

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «Операционная система Linux»

Процессы в операционной системе Linux

Студент

Посаднев В.В.

Группа АС-18

Руководитель

Кургасов В.В.

Липецк 2020 г.

Оглавление

Цель работы	3
Задание кафедры.....	4
Ход работы.....	6
Часть I.....	6
Часть II.....	12
Часть III (Вариант 7)	19
Вывод.....	24

Цель работы

Ознакомиться на практике с понятием процесса в операционной системе. Приобрести опыт и навыки управления процессами в операционной системе Linux.

Задание кафедры

Часть I

- 1) Загрузиться не root, а пользователем.
- 2) Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.
- 3) Посмотреть процессы `ps -f`. Прокомментировать. Для этого почитать `man ps`.
- 4) Написать с помощью редактора `vi` два сценария `loop` и `loop2`. Текст сценариев:

Loop:

```
while true; do true; done
```


Loop2:

```
while true; do true; echo 'Hello'; done
```
- 5) Запустить `loop2` на переднем плане: `sh loop2`.
- 6) Остановить, пошлав сигнал `STOP`.
- 7) Посмотреть, последовательно несколько раз `ps -f`. Записать сообщение, объяснить.
- 8) Убить процесс `loop2`, пошлав сигнал `kill -9 PID`. Записать сообщение. Прокомментировать.
- 9) Запустить в фоне процесс `loop`: `sh loop&`. Не останавливая, посмотреть несколько раз: `ps -f`. Записать значение, объяснить.
- 10) Завершить процесс командой `kill -15 PID`. Записать сообщение, прокомментировать.
- 11) Третий раз запустить в фоне. Не останавливая убить командой `kill -9 PID`.
- 12) Запустить еще один экземпляр оболочки: `bash`.
- 13) Запустить несколько процессов в фоне. Останавливать их и снова запускать. Записать результаты просмотра командой `ps -f`.

Часть II

- 1) Запустить в консоли на выполнение три задачи, две в интерактивном

2) режиме, одну – в фоновом.

3) Перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим.

4) Провести эксперименты по переводу задач из фонового режима в интерактивный и наоборот.

5) Создать именованный канал для архивирования и осуществить передачу в канал

- Списка файлов домашнего каталога вместе с подкаталогами (ключ -R),
- Одного каталога вместе с файлами и подкаталогами

6) В отчете предоставьте все шаги ваших действий. То есть следует привести следующее: текст задания, а следом за ним снимок экрана консоли с результатами выполнения задания. Кроме того, перед скриншотом следует привести текстовую запись использованных команд.

Часть III. Индивидуальные задания

Вариант 7

1) Вывести информацию о состоянии процессов системы в реальном режиме с обновлением один раз в 5 секунд. Отсортировать вывод по идентификатору пользователя по возрастанию и убыванию.

2) Завершить выполнение процесса, владельцем которого является текущий пользователь, с помощью сигнала SIGQUIT двумя способами: задав имя сигнала и используя комбинацию клавиш.

3) Измените на 2 единицы приоритет процесса, запущенного из командного интерпретатора.

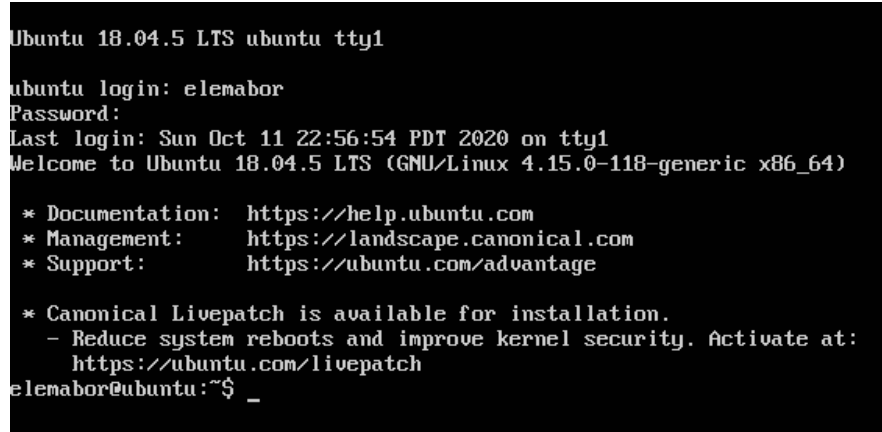
4) В отчете предоставьте все шаги ваших действий. То есть следует привести следующее: текст задания, а следом за ним снимок экрана консоли с результатами выполнения задания. Кроме того, перед скриншотом следует привести текстовую запись использованных команд. Кратко поясните результаты выполнения всех команд.

Ход работы

Часть I

1) Загрузиться не root, а пользователем

Изображение авторизации в системе под пользователем, а не root находится на рисунке 1.

A terminal window showing the login process for user 'elemabor' on Ubuntu 18.04.5 LTS. The prompt is 'ubuntu login: elemabor'. After entering the password, the system displays the last login time and a welcome message. It also provides links for documentation, management, and support, and mentions Canonical Livepatch.

```
ubuntu 18.04.5 LTS ubuntu tty1
ubuntu login: elemabor
Password:
Last login: Sun Oct 11 22:56:54 PDT 2020 on tty1
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-118-generic x86_64)

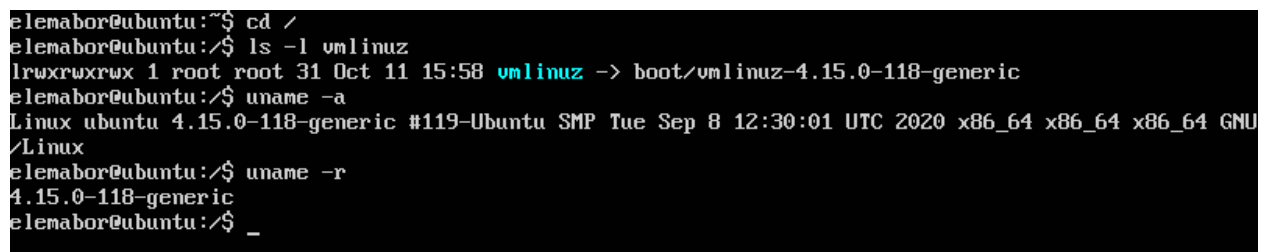
 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
elemabor@ubuntu:~$ _
```

Рисунок 1 – Авторизация пол пользователем

2) Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.

