

Отчёт по лабораторной работе №6

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

Тагиев Павел Фаикович

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Использование операторов перенаправления	8
4.2	Поиск файлов	9
4.3	Запуск работы	11
4.4	Команды ps и kill	11
4.5	Команда df	13
4.6	Команда du	15
4.7	Имена всех директорий в домашнем каталоге	16
5	Ответы на контрольные вопросы	18
6	Выводы	23
	Список литературы	24

Список иллюстраций

4.1	Команды <code>grep</code> , <code>find</code> и операторы <code>></code> , <code>>></code>	8
4.2	Поиск файла по шаблону	9
4.3	Поиск в каталоге <code>/etc</code>	10
4.4	Постраничный вывод	10
4.5	Запуск работы	11
4.6	Запуск <code>gedit</code>	12
4.7	Команды <code>kill</code> и <code>jobs</code>	12
4.8	Документация команды <code>kill</code>	13
4.9	Команда <code>df</code>	14
4.10	Документация команды <code>df</code>	14
4.11	Команда <code>du</code>	15
4.12	Документация команды <code>du</code>	16
4.13	Документация к команде <code>find</code>	17
4.14	Каталоги домашней директории	17
5.1	Утилита <code>top</code>	22
5.2	Утилита <code>htop</code>	22

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 [1].

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, начинавшиеся с символа `c`? Предложите несколько вариантов, как это сделать.
5. Выведите на экран (по странично) 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`. Как ещё можно определить идентификатор процесса?
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 директорий, имеющихся в вашем домашнем каталоге.

3 Теоретическое введение

В процессе работы с файловой системой Linux часто возникает необходимость в поиске определенных файлов по различным критериям, таким как имя файла, размер, тип и т.д. Мы рассмотрим различные инструменты командной строки, такие как `find` и `grep`, которые позволяют эффективно выполнять поиск файлов.

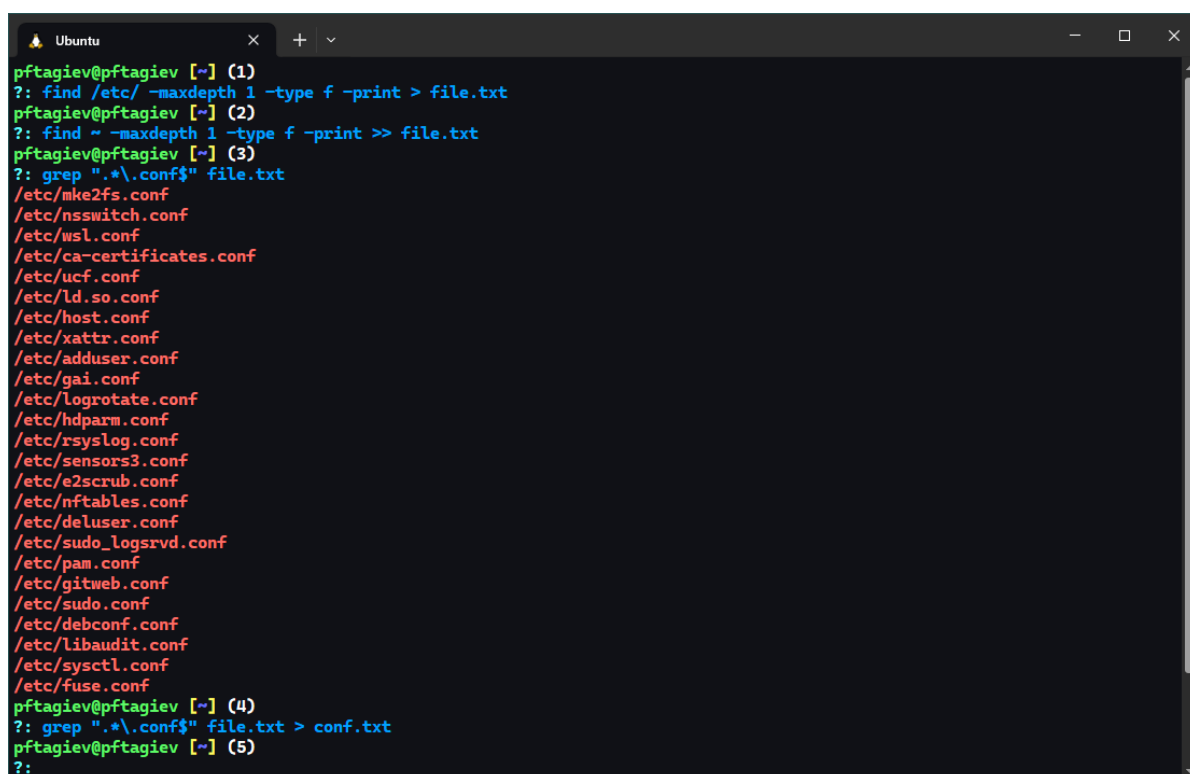
Перенаправление ввода-вывода — еще один мощный механизм командной строки, который позволяет изменять потоки данных между программами и файлами. Мы изучим основные способы перенаправления ввода-вывода, такие как использование символов перенаправления `>`, `>>`, `<<`, `<` и `|`, а также их применение в различных сценариях.

Для эффективного управления системой важно иметь возможность просматривать информацию о текущих процессах, запущенных на компьютере. Мы ознакомимся с командами `ps`, `top` и `htop`, которые предоставляют информацию о процессах и ресурсах системы в реальном времени.

4 Выполнение лабораторной работы

4.1 Использование операторов перенаправления

Запишем в файл `file.txt` названия файлов в содержащихся в каталоге `/etc`. Допишем в этот файл названия файлов, содержащихся в нашем домашнем каталоге. Так как в задании не указано нужно ли записывать в `file.txt` содержимое и вложенных в `/etc` и `~` каталогов, я указал опции `--maxdepth` глубину 1. Прделаное можно увидеть на рис. 4.1 на промтах (1) и (2).



