

Отчет по выполнению лабораторной работы №8

Операционные системы

Кузьмин Егор Витальевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	8
4	Выполнение лабораторной работы	9
5	Выводы	14
6	Ответы на контрольные вопросы	15
	Список литературы	17

Список иллюстраций

4.1	Запись в файл	9
4.2	Вывод содержимого файла	9
4.3	Добавление данных в файл, поиск файлов определенного расширения	10
4.4	Запись в файл, проверка	10
4.5	Поиск файлов, начинающихся с определенного элемента	10
4.6	Поиск файлов, начинающихся с определенного элемента	11
4.7	Поиск файлов, начинающихся с определенного элемента	11
4.8	Создание, запись, удаление фонового процесса	11
4.9	Создание фонового, поиск идентификатора процесса	12
4.10	Чтение документации	12
4.11	Удаление процесса, чтение документации	12
4.12	Утилиты df и du	13
4.13	Чтение документации	13
4.14	Вывод содержимого директорий	13

Список таблиц

1 Цель работы

Целью данной лабораторной работы является ознакомление с инструментами поиска файлов и фильтрации текстовых данных, а также приобретение практических навыков по управлению процессами, использованию диска и по обслуживанию файловых систем.

2 Задание

0. Ознакомиться с необходимой технической информацией
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 директо-
рий, имею- щихся в вашем домашнем каталоге.

3 Теоретическое введение

В интерфейсе командной строки есть очень полезная возможность перенаправления (переадресации) ввода и вывода (англ. термин I/O Redirection). Как мы уже заметили, многие программы выводят данные на экран. А ввод данных в терминале осуществляется с клавиатуры. С помощью специальных обозначений можно перенаправить вывод многих команд в файлы или иные устройства вывода (например, распечатать на принтере). То же самое и со вводом информации, вместо ввода данных с клавиатуры, для многих программ можно задать считывание символов их файла. Кроме того, можно даже вывод одной программы передать на ввод другой программе.

К каждой программе, запускаемой в командной строке, по умолчанию подключено три потока данных:

STDIN (0) — стандартный поток ввода (данные, загружаемые в программу). STDOUT (1) — стандартный поток вывода (данные, которые выводит программа). По умолчанию — терминал. STDERR (2) — стандартный поток вывода диагностических и отладочных сообщений (например, сообщениях об ошибках). По умолчанию — терминал.

Pipe (конвейер) – это однонаправленный канал межпроцессного взаимодействия. Термин был придуман Дугласом Макилроем для командной оболочки Unix и назван по аналогии с трубопроводом. Конвейеры чаще всего используются в shell-скриптах для связи нескольких команд путем перенаправления вывода одной команды (stdout) на вход (stdin) последующей, используя символ конвейера '|'.

4 Выполнение лабораторной работы

Записываю в файл file.txt названия файлов из каталога /etc с помощью перенаправления “>” (записав в него то, что могло быть выведено ls -lR /etc). В файл я включил также все файлы из подкаталогов (рис. 1).

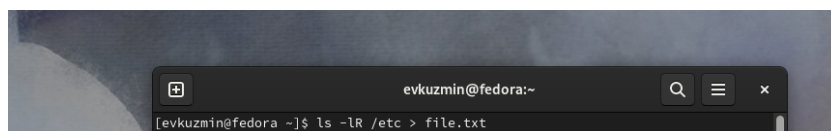


Рис. 4.1: Запись в файл

Проверяю, что в файл записались нужные значения, делая это с помощью утилиты head: она выводит первые 10 строк файла на экран (рис. 2).

