

Отчёт по лабораторной работе №6

Дисциплина: Операционные системы

Подъярова Ксения Витальевна (группа: НПМбд-02-21)

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	11
4	Ответы на контрольные вопросы:	12

Список иллюстраций

2.1	Файл file.txt	6
2.2	Файлы из file.txt, имеющие расширение .conf	7
2.3	Имена каталогов, начинающиеся с символа c	7
2.4	Имена каталогов, начинающиеся с символа c	7
2.5	Имена каталогов, начинающиеся с символа h	8
2.6	Запуск фонового режима	8
2.7	Удаление файла ~/logfile	8
2.8	Редактор gedit	8
2.9	Редактор gedit	8
2.10	Команда man kill	9
2.11	Справка команды kill	9
2.12	Справка команды df	9
2.13	Справка команды du	9
2.14	Справка команды find	10

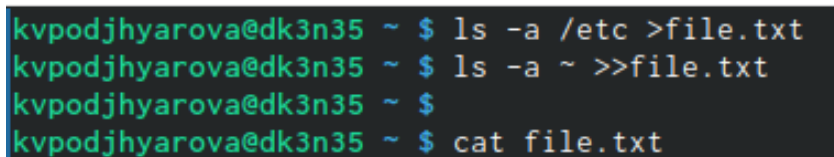
Список таблиц

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 Выполнение лабораторной работы

1. Осуществляем вход в систему, используя свое имя пользователя.
2. Записываю в файл file.txt названия файлов, содержащиеся в каталоге /etc. Дописываю в этот же файл названия файлов, содержащиеся в домашнем каталоге.(рис. 2.1)

A screenshot of a terminal window with a dark background and green text. It shows four lines of commands being executed by a user named 'kvpodjhyarova' on a machine named 'dk3n35'. The commands are: 'ls -a /etc >file.txt', 'ls -a ~ >>file.txt', a blank line, and 'cat file.txt'.

```
kvpodjhyarova@dk3n35 ~ $ ls -a /etc >file.txt
kvpodjhyarova@dk3n35 ~ $ ls -a ~ >>file.txt
kvpodjhyarova@dk3n35 ~ $
kvpodjhyarova@dk3n35 ~ $ cat file.txt
```

Рис. 2.1: Файл file.txt

3. Вывожу имена всех файлов из file.txt, имеющие расширение .conf, после чего записываю их в новый текстовый файл conf.txt.(рис. 2.2)

```

kvpodjhyarova@dk3n35 ~ $ grep -e '\.conf$' file.txt >conf.txt
kvpodjhyarova@dk3n35 ~ $ cat conf.txt
appstream.conf
brlty.conf
ca-certificates.conf
cachefilesd.conf
cfg-update.conf
cpufreq-bench.conf
dhcpcd.conf
dispatch-conf.conf
dleyna-server-service.conf
dnsmasq.conf
dracut.conf
e2fsck.conf
e2scrub.conf
etc-update.conf
fluidsynth.conf
fuse.conf
gai.conf
genkernel.conf
gssapi_mech.conf
host.conf
idmapd.conf
idn2.conf
idnalias.conf
krb5.conf

```

Рис. 2.2: Файлы из file.txt, имеющие расширение .conf

4. Определяю, какие файлы в домашнем каталоге имеют имена, начинающиеся с символа с. (рис. 2.3) (рис. 2.4)

```

kvpodjhyarova@dk3n35 ~ $ find ~ -maxdepth 1 -name "с*" -print
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/colors.css
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-active.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-backdrop-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-backdrop-active.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/gtk-3.0/assets/close-backdrop-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/kdeconnect/certificate.pem
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/kdeconnect/config
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/pulse/cookie
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/.config/caja

```

Рис. 2.3: Имена каталогов, начинающиеся с символа с

```

kvpodjhyarova@dk3n35 ~ $ ls ~/с*
/afs/.dk.sci.pfu.edu.ru/home/k/v/kvpodjhyarova/conf.txt
kvpodjhyarova@dk3n35 ~ $ ls -a ~ | grep с*
conf.txt

