РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет Физико-Математических Наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

дисциплина: Операционные системы

Студент: Манаева Варвара Евгеньевна

Группа: НФИбд-01-20

Преподаватель: Кулябов Дмитрий Сергеевич

МОСКВА

2021 г.

Техническое оснащение:

- Персональный компьютер с операционной системой Windows 7;
- Планшет для записи видеосопровождения и голосовых комментариев;
- Виртуальная коробка VirtualBox, виртуальная машина с установленной на ней операционной системой CentOS;
- Microsoft Teams, использующийся для записи скринкаста лабораторной работы;
- Приложение MarkPad 2 для редактирования файлов формата *md*;
- *pandoc* для конвертации файлов отчётов и презентаций.

Объект и предмет исследования: Инструмент поиска файлов, файловая система *Linux*.

Цель: ознакомиться с инструментами поиска файлов и фильтрации текстовых данных, приобрести практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

Задачи:

1) Научиться искать файлы различными методами по различным признакам;
2) Научиться управлять процессами и заданиями;
3) Научиться проверять использование диска и обслуживание файловых систем.

Теоретические вводные данные:

Перенаправление ввода-вывода 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 вывода 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*. Например, команда *ls* выводит в стандартный поток вывода (консоль) список файлов в текущей директории. Потоки вывода и ввода можно перенаправлять на другие файлы или устройства. Проще всего это делается с помощью символов >, >>, <, <<. Рассмотрим пример.

Перенаправление stdout (вывода) в файл.
Если файл отсутствовал, то он создаётся,
иначе -- перезаписывается.
Создаёт файл, содержащий список дерева каталогов.
ls -lR > dir-tree.list
1>filename
Перенаправление вывода (stdout) в файл "filename".
1>>filename
Перенаправление вывода (stdout) в файл "filename",
файл открывается в режиме добавления.
2>filename
Перенаправление stderr в файл "filename".
2>>filename
Перенаправление stderr в файл "filename",
файл открывается в режиме добавления.
&>filename
Перенаправление stdout и stderr в файл "filename".

Конвейер. Конвейер (*pipe*) служит для объединения простых команд или утилит в цепочки, в которых результат работы предыдущей команды передаётся последующей. Синтаксис следующий: *команда 1* | *команда 2*
означает, что вывод команды 1 передастся на ввод команде 2

Конвейеры можно группировать в цепочки и выводить с помощью перенаправления в файл, например:
ls -la |sort > sortilglist
вывод команды ls -la передаётся команде сортировки sort\verb, которая пишет результат в файл sortinglist\verb.
Чаще всего скрипты на *Bash* используются в качестве автоматизации каких-то рутинных операций в консоли, отсюда иногда возникает необходимость в обработке *stdout* одной команды и передача на *stdin* другой команде, при этом результат выполнения одной команды

должен быть неким образом обработан.

Поиск файла. Команда *find* используется для поиска и отображения имён файлов, соответствующих заданной строке символов.
Формат команды:
find путь [-опции]
Путь определяет каталог, начиная с которого по всем подкаталогам будет вестись поиск.

Примеры:
1) Вывести на экран имена файлов из вашего домашнего каталога и его подкаталогов, начинающихся на *f*:
find ~ -name "f" -print* , где ~ — обозначение вашего домашнего каталога,
-name — после этой опции указывается имя файла, который нужно найти,
"f"* — строка символов, определяющая имя файла,
-print — опция, задающая вывод результатов поиска на экран.

2) Вывести на экран имена файлов в каталоге */etc*, начинающихся с символа *p*:
find /etc -name "p" -print*

3) Найти в вашем домашнем каталоге файлы, имена которых заканчиваются символом и удалить их:
find ~ -name "~" -exec rm "{}" \;*
где опция *-exec rm "{}" \;* задаёт применение команды *rm* ко всем файлам, имена которых соответствуют указанной после опции *-name* строке символов.

Для просмотра полного списка опций команды *find* воспользуйтесь командой *man*.

Фильтрация текста. Найти в текстовом файле указанную строку символов позволяет команда *grep*.
Формат команды:
grep строка имя_файла

Кроме того, команда *grep* способна обрабатывать стандартный вывод других команд (любой текст). Для этого следует использовать конвейер, связав вывод команды с вводом *grep*.

Примеры:
1) Показать строки во всех файлах в вашем домашнем каталоге с именами, начинающимися на *f*, в которых есть слово *begin*: *grep begin f**

2) Найти в текущем каталоге все файлы, в имени которых есть буквосочетание «лаб»:
ls -l | grep лаб

Проверка использования диска. Команда *df* показывает размер каждого смонтированного раздела диска.
Формат команды:
df [-опции] [файловая_система]

Пример:
df -vi

Команда *du* показывает число килобайт, используемое каждым файлом или каталогом.
Формат команды:
du [-опции] [имя_файла...]

