

Отчёт по лабораторной работе №8

Операционные системы

Калашникова Ольга Сергеевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	8
3.1	Конвейер	8
3.2	Поиск файла	10
3.3	Управление процессами и задачами	11
3.4	Проверка использования диска	13
4	Выводы	16
5	Ответы на контрольные вопросы	17

Список иллюстраций

3.1	Запись в файл	8
3.2	Вывод содержимого файла	8
3.3	Добавление данных в файл	9
3.4	Поиск файлов определенного расширения	9
3.5	Запись в файл	9
3.6	Поиск файлов, начинающихся с определенного элемента	10
3.7	Поиск файлов, начинающихся с определенного элемента	10
3.8	Поиск файлов, начинающихся с определенного элемента	11
3.9	Создание фонового процесса	11
3.10	Проверка	11
3.11	Удаление файла	12
3.12	Создание фонового процесса	12
3.13	Поиск идентификатора процесса	12
3.14	Поиск идентификатора процесса	12
3.15	Чтение документации	13
3.16	Удаление процесса	13
3.17	Чтение документации	13
3.18	Чтение документации	14
3.19	Утилита df	14
3.20	Утилита du	14
3.21	Чтение документации	15
3.22	Название рисунка	15

Список таблиц

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 (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 Выполнение лабораторной работы

3.1 Конвейер

Я вошла в систему под соответствующим именем пользователя, открыла терминал, записала в файл file.txt названия файлов из каталога /etc с помощью перенаправления “>” (и файл создала, и записала в него то, что могло быть выведено ls -lR /etc). В файл я добавила также все файлы из подкаталогов (рис. fig. 3.1).

```
[root@oskalashnikova ~]# ls -lR /etc > file.txt
[root@oskalashnikova ~]#
```

Рис. 3.1: Запись в файл

Проверила, что в файл записались нужные значения с помощью утилиты head, она выводит первые 10 строк файла на экран (рис. fig. 3.2).

```
[root@oskalashnikova ~]# head file.txt
/etc:
итого 1356
drwxr-xr-x. 1 root root    126 ноя  1 04:07 abrt
-rw-r--r--. 1 root root    16 фев 24 16:44 adjtime
-rw-r--r--. 1 root root  1529 июл 25 2023 aliases
drwxr-xr-x. 1 root root    70 янв 29 03:00 alsa
drwxr-xr-x. 1 root root  1462 мар  2 20:37 alternatives
drwxr-xr-x. 1 root root    56 ноя  1 04:07 anaconda
-rw-r--r--. 1 root root   541 июл 19 2023 anacrontab
-rw-r--r--. 1 root root   833 фев 10 2023 appstream.conf
[root@oskalashnikova ~]#
```

Рис. 3.2: Вывод содержимого файла

Добавила в созданный файл имена файлов из домашнего каталога, используя

перенаправление “>” в режиме добавления (рис. fig. 3.3).

```
[root@oskalashnikova ~]# ls -lR ~/ >> file.txt
[root@oskalashnikova ~]#
```

Рис. 3.3: Добавление данных в файл

Вывела на экран имена всех файлов, имеющих расширение “.conf” с помощью утилиты grep (рис. fig. 3.4).

```
[root@oskalashnikova ~]# grep .conf file.txt
-rw-r--r--. 1 root root 833 фев 10 2023 appstream.conf
-rw-r--r--. 1 root root 55 янв 29 03:00 asound.conf
-rw-r--r--. 1 root root 1372 дек 5 03:00 chrony.conf
drwxr-xr-x. 1 root root 18 ноя 1 04:05 dconf
-rw-r--r--. 1 root root 28602 фев 13 03:00 dnsmasq.conf
-rw-r--r--. 1 root root 117 ноя 16 03:00 dracut.conf
drwxr-xr-x. 1 root root 0 ноя 16 03:00 dracut.conf.d
-rw-r--r--. 1 root root 20 фев 24 2022 fprintd.conf
-rw-r--r--. 1 root root 38 авг 9 2023 fuse.conf
-rw-r--r--. 1 root root 9 июл 25 2023 host.conf
```