Чтобы получить информацию о номере версии Linux необходимо найти файл с образом ядра. Для этого необходимо выполнить следующие команды: `cd /`, `ls -l vmlinuz`. После этого мы можем увидеть информацию о файле с образом ядра Linux. Также данную информацию можно узнать, выполнив команду `uname -a`. Изображение выполненных команд находится на рисунке 2. Используемая версия Linux 4.15.0.

A terminal window showing the execution of commands to find the kernel image file and check the system version. The user 'elemabor' is at the root directory. The command 'ls -l vmlinuz' shows the file 'vmlinuz' with permissions 'lrwxrwxrwx' and points to 'boot/vmlinuz-4.15.0-118-generic'. The command 'uname -a' displays the system information, including the kernel version '4.15.0-118-generic'.

```
elemabor@ubuntu:~$ cd /
elemabor@ubuntu:/$ ls -l vmlinuz
lrwxrwxrwx 1 root root 31 Oct 11 15:58 vmlinuz -> boot/vmlinuz-4.15.0-118-generic
elemabor@ubuntu:/$ uname -a
Linux ubuntu 4.15.0-118-generic #119-Ubuntu SMP Tue Sep 8 12:30:01 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
elemabor@ubuntu:/$ uname -r
4.15.0-118-generic
elemabor@ubuntu:/$ _
```

Рисунок 2 – Файл с образом ядра

3) Посмотреть процессы `ps -f`. Прокомментировать.

После выполнения команды `ps -f` мы получим результат, соответствующий рисунку 3.

```

elenabor@ubuntu:/$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elenabor  1158    1117  0  10:43 tty1        00:00:00 -bash
elenabor  1179    1158  0  10:50 tty1        00:00:00 ps -f
elenabor@ubuntu:/$

```

Рисунок 3 – Пример просмотра процессов

UID – идентификатор владельца процесса. При указании флага -f получает входное имя пользователя.

PID – идентификатор процесса в системе.

PPID – идентификатор родительского процесса.

C – доля времени центрального процесса (в процентах), выделяемому данному процессу.

STIME – время старта процесса.

TTY – указание на терминал, с которого запущен процесс.

TIME – сколько времени ЦП занял данный процесс.

CMD – командная строка запуска программы, выполняемой процессом.

При указании флага -f будет выведено полное имя команды и её аргументы.

4) Написать с помощью редактора vi два сценария loop и loop2. Текст сценариев:

Loop:

```
while true; do true; done
```

Loop2:

```
while true, do true; echo 'Hello'; done
```

Создадим сценарии loop и loop2 с помощью редактора vi и запишем в них данные. После создания данных сценариев выведем содержимое на консоль. Вывод содержимого осуществляется с помощью команды cat название_файла. Результат выполнения данных операций изображен на рисунке 4.

```

elemabor@ubuntu:~$ ls -l
total 8
-rw-rw-r-- 1 elemabor elemabor 26 Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40 Oct 28 11:03 loop2
elemabor@ubuntu:~$ cat loop
while true; do true; done
elemabor@ubuntu:~$ cat loop2
while true; do true; echo 'Hello'; done
elemabor@ubuntu:~$ _

```

Рисунок 4 – Созданные сценарии loop и loop2

5) Запустить loop2 на переднем плане: sh loop2. Остановить, пошлав сигнал STOP.

```

Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
^Z
[1]+  Stopped                  sh loop2
elemabor@ubuntu:~$

```

Рисунок 5 – Пример запуска и остановка процесса

б) Посмотреть последовательно несколько раз ps -f. Записать сообщение, объяснить.

По данным изображенным на рисунке 6 процесс sh loop2 выполнялся 10 секунд и в данный момент остановлен. Также, он потребляет от 6-7 процентов доли центрального процессора.

```

elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1206    1158  7 11:07 tty1        00:00:10 sh loop2
elemabor    1214    1158  0 11:10 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1206    1158  7 11:07 tty1        00:00:10 sh loop2
elemabor    1215    1158  0 11:10 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1206    1158  6 11:07 tty1        00:00:10 sh loop2
elemabor    1216    1158  0 11:10 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1206    1158  6 11:07 tty1        00:00:10 sh loop2
elemabor    1217    1158  0 11:10 tty1        00:00:00 ps -f
elemabor@ubuntu:~$

```

Рисунок 6 – Просмотр процессов

7) Убить процесс loop2, пошлав сигнал kill -9 PID. Записать сообщение. Прокомментировать.

Как видно из данных на рисунке 7 процесс с PID 1206 был успешно выключен. Сигнал 9 (KILL) означает немедленное прекращение выполнения процесса. [1] – это строка на которой находился процесс.

```
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1206    1158  2 11:07 tty1        00:00:10 sh loop2
elemabor    1219    1158  0 11:14 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ kill -9 1206
[1]+  Killed                  sh loop2
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1220    1158  0 11:15 tty1        00:00:00 ps -f
elemabor@ubuntu:~$
```

Рисунок 7 – Удаление процесса loop2

8) Запустить в фоне процесс loop: sh loop&. Не останавливая, посмотреть несколько раз: ps -f. Записать значение, объяснить.

Как можно наблюдать из рисунка 8 процесс loop был запущен в фоновом режиме и его PID установлен как 1230. По нагрузке на процессор можно утверждать, что процесс полностью загружает ЦП. По времени можно сделать вывод, что процесс продолжает выполняться.

```
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1227    1158  0 11:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ sh loop&
[1] 1230
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1230    1158  99 11:18 tty1        00:00:04 sh loop
elemabor    1231    1158  0 11:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1230    1158  98 11:18 tty1        00:00:07 sh loop
elemabor    1232    1158  0 11:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1230    1158  99 11:18 tty1        00:00:11 sh loop
elemabor    1233    1158  0 11:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$
```

Рисунок 8 – Запуск в фоне процесса loop и просмотр процессов

9) Завершить процесс loop командой kill -15 PID. Записать сообщение прокомментировать.

