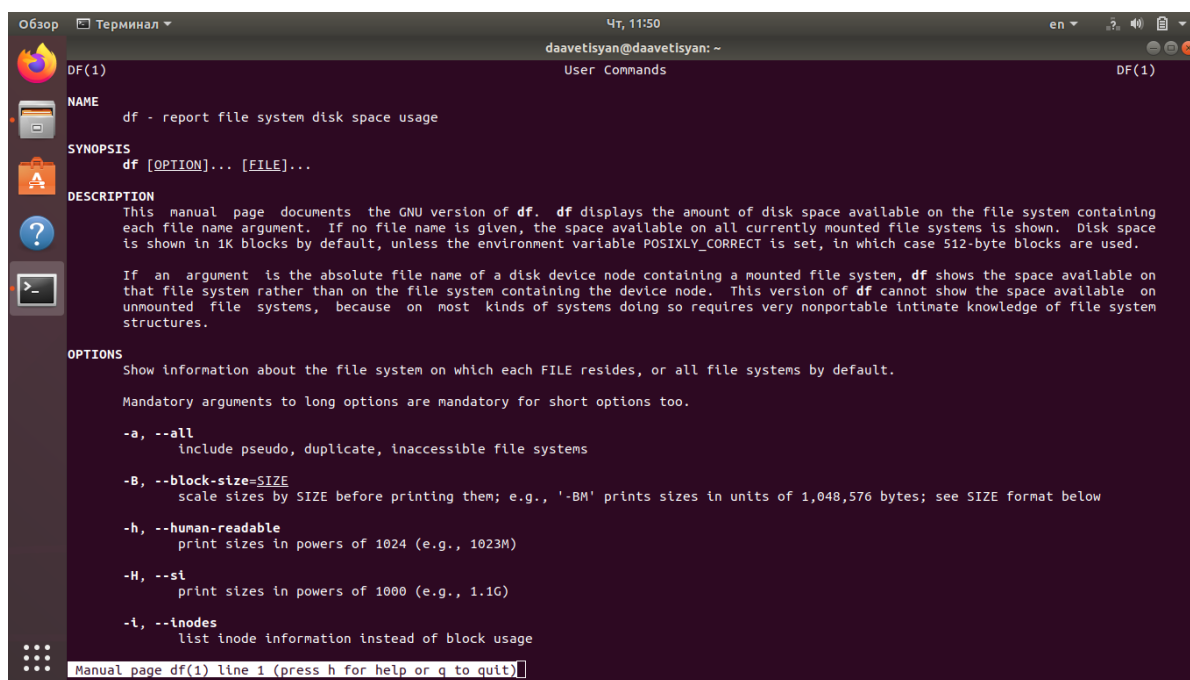
11. С помощью команд «man df» (рис. -fig. 3.13) и «man du» (рис. -fig. 3.14) узнаю информацию по необходимым командам и далее использую их (рис. -fig. 3.15).

df – утилита, показывающая список всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования.

Синтаксис: df опции устройство

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



```
Обзор Терминал 4т, 11:50
daavetisyan@daavetisyan: ~
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.

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. 3.13: Информация о команде df

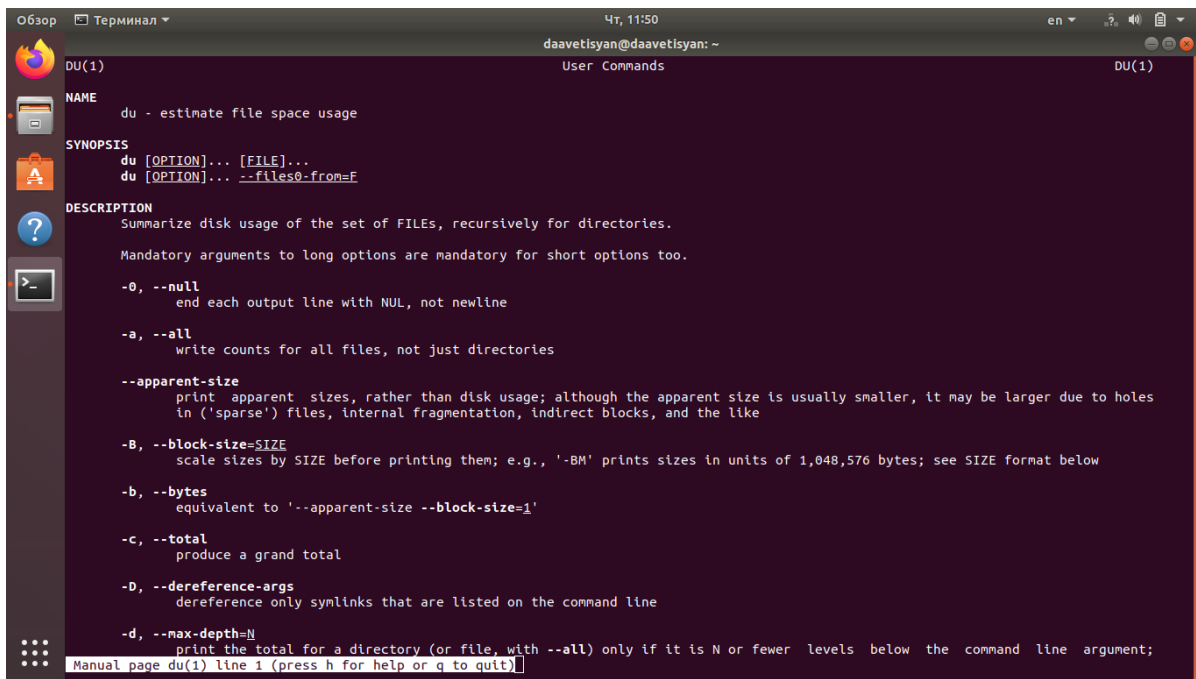


Рис. 3.14: Информация о команде du