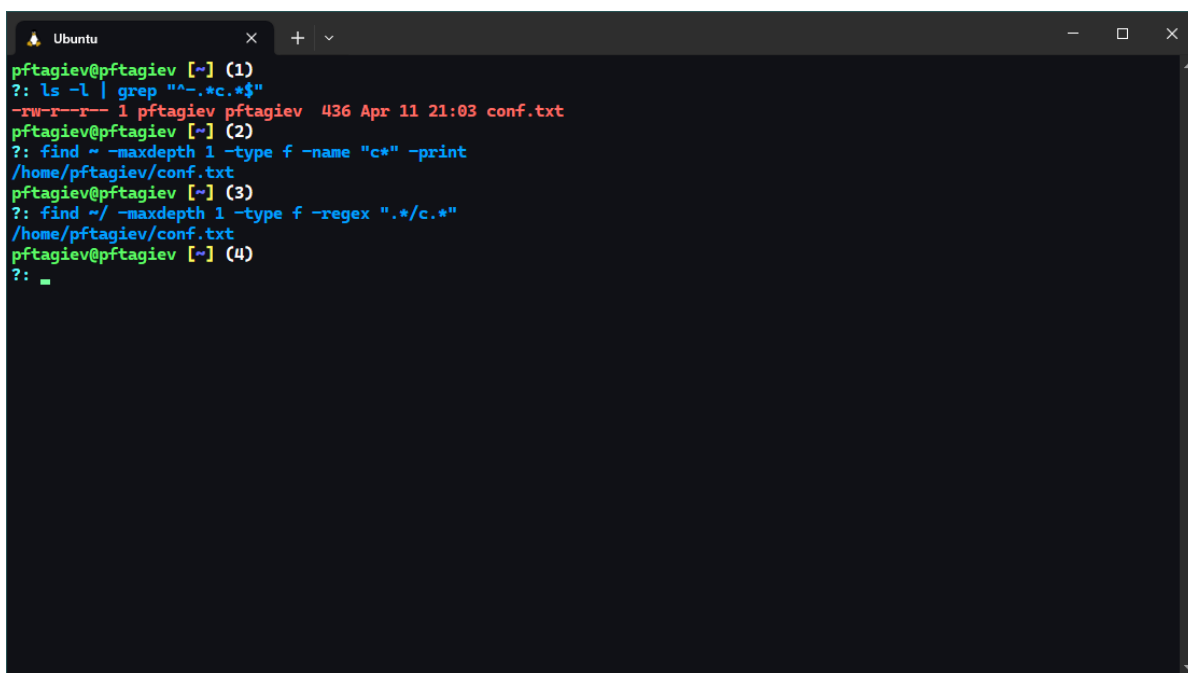
```
Ubuntu
pftagiev@pftagiev [~] (1)
?: find /etc/ -maxdepth 1 -type f -print > file.txt
pftagiev@pftagiev [~] (2)
?: find ~ -maxdepth 1 -type f -print >> file.txt
pftagiev@pftagiev [~] (3)
?: grep ".*\.conf$" file.txt
/etc/mke2fs.conf
/etc/nsswitch.conf
/etc/wsl.conf
/etc/ca-certificates.conf
/etc/ucf.conf
/etc/ld.so.conf
/etc/host.conf
/etc/xattr.conf
/etc/adduser.conf
/etc/gai.conf
/etc/logrotate.conf
/etc/hdparm.conf
/etc/rsyslog.conf
/etc/sensors3.conf
/etc/e2scrub.conf
/etc/nftables.conf
/etc/deluser.conf
/etc/sudo_logsrvd.conf
/etc/pam.conf
/etc/gitweb.conf
/etc/sudo.conf
/etc/debconf.conf
/etc/libaudit.conf
/etc/sysctl.conf
/etc/fuse.conf
pftagiev@pftagiev [~] (4)
?: grep ".*\.conf$" file.txt > conf.txt
pftagiev@pftagiev [~] (5)
?:
```

Рис. 4.1: Команды `grep`, `find` и операторы `>`, `>>`

Выведем имена всех файлов из `file.txt` имеющих расширение `.conf`, после чего запишем эти имена в файл `conf.txt` (рис. 4.1 промты (3) и (4)).

4.2 Поиск файлов

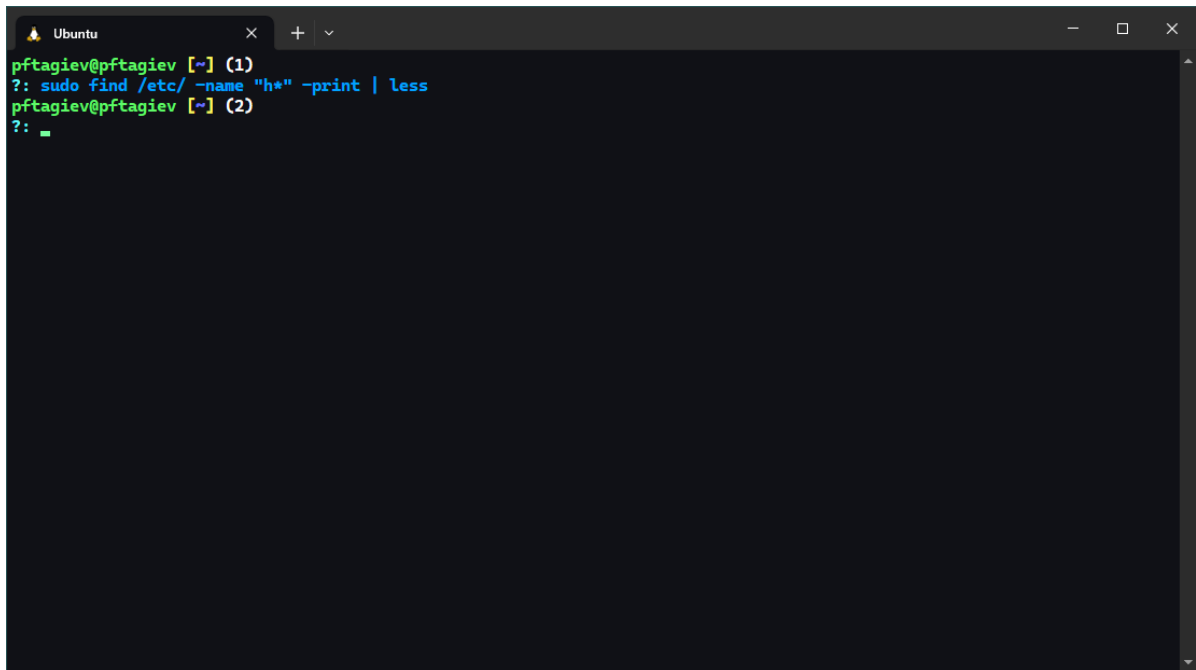
Задание требует определить, какие *файлы* в нашем домашнем каталоге имеют имена начинающиеся с символа `c`. Несколько вариантов того, как это сделать можно увидеть на рис. 4.2.



```
pftagiev@pftagiev [~] (1)
?: ls -l | grep "^-.*c.*$"
-rw-r--r-- 1 pftagiev pftagiev 436 Apr 11 21:03 conf.txt
pftagiev@pftagiev [~] (2)
?: find ~ -maxdepth 1 -type f -name "c*" -print
/home/pftagiev/conf.txt
pftagiev@pftagiev [~] (3)
?: find ~/ -maxdepth 1 -type f -regex ".*c.*"
/home/pftagiev/conf.txt
pftagiev@pftagiev [~] (4)
?: _
```