```

Рис. 2.4: Имена каталогов, начинающиеся с символа с

5. Вывожу на экран имена файлов из каталога /etc, начинающиеся с символа h.(рис. 2.5)

```
kvpodjhyarova@dk3n35 ~ $ find /etc -maxdepth 1 -name "h*" | less
```

Рис. 2.5: Имена каталогов, начинающиеся с символа h

6. Запускаю в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log.(рис. 2.6)

```
kvpodjhyarova@dk3n35 ~ $ find / -name "log*" >logfile &  
[2] 5842
```

Рис. 2.6: Запуск фонового режима

7. Удаляю файл ~/logfile.(рис. 2.7)

```
kvpodjhyarova@dk3n35 ~ $ rm -r logfile
```

Рис. 2.7: Удаление файла ~/logfile

8. Запускаю из консоли в фоновом режиме редактор gedit.(рис. 2.8) (рис. 2.9)

```
kvpodjhyarova@dk3n35 ~ $ gedit &  
[1] 6432
```

Рис. 2.8: Редактор gedit



Рис. 2.9: Редактор gedit

9. Читаю справку (man) команды kill, после чего использую её для завершения процесса gedit.(рис. 2.10) (рис. 2.11)


```
kvpodjhyarova@dk3n35 ~ $ man kill
```

Рис. 2.10: Команда man kill

```
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 -t to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and
  0. Alternate signals may be specified in three ways: -s, -SIGKILL or -KILL. Negative PID values may be used to choose whole process groups; see
  the PIDs 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.

  -s <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 <signal>(1) 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 the signal, it will receive the signal and return from the handler. If the process does not have a handler,
    it will be terminated.
```

Рис. 2.11: Справка команды kill

10. Выполняя команды df и du, предварительно получив более подробную информацию об этих командах, с помощью команды man.(рис. 2.12) (рис. 2.13)

```
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 arg-
  ument. 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 de-
  fault, 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
```

Рис. 2.12: Справка команды df

```
du(1) User Commands du(1)
NAME
  du - estimate file space usage

SYNOPSIS
  du [OPTION]... [FILE]...
  du [OPTION]... --files-from <file>

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
```

Рис. 2.13: Справка команды du

11. Воспользовавшись справкой команды find, вывожу имена всех директорий, имеющих в домашнем каталоге(рис. 2.14)

```
./os-intro/kvpodjhyarova.github.io/.git/info
./os-intro/kvpodjhyarova.github.io/.git/hooks
./os-intro/kvpodjhyarova.github.io/.git/refs
./os-intro/kvpodjhyarova.github.io/.git/refs/heads
./os-intro/kvpodjhyarova.github.io/.git/refs/tags
./os-intro/kvpodjhyarova.github.io/.git/refs/remotes
./os-intro/kvpodjhyarova.github.io/.git/refs/remotes/origin
./os-intro/kvpodjhyarova.github.io/.git/objects
./os-intro/kvpodjhyarova.github.io/.git/objects/pack
./os-intro/kvpodjhyarova.github.io/.git/objects/info
./os-intro/kvpodjhyarova.github.io/.git/objects/e6
./os-intro/kvpodjhyarova.github.io/.git/objects/f9
./os-intro/kvpodjhyarova.github.io/.git/objects/4f
./os-intro/kvpodjhyarova.github.io/.git/logs
./os-intro/kvpodjhyarova.github.io/.git/logs/refs
./os-intro/kvpodjhyarova.github.io/.git/logs/refs/heads
./os-intro/kvpodjhyarova.github.io/.git/logs/refs/remotes
./os-intro/kvpodjhyarova.github.io/.git/logs/refs/remotes/origin
kvpodjhyarova@dk3n35 ~ $ find -type d
```

Рис. 2.14: Справка команды find

3 Выводы

Я ознакомилась с инструментами поиска файлов и фильтрации текстовых данных, приобрела практические навыки: по управлению процессами, по проверке использования диска и обслуживанию файловых систем.

4 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. 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: 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. 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: команда1|команда2 (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 (processID), 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 stop, 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 дамп 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 ре-

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. команда1|команда2 (это означает, что вывод команды 1 передается 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 (`processID`), 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 фоновые 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 `stop`, 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 ~/`