```

daavetisyan@daavetisyan:~$ df
Файл.система  1K-блоков  Использовано  Доступно  Использовано%  Смонтировано в
udev           993872      0      993872      0% /dev
tmpfs          203556      1384    202172      1% /run
/dev/sda1      10253588    7570120    2142900     78% /
tmpfs          1017760      0    1017760      0% /dev/shm
tmpfs           5120        4      5116      1% /run/lock
tmpfs          1017760      0    1017760      0% /sys/fs/cgroup
/dev/loop3      640        640        0     100% /snap/gnome-logs/103
/dev/loop2      2304        2304        0     100% /snap/gnome-system-monitor/157
/dev/loop0      56704        56704        0     100% /snap/core18/1885
/dev/loop5      2560        2560        0     100% /snap/gnome-calculator/748
/dev/loop6      56832        56832        0     100% /snap/core18/1997
/dev/loop1      2304        2304        0     100% /snap/gnome-system-monitor/148
/dev/loop4      2560        2560        0     100% /snap/gnome-calculator/884
/dev/loop7      63616        63616        0     100% /snap/gtk-common-themes/1506
/dev/loop8       384         384        0     100% /snap/gnome-characters/708
/dev/loop9      66688        66688        0     100% /snap/gtk-common-themes/1515
/dev/loop10     30720        30720        0     100% /snap/snapd/8542
/dev/loop11      1024        1024        0     100% /snap/gnome-logs/100
/dev/loop12     261760        261760        0     100% /snap/gnome-3-34-1804/36
/dev/loop13      33152        33152        0     100% /snap/snapd/11588
/dev/loop14     224256        224256        0     100% /snap/gnome-3-34-1804/66
/dev/loop15       384         384        0     100% /snap/gnome-characters/550
tmpfs          203552        64    203488      1% /run/user/1000