Рис. 3.4: Поиск файлов определенного расширения

Добавила вывод прошлой команды в новый файл conf.txt с помощью перенаправления “>” (файл создается при выполнении этой команды) (рис. fig. 3.5).

```
[root@oskalashnikova ~]# grep .conf file.txt > conf.txt
[root@oskalashnikova ~]# head conf.txt
-rw-r--r--. 1 root root 833 фев 10 2023 appstream.conf
-rw-r--r--. 1 root root 55 янв 29 03:00 asound.conf
-rw-r--r--. 1 root root 1372 дек 5 03:00 chrony.conf
drwxr-xr-x. 1 root root 18 ноя 1 04:05 dconf
-rw-r--r--. 1 root root 28602 фев 13 03:00 dnsmasq.conf
-rw-r--r--. 1 root root 117 ноя 16 03:00 dracut.conf
drwxr-xr-x. 1 root root 0 ноя 16 03:00 dracut.conf.d
-rw-r--r--. 1 root root 20 фев 24 2022 fprintd.conf
-rw-r--r--. 1 root root 38 авг 9 2023 fuse.conf
-rw-r--r--. 1 root root 9 июл 25 2023 host.conf
[root@oskalashnikova ~]#
```

Рис. 3.5: Запись в файл

3.2 Поиск файла

Определяю, какие файлы в домашнем каталоге начинаются с символа “с” с помощью утилиты `find`, прописываю ей в аргументах домашнюю директорию, выбираю опцию `-name` (ищем по имени), и пишу маску, по которой будем искать имя, где `*` - любое кол-во любых символов, добавляю опцию `-print`, чтобы мне вывелся результат. Я получаю информацию даже о файлах из подкаталогов домашнего каталога. (рис. fig. 3.6).

```
[root@oskalashnikova ~]# find ~ -name "c*" -print
```

Рис. 3.6: Поиск файлов, начинающихся с определенного элемента

Второй способ использовать утилиту `ls -lR` и использовать `grep`, чтобы найти элементы с первым символом с. Однако этот способ не работает для поиска файлов из подкаталогов каталога (рис. fig. 3.7).

```
[root@oskalashnikova ~]# ls -lR | grep c*  
-rw-r--r--. 1 root root 42506 map 30 18:23 conf.txt  
[root@oskalashnikova ~]#
```

Рис. 3.7: Поиск файлов, начинающихся с определенного элемента

С помощью метода `find`, чьи опции я расписала ранее, ищу все файлы, начинающиеся с буквы “h” из каталога `/etc` (рис. fig. 3.8).

```
[root@oskalashnikova ~]# find /etc -name "h*" -print
/etc/avahi/hosts
/etc/firewalld/helpers
/etc/libibverbs.d/hfi1verbs.driver
/etc/libibverbs.d/hns.driver
/etc/systemd/homed.conf
/etc/udev/hwdb.d
/etc/udev/hwdb.bin
/etc/host.conf
/etc/hosts
/etc/hostname
/etc/mercurial/hgrc.d
[root@oskalashnikova ~]#
```

Рис. 3.8: Поиск файлов, начинающихся с определенного элемента

3.3 Управление процессами и задачами

Запускаю в фоновом режиме (на это указывает символ &) процесс, который будет записывать в файл logfile (с помощью перенаправления >) файлы, имена которых начинаются с log (рис. fig. 3.9).

```
[root@oskalashnikova ~]# find ~ -name "log*" -print > logfile &
[1] 3229
[root@oskalashnikova ~]#
```

Рис. 3.9: Создание фонового процесса

Проверяю, что файл создан (рис. fig. 3.10).

```
[root@oskalashnikova ~]# ls
anaconda-ks.cfg      LICENSE
bin                  logfile
conf.txt             pandoc-1.19.2.1-1-amd64.deb
file.txt             revoke.asc
'g | grep -i "Linux version"'  work
git-extended
[1]+  Завершён      find ~ -name "log*" -print > logfile
[root@oskalashnikova ~]#
```

Рис. 3.10: Проверка

Удаляю файл, проверяю, что файл удален (рис. fig. 3.11).

```
[root@oskalashnikova ~]# rm logfile
[root@oskalashnikova ~]# ls
anaconda-ks.cfg  'g | grep -i "Linux version"'  revoke.asc
bin              git-extended                   work
conf.txt         LICENSE
file.txt         pandoc-1.19.2.1-1-amd64.deb
[root@oskalashnikova ~]#
```

Рис. 3.11: Удаление файла

Запускаю в консоли в фоновом режиме (с помощью символа &) редактор nano (рис. fig. 3.12).

```
[root@oskalashnikova ~]# nano &
[1] 3497
[root@oskalashnikova ~]#
```