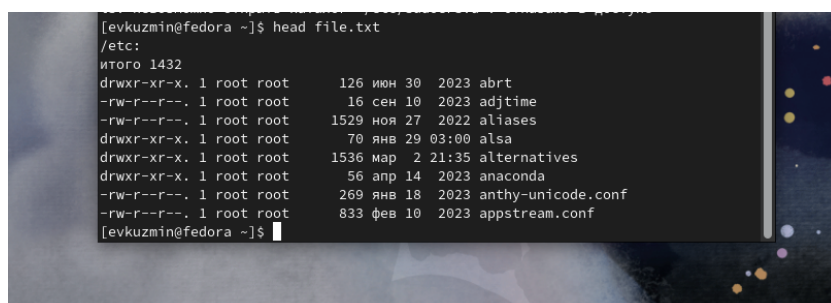
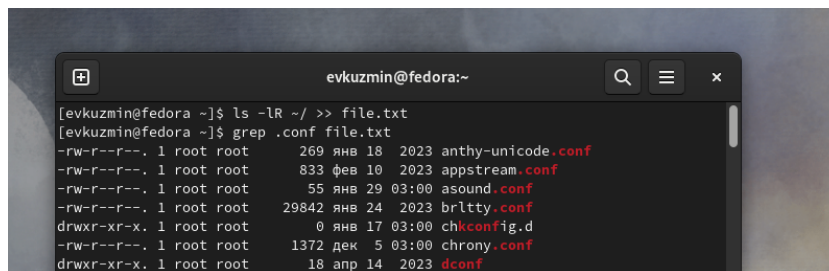


Рис. 4.2: Вывод содержимого файла

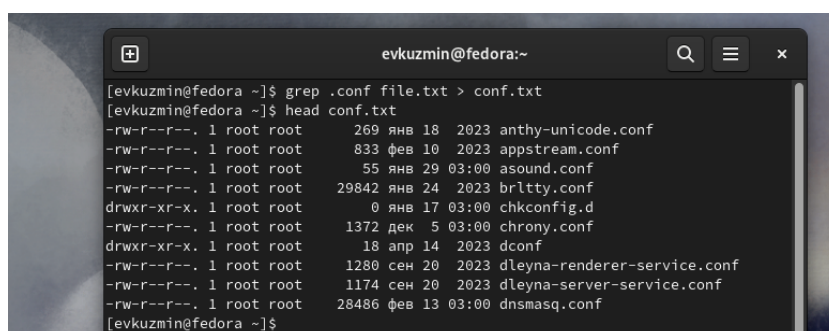
Добавляю в созданный файл имена файлов из домашнего каталога, используя перенаправление “>” в режиме добавления, вывожу на экран имена всех файлов, имеющих расширение “.conf” с помощью утилиты grep (рис. 3).



```
evkuzmin@fedora:~  
[evkuzmin@fedora ~]$ ls -lR ~/ >> file.txt  
[evkuzmin@fedora ~]$ grep .conf file.txt  
-rw-r--r--. 1 root root 269 янв 18 2023 anthy-unicode.conf  
-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 29842 янв 24 2023 brltty.conf  
drwxr-xr-x. 1 root root 0 янв 17 03:00 chkconfig.d  
-rw-r--r--. 1 root root 1372 дек 5 03:00 chrony.conf  
drwxr-xr-x. 1 root root 18 апр 14 2023 dconf
```

Рис. 4.3: Добавление данных в файл, поиск файлов определенного расширения

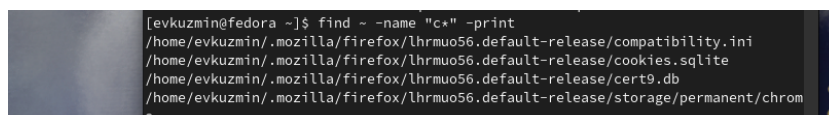
Добавляю вывод прошлой команды в новый файл conf.txt с помощью перенаправления “>” (файл создается при выполнении этой команды). Делаю проверку (рис. 4).



```
evkuzmin@fedora:~  
[evkuzmin@fedora ~]$ grep .conf file.txt > conf.txt  
[evkuzmin@fedora ~]$ head conf.txt  
-rw-r--r--. 1 root root 269 янв 18 2023 anthy-unicode.conf  
-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 29842 янв 24 2023 brltty.conf  
drwxr-xr-x. 1 root root 0 янв 17 03:00 chkconfig.d  
-rw-r--r--. 1 root root 1372 дек 5 03:00 chrony.conf  
drwxr-xr-x. 1 root root 18 апр 14 2023 dconf  
-rw-r--r--. 1 root root 1280 сен 20 2023 dleyna-renderer-service.conf  
-rw-r--r--. 1 root root 1174 сен 20 2023 dleyna-server-service.conf  
-rw-r--r--. 1 root root 28486 фев 13 03:00 dnsmasq.conf  
[evkuzmin@fedora ~]$
```