Как видно по рисунку 9 процесс loop с PID 1230 после выполнения команды kill -15 1230 успешно завершился. Сигнал 15 (TERM) означает корректное завершение процесса.

```
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1230    1158  99 11:18 tty1        00:00:11 sh loop
elemabor    1233    1158  0 11:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ kill -15 1230
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1235    1158  0 11:21 tty1        00:00:00 ps -f
[1]+  Terminated                  sh loop
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1236    1158  0 11:22 tty1        00:00:00 ps -f
elemabor@ubuntu:~$
```

Рисунок 9 – Завершение процесса loop и просмотр процессов

10) Третий раз запустить в фоне. Не останавливая убить командой kill -9 PID.

На рисунке 10 видно вызов команды просмотра процесса (где еще не находится процесс loop), запуск процесса в фоне с назначенным ему PID 1240 и завершение процесса сигналом 9 (KILL – немедленное прекращение выполнения).

```
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1239    1158  0 11:27 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ sh loop&
[1] 1240
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1240    1158  96 11:27 tty1        00:00:03 sh loop
elemabor    1241    1158  0 11:27 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ kill -9 1240
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
elemabor    1158    1117  0 10:43 tty1        00:00:00 -bash
elemabor    1242    1158  0 11:27 tty1        00:00:00 ps -f
[1]+  Killed                      sh loop
elemabor@ubuntu:~$
```

Рисунок 10 – Запуск процесса в фоне и его завершение

11) Запустить еще один экземпляр оболочки: bash

На рисунке 11 можно наблюдать запуск второго экземпляра оболочки. Bash – это стандартный командный интерпретатор в юниксподобных системах.

```
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:00 -bash
elemabor  1247    1158   0  11:31 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ echo $$
1158
elemabor@ubuntu:~$ bash
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:01 -bash
elemabor  1248    1158   1  11:32 tty1        00:00:00 bash
elemabor  1256    1248   0  11:32 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ echo $$
1248
elemabor@ubuntu:~$ _
```

Рисунок 11 – Запуск второй оболочки

12) Запустить несколько процессов в фоне. Останавливать их и снова запускать. Записать результаты просмотра командой ps -f.

```
elemabor@ubuntu:~$ kill -9 1307
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:01 -bash
elemabor  1248    1158   0  11:32 tty1        00:00:00 bash
elemabor  1308    1248  42  11:49 tty1        00:00:09 sh loop
elemabor  1314    1248  37  11:49 tty1        00:00:04 sh loop
elemabor  1315    1248   0  11:49 tty1        00:00:00 ps -f
[11] Killed                  sh loop
elemabor@ubuntu:~$ sh loop&
[41] 1316
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:01 -bash
elemabor  1248    1158   0  11:32 tty1        00:00:00 bash
elemabor  1308    1248  41  11:49 tty1        00:00:12 sh loop
elemabor  1314    1248  37  11:49 tty1        00:00:06 sh loop
elemabor  1316    1248  30  11:50 tty1        00:00:01 sh loop
elemabor  1317    1248   0  11:50 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ kill -15 1314
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:01 -bash
elemabor  1248    1158   0  11:32 tty1        00:00:00 bash
elemabor  1308    1248  39  11:49 tty1        00:00:16 sh loop
elemabor  1316    1248  33  11:50 tty1        00:00:05 sh loop
elemabor  1318    1248   0  11:50 tty1        00:00:00 ps -f
[31]- Terminated            sh loop
elemabor@ubuntu:~$ kill -9 1308 1316
elemabor@ubuntu:~$ ps -f
  UID      PID    PPID  C  STIME TTY          TIME CMD
elemabor  1158    1117   0  10:43 tty1        00:00:01 -bash
elemabor  1248    1158   0  11:32 tty1        00:00:00 bash
elemabor  1320    1248   0  11:51 tty1        00:00:00 ps -f
[21]- Killed                  sh loop
[41]+ Killed                  sh loop
elemabor@ubuntu:~$ _
```

Рисунок 12 – Запуск нескольких процессов в фоне и их завершение

Часть II

1) Запустить в консоли на выполнение три задачи, две в интерактивном режиме, одну – в фоновом.

Запустим в консоли две задачи в интерактивном режиме (PID задач в интерактивном режиме 1412 и 1413) и одну задачу в фоновом режиме (PID задачи 1414). Результат запуска изображен на рисунке 13.

```
UID      PID    PPID  C STIME TTY          TIME CMD
elemabor 1376   1341  0 14:15 tty1        00:00:00 -bash
elemabor 1387   1376  0 14:15 tty1        00:00:00 bash
elemabor 1411   1387  0 14:17 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ sh loop
^Z
[11]+  Stopped                  sh loop
elemabor@ubuntu:~$ sh loop
^Z
[21]+  Stopped                  sh loop
elemabor@ubuntu:~$ sh loop&
[3] 1414
elemabor@ubuntu:~$ ps -f
UID      PID    PPID  C STIME TTY          TIME CMD
elemabor 1376   1341  0 14:15 tty1        00:00:00 -bash
elemabor 1387   1376  0 14:15 tty1        00:00:00 bash
elemabor 1412   1387  14 14:17 tty1        00:00:03 sh loop
elemabor 1413   1387  27 14:17 tty1        00:00:05 sh loop
elemabor 1414   1387  99 14:17 tty1        00:00:04 sh loop
elemabor 1415   1387  0 14:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID      PID    PPID  C STIME TTY          TIME CMD
elemabor 1376   1341  0 14:15 tty1        00:00:00 -bash
elemabor 1387   1376  0 14:15 tty1        00:00:00 bash
elemabor 1412   1387  13 14:17 tty1        00:00:03 sh loop
elemabor 1413   1387  24 14:17 tty1        00:00:05 sh loop
elemabor 1414   1387  95 14:17 tty1        00:00:05 sh loop
elemabor 1416   1387  0 14:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID      PID    PPID  C STIME TTY          TIME CMD
elemabor 1376   1341  0 14:15 tty1        00:00:00 -bash
elemabor 1387   1376  0 14:15 tty1        00:00:00 bash
elemabor 1412   1387  11 14:17 tty1        00:00:03 sh loop
elemabor 1413   1387  21 14:17 tty1        00:00:05 sh loop
elemabor 1414   1387  99 14:17 tty1        00:00:09 sh loop
elemabor 1417   1387  0 14:18 tty1        00:00:00 ps -f
elemabor@ubuntu:~$
```

Рисунок 13 – Запуск двух приложений в интерактивном и одно в фоновом

2) Перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим.

