

Отчёт по лабораторной работе №7

дисциплина: Операционные системы

Быстров Глеб Андреевич

Содержание

1	Цель работы	3
2	Теория	4
3	Задание	5
4	Выполнение лабораторной работы	6
5	Контрольные вопросы	13
6	Выводы	17
7	Библиографический список	18

1 Цель работы

В данной лабораторной работе мне будет необходимо ознакомиться с инструментами поиска файлов и фильтрации текстовых данных. Будет необходимо преобрести практические навыки по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

2 Теория

- 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.
- Конвейер (`pipe`) служит для объединения простых команд или утилит в цепочки, в которых результат работы предыдущей команды передаётся последующей.
- Команда `find` используется для поиска и отображения имён файлов, соответствующих заданной строке символов.
- Найти в текстовом файле указанную строку символов позволяет команда `grep`.
- Команда `df` показывает размер каждого смонтированного раздела диска.
- Команда `ps` используется для получения информации о процессах.

3 Задание

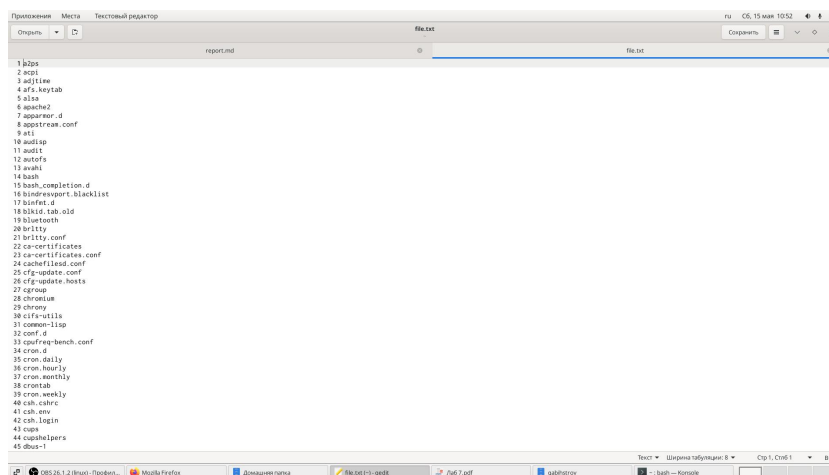
Провести работу в домашнем каталоге согласно инструкции. Использовать команды для создания, изменения, поиска, исследования и перемещения файлов.

4 Выполнение лабораторной работы

1. Записал в файл file.txt названия файлов, содержащихся в каталоге /etc. Дописал в этот же файл названия файлов, содержащихся в вашем домашнем каталоге. (рис. 4.1) (рис. 4.2)

```
gabihstrov@dk6n64 ~ $ touch file.txt
gabihstrov@dk6n64 ~ $ ls /etc >file.txt
gabihstrov@dk6n64 ~ $ ls >>file.txt
gabihstrov@dk6n64 ~ $
```

Figure 4.1: Работа с файлом file.txt



```
1 h2ps
2 apt
3 addline
4 a/s.keytab
5 x11
6 apache2
7 apparmor.d
8 appstream.conf
9 x11
10 audisp
11 audit
12 autofs
13 avahi
14 bash
15 bash_completion.d
16 bindresolv.conf
17 binfmt.d
18 blist.tab.old
19 bluetooth
20 brltty
21 brltty.conf
22 ca-certificates
23 ca-certificates.conf
24 castrim.conf
25 cfm-update.conf
26 cfm-update.hosts
27 cgroup
28 chromium
29 cron
30 cifs-utils
31 connman
32 conf.d
33 confd
34 cron.d
35 cron.daily
36 cron.hourly
37 cron.monthly
38 cronstab
39 cron.weekly
40 csh.cshrc
41 csh.env
42 csh.login
43 cups
44 cupsd.conf
45 dbus-1
```