Рис. 4.4: Запись в файл, проверка

Первый способ состоит в том, чтобы определить, какие файлы в домашнем каталоге начинаются с символа “с”, делая это с помощью утилиты find. Прописываю ей в аргументах домашнюю директорию, выбираю опцию -name, и пишу маску, по которой будем искать имя, где * - любое кол-во любых символов, Аналогично добавляю опцию -print, чтобы мне вывелся результат (рис. 5)



```
evkuzmin@fedora ~]$ find ~ -name "c*" -print  
/home/evkuzmin/.mozilla/firefox/lhrmuo56.default-release/compatibility.ini  
/home/evkuzmin/.mozilla/firefox/lhrmuo56.default-release/cookies.sqlite  
/home/evkuzmin/.mozilla/firefox/lhrmuo56.default-release/cert9.db  
/home/evkuzmin/.mozilla/firefox/lhrmuo56.default-release/storage/permanent/chrom  
e
```

Рис. 4.5: Поиск файлов, начинающихся с определенного элемента

Второй способ - это 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. 6).

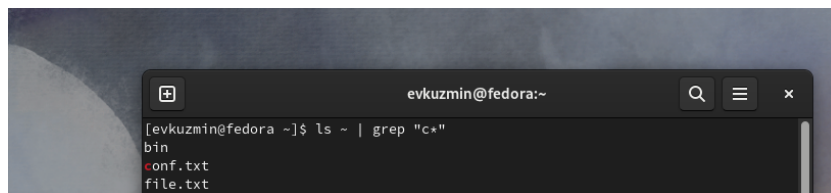


Рис. 4.6: Поиск файлов, начинающихся с определенного элемента

С помощью метода `find` ищу все файлы, начинающиеся с буквы “h” (рис. 7).

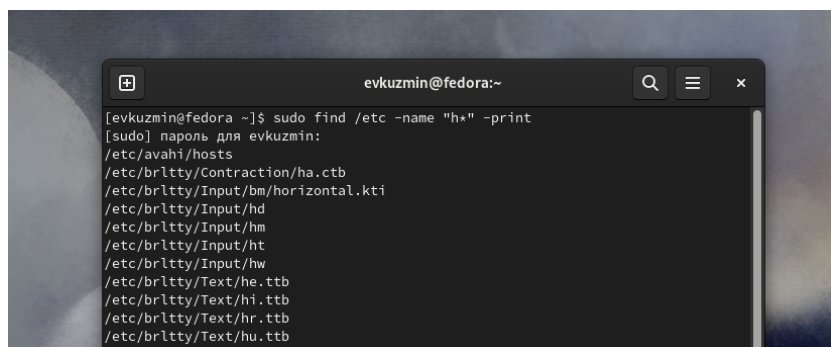


Рис. 4.7: Поиск файлов, начинающихся с определенного элемента

Запускаю в фоновом режиме, с помощью “&” процесс, который “>” будет записывать в logfile файлы, имена которых начинаются с log. Проверяю, что файл создан, удаляю его, проверяю, что файл удален (рис. 8).