Для перевода задачи, выполняющейся в интерактивном режиме, в фоновый режим необходимо выполнить команду `bg 1` (по номеру в выводе утилиты `jobs`). Пример выполнения данной команды и перевод процесса `sh loop` в фоновый режим приведен на рисунке 14.

```

elemabor@ubuntu:~$ bg --help
bg: bg [job_spec ...]
    Move jobs to the background.

    Place the jobs identified by each JOB_SPEC in the background, as if they
    had been started with '&'. If JOB_SPEC is not present, the shell's notion
    of the current job is used.

    Exit Status:
    Returns success unless job control is not enabled or an error occurs.
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C  STIME TTY          TIME CMD
elemabor    1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor    1387    1376  0  14:15 tty1        00:00:00 bash
elemabor    1412    1387  0  14:17 tty1        00:00:03 sh loop
elemabor    1413    1387  0  14:17 tty1        00:00:05 sh loop
elemabor    1433    1387  97 14:51 tty1        00:02:21 sh loop
elemabor    1444    1387  0  14:53 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ jobs
[1]-  Stopped                  sh loop
[2]+  Stopped                  sh loop
[3]   Running                  sh loop &
elemabor@ubuntu:~$ bg 1
[1]-  sh loop &
elemabor@ubuntu:~$ jobs
[1]   Running                  sh loop &
[2]+  Stopped                  sh loop
[3]-  Running                  sh loop &
elemabor@ubuntu:~$ ps -f
UID          PID    PPID  C  STIME TTY          TIME CMD
elemabor    1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor    1387    1376  0  14:15 tty1        00:00:00 bash
elemabor    1412    1387  0  14:17 tty1        00:00:07 sh loop
elemabor    1413    1387  0  14:17 tty1        00:00:05 sh loop
elemabor    1433    1387  95 14:51 tty1        00:02:41 sh loop
elemabor    1445    1387  3  14:54 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ _

```

Рисунок 14 – Перевод задачи из интерактивного режима в фоновый

3) Провести эксперименты по переводу задач из фонового режима в интерактивный и наоборот.

Посмотрим на список процессов и поставим процесс с индексом 3 на исполнение в интерактивном режиме, для этого выполним команду fg 3, а после остановим его. Переведем процесс с индексом 2 в фоновый режим, для этого выполним команду bg 2, и посмотрим на список процессов с помощью команды ps -f. Пример выполнения данных операций изображен на рисунке 15.

```

elemabor@ubuntu:~$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:00 bash
elemabor 1412    1387  3  14:17 tty1        00:01:30 sh loop
elemabor 1413    1387  0  14:17 tty1        00:00:05 sh loop
elemabor 1433    1387 72  14:51 tty1        00:04:04 sh loop
elemabor 1447    1387  0  14:56 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ jobs
[1]  Running                  sh loop &
[2]+  Stopped                  sh loop
[3]-  Running                  sh loop &
elemabor@ubuntu:~$ fg 3
sh loop
^Z
[3]+  Stopped                  sh loop
elemabor@ubuntu:~$ jobs
[1]  Running                  sh loop &
[2]-  Stopped                  sh loop
[3]+  Stopped                  sh loop
elemabor@ubuntu:~$ bg 2
[2]- sh loop &
elemabor@ubuntu:~$ jobs
[1]  Running                  sh loop &
[2]-  Running                  sh loop &
[3]+  Stopped                  sh loop
elemabor@ubuntu:~$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:00 bash
elemabor 1412    1387  5  14:17 tty1        00:02:00 sh loop
elemabor 1413    1387  0  14:17 tty1        00:00:08 sh loop
elemabor 1433    1387 67  14:51 tty1        00:04:19 sh loop
elemabor 1449    1387  0  14:57 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ _

```

Рисунок 15 – Пример перевода задач из интерактивного режима в фоновый и наоборот

4) Создать именованный канал для архивирования и осуществить передачу в канал

Именованный канал или именованный конвейер – один из методов межпроцессного взаимодействия. Именованный канал позволяет различным процессам обмениваться данными даже если программы, выполняющиеся в этих процессах, изначально не были написаны для взаимодействия с другими программами. Для создания именованного канала используется команда `mkfifo`. Результат попадает в файл `result`, также используем максимальное сжатие (флаг `-9`). Пример создания именованного канала изображен на рисунке 16.

```

elemabor@ubuntu:~$ ls -la
total 40
drwxr-xr-x 3 elemabor elemabor 4096 Oct 28 11:03 .
drwxr-xr-x 4 root     root     4096 Oct 14 05:24 ..
-rw-r----- 1 elemabor elemabor 1428 Oct 28 15:14 .bash_history
-rw-r--r-- 1 elemabor elemabor 220 Oct 11 15:46 .bash_logout
-rw-r--r-- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
-rw-rw-r-- 1 elemabor elemabor 26 Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40 Oct 28 11:03 loop2
-rw-r--r-- 1 elemabor elemabor 807 Oct 11 15:46 .profile
-rw-r--r-- 1 elemabor elemabor 0 Oct 11 15:50 .sudo_as_admin_successful
-rw-r----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ mkfifo channel
elemabor@ubuntu:~$ ls -la
total 40
drwxr-xr-x 3 elemabor elemabor 4096 Oct 28 15:16 .
drwxr-xr-x 4 root     root     4096 Oct 14 05:24 ..
-rw-r----- 1 elemabor elemabor 1428 Oct 28 15:14 .bash_history
-rw-r--r-- 1 elemabor elemabor 220 Oct 11 15:46 .bash_logout
-rw-r--r-- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
prw-rw-r-- 1 elemabor elemabor 0 Oct 28 15:16 channel
-rw-rw-r-- 1 elemabor elemabor 26 Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40 Oct 28 11:03 loop2
-rw-r--r-- 1 elemabor elemabor 807 Oct 11 15:46 .profile
-rw-r--r-- 1 elemabor elemabor 0 Oct 11 15:50 .sudo_as_admin_successful
-rw-r----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ gzip -9 -c < channel > result