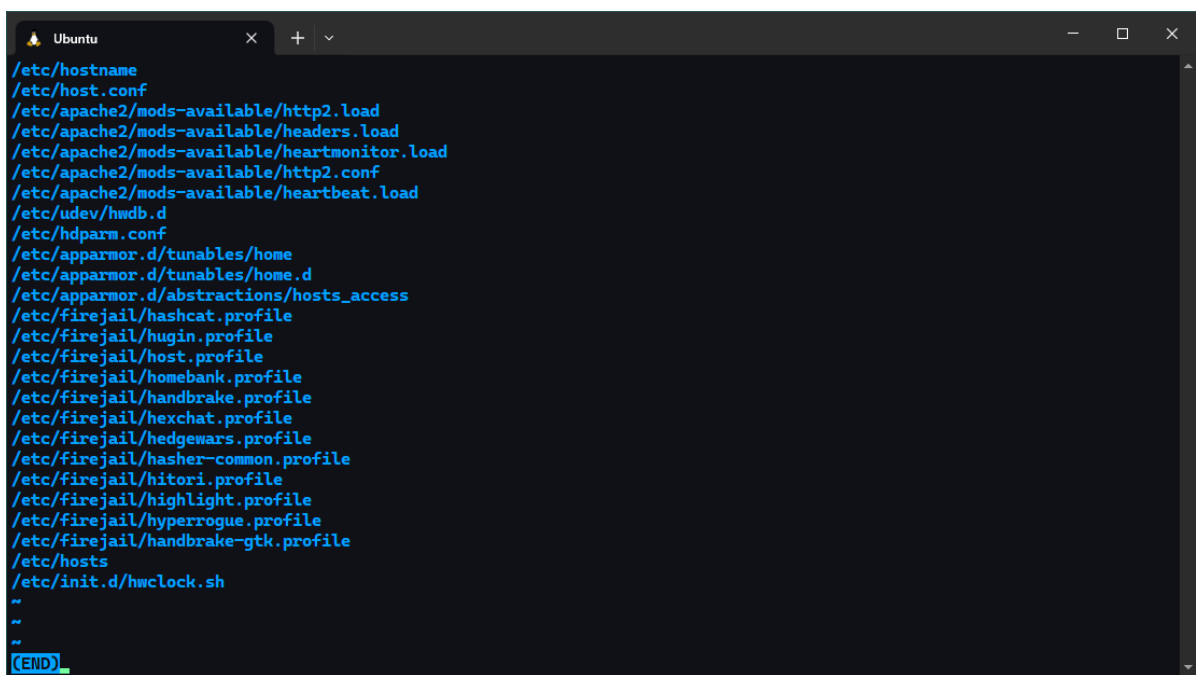
Рис. 4.2: Поиск файла по шаблону

Далее нужно постранично вывести файлы из каталога `/etc` имена которых начинаются с символа `h` (рис. 4.3, 4.4).

A terminal window titled 'Ubuntu' with a dark background. The prompt is 'pftagiev@pftagiev [~] (1)'. The user enters the command '?: sudo find /etc/ -name "h*" -print | less'. The prompt changes to 'pftagiev@pftagiev [~] (2)' and the user enters '?:'.

```
pftagiev@pftagiev [~] (1)
?: sudo find /etc/ -name "h*" -print | less
pftagiev@pftagiev [~] (2)
?:
```

Рис. 4.3: Поиск в каталоге /etc

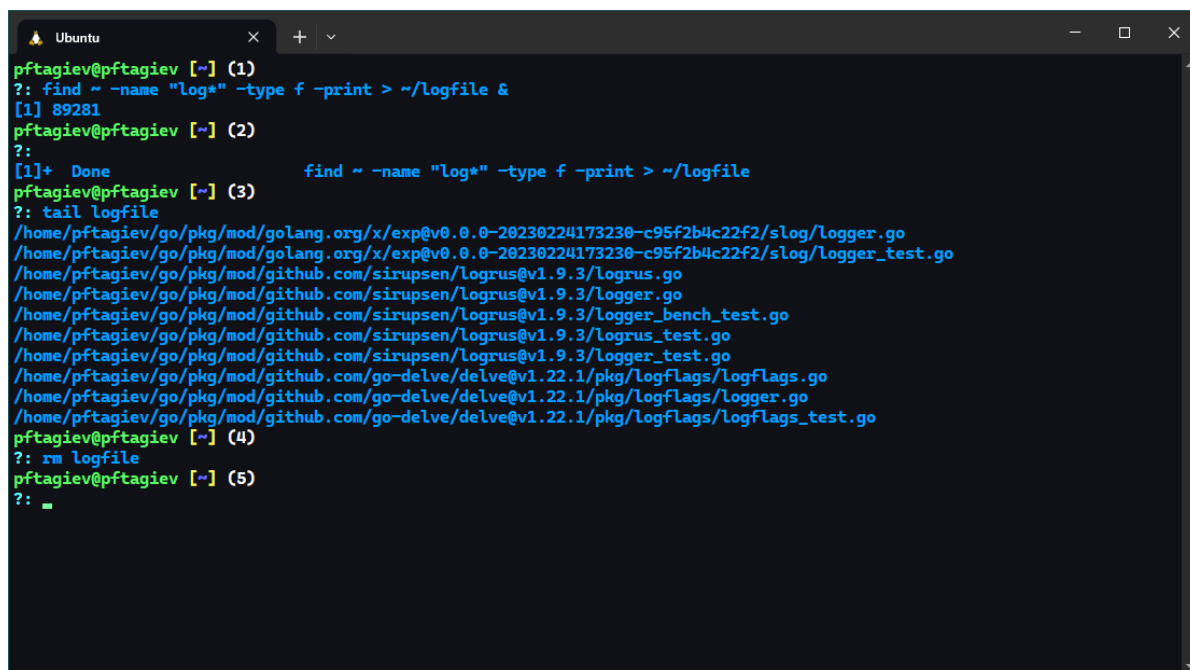
A terminal window titled 'Ubuntu' with a dark background. It shows the output of the search command from the previous image. The output lists various files in /etc, including /etc/hostname, /etc/host.conf, and several files in /etc/apache2/mods-available and /etc/firejail. The output ends with '(END)'.

```
/etc/hostname
/etc/host.conf
/etc/apache2/mods-available/http2.load
/etc/apache2/mods-available/headers.load
/etc/apache2/mods-available/heartmonitor.load
/etc/apache2/mods-available/http2.conf
/etc/apache2/mods-available/heartbeat.load
/etc/udev/hwdb.d
/etc/hdparm.conf
/etc/apparmor.d/tunables/home
/etc/apparmor.d/tunables/home.d
/etc/apparmor.d/abstractions/hosts_access
/etc/firejail/hashcat.profile
/etc/firejail/hugin.profile
/etc/firejail/host.profile
/etc/firejail/homebank.profile
/etc/firejail/handbrake.profile
/etc/firejail/hexchat.profile
/etc/firejail/hedgewars.profile
/etc/firejail/hasher-common.profile
/etc/firejail/hitori.profile
/etc/firejail/highlight.profile
/etc/firejail/hyperrogue.profile
/etc/firejail/handbrake-gtk.profile
/etc/hosts
/etc/init.d/hwclock.sh
...
...
...
(END)
```