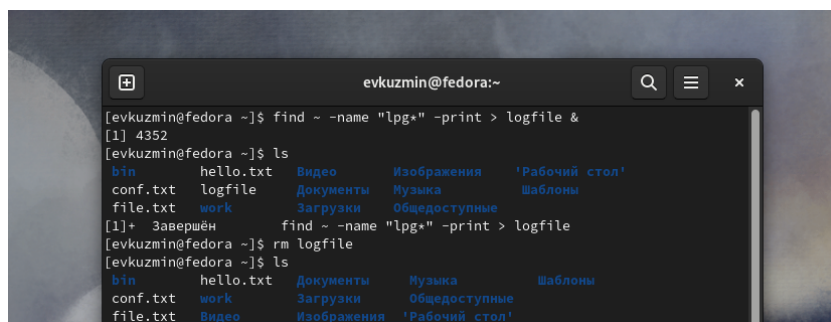


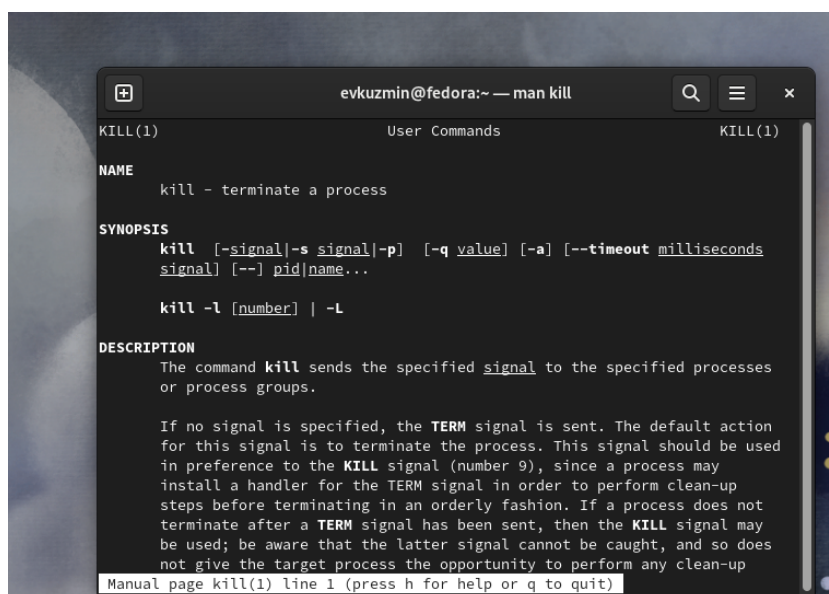
Рис. 4.8: Создание, запись, удаление фонового процесса

Запускаю в консоли в фоновом режиме редактор `gedit`. С помощью утилиты `ps` определяю идентификатор процесса. Также мы можем определить это значение с помощью `pgrep`, `pidof`. (рис. 9).

```
[evkuzmin@fedora ~]$ gedit &
[1] 4372
[evkuzmin@fedora ~]$ ps aux | grep "gedit"
evkuzmin  4372  3.6  3.6 854768 73572 pts/0    Sl   21:31   0:01 gedit
evkuzmin  4398  0.0  0.1 222436  2304 pts/0    S+   21:31   0:00 grep --color=
auto gedit
[evkuzmin@fedora ~]$ ps -fc gedit
    UID      PID  PPID  C  STIME TTY          TIME CMD
evkuzmin  4372   4091  2   21:31 pts/0        00:00:01 gedit
[evkuzmin@fedora ~]$ pidof gedit
4372
[evkuzmin@fedora ~]$
```

Рис. 4.9: Создание фонового, поиск идентификатора процесса

Читаю справку команды kill (рис. 10).



```
evkuzmin@fedora:~ — man kill
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

DESCRIPTION
    The command kill sends the specified signal to the specified processes
    or process groups.

    If no signal is specified, the TERM signal is sent. The default action
    for this signal is to terminate the process. This signal should be used
    in preference to the KILL signal (number 9), since a process may
    install a handler for the TERM signal in order to perform clean-up
    steps before terminating in an orderly fashion. If a process does not
    terminate after a TERM signal has been sent, then the KILL signal may
    be used; be aware that the latter signal cannot be caught, and so does
    not give the target process the opportunity to perform any clean-up

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

Рис. 4.10: Чтение документации

Использую команду kill и идентификатор процесса, чтобы его удалить. Процесс gedit закрылся. Далее читаю документацию про функции du и df (рис. 11).