Рис. 3.12: Создание фонового процесса

С помощью утилиты ps определяю идентификатор процесса mousepad(3497) (рис. fig. 3.13).

```
[root@oskalashnikova ~]# ps aux | grep nano
root      3497  0.0  0.0 223748 3584 pts/1    T   19:51   0:00 nano
root      3513  0.0  0.0 222456 2304 pts/1    S+  19:52   0:00 grep
--color=auto nano
[1]+  Остановлен  nano
[root@oskalashnikova ~]#
```

Рис. 3.13: Поиск идентификатора процесса

Второй способ (рис. fig. 3.14).

```
[root@oskalashnikova ~]# ps aux | grep nano | grep -v grep
root      3497  0.0  0.0 223748 3584 pts/1    T   19:51   0:00 nano
[root@oskalashnikova ~]#
```

Рис. 3.14: Поиск идентификатора процесса

Прочитала справку команды kill (рис. fig. 3.15).

```
KILL(1)                                User Commands                                KILL(1)

NAME
    kill - terminate a process

SYNOPSIS
    kill [-signal|-s signal|-p] [-q value] [-a] [--timeout
    milliseconds signal] [--] pid|name...

    kill -l [number] | -L
```

Рис. 3.15: Чтение документации

Используя команду kill и идентификатор процесса, чтобы его удалить (рис. fig. 3.16)

```
[root@oskalashnikova ~]# man kill
[root@oskalashnikova ~]# kill 3497
[root@oskalashnikova ~]#
```

Рис. 3.16: Удаление процесса

3.4 Проверка использования диска

Прочитала документацию про функции df (рис. fig. 3.17).

```
DF(1)                                User Commands                                DF(1)

NAME
    df - report file system space usage

SYNOPSIS
    df [OPTION]... [FILE]...
```

Рис. 3.17: Чтение документации

Прочитала документацию про функции du (рис. fig. 3.18).

```
DU(1)                                User Commands                                DU(1)

NAME
    du - estimate file space usage

SYNOPSIS
    du [OPTION]... [FILE]...
    du [OPTION]... --files0-from=F
```

Рис. 3.18: Чтение документации

Используя утилиту `df` опции `-iv` позволяют увидеть информацию об инодах и сделать вывод читаемым, игнорируя сообщения системы о нем. Эта утилита нам нужна, чтобы выяснить, сколько свободного места есть у нашей системы (рис. fig. 3.19).

```
[root@oskalashnikova ~]# man df
[root@oskalashnikova ~]# man du
[root@oskalashnikova ~]# df -vi
Файловая система  Инодов  ИИспользовано  ИСвободно  ИИспользовано%  Смонтиро
вано в
/dev/sda3          0          0          0          - /
devtmpfs           493258        535    492723        1% /dev
tmpfs              498519         8    498511        1% /dev/shm
efivarfs           0          0          0          - /sys/fir
mware/efi/efivars
tmpfs              819200        881    818319        1% /run
tmpfs              1048576        31   1048545        1% /tmp
/dev/sda3          0          0          0          - /home
/dev/sda2          65536         37    65499        1% /boot
/dev/sda1          0          0          0          - /boot/ef
i
work               1000       -999000   1000000          - /media/s
f_work
tmpfs              99703         107    99596        1% /run/use
r/1000
```

Рис. 3.19: Утилита `df`

Используя утилиту `du`. Она нужна чтобы просмотреть, сколько места занимают файлы в определенной директории и найти самые большие из них (рис. fig. 3.20).

```
[root@oskalashnikova ~]# du -a /home/oskalashnikova/monthly1/
4      /home/oskalashnikova/monthly1/april
4      /home/oskalashnikova/monthly1/may
4      /home/oskalashnikova/monthly1/june
12     /home/oskalashnikova/monthly1/
[root@oskalashnikova ~]#
```

Рис. 3.20: Утилита `du`

Прочитала документацию о команде `find` (рис. fig. 3.21).

```
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, accord-
    ing to the rules of precedence (see section OPERATORS), until
    the outcome is known (the left hand side is false for and oper-
    ations, true for or), at which point find moves on to the next
    file name.  If no starting-point is specified, .' is assumed.
```

Рис. 3.21: Чтение документации

Вывела имена всех директорий, имеющиххся в моем домашнем каталоге, используя аргумент `d` у утилиты `find` опции `-type`, то есть указываю тип файлов, который мне нужен и этот тип Директория (рис. fig. 3.22).

```
/home/oskalashnikova/reports/monthly/monthly
/home/oskalashnikova/monthly1
/home/oskalashnikova/ski.places
/home/oskalashnikova/ski.places/equipment
/home/oskalashnikova/ski.places/plans
/home/oskalashnikova/play
/home/oskalashnikova/play/games
[oskalashnikova@oskalashnikova ~]$ find ~ -type d
```

Рис. 3.22: Название рисунка

4 Выводы

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

5 Ответы на контрольные вопросы

1. Какие потоки ввода вывода вы знаете?