```

Рисунок 16 – Пример создания именного канала для архивирования данных

Откроем второй терминал для передачи данных по именованному каналу. Для этого необходимо выполнить комбинацию клавиш: CTRL+ALT+F_номер. Например: для открытия второго терминала необходимо выполнить команду CTRL+ALT+F2. Результат открытия и передачи домашнего каталога вместе с подкаталогами изображен на рисунке 17.

```

Ubuntu 18.04.5 LTS ubuntu tty2

ubuntu login: elemabor
Password:
Last login: Wed Oct 28 15:14:28 PDT 2020 on tty2
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-118-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

elemabor@ubuntu:~$ ls -R /home > channel
elemabor@ubuntu:~$ _

```

Рисунок 17 – Вход во втором терминале и передача папки home со всеми файлами и подкаталогами (ключ -R)

После успешной передачи информации вернемся к первой консоли (комбинация клавиш CTRL+ALT+F1) и посмотрим на содержимое после сжатия, полученное содержимое изображено на рисунке 18.

```

-rw----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ mkfifo channel
elemabor@ubuntu:~$ ls -la
total 40
drwxr-xr-x 3 elemabor elemabor 4096 Oct 28 15:16 .
drwxr-xr-x 4 root      root      4096 Oct 14 05:24 ..
-rw----- 1 elemabor elemabor 1428 Oct 28 15:14 .bash_history
-rw-r--r-- 1 elemabor elemabor 220  Oct 11 15:46 .bash_logout
-rw-r--r-- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
prw-rw-r-- 1 elemabor elemabor   0 Oct 28 15:16 channel
-rw-rw-r-- 1 elemabor elemabor 26   Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40   Oct 28 11:03 loop2
-rw-r--r-- 1 elemabor elemabor 807  Oct 11 15:46 .profile
-rw-r--r-- 1 elemabor elemabor   0 Oct 11 15:50 .sudo_as_admin_successful
-rw----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ gzip -9 -c < channel > result
elemabor@ubuntu:~$ ls -la
total 44
drwxr-xr-x 3 elemabor elemabor 4096 Oct 28 15:19 .
drwxr-xr-x 4 root      root      4096 Oct 14 05:24 ..
-rw----- 1 elemabor elemabor 1428 Oct 28 15:14 .bash_history
-rw-r--r-- 1 elemabor elemabor 220  Oct 11 15:46 .bash_logout
-rw-r--r-- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
prw-rw-r-- 1 elemabor elemabor   0 Oct 28 15:19 channel
-rw-rw-r-- 1 elemabor elemabor 26   Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40   Oct 28 11:03 loop2
-rw-r--r-- 1 elemabor elemabor 807  Oct 11 15:46 .profile
-rw-rw-r-- 1 elemabor elemabor 98   Oct 28 15:19 result
-rw-r--r-- 1 elemabor elemabor   0 Oct 11 15:50 .sudo_as_admin_successful
-rw----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ cat result
m_5♦♦
/♦♦♦♦♦dALyQ♦LucU♦♦
♦f♦(p)♦E♦J♦ri♦Y♦Sy
♦♦♦♦♦L♦♦>♦♦hmelemabor@ubuntu:~$

```

Рисунок 18 – Результат сжатия

Для просмотра содержимого данных без сжатия необходимо выполнить команду zcat и указанное ранее название. Содержимое без сжатия изображено на рисунке 19.

```

elemabor@ubuntu:~$ cat result
m_5♦♦
/♦♦♦♦♦dALyQ♦LucU♦♦
♦f♦(p)♦E♦J♦ri♦Y♦Sy
♦♦♦♦♦L♦♦>♦♦hmelemabor@ubuntu:~$ zcat result
/home:
elemabor
user

/home/elemabor:
channel
loop
loop2
result

/home/user:
1.txt
3.txt
hard_link
symb_link
elemabor@ubuntu:~$ _

```

Рисунок 19 – Содержимое архива

Откроем еще одно соединение и передадим туда ранее подготовленные данные. Пример подготовленных данных изображено на рисунке 21, а открытие соединения на рисунке 20. Полученный результат передачи можно наблюдать на рисунке 22.

```
elemabor@ubuntu:~$ ls -la
total 44
drwxr-xr-x 3 elemabor elemabor 4096 Oct 28 15:19 .
drwxr-xr-x 4 root      root      4096 Oct 14 05:24 ..
-rw-r----- 1 elemabor elemabor 1428 Oct 28 15:14 .bash_history
-rw-r----- 1 elemabor elemabor 220  Oct 11 15:46 .bash_logout
-rw-r----- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
prw-rw-r-- 1 elemabor elemabor   0  Oct 28 15:19 channel
-rw-rw-r-- 1 elemabor elemabor 26   Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40   Oct 28 11:03 loop2
-rw-r----- 1 elemabor elemabor 807  Oct 11 15:46 .profile
-rw-rw-r-- 1 elemabor elemabor 98   Oct 28 15:19 result
-rw-r----- 1 elemabor elemabor   0  Oct 11 15:50 .sudo_as_admin_successful
-rw-r----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ gzip -9 -c < channel > folder_result
```

Рисунок 20 – Создание именного канала для архивирования данных (для передачи одного каталога)

```
Ubuntu 18.04.5 LTS ubuntu tty2

ubuntu login: elemabor
Password:
Last login: Wed Oct 28 15:17:31 PDT 2020 on tty2
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-118-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

elemabor@ubuntu:~$ mkdir test
elemabor@ubuntu:~$ cd test/
elemabor@ubuntu:~/test$ touch first_file_test.txt
elemabor@ubuntu:~/test$ touch second_file_test.txt
elemabor@ubuntu:~/test$ echo "test123" > first_file_test.txt
elemabor@ubuntu:~/test$ echo "321test" > second_file_test.txt
elemabor@ubuntu:~/test$ cat first_file_test.txt
test123
elemabor@ubuntu:~/test$ cat second_file_test.txt
321test
elemabor@ubuntu:~/test$ cd ..
elemabor@ubuntu:~$ test > channel
elemabor@ubuntu:~$ ls test > channel
elemabor@ubuntu:~$ _
```