Figure 4.2: Названия файлов из домашнего каталога

2. Вывел имена всех файлов из file.txt, имеющих расширение .conf, после чего записал их в новый текстовый файл conf.txt. (рис. 4.3) (рис. 4.4)

```

gabihstrov@dk6n64 ~ $ grep .conf file.txt
appstream.conf
brltty.conf
ca-certificates.conf
cachefilesd.conf
cfg-update.conf
cpufreq-bench.conf
dconf
dhcpcd.conf
dispatch-conf.conf
dleyna-server-service.conf
dnsmasq.conf
e2fsck.conf
e2scrub.conf
etc-update.conf
fluidsynth.conf

```

Figure 4.3: Работа с файлами file.txt и conf.txt

```

gabihstrov@dk6n64 ~ $ grep .conf file.txt >conf.txt
gabihstrov@dk6n64 ~ $

```

Figure 4.4: Записал в новый файл conf.txt

3. Определил, какие файлы в вашем домашнем каталоге имеют имена, начинающиеся с символа с. Сделал это двумя способами. (рис. 4.5) (рис. 4.6)

```

gabihstrov@dk6n64 ~ $ find ~ -name "c*" -print
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-active.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-backdrop-normal.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-backdrop-active.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/assets/close-backdrop-hover.svg
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/gtk-3.0/colors.css
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/pulse/cookie
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/kdeconnect/certificate.pem
/afs/.dk.sci.pfu.edu.ru/home/g/a/gabihstrov/.config/libreoffice/4/user/config

```

Figure 4.5: Файлы начинавшиеся с символа с

```

gabihstrov@dk6n64 ~ $ ls -l | grep c*
-rw-r--r-- 1 gabihstrov studsci 1233 мая 15 10:55 conf.txt
gabihstrov@dk6n64 ~ $

```

Figure 4.6: Файлы начинавшиеся с символа с

4. Вывел на экран (по странично) имена файлов из каталога /etc, начинающиеся с символа h. (рис. 4.7)

```
gabihstrov@dk6n64 ~ $ find /etc -name "h*" -print | more
find: '/etc/cron.daily': Отказано в доступе
find: '/etc/polkit-1/rules.d': Отказано в доступе
find: '/etc/cron.monthly': Отказано в доступе
find: '/etc/cron.weekly': Отказано в доступе
find: '/etc/lvm/cache': Отказано в доступе
find: '/etc/mail/spamassassin/sa-update-keys': Отказано в доступе
find: '/etc/audisp/plugins.d': Отказано в доступе
find: '/etc/sudoers.d': Отказано в доступе
find: '/etc/cups/certs': Отказано в доступе
find: '/etc/cups/ssl': Отказано в доступе
find: '/etc/cron.hourly': Отказано в доступе
find: '/etc/fcron': Отказано в доступе
find: '/etc/skey': Отказано в доступе
find: '/etc/munge': Отказано в доступе
find: '/etc/.git': Отказано в доступе
/etc/httpd
/etc/hosts.allow
/etc/brltty/Text/he.ttb
/etc/brltty/Text/hr.ttb
/etc/brltty/Text/hu.ttb
/etc/brltty/Text/hy.ttb
/etc/brltty/Text/hi.ttb
/etc/brltty/Input/hm
/etc/brltty/Input/ht
/etc/brltty/Input/bm/horizontal.kti
/etc/brltty/Input/hd
/etc/brltty/Input/hw
/etc/brltty/Config/brltty.conf
```

Figure 4.7: Имена файлов из каталога /etc

5. Запустил в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log. (рис. 4.8)

```
gabihstrov@dk6n64 ~ $ find -name "log*" -print >logfile &
[1] 9078
gabihstrov@dk6n64 ~ $
```

Figure 4.8: Процесс для записи в файл ~/logfile

6. Удалил файл ~/logfile. (рис. 4.9)

```
[1]+  Завершён      find -name "log*" -print > logfile
gabihstrov@dk6n64 ~ $
```