Рис. 4.4: Постраничный вывод

4.3 Запуск работы

Запустим в *фоновом режиме* процесс, который будет записывать в файл `~/logfile` файлы, имена которых начинаются с `log`. Убедимся что процесс отработал правильно выведя последние 10 строк файла `logfile` командой `tail`. Затем удалим его, как того требует задание (рис. 4.5).



```
Ubuntu
pftagiev@pftagiev [~] (1)
?: find ~ -name "log*" -type f -print > ~/logfile &
[1] 89281
pftagiev@pftagiev [~] (2)
?:
[1]+  Done                  find ~ -name "log*" -type f -print > ~/logfile
pftagiev@pftagiev [~] (3)
?: tail logfile
/home/pftagiev/go/pkg/mod/golang.org/x/exp@v0.0.0-20230224173230-c95f2b4c22f2/slog/logger.go
/home/pftagiev/go/pkg/mod/golang.org/x/exp@v0.0.0-20230224173230-c95f2b4c22f2/slog/logger_test.go
/home/pftagiev/go/pkg/mod/github.com/sirupsen/logrus@v1.9.3/logrus.go
/home/pftagiev/go/pkg/mod/github.com/sirupsen/logrus@v1.9.3/logger.go
/home/pftagiev/go/pkg/mod/github.com/sirupsen/logrus@v1.9.3/logger_bench_test.go
/home/pftagiev/go/pkg/mod/github.com/sirupsen/logrus@v1.9.3/logger_test.go
/home/pftagiev/go/pkg/mod/github.com/sirupsen/logrus@v1.9.3/logger_test.go
/home/pftagiev/go/pkg/mod/github.com/go-delve/delve@v1.22.1/pkg/logflags/logflags.go
/home/pftagiev/go/pkg/mod/github.com/go-delve/delve@v1.22.1/pkg/logflags/logger.go
/home/pftagiev/go/pkg/mod/github.com/go-delve/delve@v1.22.1/pkg/logflags/logflags_test.go
pftagiev@pftagiev [~] (4)
?: rm logfile
pftagiev@pftagiev [~] (5)
?: _
```

Рис. 4.5: Запуск работы

4.4 Команды `ps` и `kill`

В терминале запустим редактор `gedit` в *фоновом режиме* как показано на рис. 4.6. На моей системе следует запускать `gedit` с правами суперпользователя, почему это так можно узнать в [2]. Определим PID процесса `gedit` используя комбинацию команд `ps` и `grep`, так же можно использовать `pgrep <имя>`, который производит поиск шаблона по списку процессов, т. е. является аналогом комбинации `ps aux | grep "имя"`. Или можно вывести запущенные работы командой `jobs` (рис. 4.7).