Пример.
du -a ~/

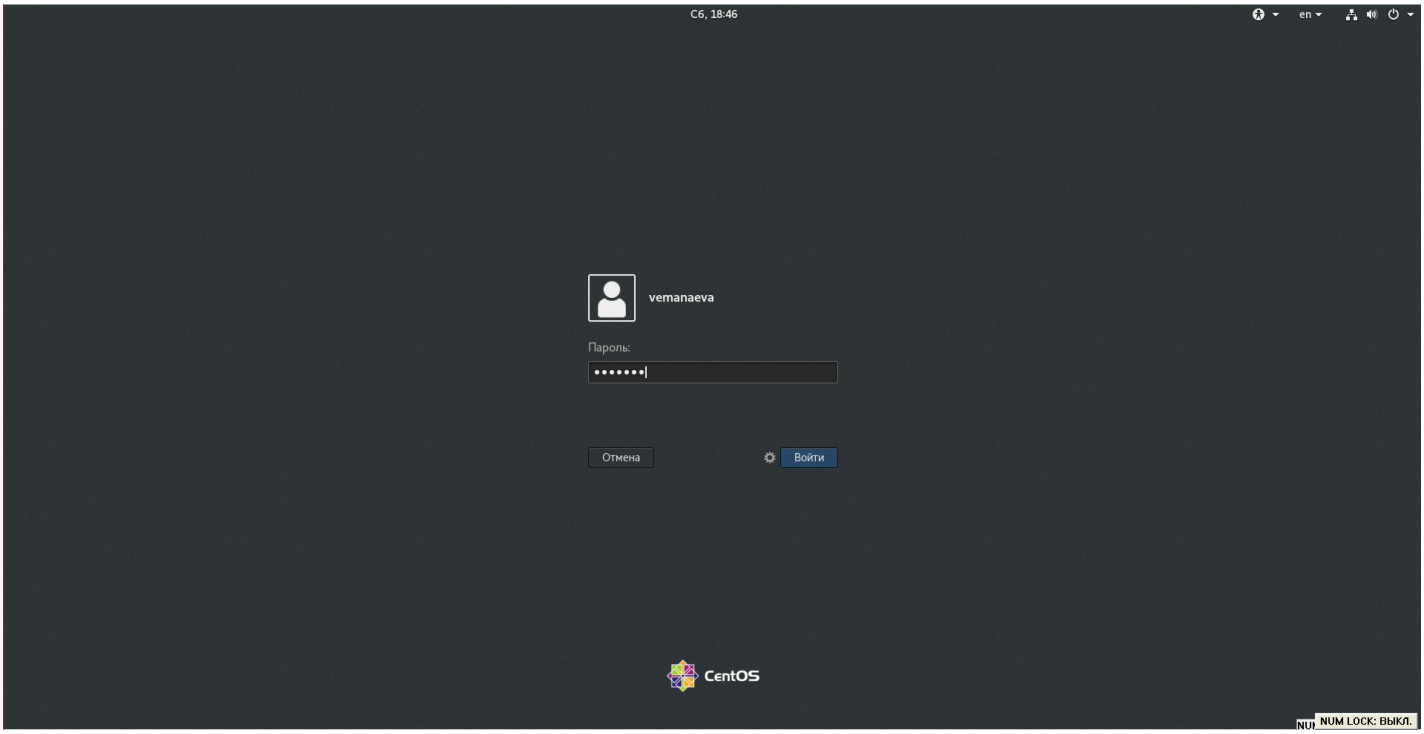
На *afs* можно посмотреть использованное пространство командой
fs quota

Управление задачами. Любую выполняющуюся в консоли команду или внешнюю программу можно запустить в фоновом режиме. Для этого следует в конце имени команды указать знак амперсанда (*&*). Например:
gedit &
Будет запущен текстовой редактор *gedit* в фоновом режиме. Консоль при этом не будет заблокирована.

Запущенные фоном программы называются задачами (*jobs*). Ими можно управлять с помощью команды *jobs*, которая выводит список запущенных в данный момент задач. Для завершения задачи необходимо выполнить команду
kill %номер задачи

Этапы работы:

1) Осуществила вход в систему, используя соответствующее имя пользователя.



2) Записала в файл *file.txt* названия файлов, содержащихся в каталоге */etc*. Дописала в этот же файл названия файлов, содержащихся в моём домашнем каталоге.

```
[vemanaeva@vemanaeva ~]$ ls /etc > /home/vemanaeva/file.txt
[vemanaeva@vemanaeva ~]$ ls ~ >> file.txt
```

3) Вывела имена всех файлов из *file.txt*, имеющих расширение *.conf*, после чего записала их в новый текстовой файл *conf.txt*.

```
asound.conf
brltty.conf
chkconfig.d
chrony.conf
dconf
dleyna-server-service.conf
dnsmasq.conf
dracut.conf
dracut.conf.d
e2fsck.conf
fprintd.conf
fuse.conf
gconf
GeoIP.conf
host.conf
idmapd.conf
ipsec.conf
kdump.conf
krb5.conf
krb5.conf.d
ksmtuned.conf
ld.so.conf
ld.so.conf.d
libaudit.conf
libuser.conf
locale.conf
logrotate.conf
makedumpfile.conf.sample
man_db.conf
mke2fs.conf
mtools.conf
netconfig
nfs.conf
nfsmount.conf
nsswitch.conf
nsswitch.conf.bak
numad.conf
oddjobd.conf
oddjobd.conf.d
pbm2ppa.conf
pnm2ppa.conf
prelink.conf.d
radvd.conf
request-key.conf
resolv.conf
resolv.conf.7ARE20
rsyncd.conf
rsyslog.conf
sestatus.conf
sos.conf
sudo.conf
sudo-ldap.conf
sysconfig
sysctl.conf
tcsd.conf
updatedb.conf
usb_modeswitch.conf
vconsole.conf
wvdial.conf
yum.conf
[vemanaeva@vemanaeva ~]$ grep .conf file.txt > conf.txt
[vemanaeva@vemanaeva ~]$
```