Figure 4.9: Удаление файла ~/logfile

7. Запустил из консоли в фоновом режиме редактор gedit. (рис. 4.10) (рис. 4.11)


```
gabihstrov@dk6n64 ~ $ gedit &
[1] 9420
gabihstrov@dk6n64 ~ $
```

Figure 4.10: Команда для запуска gedit

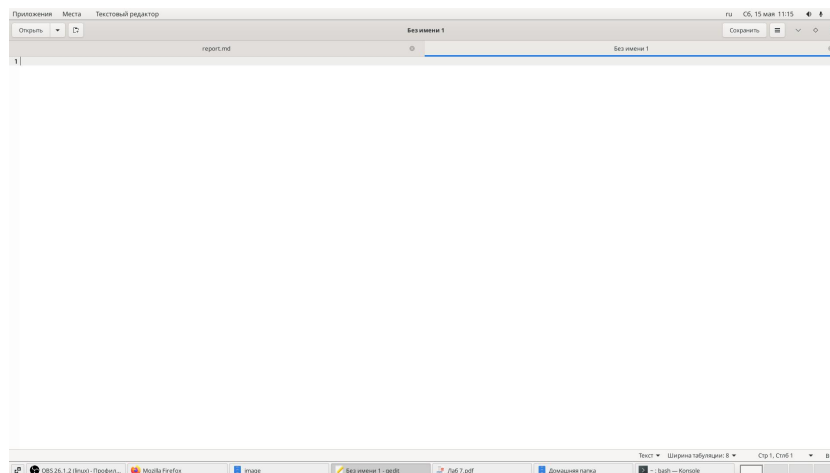


Figure 4.11: Редактор gedit

8. Определил идентификатор процесса gedit, используя команду ps, конвейер фильтр grep. (рис. 4.12)

```
gabihstrov@dk6n64 ~ $ ps aux | grep gedit
gabihst+ 3528  4.6  1.7 783924 141444 ?        S1   09:39   4:31 /usr/bin/gedit --gapplication-service
gabihst+ 9792  0.0  0.0 14316  2396 pts/0      S+   11:17   0:00 grep --colour=auto gedit
[1]+  Завершён gedit
gabihstrov@dk6n64 ~ $
```

Figure 4.12: Идентификатор процесса gedit

9. Прочитал справку (man) команды kill, после чего использовал её для завершения процесса gedit. (рис. 4.13) (рис. 4.14)

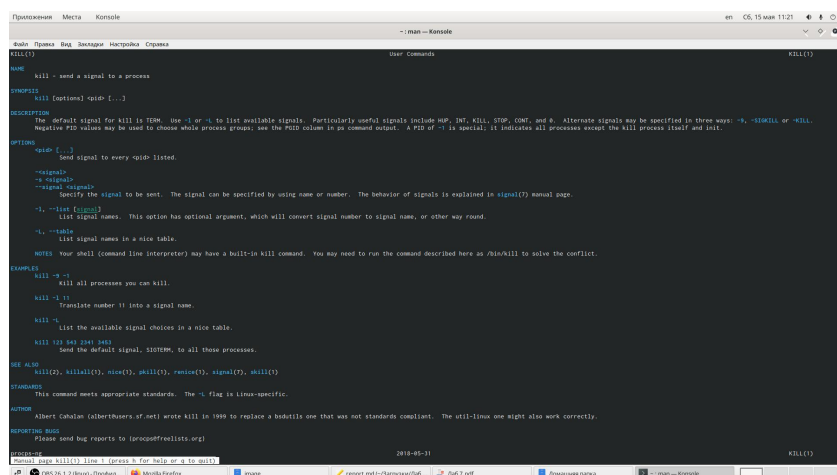


Figure 4.13: Справка (man) команды kill

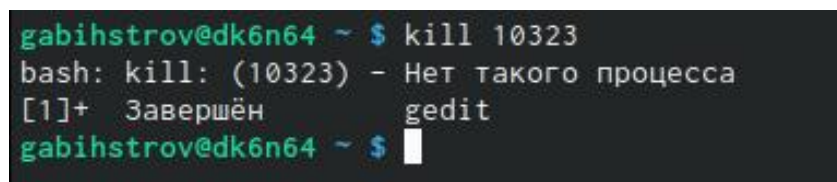


Figure 4.14: Завершил процесс gedit

- Выполнил команды `df` и `du`, предварительно получив более подробную информацию об этих командах, с помощью команды `man`. (рис. 4.15) (рис. 4.16)