```
[evkuzmin@fedora ~]$ kill 4509
[evkuzmin@fedora ~]$ man df
[2]+  Завершено      gedit
[evkuzmin@fedora ~]$ man du
[evkuzmin@fedora ~]$
```

Рис. 4.11: Удаление процесса, чтение документации

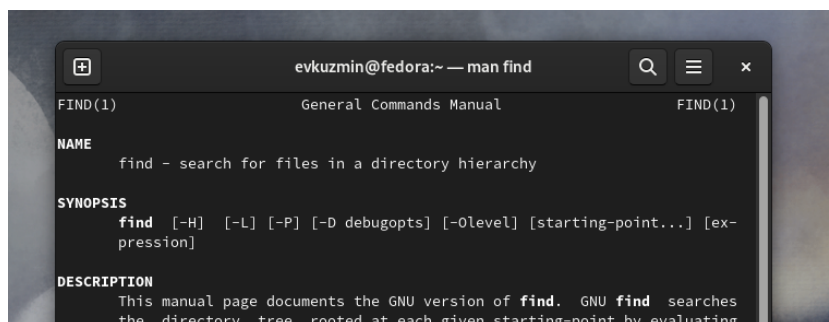
Использую утилиту df с опцией -vi, которая позволяет выяснить, сколько свободного места есть у нашей системы. Использую утилиту du. Она нужна чтобы

просмотреть, сколько места занимают файлы в определенной директории (рис. 12).

```
[evkuzmin@fedora ~]$ man du
[evkuzmin@fedora ~]$ df -vi
Файловая система  Инодов  ИИспользовано  Исвободно  ИИспользовано%  Смонтировано в
devtmpfs          246150      512      245638          1% /dev
tmpfs             251289        2      251287          1% /dev/shm
tmpfs            819200      981      818219          1% /run
/dev/sda3          0            0            0            - /
tmpfs           1048576       53     1048523          1% /tmp
/dev/sda2         65536       401      65135          1% /boot
/dev/sda3          0            0            0            - /home
tmpfs            50257       153      50104          1% /run/user/1000
/dev/sr0           0            0            0            - /run/media/evkuz
min/VBox_GAs_7.0.10
[evkuzmin@fedora ~]$ du -a file.txt
488      file.txt
[evkuzmin@fedora ~]$
```

Рис. 4.12: Утилиты df и du

Читаю документацию о команде find (рис. 13).



```
evkuzmin@fedora:~ — man find
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...] [ex-
    pression]

DESCRIPTION
    This manual page documents the GNU version of find. GNU find searches
    the directory tree rooted at each given starting-point by evaluating
```

Рис. 4.13: Чтение документации

Вывожу имена всех директорий, имеющихся в моем домашнем каталоге, используя аргумент `d` у утилиты `find`, а также опцию `-type`, указывающую тип файлов, который мне нужен (рис. 14)

```
[evkuzmin@fedora ~]$ find ~ -type d
/home/evkuzmin
/home/evkuzmin/.mozilla
/home/evkuzmin/.mozilla/extensions
/home/evkuzmin/.mozilla/extensions/{ec8030f7-c20a-464f-9b0e-13a3a9e97384}
/home/evkuzmin/.mozilla/plugins
/home/evkuzmin/.mozilla/firefox
```

Рис. 4.14: Вывод содержимого директорий

5 Выводы

В результате данной лабораторной работы я ознакомился с инструментами поиска файлов и фильтрации текстовых данных, а также приобрел практические навыки по управлению процессами, по проверке дисков и обслуживанию файловых систем

6 Ответы на 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.
2. Этот знак > - перенаправление ввода/вывода, а » - перенаправление в режиме добавления.
3. Конвейер (pipe) служит для объединения простых команд или утилит в цепочки, в которых результат работы предыдущей команды передаётся последующей.
4. 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. PPID - (parent process ID) идентификатор родительского процесса. Процесс может порождать и другие процессы. UID, GID - реальные идентификаторы пользователя и его группы, запустившего данный процесс.
6. Запущенные фоном программы называются задачами (jobs). Ими можно управлять с помощью команды jobs, которая выводит список запущенных в данный момент задач.

7. Команда `htop` похожа на команду `top` по выполняемой функции: они 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`

9. `find / -type f -exec grep -H 'текстДляПоиска' {} ;`
10. С помощью команды `df -h`.
11. С помощью команды `du -s`.
12. С помощью команды `kill` номер задачи.

Список литературы

Архитектура компьютеров и ОС/Электронный ресурс