4) Определила, какие файлы в вашем домашнем каталоге имеют имена, начинавшиеся с символа "c". Несколько вариантов, как это сделать:

```
[vemanaeva@vemanaeva ~]$ ls -l | grep c*
-rw-rw-r--. 1 vemanaeva vemanaeva  769 май 15 18:54 conf.txt
[vemanaeva@vemanaeva ~]$ find ~/c* -name "c*" -print
/home/vemanaeva/conf.txt
[vemanaeva@vemanaeva ~]$
```

5) Вывела на экран (постранично) имена файлов из каталога */etc*, начинающиеся с символа "h".

```
[vemanaeva@vemanaeva etc]$ ls -l | grep h* | less
[vemanaeva@vemanaeva etc]$
```

6) Запустила в фоновом режиме процесс, который будет записывать в файл *~/logfile* файлы, имена которых начинаются с "log".

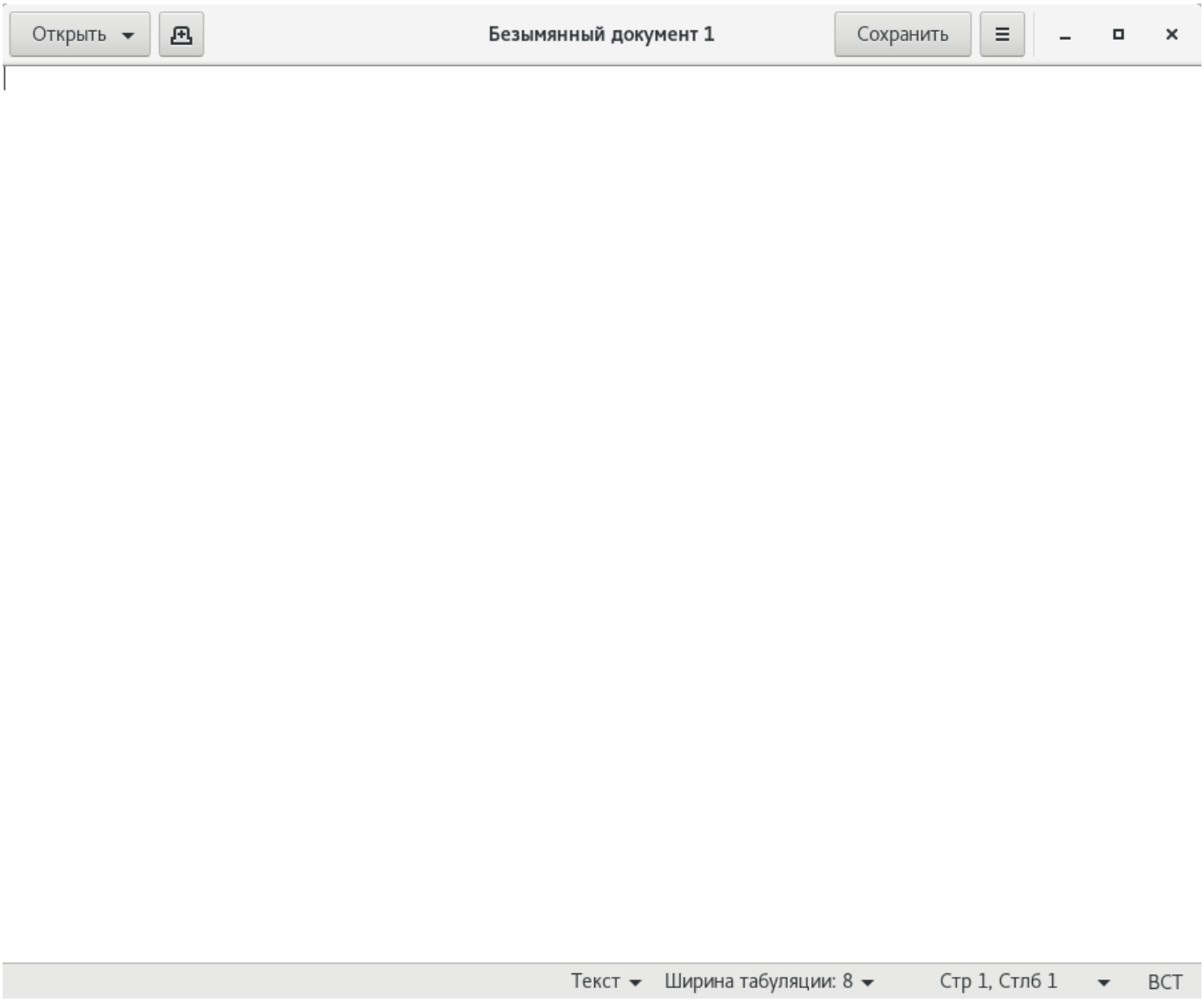
```
[vemanaeva@vemanaeva etc]$ find ~ -name "log*" -print > ~/logfile &
[1] 6496
```