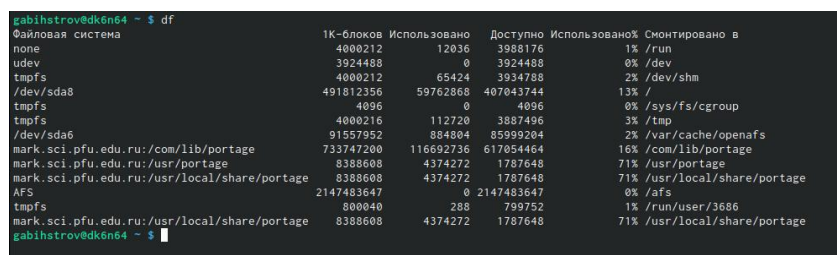


Figure 4.15: Команда df

```

gabihstrov@dk6n64 ~ $ du
2      ./public/public_html
4      ./public
18     ./texlive2020/texmf-var/fonts/tfm/lh/lh-t2a
20     ./texlive2020/texmf-var/fonts/tfm/lh
22     ./texlive2020/texmf-var/fonts/tfm
6      ./texlive2020/texmf-var/fonts/source/lh/lh-t2a
8      ./texlive2020/texmf-var/fonts/source/lh
10     ./texlive2020/texmf-var/fonts/source
163    ./texlive2020/texmf-var/fonts/pk/ljfour/lh/lh-t2a
165    ./texlive2020/texmf-var/fonts/pk/ljfour/lh
167    ./texlive2020/texmf-var/fonts/pk/ljfour
169    ./texlive2020/texmf-var/fonts/pk
203    ./texlive2020/texmf-var/fonts
3416   ./texlive2020/texmf-var/luatex-cache/generic/names
25900  ./texlive2020/texmf-var/luatex-cache/generic/fonts/otl
25902  ./texlive2020/texmf-var/luatex-cache/generic/fonts
29320  ./texlive2020/texmf-var/luatex-cache/generic
29322  ./texlive2020/texmf-var/luatex-cache
29527  ./texlive2020/texmf-var
29529  ./texlive2020
2      ./config/gnome-session/saved-session
4      ./config/gnome-session
3      ./config/ibus/bus
5      ./config/ibus
9      ./config/dconf

```

Figure 4.16: Команда du

11. Воспользовавшись справкой команды find, вывел имена всех директорий, имеющих в вашем домашнем каталоге. (рис. 4.17)

```
gabihstrov@dk6n64 ~ $ find -type d
.
./public
./public/public_html
./.texlive2020
./.texlive2020/texmf-var
./.texlive2020/texmf-var/fonts
./.texlive2020/texmf-var/fonts/tfm
./.texlive2020/texmf-var/fonts/tfm/lh
./.texlive2020/texmf-var/fonts/tfm/lh/lh-t2a
./.texlive2020/texmf-var/fonts/source
./.texlive2020/texmf-var/fonts/source/lh
./.texlive2020/texmf-var/fonts/source/lh/lh-t2a
./.texlive2020/texmf-var/fonts/pk
./.texlive2020/texmf-var/fonts/pk/ljfour
./.texlive2020/texmf-var/fonts/pk/ljfour/lh
./.texlive2020/texmf-var/fonts/pk/ljfour/lh/lh-t2a
./.texlive2020/texmf-var/luatex-cache
./.texlive2020/texmf-var/luatex-cache/generic
./.texlive2020/texmf-var/luatex-cache/generic/names
./.texlive2020/texmf-var/luatex-cache/generic/fonts
./.texlive2020/texmf-var/luatex-cache/generic/fonts/otl
./.config
./.config/gnome-session
./.config/gnome-session/saved-session
./.config/ibus
./.config/ibus/bus
./.config/dconf
./.config/evolution
./.config/evolution/sources
```

Figure 4.17: Имена директорий в домашнем каталоге.