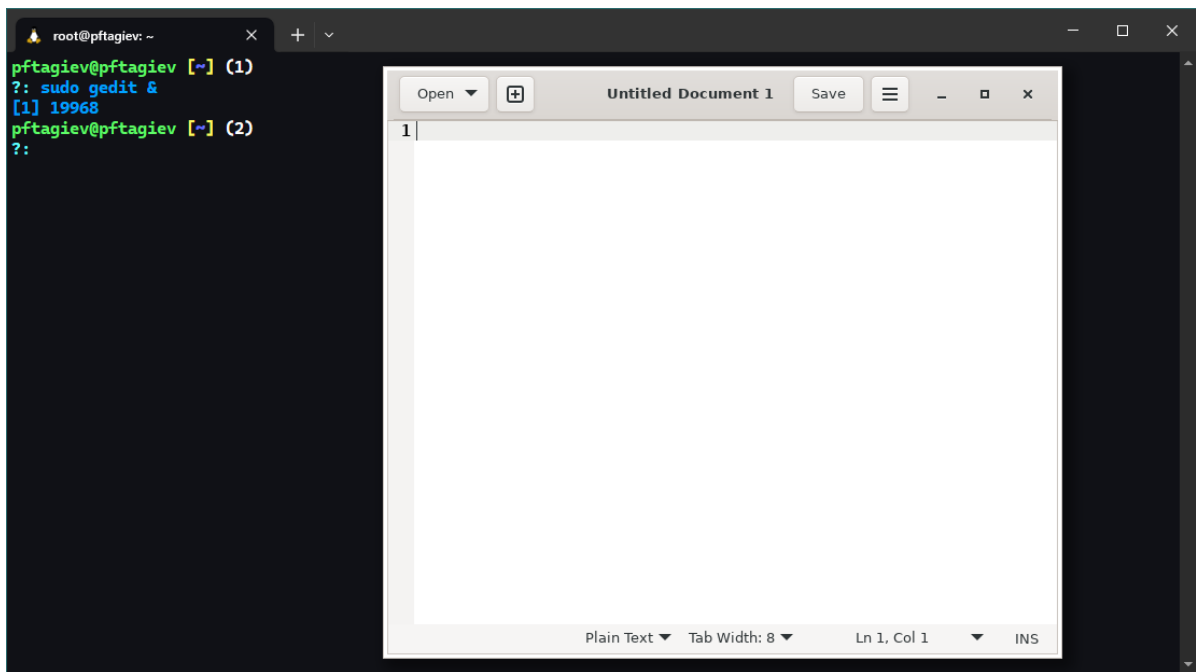


Рис. 4.6: Запуск gedit

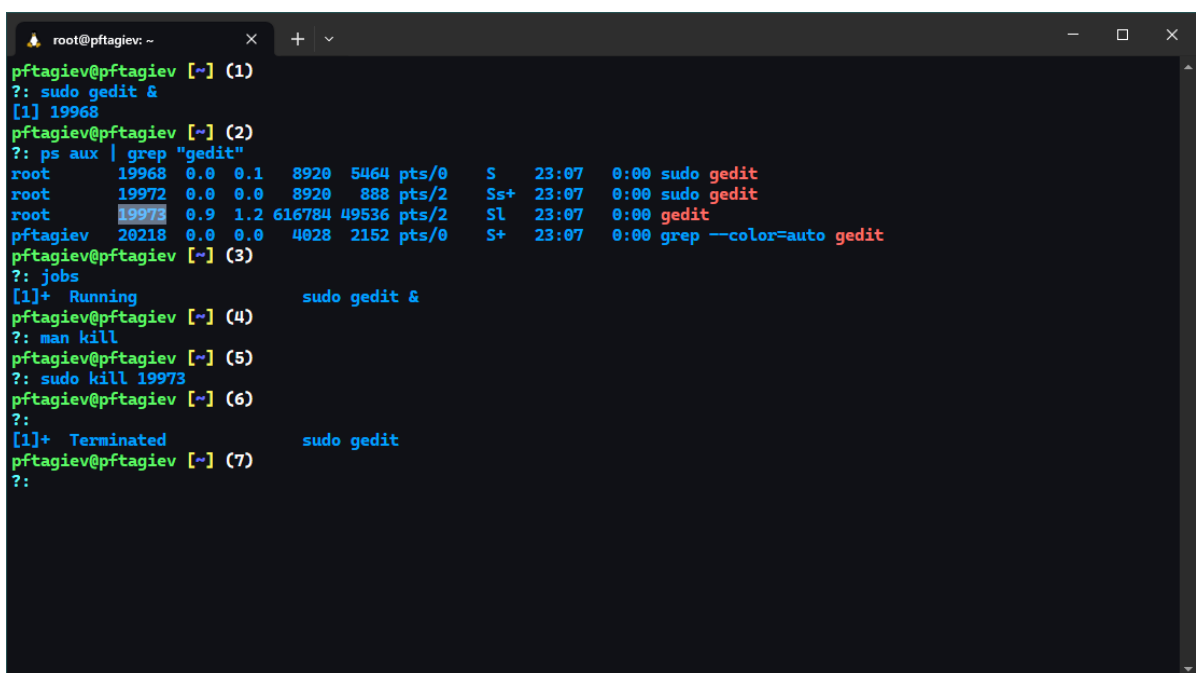
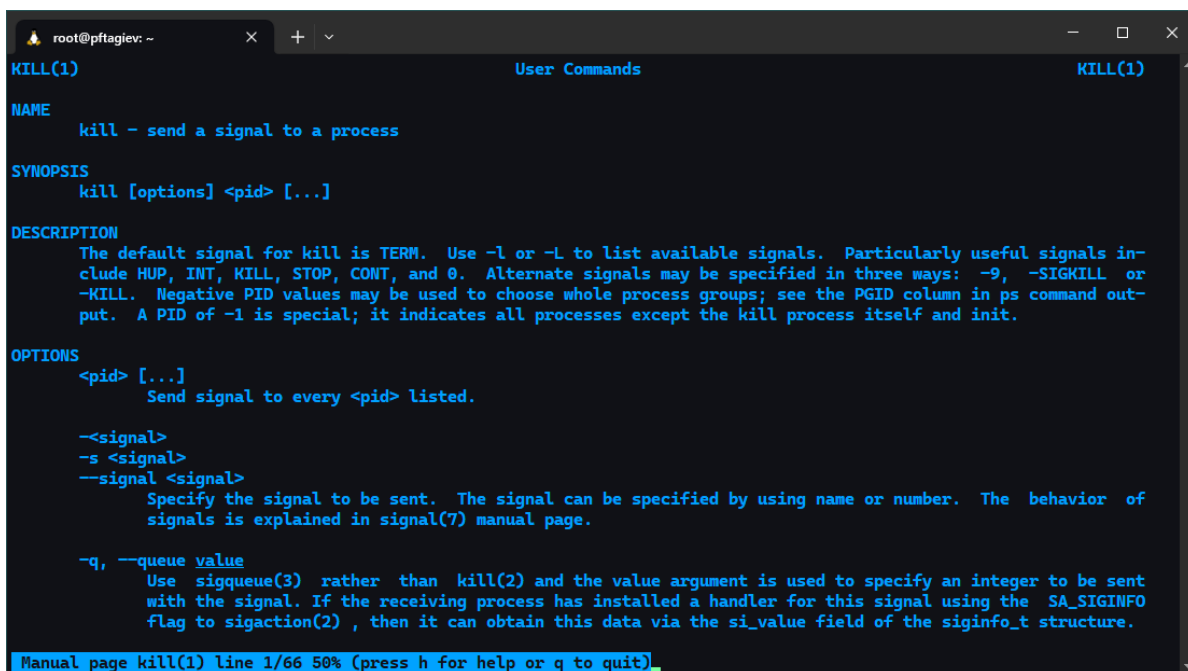


Рис. 4.7: Команды kill и jobs

Воспользовавшись документацией к команде `kill` (рис. 4.8). Завершим процесс `gedit` как показано на рис. 4.7 на промте (5) (так как я запустил

gedit с правами суперпользователя, чтобы завершить процесс тоже нужны эти права).



```
root@pftagiev: ~
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.

  -q, --queue value
    Use sigqueue(3) 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.

Manual page kill(1) line 1/66 50% (press h for help or q to quit)
```

Рис. 4.8: Документация команды kill

4.5 Команда df

Воспользовавшись документацией к команде `df` (рис. 4.10), запустим ее с флагом `-h` для вывода размеров монтированных файловых систем в человекочитаемом формате (рис. 4.9).

```
root@pftagiev: ~  
pftagiev@pftagiev [~] (1)  
?: man df  
pftagiev@pftagiev [~] (2)  
?: df -h  
Filesystem      Size  Used Avail Use% Mounted on  
none            2.0G  4.0K  2.0G   1% /mnt/wsl  
none            447G  195G  253G  44% /usr/lib/wsl/drivers  
none            2.0G   0  2.0G   0% /usr/lib/modules  
none            2.0G   0  2.0G   0% /usr/lib/modules/5.15.146.1-microsoft-standard-WSL2  
/dev/sdc        1007G  11G  946G   2% /  
none            2.0G  112K  2.0G   1% /mnt/wslg  
none            2.0G   0  2.0G   0% /usr/lib/wsl/lib  
rootfs          1.9G  1.9M  1.9G   1% /init  
none            2.0G  868K  2.0G   1% /run  
none            2.0G   0  2.0G   0% /run/lock  
none            2.0G   0  2.0G   0% /run/shm  
tmpfs           4.0M   0  4.0M   0% /sys/fs/cgroup  
none            2.0G  1.4M  1.9G   1% /mnt/wslg/versions.txt  
none            2.0G  1.4M  1.9G   1% /mnt/wslg/doc  
C:\             447G  195G  253G  44% /mnt/c  
D:\             918G  83G  835G  10% /mnt/d  
E:\             13G   43M   13G   1% /mnt/e  
snapfuse        128K  128K   0 100% /snap/bare/5  
snapfuse         74M   74M   0 100% /snap/core22/864  
snapfuse         75M   75M   0 100% /snap/core22/1122  
snapfuse         41M   41M   0 100% /snap/snapd/20290  
snapfuse         92M   92M   0 100% /snap/gtk-common-themes/1535  
snapfuse         40M   40M   0 100% /snap/snapd/21184  
snapfuse        132M  132M   0 100% /snap/ubuntu-desktop-installer/1276  
snapfuse        132M  132M   0 100% /snap/ubuntu-desktop-installer/1286  
pftagiev@pftagiev [~] (3)  
?:
```