7) Удалила файл *~/logfile*.

```
[vemanaeva@vemanaeva ~]$ rm logfile
[vemanaeva@vemanaeva ~]$
```

8) Запустила из консоли в фоновом режиме редактор *gedit*.

```
[vemanaeva@vemanaeva ~]$ gedit &
[1] 6556
[vemanaeva@vemanaeva ~]$
```



9) Определила идентификатор процесса *gedit*, используя команду *ps*, конвейер и фильтр *grep*. Более простым способом определить этот идентификатор получилось: просто введя команду ps aux и глазками отыскав её в списке. Насколько этот способ проще судить не берусь.

```
[vemanaeva@vemanaeva ~]$ ps aux | grep -i gedit
vemanae+  6556  0.5  4.0 752052 41572 pts/0    Sl   19:03   0:00 gedit
vemanae+  6580  0.0  0.0 112832   996 pts/0    R+   19:04   0:00 grep --color=auto -i gedit
```

```
[vemanaeva@vemanaeva ~]$ ps aux
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root          1  0.0  0.4 193964  4780 ?        Ss   18:04   0:01 /usr/lib/systemd/systemd --switched-root --system --deserialize 22
root          2  0.0  0.0      0     0 ?        S    18:04   0:00 [kthreadd]
root          4  0.0  0.0      0     0 ?        S<   18:04   0:00 [kworker/0:0H]
root          6  0.0  0.0      0     0 ?        S    18:04   0:00 [ksoftirqd/0]
root          7  0.0  0.0      0     0 ?        S    18:04   0:00 [migration/0]
root          8  0.0  0.0      0     0 ?        S    18:04   0:00 [rcu_bh]
root          9  0.0  0.0      0     0 ?        R    18:04   0:01 [rcu_sched]
root         10  0.0  0.0      0     0 ?        S<   18:04   0:00 [lru-add-drain]
root         11  0.0  0.0      0     0 ?        S    18:04   0:00 [watchdog/0]
root         13  0.0  0.0      0     0 ?        S    18:04   0:00 [kdevtmpfs]
root         14  0.0  0.0      0     0 ?        S<   18:04   0:00 [netns]
root         15  0.0  0.0      0     0 ?        S    18:04   0:00 [khungtaskd]
root         16  0.0  0.0      0     0 ?        S<   18:04   0:00 [writeback]
root         17  0.0  0.0      0     0 ?        S<   18:04   0:00 [kintegrityd]
root         18  0.0  0.0      0     0 ?        S<   18:04   0:00 [bioset]
root         19  0.0  0.0      0     0 ?        S<   18:04   0:00 [bioset]
root         20  0.0  0.0      0     0 ?        S<   18:04   0:00 [bioset]
root         21  0.0  0.0      0     0 ?        S<   18:04   0:00 [kblockd]
```

```
vemanae+  6556  0.2  4.0 752052 41572 pts/0    Sl   19:03   0:00 gedit
```

10) Прочитала справку (*man*) команды *kill*, после чего использовала её для завершения процесса *gedit*.

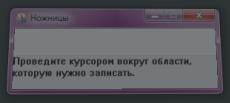
```
KILL(1)                                                    User Commands                                                    KILL(1)

NAME
       kill - terminate a process

SYNOPSIS
       kill [-s signal|-p] [-q sigval] [-a] [--] pid...
       kill -l [signal]

DESCRIPTION
       The  command  kill  sends the specified signal to the specified process or process group.  If no signal is specified, the TERM signal is sent.  The TERM signal will kill processes which do not catch
       this signal.  For other processes, it may be necessary to use the KILL (9) signal, since this signal cannot be caught.

       Most modern shells have a builtin kill function, with a usage rather similar to that of the command described here.  The '-a' and '-p' options, and the possibility to specify  processes  by  command
       name are a local extension.

       If sig is 0, then no signal is sent, but error checking is still performed.

OPTIONS
       pid... Specify the list of processes that kill should signal.  Each pid can be one of five things:

              n      where n is larger than 0.  The process with pid n will be signaled.

              0      All processes in the current process group are signaled.

              -1     All processes with pid larger than 1 will be signaled.

              -n     where n is larger than 1.  All processes in process group n are signaled.  When an argument of the form '-n' is given, and it is meant to denote a process group, either the signal must
                     be specified first, or the argument must be preceded by a '--' option, otherwise it will be taken as the signal to send.

              commandname
                     All processes invoked using that name will be signaled.

       -s, --signal signal
              Specify the signal to send.  The signal may be given as a signal name or number.

       -l, --list [signal]
              Print a list of signal names, or convert signal given as argument to a name.  The signals are found in /usr/include/linux/signal.h

       -L, --table
              Similar to -l, but will print signal names and their corresponding numbers.

       -a, --all
              Do not restrict the commandname-to-pid conversion to processes with the same uid as the present process.
```

```
[vemanaeva@vemanaeva ~]$ man kill
[vemanaeva@vemanaeva ~]$ kill 6556
[1]+  Завершено      gedit
```

