

# **Отчет по лабораторной работе №6**

**Операционные системы**

Ничипорова Елена Дмитриевна

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Контрольные вопросы	12

# Список иллюстраций

2.1	Команды df и du . . . . .	10
2.2	Команды df . . . . .	10
2.3	Команды du . . . . .	11

## **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

- Осуществила вход в систему, используя соответствующее имя пользователя
- Записала в файл file.txt название файлов, содержащихся в определенном каталоге, далее дописываю названия файлов, содержащихся в домашнем каталоге просматриваю файл, чтобы убедиться в правильности действий(рис. ??)

```
ednichiporova@dk3n53 ~ $ ls -a /etc >file.txt
ednichiporova@dk3n53 ~ $ ls -a ~ - >>file.txt
ednichiporova@dk3n53 ~ $ cat file.txt
.
..
a2ps
acpi
adjtime
afs.keytab
alsa
apache2
apparmor.d
appstream.conf
ati
audisp
audit
autofs
avahi
bash
bash_completion.d
bindresvport.blacklist
binfmt.d
blkid.tab.old
bluetooth
brlty
brlty.conf
ca-certificates
ca-certificates.conf
cachefilesd.conf
cfg-update.conf
cfg-update.hosts
cgroup
chromium
chrony
cifs-utils
common-lisp
conf.d
cpufreq-bench.conf
cron.d
cron.daily
cron.hourly
cron.monthly
crontab
cron.weekly
csh.cshrc
csh.env
csh.login
cups
cupshelpers
dbus-1
dconf
default
```

- Вывела имена всех

файлов из file.txt, имеющих расширение .conf и записала их в новый текстовый файл.(рис. ??)

```
ednichiporova@dk3n53 ~ $ grep -e '\.conf$' file.txt > conf.txt
ednichiporova@dk3n53 ~ $ cat conf.txt
appstream.conf
brltty.conf
ca-certificates.conf
cachefilesd.conf
cfg-update.conf
cpufreq-bench.conf
dhcpcd.conf
dispatch-conf.conf
dleyna-server-service.conf
dnsmasq.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
ldap.conf
ld.so.conf
libaudit.conf
lightdm.conf
locale.conf
logrotate.conf
mailutils.conf
make.conf
man.conf
man_db.conf
mdadm.conf
metalog.conf
mke2fs.conf
mlocate-cron.conf
modules.conf
mplayer.conf
nscd.conf
nslcd.conf
nss-ldapd.conf
```

- Определяю, ка-

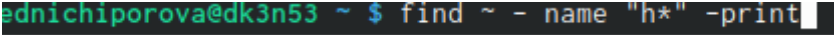
кие файлы в домашнем каталоге имеют имена, начинающиеся с символа c несколькими командами(рис. ??)

```
ednichiporova@dk3n53 ~ $ find ~ -maxdepth 1 -name "c*" -print
ednichiporova@dk3n53 ~ $ find ~ -name "c*" -print
ednichiporova@dk3n53 ~ $ ls ~/c*
bash: ls ~/c*: Нет такого файла или каталога
ednichiporova@dk3n53 ~ $ ls ~/c*
/afs/.dk.sci.pfu.edu.ru/home/e/d/ednichiporova/conf.txt
ednichiporova@dk3n53 ~ $ ls -a ~ | grep c*
conf.txt
ednichiporova@dk3n53 ~ $
```

- Вывожу на экран

файлы из каталога /etc , начинающиеся с символа h(рис. ??)



 - Запускаю в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log. Так как в фоновом режиме запустился непрерывный процесс записывания файла, я сделала скриншот некоторой его части.(рис. ??)

```
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/KONEC.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/WWWWW.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGA.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/2383MART.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGAZA2.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/RRRRRR.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/KONTROLL.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/XZ.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/ANKA.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/MATRIX.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/IA20.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/APR6.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/APREL.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/6APR.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/DATA.TXT': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/QQ.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/23MART.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/FRF.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/RANDOM1.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/P_9.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGAZA.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/99MART.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/RESULT.TXT': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/UUYUY.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/KONTROL2.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGAZZ.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/NINA.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/TYUI.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGAZ.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/RRR.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/ANKA.RES': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/PROGAZAS.PAS': Отказано в доступе
find: '/afs/dk.sci.pfu.edu.ru/home/a/a/aakuchina/work/TITTTTTI.PAS': Отказано в доступе
```

- Удаляю файл

~/logfile (рис. ??)

```
ednichiporova@dk3n53 ~ $ rm logfile
```

- Запускаю редактор

gedit в фоновом режиме. После этого на экране появляется окно редактора(рис. ??)

```
ednichiporova@dk3n53 ~ $ gedit &
[1] 8138
```

- Определяю идентификатор процесса gedit

Читаю информацию о команде kill с помощью команды “man kill”(рис. ??)

```
kill(1)                                User Commands
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:
  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.

  -s[signal]
  -S -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.

  -l, --show-signal
    Use show-signal(2) 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_tinfo(2), then it can obtain this data via the si_value field of the siginfo_t structure.

  -L, --list [-,+]
    List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.

  -t, --table
    List signal names in a nice table.

NOTES
  Your shell (command line interpreter) may have a built-in kill command. You may need to run the command described here as /bin/kill to solve the conflict.

EXAMPLES
  kill -9 -1
    Kill all processes you can kill.

  kill -1 11
    Translate number 11 into a signal name.

  kill -t
    List the available signal choices in a nice table.

  kill 123 543 7141 8451
    Send the default signal, SIGTERM, to all those processes.

SEE ALSO
  kill(2), killall(1), nice(1), shill(1), renice(1), signal(7), signame(3), shill(1)
```