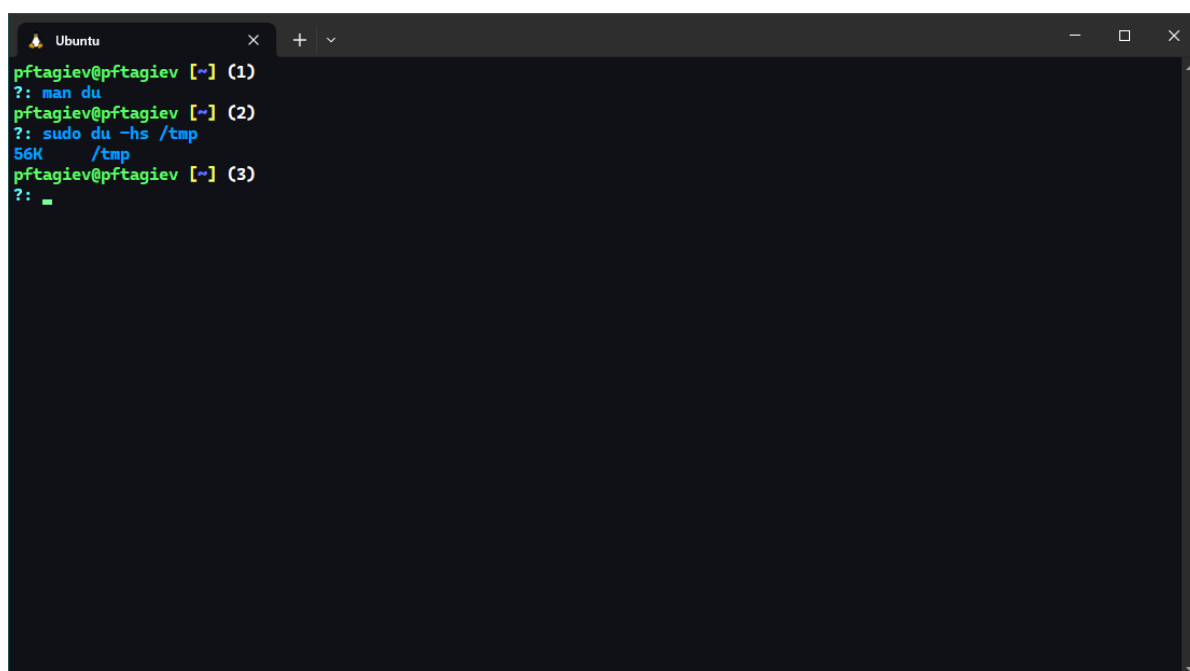
Рис. 4.9: Команда df

```
root@pftagiev: ~  
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 Manual page df(1) line 1 (press h for help or q to quit).
```

Рис. 4.10: Документация команды df

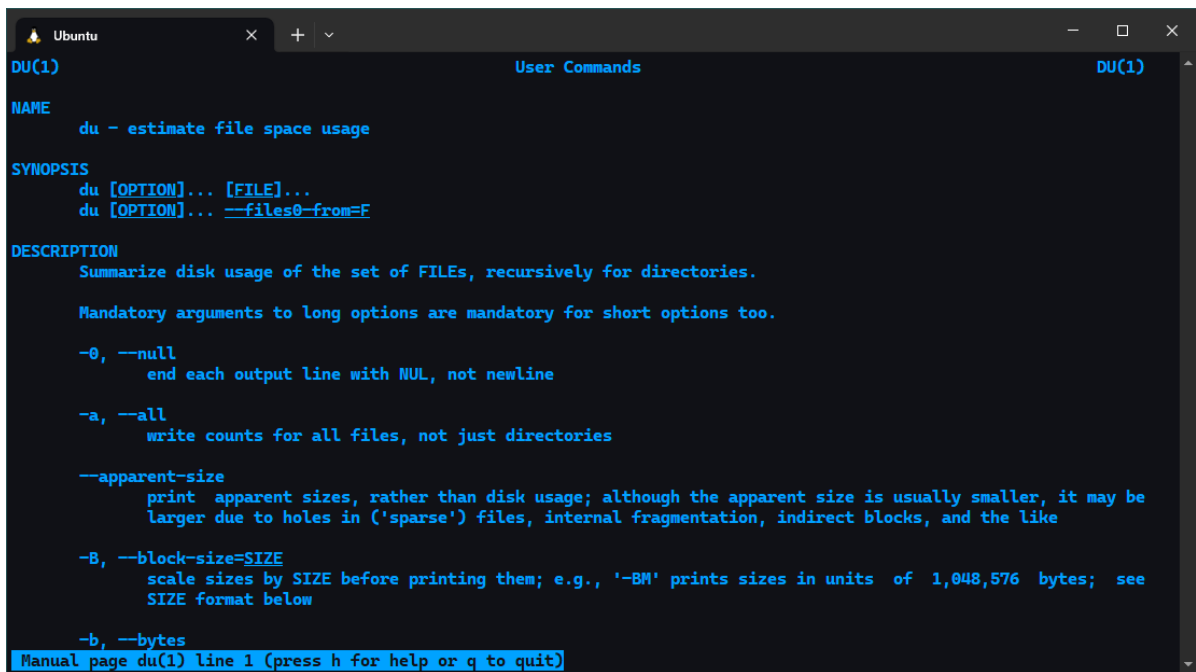
4.6 Команда du

Откроем документацию к команде `du` (рис. 4.12). Узнав новые для себя опции воспользуемся этой командой чтобы узнать размер каталога `/tmp` как показано на рис. 4.11. Флаг `-s` означает *summarize* т. е. сумма размеров всех файлов и каталогов, `-h` — уже знакомый нам флаг *human readable format*. Для каталога `/tmp` запускать эту команду нужно с правами суперпользователя так как там могут встретиться файлы которые запрещено читать кому-то кроме пользователя `root`.



```
pftagiev@pftagiev [~] (1)
?: man du
pftagiev@pftagiev [~] (2)
?: sudo du -hs /tmp
56K    /tmp
pftagiev@pftagiev [~] (3)
?:
```