11) Выполнила команды *df* и *du*, предварительно получив более подробную информацию об этих командах, с помощью команды *man*.

```
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 avail-
       able 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 SIZE format below

       --direct
              show statistics for a file instead of mount point

       --total
              produce a grand total

       -h, --human-readable
              print sizes in human readable format (e.g., 1K 234M 2G)

       -H, --si
              likewise, but use powers of 1000 not 1024

       -i, --inodes
              list inode information instead of block usage

       -k     like --block-size=1K

       -l, --local
              limit listing to local file systems
```

```
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 each FILE, recursively for directories.

       Mandatory arguments to long options are mandatory for short options too.

       -0, --null
              end each output line with 0 byte rather than 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
              equivalent to '--apparent-size --block-size=1'

       -c, --total
              produce a grand total

       -D, --dereference-args
              dereference only symlinks that are listed on the command line

       -d, --max-depth=N
              print the total for a directory (or file, with --all) only if it is N or fewer levels below the command line argument;  --max-depth=0 is the same as --summarize

       --files0-from=F
              summarize disk usage of the NUL-terminated file names specified in file F; if F is -, then read names from standard input

       -H     equivalent to --dereference-args (-D)

       -h, --human-readable
```




```
[vemanaeva@vemanaeva ~]$ man du
[vemanaeva@vemanaeva ~]$ man df
[vemanaeva@vemanaeva ~]$ df
Файловая система        1K-блоков Использовано Доступно Использовано% Смонтировано в
devtmpfs                   490560            0   490560            0% /dev
tmpfs                      507376            0   507376            0% /dev/shm
tmpfs                      507376         7812   499564            2% /run
tmpfs                      507376            0   507376            0% /sys/fs/cgroup
/dev/mapper/centos-root  38770180      5040448 33729732           14% /
/dev/sda1                 1038336       175100   863236           17% /boot
tmpfs                      101476           28   101448            1% /run/user/1000
[vemanaeva@vemanaeva ~]$ du
0       ./.mozilla/extensions
0       ./.mozilla/plugins
0       ./.mozilla
0       ./.cache/gdm
8       ./.cache/imsettings
0       ./.cache/libgweather
0       ./.cache/evolution/addressbook/trash
0       ./.cache/evolution/addressbook
0       ./.cache/evolution/calendar/trash
0       ./.cache/evolution/calendar
0       ./.cache/evolution/mail/trash
0       ./.cache/evolution/mail
0       ./.cache/evolution/memos/trash
0       ./.cache/evolution/memos
0       ./.cache/evolution/sources/trash
0       ./.cache/evolution/sources
0       ./.cache/evolution/tasks/trash
0       ./.cache/evolution/tasks
0       ./.cache/evolution
0       ./.cache/gnome-shell
4       ./.cache/abrt
3340    ./.cache/tracker
0       ./.cache/flatpak/system-cache
0       ./.cache/flatpak
0       ./.cache/media-art
488     ./.cache/gstreamer-1.0
740     ./.cache/gnome-software/shell-extensions
1056    ./.cache/gnome-software/odrs
1796    ./.cache/gnome-software
0       ./.cache/yelp/WebKitCache/Version 16/Blobs
4       ./.cache/yelp/WebKitCache/Version 16
4       ./.cache/yelp/WebKitCache
4       ./.cache/yelp
32      ./.cache/thumbnails/large
32      ./.cache/thumbnails
```

```
5688    ./.cache
4       ./.dbus/session-bus
4       ./.dbus
0       ./.config/imsettings
0       ./.config/gnome-session/saved-session
0       ./.config/gnome-session
84      ./.config/pulse
4       ./.config/ibus/bus
4       ./.config/ibus
12      ./.config/dconf
4       ./.config/evolution/sources
4       ./.config/evolution
0       ./.config/goa-1.0
0       ./.config/gconf
4       ./.config/gtk-3.0
0       ./.config/abrt
4       ./.config/nautilus
4       ./.config/yelp
0       ./.config/enchant
128     ./.config
8       ./.local/share/keyrings
4       ./.local/share/gnome-shell
0       ./.local/share/evolution/addressbook/trash
0       ./.local/share/evolution/addressbook/system/photos
24      ./.local/share/evolution/addressbook/system
24      ./.local/share/evolution/addressbook
0       ./.local/share/evolution/calendar/trash
4       ./.local/share/evolution/calendar/system
4       ./.local/share/evolution/calendar
0       ./.local/share/evolution/mail/trash
0       ./.local/share/evolution/mail
0       ./.local/share/evolution/memos/trash
0       ./.local/share/evolution/memos
0       ./.local/share/evolution/tasks/trash
0       ./.local/share/evolution/tasks
28      ./.local/share/evolution
8       ./.local/share/telepathy/mission-control
8       ./.local/share/telepathy
0       ./.local/share/sounds
0       ./.local/share/applications
0       ./.local/share/icc
0       ./.local/share/gnome-settings-daemon
232     ./.local/share/tracker/data
232     ./.local/share/tracker
0       ./.local/share/flatpak/db
0       ./.local/share/flatpak/repo/tmp/cache
```

```
0       ./.local/share/flatpak/repo/tmp
0       ./.local/share/flatpak/repo/extensions
0       ./.local/share/flatpak/repo/state
0       ./.local/share/flatpak/repo/refs/heads
0       ./.local/share/flatpak/repo/refs/mirrors
0       ./.local/share/flatpak/repo/refs/remotes
0       ./.local/share/flatpak/repo/refs
0       ./.local/share/flatpak/repo/objects
4       ./.local/share/flatpak/repo
4       ./.local/share/flatpak
72      ./.local/share/gvfs-metadata
0       ./.local/share/nautilus/scripts
0       ./.local/share/nautilus
1128    ./.local/share/app-info/xmls
1128    ./.local/share/app-info
0       ./.local/share/webkitgtk/deviceidhashsalts/1
0       ./.local/share/webkitgtk/deviceidhashsalts
0       ./.local/share/webkitgtk/databases/indexeddb/v
0       ./.local/share/webkitgtk/databases/indexeddb
0       ./.local/share/webkitgtk/databases
0       ./.local/share/webkitgtk/localstorage
0       ./.local/share/webkitgtk
1488    ./.local/share
1488    ./.local
0       ./Рабочий стол
0       ./Загрузки
0       ./Шаблоны
0       ./Общедоступные
0       ./Документы
0       ./Музыка
0       ./Изображения
0       ./Видео
12      ./.ssh
0       ./.pki/nssdb
0       ./.pki
0       ./lab08
7380    .
[vemanaeva@vemanaeva ~]$
```