Рисунок 21 – Подготовка второго терминала для передачи файлов по каналу

```

elemabor@ubuntu:~$ ls -la
total 48
drwxr-xr-x 4 elemabor elemabor 4096 Oct 28 15:36 .
drwxr-xr-x 4 root     root     4096 Oct 14 05:24 ..
-rw----- 1 elemabor elemabor 1462 Oct 28 15:24 .bash_history
-rw-r--r-- 1 elemabor elemabor 220  Oct 11 15:46 .bash_logout
-rw-r--r-- 1 elemabor elemabor 3771 Oct 11 15:46 .bashrc
drwx----- 2 elemabor elemabor 4096 Oct 11 15:50 .cache
prw-rw-r-- 1 elemabor elemabor   0  Oct 28 15:34 channel
-rw-rw-r-- 1 elemabor elemabor 26   Oct 28 11:02 loop
-rw-rw-r-- 1 elemabor elemabor 40   Oct 28 11:03 loop2
-rw-r--r-- 1 elemabor elemabor 807  Oct 11 15:46 .profile
-rw-rw-r-- 1 elemabor elemabor 98   Oct 28 15:19 result
-rw-r--r-- 1 elemabor elemabor   0  Oct 11 15:50 .sudo_as_admin_successful
drwxrwxr-x 2 elemabor elemabor 4096 Oct 28 15:36 test
-rw----- 1 elemabor elemabor 1266 Oct 28 11:03 .viminfo
elemabor@ubuntu:~$ gzip -9 -c < channel > folder_result
elemabor@ubuntu:~$ zcat folder_result
channel
first_file_test.txt
second_file_test.txt
elemabor@ubuntu:~$ cat folder_result
♦♦_K♦H♦♦K♦♦J♦,*.♦0♦♦I♦/I-♦♦♦(♦♦NM♦♦K♦ ♦E♦1elemabor@ubuntu:~$ _

```

Рисунок 22 – Просмотр переданной информации

Часть III (Вариант 7)

1) Вывести информацию о состоянии процессов системы в реальном режиме с обновлением один раз в 5 секунд. Отсортировать вывод по идентификатору пользователя по возрастанию по убыванию.

Для просмотра информации о состоянии процессов системы в реальном режиме необходимо выполнить команду `top`. Обратимся за помощью по поводу аргументов данной команды, для этого выполним команду `top -help`. Как видно из результата выполнения предыдущей команды, изображенной на рисунке 23, чтобы установить период обновления необходимо использовать аргумент `-d` и значение для обновления. В нашем случае готовая команда будет следующая: `top -d 5`.

```
elemabor@ubuntu:~$ top -help
procs-ng 3.3.12
Usage:
  top -hv l -bcHiOSs -d secs -n max -uIU user -p pid(s) -o field -w [cols]
elemabor@ubuntu:~$ _
```

Рисунок 23 – Помощь по команде `top`

```
top - 16:05:20 up 3:12, 2 users, load average: 2.02, 2.03, 2.00
Tasks: 141 total, 3 running, 74 sleeping, 1 stopped, 0 zombie
%Cpu(s): 98.8 us, 0.8 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.4 si, 0.0 st
KiB Mem : 2017456 total, 1447180 free, 98236 used, 472040 buff/cache
KiB Swap: 483800 total, 483800 free, 0 used, 1770308 avail Mem
```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1412	elemabor	20	0	4628	776	708	R	48.8	0.0	35:18.23	sh
1413	elemabor	20	0	4628	800	732	R	48.8	0.0	33:25.82	sh
1836	elemabor	20	0	44076	4044	3416	R	0.8	0.2	0:00.36	top
429	root	0	-20	218216	7104	6088	S	0.4	0.4	1:31.14	umtoolsd
1610	root	20	0	0	0	0	I	0.4	0.0	0:06.89	kworker/0:1
333	systemd+	20	0	80080	5420	4820	S	0.2	0.3	0:00.57	systemd-network
1	root	20	0	77388	8424	6604	S	0.0	0.4	0:09.07	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.03	kthreadd
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
7	root	20	0	0	0	0	S	0.0	0.0	0:02.01	ksoftirqd/0
8	root	20	0	0	0	0	I	0.0	0.0	0:12.02	rcu_sched
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_bh
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.25	watchdog/0
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
13	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kdevtmpfs
14	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
15	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_kthre
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kauditd
17	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper
19	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeback
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kcompactd0
21	root	25	5	0	0	0	S	0.0	0.0	0:00.00	ksmd
22	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
23	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	crypto
24	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd
25	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kblockd
26	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	ata_sff