5 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 `»`. 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”.
3. Что такое 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 передаётся на ввод команде 2. Конвейеры можно группировать в цепочки и выводить с помощью перенаправления в файл, например: `ls -la |sort > sortilg_list` вывод команды `ls -la` передаётся команде сортировки `sort\verb`, которая пишет результат в файл `sorting_list\verb`.
4. Что такое процесс? Чем это понятие отличается от программы? (Библиографический список. Источник 1) 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. Программа и процесс являются родственными терминами. Основное различие между программой и процессом заключается в том, что программа представляет собой группу инструкций для выполнения определенной задачи, тогда как процесс представляет собой программу в процессе выполнения. Хотя процесс является активной сущностью, программа считается пассивной. Между процессом и программой существует отношение многие-к-одному, что означает, что одна программа может вызывать несколько процессов или, другими словами, несколько процессов могут быть частью одной и той же программы.
5. Что такое PID и GID? (Библиографический список. Источник 2)
- PID: В Linux исполняемый файл, хранящийся на диске, называется программой, а программа, загруженная в память и работающая, называется про-

цессом. Когда процесс запускается, ему присваивается уникальный номер, называемый идентификатором процесса (PID), который идентифицирует этот процесс в системе. Если вам когда-нибудь понадобится, например, убить процесс, вы можете обратиться к нему по его PID.

- GID: Идентификатор группы, часто сокращенно GID, представляет собой числовое значение, используемое для представления определенной группы. Диапазон значений GID варьируется в разных системах; по крайней мере, GID может быть от 0 до 32 767, с одним ограничением: группа входа для суперпользователя должна иметь GID 0.

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

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

9. Можно ли по контексту (содержанию) найти файл? Если да, то как?
10. Как определить объем свободной памяти на жёстком диске? Команда `df` показывает размер каждого смонтированного раздела диска. Формат команды: `df [-опции] [файловая_система]`
11. Как определить объем вашего домашнего каталога? Команда `du` показывает число килобайт, используемое каждым файлом или каталогом. Формат команды: `du [-опции] [имя_файла...]`
12. Как удалить зависший процесс? (Библиографический список. Источник 3) Каждый процесс в Linux имеет свой идентификатор, называемый PID. Перед тем, как выполнить остановку процесса, нужно определить его PID. Для этого воспользуемся командами `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 результатам команды `ps`. Когда известен PID процесса, мы можем убить его командой `kill`. Команда `kill` принимает в качестве параметра PID процесса. Команда `killall` в Linux предназначена для «убийства» всех процессов, имеющих одно и то же имя. Это удобно, так как нам не нужно знать PID процесса. Некоторые процессы не удастся остановить под обычным пользователем. Например, если процесс был запущен от имени пользователя `root` или от имени другого пользователя системы, то команды `kill` и `killall` нужно выполнять от имени суперпользователя, добавляя `sudo`.

6 Выводы

В данной лабораторной работе мне успешно удалось ознакомиться с инструментами поиска файлов и фильтрации текстовых данных. Получилось преобрести практические навыки по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.

7 Библиографический список

1. Разница между программой и процессом (<https://ru.gadget-info.com/difference-between-program>)
2. Разница между pid, ppid, uid, euid, gid и egid процесса (<https://question-it.com/questions/2399204/v-chem-raznitsa-mezhdu-pid-ppid-uid-euid-gid-i-egid-protssessa>)
3. Команды ps, kill и killall (<https://pingvinus.ru/note/ps-kill-killall>)