12) Воспользовавшись справкой команды *find*, вывела имена всех директорий, имеющихся в моём домашнем каталоге.

```
FIND(1)                                                    General Commands Manual                                                    FIND(1)

NAME
       find - search for files in a directory hierarchy

SYNOPSIS
       find [-H] [-L] [-P] [-D debugopts] [-Olevel] [path...] [expression]

DESCRIPTION
       This manual page documents the GNU version of find.  GNU find searches the directory tree rooted at each given file name by evaluating the given expression from left to right, according to the rules
       of precedence (see section OPERATORS), until the outcome is known (the left hand side is false for and operations, true for or), at which point find moves on to the next file name.

       If you are using find in an environment where security is important (for example if you are using it to search directories that are writable by other users), you should read the "Security Considera-
       tions"  chapter  of  the findutils documentation, which is called Finding Files and comes with findutils.   That document also includes a lot more detail and discussion than this manual page, so you
       may find it a more useful source of information.

OPTIONS
       The -H, -L and -P options control the treatment of symbolic links.  Command-line arguments following these are taken to be names of files or directories to be examined, up to the first argument that
       begins  with  `-',  or  the  argument `(' or `!'.  That argument and any following arguments are taken to be the expression describing what is to be searched for.  If no paths are given, the current
       directory is used.  If no expression is given, the expression -print is used (but you should probably consider using -print0 instead, anyway).

       This manual page talks about `options' within the expression list.  These options control the behaviour of find but are specified immediately after the last path name.  The five `real'  options  -H,
       -L,  -P, -D and -O must appear before the first path name, if at all.  A double dash -- can also be used to signal that any remaining arguments are not options (though ensuring that all start points
       begin with either `./' or `/' is generally safer if you use wildcards in the list of start points).

       -P     Never follow symbolic links.  This is the default behaviour.  When find examines or prints information a file, and the file is a symbolic link, the information used shall be  taken  from  the
              properties of the symbolic link itself.

       -L     Follow  symbolic  links.  When find examines or prints information about files, the information used shall be taken from the properties of the file to which the link points, not from the link
              itself (unless it is a broken symbolic link or find is unable to examine the file to which the link points).  Use of this option implies -noleaf.  If you later use the -P option, -noleaf will
              still be in effect.  If -L is in effect and find discovers a symbolic link to a subdirectory during its search, the subdirectory pointed to by the symbolic link will be searched.

              When  the  -L  option  is in effect, the -type predicate will always match against the type of the file that a symbolic link points to rather than the link itself (unless the symbolic link is
              broken).  Using -L causes the -lname and -ilname predicates always to return false.

       -H     Do not follow symbolic links, except while processing the command line arguments.  When find examines or prints information about files, the information used shall be taken from  the  proper-
              ties  of  the  symbolic link itself.   The only exception to this behaviour is when a file specified on the command line is a symbolic link, and the link can be resolved.  For that situation,
              the information used is taken from whatever the link points to (that is, the link is followed).  The information about the link itself is used as a fallback if the file pointed to by the sym-
              bolic  link  cannot  be  examined.   If  -H  is in effect and one of the paths specified on the command line is a symbolic link to a directory, the contents of that directory will be examined
              (though of course -maxdepth 0 would prevent this).

       If more than one of -H, -L and -P is specified, each overrides the others; the last one appearing on the command line takes effect.  Since it is the default, the -P option should be considered to be
       in effect unless either -H or -L is specified.

       GNU  find  frequently stats files during the processing of the command line itself, before any searching has begun.  These options also affect how those arguments are processed.  Specifically, there