daavetisyan@daavetisyan:~$ du
4      ./local/share/flatpak/db
8      ./local/share/flatpak
4      ./local/share/sounds
80     ./local/share/gvfs-metadata
8      ./local/share/gnome-shell
4      ./local/share/Trash/expunged
4      ./local/share/Trash/files
4      ./local/share/Trash/info
16     ./local/share/Trash
4      ./local/share/mc/mcedit
16     ./local/share/mc
4      ./local/share/icc
4      ./local/share/gnome-settings-daemon

```

Рис. 3.15: Используем df и du

12. Получаем информацию с помощью команды «man find» (рис. -fig. 3.16) и выводим имена всех директорий, имеющихя в домашнем каталоге с помощью команды «find ~ -type d» (рис. -fig. 3.17).



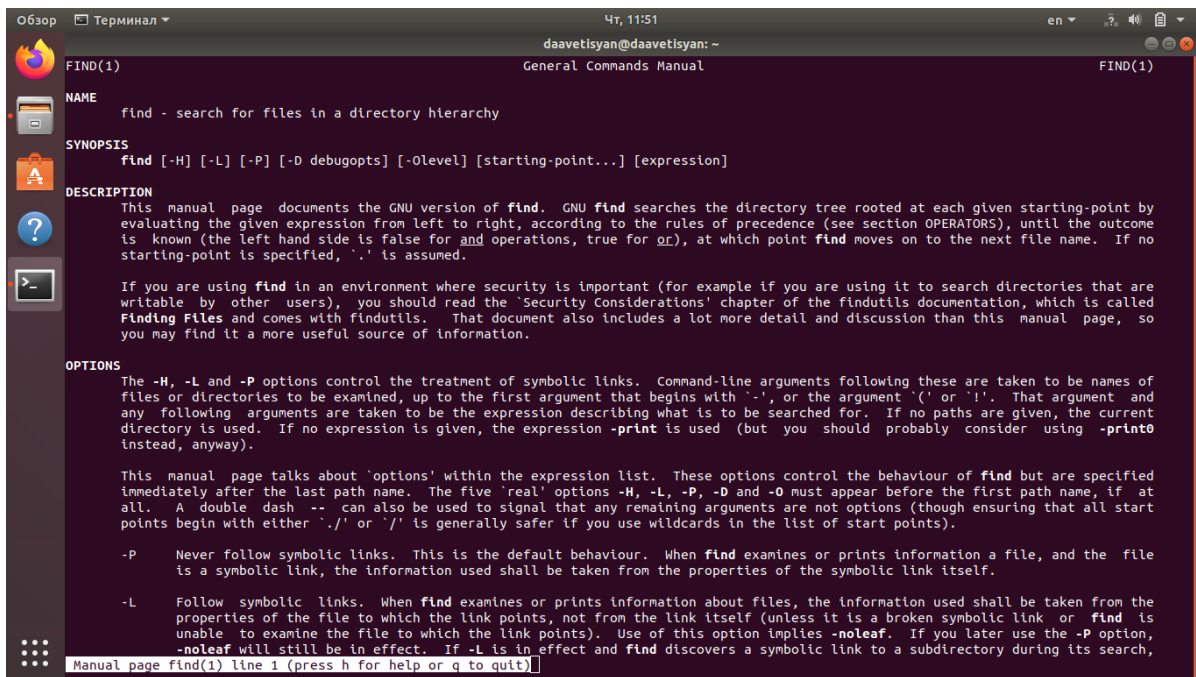


Рис. 3.16: Информация о команде `find`

```
daavetisyan@daavetisyan:~$ man find
daavetisyan@daavetisyan:~$ find ~ -type d
/home/daavetisyan
/home/daavetisyan/.local
/home/daavetisyan/.local/share
/home/daavetisyan/.local/share/flatpak
/home/daavetisyan/.local/share/flatpak/db
/home/daavetisyan/.local/share/sounds
/home/daavetisyan/.local/share/gvfs-metadata
/home/daavetisyan/.local/share/gnome-shell
/home/daavetisyan/.local/share/Trash
/home/daavetisyan/.local/share/Trash/expunged
/home/daavetisyan/.local/share/Trash/files
/home/daavetisyan/.local/share/Trash/info
/home/daavetisyan/.local/share/mc
/home/daavetisyan/.local/share/mc/mcedit
/home/daavetisyan/.local/share/icc
/home/daavetisyan/.local/share/gnome-settings-daemon
/home/daavetisyan/.local/share/nano
/home/daavetisyan/.local/share/nautilus
/home/daavetisyan/.local/share/nautilus/scripts
/home/daavetisyan/.local/share/keyrings
/home/daavetisyan/.local/share/app-info
/home/daavetisyan/.local/share/app-info/xmls
/home/daavetisyan/.local/share/xorg
/home/daavetisyan/.local/share/ibus-table
/home/daavetisyan/.local/share/evolution
/home/daavetisyan/.local/share/evolution/addressbook
/home/daavetisyan/.local/share/evolution/addressbook/system
/home/daavetisyan/.local/share/evolution/addressbook/system/photos
/home/daavetisyan/.local/share/evolution/addressbook/trash
/home/daavetisyan/.local/share/evolution/memos
/home/daavetisyan/.local/share/evolution/memos/trash
/home/daavetisyan/.local/share/evolution/tasks
/home/daavetisyan/.local/share/evolution/tasks/trash
/home/daavetisyan/.local/share/evolution/calendar
/home/daavetisyan/.local/share/evolution/calendar/system
/home/daavetisyan/.local/share/evolution/calendar/trash
/home/daavetisyan/.local/share/evolution/mail
```

Рис. 3.17: Вывод имен всех директорий, имеющих в домашнем каталоге

## 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.  
Программа 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+/\`;
  - 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.

`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 данной лабораторной работы я изучил инструменты поиска файлов и фильтрации текстовых данных, а также приобрёл практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.