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.

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

- используется для перенаправления вывода в файл. Если файл уже существует, он будет перезаписан.
- также используется для перенаправления вывода в файл, но если файл существует, данные будут добавлены в конец файла без его перезаписи.

3. Что такое конвейер?

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

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

- Программа - это исполняемый файл, который хранится на диске и представляет из себя инструкции для выполнения определенной задачи.

- Процесс - это экземпляр программы, который выполняется в оперативной памяти компьютера. Процесс имеет своё пространство памяти, состояние и другие атрибуты, отличающие его от других процессов.

Главное отличие между программой и процессом заключается в том, что программа - это набор инструкций, который позволяет ЦПУ выполнять определенную задачу, в то время как процесс - это исполняемая программа.

5. Что такое PID и GID?

- PID (Process ID): Это уникальный идентификатор процесса в операционной системе Linux. Каждому запущенному процессу назначается свой уникальный PID. Процесс может порождать и другие процессы.
- GID (Group ID): Это идентификатор группы, к которой принадлежит пользователь. GID используется для управления доступом к файлам и ресурсам в Linux.

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

Задача в Linux - это отдельно выполняемый процесс или команда. Команда `ps` позволяет просматривать информацию о текущих задачах и процессах. Для управления задачами используются команды, такие как `kill` для завершения процессов.

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

- `top`: Утилита `top` отображает список процессов и ресурсоемкость системы в реальном времени. Позволяет просматривать и управлять запущенными процессами.
- `htop`: `htop` - это интерактивная версия `top`, предоставляющая дополнительные возможности для мониторинга процессов и ресурсов, такие как сортировка, фильтрация и управление процессами.

Обе показывают информацию о процессах в реальном времени, выводят данные о потреблении системных ресурсов и позволяют искать, останавливать и управлять процессами.

У обеих команд есть свои преимущества. Например, в программе `htop` реализован очень удобный поиск по процессам, а также их фильтрация. В команде `top` это не так удобно — нужно знать кнопку для вывода функции поиска.

Зато в `top` можно разделять область окна и выводить информацию о процессах в соответствии с разными настройками. В целом `top` намного более гибкая в настройке отображения процессов.

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

Команда `find` - это одна из наиболее важных и часто используемых утилит системы Linux. 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.

Утилита `find` предустановлена по умолчанию во всех Linux дистрибутивах, поэтому вам не нужно будет устанавливать никаких дополнительных пакетов. Это очень важная находка для тех, кто хочет использовать командную строку наиболее эффективно.

Команда `find` имеет такой синтаксис: `find [папка] [параметры] критерий шаблон [действие]` Пример: `find /etc -name "p*" -print`

Примеры использования:

- `find /home/user -name "*.txt"`: Найти все файлы с расширением `.txt` в директории `/home/user`.
- `find /var/log -type f -name "access.log"`: Найти файл `access.log` в директории `/var/log`.

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

```
find / -type f -exec grep -H 'текстДляПоиска' {} ;
```

Для поиска файла по контексту или содержанию можно использовать команду `grep`. Например, чтобы найти строку в файлах в текущем каталоге, вы можете выполнить:

```
grep "искомый_текст" *
```

Где "искомый_текст" - это текст, который нужно найти.

10. Как определить объем свободной памяти на жёстком диске?

Для определения объема свободной памяти на жестком диске можно использовать команду `df -h`, которая покажет информацию о дисковом пространстве в удобном для чтения формате.

11. Как определить объем вашего домашнего каталога?

Для определения объема вашего домашнего каталога вы можете воспользоваться командой `du -sh ~`, которая покажет размер вашего домашнего каталога в удобном формате.

12. Как удалить зависший процесс?

Чтобы удалить зависший процесс, можно воспользоваться командой `kill PID`, где PID - идентификатор процесса, который можно узнать с помощью команды `ps -aux`. Выполнение `kill` принудительно завершит процесс.