```

```
[vemanaeva@vemanaeva ~]$ man find
[vemanaeva@vemanaeva ~]$ find ~ -type d -print
/home/vemanaeva
/home/vemanaeva/.mozilla
/home/vemanaeva/.mozilla/extensions
/home/vemanaeva/.mozilla/plugins
/home/vemanaeva/.cache
/home/vemanaeva/.cache/gdm
/home/vemanaeva/.cache/imsettings
/home/vemanaeva/.cache/libgweather
/home/vemanaeva/.cache/evolution
/home/vemanaeva/.cache/evolution/addressbook
/home/vemanaeva/.cache/evolution/addressbook/trash
/home/vemanaeva/.cache/evolution/calendar
/home/vemanaeva/.cache/evolution/calendar/trash
/home/vemanaeva/.cache/evolution/mail
/home/vemanaeva/.cache/evolution/mail/trash
/home/vemanaeva/.cache/evolution/memos
/home/vemanaeva/.cache/evolution/memos/trash
/home/vemanaeva/.cache/evolution/sources
/home/vemanaeva/.cache/evolution/sources/trash
/home/vemanaeva/.cache/evolution/tasks
/home/vemanaeva/.cache/evolution/tasks/trash
/home/vemanaeva/.cache/gnome-shell
/home/vemanaeva/.cache/abrt
/home/vemanaeva/.cache/tracker
/home/vemanaeva/.cache/flatpak
/home/vemanaeva/.cache/flatpak/system-cache
/home/vemanaeva/.cache/media-art
/home/vemanaeva/.cache/gstreamer-1.0
/home/vemanaeva/.cache/gnome-software
/home/vemanaeva/.cache/gnome-software/shell-extensions
/home/vemanaeva/.cache/gnome-software/odrs
/home/vemanaeva/.cache/yelp
/home/vemanaeva/.cache/yelp/WebKitCache
/home/vemanaeva/.cache/yelp/WebKitCache/Version 16
/home/vemanaeva/.cache/yelp/WebKitCache/Version 16/Blobs
/home/vemanaeva/.cache/thumbnails
/home/vemanaeva/.cache/thumbnails/large
/home/vemanaeva/.dbus
/home/vemanaeva/.dbus/session-bus
/home/vemanaeva/.config
/home/vemanaeva/.config/imsettings
/home/vemanaeva/.config/gnome-session
/home/vemanaeva/.config/gnome-session/saved-session
/home/vemanaeva/.config/pulse
/home/vemanaeva/.config/ibus
```

```
/home/vemanaeva/.config/ibus/bus
/home/vemanaeva/.config/dconf
/home/vemanaeva/.config/evolution
/home/vemanaeva/.config/evolution/sources
/home/vemanaeva/.config/goa-1.0
/home/vemanaeva/.config/gconf
/home/vemanaeva/.config/gtk-3.0
/home/vemanaeva/.config/abrt
/home/vemanaeva/.config/nautilus
/home/vemanaeva/.config/yelp
/home/vemanaeva/.config/enchant
/home/vemanaeva/.local
/home/vemanaeva/.local/share
/home/vemanaeva/.local/share/keyrings
/home/vemanaeva/.local/share/gnome-shell
/home/vemanaeva/.local/share/evolution
/home/vemanaeva/.local/share/evolution/addressbook
/home/vemanaeva/.local/share/evolution/addressbook/trash
/home/vemanaeva/.local/share/evolution/addressbook/system
/home/vemanaeva/.local/share/evolution/addressbook/system/photos
/home/vemanaeva/.local/share/evolution/calendar
/home/vemanaeva/.local/share/evolution/calendar/trash
/home/vemanaeva/.local/share/evolution/calendar/system
/home/vemanaeva/.local/share/evolution/mail
/home/vemanaeva/.local/share/evolution/mail/trash
/home/vemanaeva/.local/share/evolution/memos
/home/vemanaeva/.local/share/evolution/memos/trash
/home/vemanaeva/.local/share/evolution/tasks
/home/vemanaeva/.local/share/evolution/tasks/trash
/home/vemanaeva/.local/share/telepathy
/home/vemanaeva/.local/share/telepathy/mission-control
/home/vemanaeva/.local/share/sounds
/home/vemanaeva/.local/share/applications
/home/vemanaeva/.local/share/icc
/home/vemanaeva/.local/share/gnome-settings-daemon
/home/vemanaeva/.local/share/tracker
/home/vemanaeva/.local/share/tracker/data
/home/vemanaeva/.local/share/flatpak
/home/vemanaeva/.local/share/flatpak/db
/home/vemanaeva/.local/share/flatpak/repo
/home/vemanaeva/.local/share/flatpak/repo/tmp
/home/vemanaeva/.local/share/flatpak/repo/tmp/cache
/home/vemanaeva/.local/share/flatpak/repo/extensions
/home/vemanaeva/.local/share/flatpak/repo/state
/home/vemanaeva/.local/share/flatpak/repo/refs
/home/vemanaeva/.local/share/flatpak/repo/refs/heads
```

```
/home/vemanaeva/.local/share/flatpak/repo/refs/mirrors
/home/vemanaeva/.local/share/flatpak/repo/refs/remotes
/home/vemanaeva/.local/share/flatpak/repo/objects
/home/vemanaeva/.local/share/gvfs-metadata
/home/vemanaeva/.local/share/nautilus
/home/vemanaeva/.local/share/nautilus/scripts
/home/vemanaeva/.local/share/app-info
/home/vemanaeva/.local/share/app-info/xmls
/home/vemanaeva/.local/share/webkitgtk
/home/vemanaeva/.local/share/webkitgtk/deviceidhashsalts
/home/vemanaeva/.local/share/webkitgtk/deviceidhashsalts/1
/home/vemanaeva/.local/share/webkitgtk/databases
/home/vemanaeva/.local/share/webkitgtk/databases/indexeddb
/home/vemanaeva/.local/share/webkitgtk/databases/indexeddb/v1
/home/vemanaeva/.local/share/webkitgtk/localstorage
/home/vemanaeva/Рабочий стол
/home/vemanaeva/Загрузки
/home/vemanaeva/Шаблоны
/home/vemanaeva/Общедоступные
/home/vemanaeva/Документы
/home/vemanaeva/Музыка
/home/vemanaeva/Изображения
/home/vemanaeva/Видео
/home/vemanaeva/.ssh
/home/vemanaeva/.pki
/home/vemanaeva/.pki/nssdb
/home/vemanaeva/lab08
[vemanaeva@vemanaeva ~]$
```