-С помощью команд

“man df” “man du” узнаю информацию по необходимым командам и далее ис-

пользую ее(рис. 2.1)

```
ednichiporova@dk3n53 ~ $ man df
ednichiporova@dk3n53 ~ $ man du
ednichiporova@dk3n53 ~ $
```

Рис. 2.1: Команды df и du

- df -утилита, показывающая список всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования(рис. 2.2)

```
df(1)                                User Commands
NAME
  df - report file system disk space usage
SYNOPSIS
  df [-H] [-k] [-P] [-t FILESType]...
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 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.  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., 1024M)
  -k, --k
      print sizes in powers of 1000 (e.g., 1.1G)
  -l, --inodes
      list inode information instead of block usage
  -x, --file-type=FILESType
      like --block-size=
  -L, --local
      limit listing to local file systems
  --no-sync
      do not invoke sync before getting usage info (default)
  --output=FIELD_LIST
      use the output format defined by FIELD_LIST, or print all fields if FIELD_LIST is omitted.
  -P, --portability
      use the POSIX output format
  --sync
      invoke sync before getting usage info
  --info
```

Рис. 2.2: Команды df

- du- утилита, предназначенная для вывода информации об объеме дискового пространства, занятого файлами и директориями(рис. 2.3)

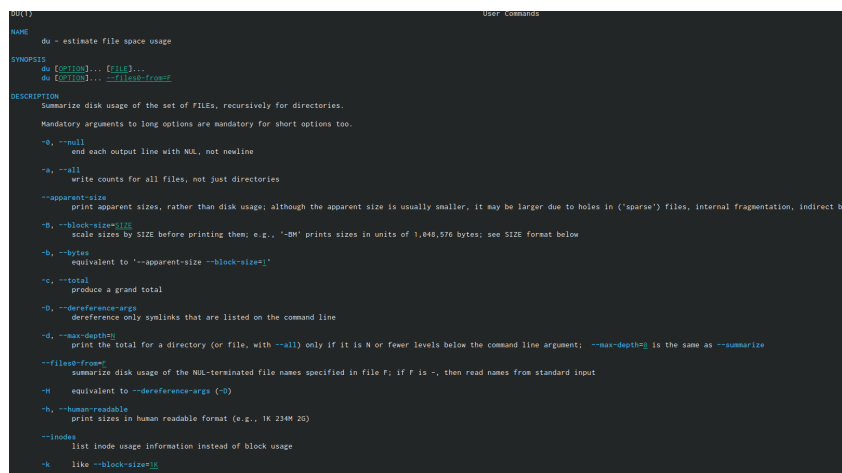
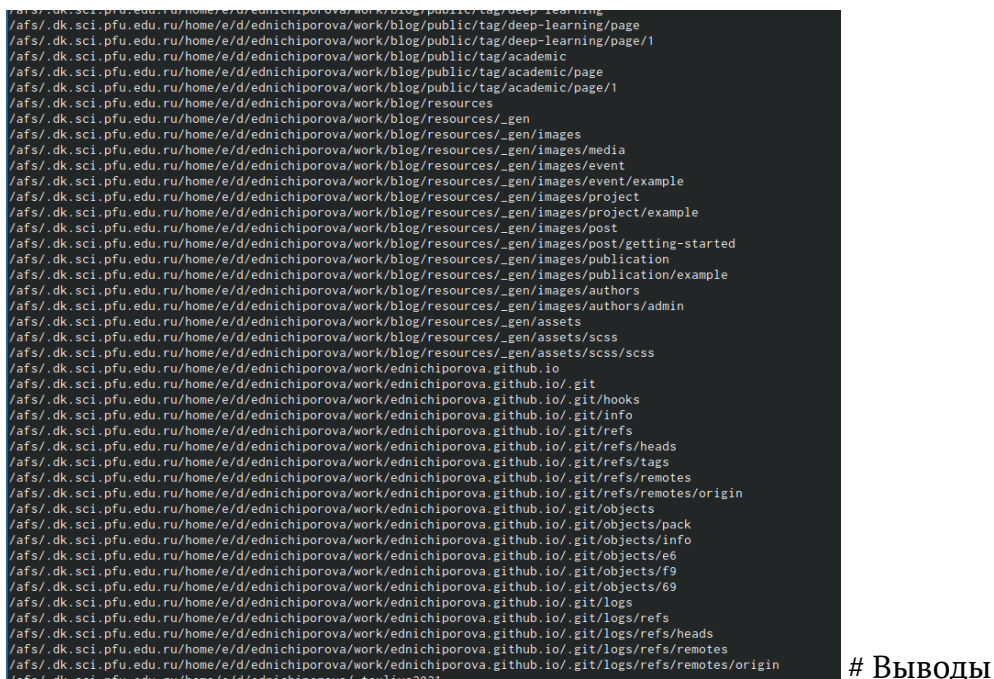


Рис. 2.3: Команды du

- Вывела имена всех директорий, имеющих в домашнем каталоге с помощью команды “find ~ -type d”, предварительно получив информацию с помощью команды “man kill”(рис. ??)



В ходе выполнения данной лабораторной работы я изучила инструменты поиска файлофильтрации текстовых данных, а также приобрела практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

### 3 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

-typef - искать только файлы

-typed - 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 пр

`SIGQUIT`–это еще один сигнал, который отправляется с помощью сочетания клавиш, пр

`SIGHUP`–сообщает процессу, что соединение с управляющим терминалом разорвано, отпр

`SIGTERM`–немедленно завершает процесс, но обрабатывается программой, поэтому позво

`SIGKILL`–тоже немедленно завершает процесс, но, в отличие от предыдущего варианта,

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` (вканале) и будет выполнять поиск по 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.