Рисунок 24 – Информация о состоянии системы раз в 5 секунд

Следующим шагом нам необходимо отсортировать данную информацию по идентификатору пользователя (UID). Для этого перейдем в меню с помощью клавиши F, соответствующее рисунку 25. Выберем пункт UID и нажмем на клавишу 'd', чтобы показывать его в информации о состоянии системы, также выполним сортировку по этому значению клавиша 's'. Таким образом информация о состоянии системы будет отсортировано по убыванию по ключу UID, пример результата изображен на рисунке 26. Для выбора иного способа сортировки (по убыванию или по возрастанию) необходимо выполним комбинацию клавиш SHIFT+R.

```
Fields Management for window 1:Def, whose current sort field is UID
  Navigate with Up/Dn, Right selects for move then <Enter> or Left commits,
  'd' or <Space> toggles display, 's' sets sort. Use 'q' or <Esc> to end!

* PID      = Process Id          WCHAN      = Sleeping in Function
* USER     = Effective User Name  Flags      = Task Flags <sched.h>
* PR       = Priority            CGROUPS    = Control Groups
* NI       = Nice Value          SUPGIDS    = Supp Groups IDs
* VIRT     = Virtual Image (KiB) SUPGRPS    = Supp Groups Names
* RES      = Resident Size (KiB) TGID       = Thread Group Id
* SHR      = Shared Memory (KiB) OOMa       = OOMEM Adjustment
* S        = Process Status      OOMs       = OOMEM Score current
* %CPU     = CPU Usage           ENVIRON    = Environment vars
* %MEM     = Memory Usage (RES)  vMj        = Major Faults delta
* TIME+    = CPU Time, hundredths vMn        = Minor Faults delta
* COMMAND  = Command Name/Line  USED       = Res+Swap Size (KiB)
* PPID     = Parent Process pid  nsIPC      = IPC namespace Inode
* UID      = Effective User Id   nsMNT      = MNT namespace Inode
* RUID     = Real User Id        nsNET      = NET namespace Inode
* RUSER    = Real User Name      nsPID      = PID namespace Inode
* SUID     = Saved User Id       nsUSER     = USER namespace Inode
* SUSER    = Saved User Name     nsUTS      = UTS namespace Inode
* GID      = Group Id           LXC        = LXC container name
* GROUP    = Group Name         RSan       = RES Anonymous (KiB)
* PGRP     = Process Group Id    RSfd       = RES File-based (KiB)
* TTY      = Controlling Tty     RSlk       = RES Locked (KiB)
* TPGID    = Tty Process Grp Id  RSsh       = RES Shared (KiB)
* SID      = Session Id         CGNAME     = Control Group name
* nTH      = Number of Threads
* P        = Last Used Cpu (SMP)
* TIME     = CPU Time
* SWAP     = Swapped Size (KiB)
* CODE     = Code Size (KiB)
* DATA    = Data+Stack (KiB)
* nMaj     = Major Page Faults
* nMin     = Minor Page Faults
* nDRT     = Dirty Pages Count
```

Рисунок 25 – Выбор UID для сортировки значений

```
top - 16:11:12 up 3:18, 2 users, load average: 2.00, 2.03, 2.00
Tasks: 141 total, 3 running, 74 sleeping, 1 stopped, 0 zombie
%Cpu(s): 99.4 us, 0.6 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 2017456 total, 1447180 free, 98236 used, 472040 buff/cache
KiB Swap: 483800 total, 483800 free, 0 used, 1770308 avail Mem
```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND	UID
421	systemd+	20	0	141956	3284	2752	S	0.0	0.2	0:00.42	systemd-timesyn	62583
1153	elemabor	20	0	76560	7076	6164	S	0.0	0.4	0:00.11	systemd	1000
1154	elemabor	20	0	111368	2032	52	S	0.0	0.1	0:00.00	(sd-pam)	1000
1331	elemabor	20	0	20588	2520	2356	S	0.2	0.1	0:07.89	ping	1000
1376	elemabor	20	0	22352	4944	3576	S	0.0	0.2	0:00.15	bash	1000
1387	elemabor	20	0	22568	5116	3520	S	0.0	0.3	0:00.95	bash	1000
1412	elemabor	20	0	4628	776	708	R	49.3	0.0	38:12.01	sh	1000
1413	elemabor	20	0	4628	800	732	R	49.5	0.0	36:19.60	sh	1000
1433	elemabor	20	0	4628	800	732	T	0.0	0.0	4:19.37	sh	1000
1462	elemabor	20	0	22568	5132	3652	S	0.0	0.3	0:00.94	bash	1000
1648	elemabor	20	0	22908	5744	3804	S	0.0	0.3	0:02.02	bash	1000
1836	elemabor	20	0	44076	4044	3416	R	0.4	0.2	0:01.65	top	1000
440	message+	20	0	49928	4428	3848	S	0.0	0.2	0:00.88	dbus-daemon	103
438	syslog	20	0	263036	4264	3492	S	0.0	0.2	0:00.20	rsyslogd	102
427	systemd+	20	0	70660	5340	4780	S	0.0	0.3	0:00.53	systemd-resolve	101
333	systemd+	20	0	80080	5420	4820	S	0.0	0.3	0:00.57	systemd-network	100
1	root	20	0	77388	8424	6604	S	0.0	0.4	0:09.08	systemd	0
2	root	20	0	0	0	0	S	0.0	0.0	0:00.03	kthreadd	0
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H	0
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq	0
7	root	20	0	0	0	0	S	0.0	0.0	0:02.05	ksoftirqd/0	0
8	root	20	0	0	0	0	I	0.0	0.0	0:12.08	rcu_sched	0
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_bh	0
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0	0
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.25	watchdog/0	0
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0	0
13	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kdevtmpfs	0
14	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns	0
15	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_kthre	0
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kauditd	0

Рисунок 26 – Информация о состоянии системы отсортированное по UID раз
в 5 секунд, сортировка по убыванию

```
top - 16:15:46 up 3:23, 2 users, load average: 2.01, 2.02, 2.00
Tasks: 141 total, 3 running, 74 sleeping, 1 stopped, 0 zombie
%Cpu(s): 98.0 us, 1.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.5 si, 0.0 st
KiB Mem : 2017456 total, 1447180 free, 98212 used, 472064 buff/cache
KiB Swap: 483800 total, 483800 free, 0 used, 1770332 avail Mem
```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND	UID
1	root	20	0	77388	8424	6604	S	0.0	0.4	0:09.08	systemd	0
2	root	20	0	0	0	0	S	0.0	0.0	0:00.03	kthreadd	0
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H	0
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq	0
7	root	20	0	0	0	0	S	0.0	0.0	0:02.09	ksoftirqd/0	0
8	root	20	0	0	0	0	I	0.5	0.0	0:12.16	rcu_sched	0
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_bh	0
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0	0
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.25	watchdog/0	0
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0	0
13	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kdevtmpfs	0
14	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns	0
15	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_kthre	0
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kauditd	0
17	root	20	0	0	0	0	S	0.0	0.0	0:00.03	khungtaskd	0
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper	0
19	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeback	0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kcompactd0	0
21	root	25	5	0	0	0	S	0.0	0.0	0:00.00	ksmd	0
22	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged	0
23	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	crypto	0
24	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd	0
25	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kblockd	0
26	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	ata_sff	0
27	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	md	0
28	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	edac-poller	0
29	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	devfreq_wq	0
30	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	watchdogd	0
34	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kswapd0	0
35	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/u257:0	0

Рисунок 27 – Информация о состоянии системы отсортированное по UID раз
в 5 секунд, сортировка по возрастанию

2) Завершить выполнение процесса, владельцем которого является текущий пользователь, с помощью сигнала SIGQUIT двумя способами, задав имя сигнала и используя комбинацию клавиш.