Выводы: я ознакомилась с инструментами поиска файлов и фильтрацией текстовых данных, приобрела практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

Контрольные вопросы:

1) Какие потоки ввода вывода вы знаете?

Ответ: Существует несколько потоков ввода вывода:

- stdin — стандартный поток ввода (по умолчанию: клавиатура), файловый дескриптор 0;
- *stdout* — стандартный поток вывода (по умолчанию: консоль), файловый дескриптор 1;
- *stderr* — стандартный поток вывода сообщений об ошибках (по умолчанию: консоль), файловый дескриптор 2.

2) Объясните разницу между операцией > и >>.

Ответ: > - перенаправление stdout (вывода) в файл. Если файл отсутствовал, то он создаётся, иначе - перезаписывается.
>> - перенаправление stdout (вывода) в файл. Если файл отсутствовал, то он создаётся, иначе - добавляется.

3) Что такое конвейер?

Ответ: 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?

Ответ: Процесс - это совокупность программного кода и данных, загруженных в память ЭВМ. Любой команде, выполняемой в системе, присваивается идентификатор процесса (*process ID*). Получить информацию о процессе и управлять им, пользуясь идентификатором процесса, можно из любого окна командного интерпретатора. Процессом называют выполняющуюся программу и все её элементы: адресное пространство, глобальные переменные, регистры, стек, открытые файлы и так далее.

5) Что такое PID и GID?

Ответ: *PID* — уникальный номер (идентификатор) процесса в многозадачной ОС.
GID – идентификатор группы.

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

Ответ: Запущенные фоном программы называются задачами (*jobs*). 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?

Ответ: Команда *top* в *Linux* системах позволяет вывести в виде таблицы перечень запущенных процессов и оценить, какой объем ресурсов они потребляют, т.е., какую нагрузку создают на сервер и дисковую подсистему.

Команда *htop* — продвинутый монитор процессов, показывает динамический список системных процессов, список обычно выравнивается по использованию ЦПУ. В отличие от *top*, *htop* показывает все процессы в системе. Также показывает время непрерывной работы, использование процессоров и памяти. *htop* часто применяется в тех случаях, когда информации, даваемой утилитой *top* недостаточно, например, при поиске утечек памяти в процессах.

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

Ответ: Команда *find* используется для поиска и отображения имён файлов, соответствующих заданной строке символов. Формат команды:
find путь [-опции]
Путь определяет каталог, начиная с которого по всем подкаталогам будет вестись поиск.
Примеры:

- вывести на экран имена файлов из вашего домашнего каталога и его подкаталогов, начинающихся на f: *find ~ -name "f*" -print*
- вывести на экран имена файлов в каталоге */etc*, начинающихся с символа p: *find /etc -name "p*" -print*
- найти в вашем домашнем каталоге файлы, имена которых заканчиваются символом и удалить их: *find ~ -name "*~" -exec rm "{}" \;*

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

Ответ: Найти файл по контексту (содержанию) позволяет команда *grep*.
Формат команды: *grep строка имя_файла*
Примеры:

- показать строки во всех файлах в вашем домашнем каталоге с именами, начинающимися на f, в которых есть слово begin:
  *grep begin f**
- найти в текущем каталоге все файлы, в имени которых есть буквосочетание «лаб»:
  ls -l | grep лаб

10) Как определить объём свободной памяти на жёстком диске?

Ответ: Определить объём свободной памяти на жёстком диске позволяет команда *df*.

11) Как определить объём вашего домашнего каталога?

Ответ: Определить объём домашнего каталога позволяет команда *df /home/*.

12) Как удалить зависший процесс?

Ответ: Удалить зависший процесс можно командой *kill %номер задачи*.