Рис. 4.11: Команда du



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

Рис. 4.12: Документация команды du

4.7 Имена всех директорий в домашнем каталоге

Прочитав документацию к команде `find` (рис. 4.13), становится ясно что можно указать тип файлов для которых будет производиться поиск. Для директорий нужно указать флаг `-type` со значением `d`, еще я указал глубину поиска 1, чтобы искать только в домашней директории и не учитывать вложенные каталоги. Результат можно увидеть на рис. 4.14.


```
1,048,575 bytes.

-true Always true.

-type c
File is of type c:

b    block (buffered) special
c    character (unbuffered) special
d    directory
p    named pipe (FIFO)
f    regular file
l    symbolic link; this is never true if the -L option or the -follow option is in effect, unless
    the symbolic link is broken. If you want to search for symbolic links when -L is in effect, use
    -xtype.
s    socket
D    door (Solaris)

To search for more than one type at once, you can supply the combined list of type letters separated by
a comma ',' (GNU extension).

-uid n File's numeric user ID is less than, more than or exactly n.

/-type_
```

Рис. 4.13: Документация к команде find

```
pftagiev@pftagiev [~] (1)
?: man find
pftagiev@pftagiev [~] (2)
?: find ./ -maxdepth 1 -type d -print
./
./repos
./dotnet
./vscode-server
./local
./config
./cdir
./downloads
./cache
./go
./work
./gnupg
./texlive2021
./ssh
pftagiev@pftagiev [~] (3)
?: _
```

Рис. 4.14: Каталоги домашней директории

5 Ответы на контрольные вопросы

1. Какие потоки ввода вывода вы знаете?

- `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.

2. Объясните разницу между операцией `>` и `>>`.

- `>` — Перенаправление вывода в файл, содержимое файла будет перезаписано.
- `>>` — Перенаправление вывода в файла, новая информация будет добавляться в конец файла.

3. Что такое конвейер?

Конвейер или пайп 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 будет передан на ввод команде 2.

4. Что такое процесс? Чем это понятие отличается от программы?

Процесс - это экземпляр программы, который выполняется на компьютере в определенный момент времени. Программа, с другой стороны,

представляет собой статический набор инструкций и данных, который сохранен на диске и ожидает выполнения.

5. Что такое PID и GID?

- PID — Это уникальный числовой идентификатор, присваиваемый операционной системой каждому процессу при его создании. PID используется для идентификации и управления процессами в системе. Когда вы запускаете программу или команду в терминале, операционная система назначает ей уникальный PID, который может быть использован для мониторинга, завершения или взаимодействия с процессом.
- GID — Это числовой идентификатор, связанный с определенной группой пользователей на операционной системе. Каждый пользователь может принадлежать одной или нескольким группам, и GID используется для определения принадлежности пользователей к этим группам. GID может использоваться для управления правами доступа к файлам и ресурсам, которые принадлежат определенной группе пользователей.

6. Что такое задачи и какая команда позволяет ими управлять?

Задачами называются запущенные фоном программы, например `gedit &`. Ими можно управлять с помощью команды `jobs`, которая выводит список запущенных в данный момент задач. Для завершения задачи необходимо выполнить команду `kill %номер_задачи`.

7. Найдите информацию об утилитах `top` и `htop`. Каковы их функции?

Утилиты `top` и `htop` это в первую очередь, более удобная альтернатива командам `ps` и `kill`. Интерфейс обеих утилит можно увидеть на рис. 5.1, 5.2. Ниже можно увидеть более подробную информацию по этим утилитам:

- `top` — это утилита которая предоставляет информацию о запущенных

процессах и использовании системных ресурсов. Она отображает список процессов в реальном времени, упорядоченных по использованию процессора по умолчанию. `top` предоставляет информацию о загрузке процессора, памяти, `swap`-памяти, а также общее количество процессов и их состояние. Пользователь может взаимодействовать с `top`, например, изменять порядок сортировки, убивать процессы и так далее, используя различные команды [3].

- `htop` — это интерактивная утилита командной строки, которая предоставляет более удобный и информативный способ отображения информации о процессах и ресурсах системы по сравнению с `top`. Она предоставляет аналогичную информацию о процессах, загрузке процессора, памяти и других системных ресурсах, но с более удобным интерфейсом и возможностями. `htop` позволяет пользователю взаимодействовать с процессами и ресурсами через графический интерфейс в терминале, что делает ее более удобной и интуитивно понятной для использования. Она поддерживает прокрутку, цветовую кодировку, динамическое обновление и другие функции, которые делают мониторинг и управление процессами более эффективными [4].

8. Назовите и дайте характеристику команде поиска файлов. Приведите примеры использования этой команды.

Для поиска файлов используется команда `find`. `find` — это мощная утилита командной строки, которая предназначена для поиска файлов и директорий в файловой системе на основе различных критериев. Далее можно увидеть несколько примеров ее использования:

- `find ~ -maxdepth 1 -type f -name "*.rc" -print` — Найти все файлы в домашней директории, которые заканчиваются на `rc`.
- `sudo find /etc -name ".*" -type f -print` — Вывести скрытые файлы директории `/etc` и всех директорий вложенных в нее.

- `find ~ -maxdepth 1 -type f -exec head -1 {} \;` — Вывести первую строку каждого файла в домашней директории.

9. Можно ли по контексту (содержанию) найти файл? Если да, то как?

Да можно, для этого нужно использовать команду `grep`. Например команда `grep -rn "int main()" --include=*.c,*.cpp` выведет имена всех файлов с расширением `.c` и `.cpp`, где встретилась строка `"int main()"`.

10. Как определить объем свободной памяти на жёстком диске?

Можно воспользоваться командой `df` с флагом `-h` (***human-readable*** — человекочитаемый формат), она выведет список всех монтированных файловых систем. В этом списке в колонке `Avail` будет написано количество свободной памяти.

11. Как определить объем вашего домашнего каталога?

Можно воспользоваться командой `du ~ -hs`, где флаг `-h` означает ***human-readable***, а флаг `-s` ***summarize***, т. е. выводить суммарный объем директории.

12. Как удалить зависший процесс?