Посмотрим на процессы принадлежащих текущему пользователю с помощью команды `ps -f`. Чтобы завершить выполнение процесса с помощью сигнала SIGQUIT необходимо выполним команду `kill -SIGQUIT 1412`, где 1412 – UID процесса, которое необходимо завершить. Для завершения процесса с помощью комбинаций клавиш необходимо чтобы процесс был интерактивным, для этого выполним команду `fg 2`, где 2 – это номер процесса. Для завершения процесса с помощью сигнала SIGQUIT используя комбинацию клавиш необходимо использовать следующую комбинацию: `CTRL+\`. Пример выполнения данных команд приведен на рисунке 28.

```

UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:00 bash
elemabor 1412    1387 34  14:17 tty1        00:42:40 sh loop
elemabor 1413    1387 33  14:17 tty1        00:40:48 sh loop
elemabor 1433    1387  4  14:51 tty1        00:04:19 sh loop
elemabor 1848    1387  0  16:20 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:01 bash
elemabor 1412    1387 35  14:17 tty1        00:45:48 sh loop
elemabor 1413    1387 34  14:17 tty1        00:43:55 sh loop
elemabor 1433    1387  4  14:51 tty1        00:04:19 sh loop
elemabor 1849    1387  0  16:26 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ kill -SIGQUIT 1412
elemabor@ubuntu:~$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:01 bash
elemabor 1413    1387 34  14:17 tty1        00:44:07 sh loop
elemabor 1433    1387  4  14:51 tty1        00:04:19 sh loop
elemabor 1850    1387  0  16:26 tty1        00:00:00 ps -f
[1]  Quit                               sh loop
elemabor@ubuntu:~$ jobs
[2]-  Running                          sh loop &
[3]+  Stopped                          sh loop
elemabor@ubuntu:~$ fg 2
sh loop
^\\Quit
elemabor@ubuntu:~$ ps -f
UID      PID     PPID  C  STIME TTY          TIME CMD
elemabor 1376    1341  0  14:15 tty1        00:00:00 -bash
elemabor 1387    1376  0  14:15 tty1        00:00:01 bash
elemabor 1433    1387  4  14:51 tty1        00:04:19 sh loop
elemabor 1851    1387  0  16:28 tty1        00:00:00 ps -f
elemabor@ubuntu:~$ _

```

Рисунок 28 – Завершение процессов с помощью имени сигнала и с помощью комбинации клавиш

3) Измените на 2 единицы приоритет процесса, запущенного из командного интерпретатора.

Для изменения единиц приоритета процесса запущенного из командного интерпретатора необходимо посмотреть на приоритеты текущих процессов, для этого выполним следующую команду: `ps -f -eo uid,pid,ppid,c,time,tt,time,cmd,ni`. В данной команде NI – это приоритет процесса. Изменим приоритет процесса на 2 единицы, для этого необходимо выполнить команду `renice`. Обратимся к помощи по аргументам данной команды, нам необходим ключ `-n` (на сколько увеличивать приоритет процесса) и `-p` (PID процесса). После выполнения снова посмотрим на список текущих процессов пользователя, для этого выполним команду: `ps -f -eo uid,pid,ppid,c,time,tt,time,cmd,ni`. Полученный результат выполнения команд изображен на рисунке 29.

```
elemabor@ubuntu:~$ ps -f -eo uid,pid,ppid,c,time,tt,time,cmd,ni
UID    PID    PPID  C   TIME TT      CMD                                NI
1000   1648   1612  0 00:00:02 tty2    -bash INVOCATION_ID=219b03b      0
1000   1376   1341  0 00:00:00 tty1    -bash INVOCATION_ID=95d9437      0
1000   1911   1376  99 00:00:08 tty1    \_ sh loop LS_COLORS=rs=0:      0
1000   1913   1376  0 00:00:00 tty1    \_ ps -f -eo uid,pid,ppid,      0
elemabor@ubuntu:~$ renice --help

Usage:
renice [-n] <priority> [-p|--pid] <pid>...
renice [-n] <priority> -g|--pgrp <pgrp>...
renice [-n] <priority> -u|--user <user>...

Alter the priority of running processes.

Options:
-n, --priority <num>    specify the nice increment value
-p, --pid <id>           interpret argument as process ID (default)
-g, --pgrp <id>         interpret argument as process group ID
-u, --user <name>|<id>  interpret argument as username or user ID

-h, --help              display this help
-V, --version            display version

For more details see renice(1).
elemabor@ubuntu:~$ renice -n 2 -p 1911
1911 (process ID) old priority 0, new priority 2
elemabor@ubuntu:~$ ps -f -eo uid,pid,ppid,c,time,tt,time,cmd,ni
UID    PID    PPID  C   TIME TT      CMD                                NI
1000   1648   1612  0 00:00:02 tty2    -bash INVOCATION_ID=219b03b      0
1000   1376   1341  0 00:00:00 tty1    -bash INVOCATION_ID=95d9437      0
1000   1911   1376  97 00:00:28 tty1    \_ sh loop LS_COLORS=rs=0:      2
1000   1916   1376  0 00:00:00 tty1    \_ ps -f -eo uid,pid,ppid,      0
elemabor@ubuntu:~$
```

Рисунок 29 – Пример изменения приоритета процесса

Вывод

В ходе выполнения данной лабораторной работы была получена общая информация о понятии процессов в операционной системе и получены навыки управления процессами в операционной системе Linux.