Зависший процесс может быть удален командой `kill` с флагом `-9` или с флагом `-s` и значением `KILL`. Например, `kill -s KILL <id_процесса>`. Узнать `id` процесса можно выведя их список в терминал с помощью команды `ps aux`. Также можно передать этот список через пайп утилите `grep`, чтобы разобрать его регулярным выражением и найти `id` нужного процесса. Пример: `ps aux | grep -i "my_app"`.

```

top - 11:23:23 up 1:25, 1 user, load average: 0.15, 0.10, 0.08
Tasks: 59 total, 1 running, 58 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.2 us, 0.8 sy, 0.0 ni, 97.9 id, 0.0 wa, 0.0 hi, 0.1 si, 0.0 st
MiB Mem : 3892.9 total, 1282.9 free, 775.2 used, 1834.8 buff/cache
MiB Swap: 1024.0 total, 1024.0 free, 0.0 used, 2876.6 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
  770 pftagiev  20   0 1003656 138984 42336 S   2.7   3.5   3:26.22 node
    1 root      20   0 167116 12672  8352 S   1.3   0.3   1:33.30 systemd
  512 root      20   0 44096 37672 10292 S   0.7   0.9   0:38.68 python3
    2 root      20   0 2280 1304 1188 S   0.0   0.0   0:00.02 init-systemd(Ub
    8 root      20   0 2296 132 132 S   0.0   0.0   0:00.00 init
   40 root      19  -1 47752 15456 14412 S   0.0   0.4   0:00.23 systemd-journal
   63 root      20   0 22284 6064 4552 S   0.0   0.2   0:00.24 systemd-udev
   77 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse
   79 root      20   0 4768 1976 1432 S   0.0   0.0   0:01.14 snapfuse
   80 root      20   0 4496 160 16 S   0.0   0.0   0:00.00 snapfuse
   82 root      20   0 4628 164 16 S   0.0   0.0   0:00.00 snapfuse
   85 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse
   94 root      20   0 4712 1916 1432 S   0.0   0.0   0:02.96 snapfuse
   95 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse
   96 root      20   0 4968 2088 1588 S   0.0   0.1   0:01.52 snapfuse
  149 systemd+ 20   0 25540 12740 8548 S   0.0   0.3   0:00.18 systemd-resolve
  177 root      20   0 4308 2800 2564 S   0.0   0.1   0:00.00 cron
  179 message+ 20   0 8664 4740 4152 S   0.0   0.1   0:00.24 dbus-daemon
  182 root      20   0 35796 21236 11788 S   0.0   0.5   0:00.15 networkd-dispat
  184 syslog    20   0 222404 5336 4512 S   0.0   0.1   0:00.05 rsyslogd
  185 root      20   0 1466680 40816 19548 S   0.0   1.0   0:01.22 snapd
  186 root      20   0 15336 7208 6260 S   0.0   0.2   0:00.17 systemd-logind
  213 root      20   0 4784 3408 3172 S   0.0   0.1   0:00.10 subiquity-serve

```

Рис. 5.1: Утилита top

```

0[|
1[|
2[|
3[|
Mem[|||||] 787M/3.80G
Swp[|] 0K/1.00G

Tasks: 59, 145 thr; 2 running
Load average: 0.12 0.09 0.08
Uptime: 01:26:22

  PID USER      PRI  NI   VIRT   RES   SHR  S  CPU% MEM%    TIME+  Command
    1 root      20   0 163M 12672  8352 S   2.0   0.3   1:34.47 /sbin/init
  512 root      20   0 44096 37672 10292 S   0.7   0.9   0:39.12 python3 /snap/ubuntu-desktop-installer/1286/usr/bin/clo
  770 pftagiev  20   0 980M 135M 42336 R   0.7   3.5   3:28.10 /home/pftagiev/.vscode-server/bin/e170252f762678dec6ca2
  779 pftagiev  20   0 980M 135M 42336 S   0.7   3.5   0:07.51 /home/pftagiev/.vscode-server/bin/e170252f762678dec6ca2
    2 root      20   0 2280 1304 1188 S   0.0   0.0   0:00.02 /init
    8 root      20   0 2296 132 132 S   0.0   0.0   0:00.00 plan9 --control-socket 6 --log-level 4 --server-fd 7 --
    9 root      20   0 2296 132 132 S   0.0   0.0   0:00.00 plan9 --control-socket 6 --log-level 4 --server-fd 7 --
   10 root      20   0 2280 1304 1188 S   0.0   0.0   0:00.00 /init
   40 root      19  -1 47752 15456 14412 S   0.0   0.4   0:00.23 /lib/systemd/systemd-journald
   63 root      20   0 22284 6064 4552 S   0.0   0.2   0:00.24 /lib/systemd/systemd-udev
   77 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse /var/lib/snapd/snaps/bare_5.snap /snap/bare/5
   79 root      20   0 4768 1976 1432 S   0.0   0.0   0:01.14 snapfuse /var/lib/snapd/snaps/core22_1122.snap /snap/co
   80 root      20   0 4496 160 16 S   0.0   0.0   0:00.00 snapfuse /var/lib/snapd/snaps/core22_864.snap /snap/cor
   82 root      20   0 4628 164 16 S   0.0   0.0   0:00.00 snapfuse /var/lib/snapd/snaps/gtk-common-themes_1535.sn
   85 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse /var/lib/snapd/snaps/snapd_20290.snap /snap/sn
   94 root      20   0 4712 1916 1432 S   0.0   0.0   0:02.96 snapfuse /var/lib/snapd/snaps/snapd_21184.snap /snap/sn
   95 root      20   0 4496 164 16 S   0.0   0.0   0:00.00 snapfuse /var/lib/snapd/snaps/ubuntu-desktop-installer_
   96 root      20   0 4968 2088 1588 S   0.0   0.1   0:01.52 snapfuse /var/lib/snapd/snaps/ubuntu-desktop-installer_
  149 systemd-r 20   0 25540 12740 8548 S   0.0   0.3   0:00.18 /lib/systemd/systemd-resolved
  177 root      20   0 4308 2800 2564 S   0.0   0.1   0:00.00 /usr/sbin/cron -f -P
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice F9Kill F10Quit

```

Рис. 5.2: Утилита htop

6 Выводы

В этой работе мы более подробно разобрали работу с файлами в Linux. Научились перенаправлять вывод специальными операторами, познакомились с мощной утилитой `find` и использовали ее на практике. Научились запускать процессы в фоне и управлять ими.

Список литературы

1. Кулябов. Операционные системы. Москва: РУДН, 2016. 118 с.
2. dconf-WARNING **: failed to commit changes to dconf: The connection is closed [Электронный ресурс]. 2015. URL: <https://unix.stackexchange.com/questions/182925/dconf-warning-failed-to-commit-changes-to-dconf-the-connection-is-closed>.
3. top (software) [Электронный ресурс]. 2024. URL: [https://en.wikipedia.org/wiki/Top_\(software\)](https://en.wikipedia.org/wiki/Top_(software)).
4. htop [Электронный ресурс]. 2024. URL: <https://en.wikipedia.org/wiki/Htop>.