Липецкий государственный технический университет

Кафедра прикладной математики

Отчет по лабораторной работе №2
«Процессы в операционной системе Linux.»

Студент _____ Стукановский А.О.
подпись, дата фамилия, инициалы

Группа ПМ-18

Руководитель
доц., к.п.н. кафедры АСУ _____ Кургасов В. В.
ученая степень, ученое звание подпись, дата фамилия, инициалы

Липецк 2020 г.

Содержание

Цель работы **3**

Задание кафедры **3**

Выполнение работы **5**

Часть I: . . . 5

Загрузка пользователем. . . . 5

Поиск файла с образом ядра. Определение по имени файла номера версии Linux. . . . 5

Просмотр процессов ps –f. . . . 5

Создание с помощью редактора vi двух сценариев loop и loop2. . . . 5

Запустить loop2 на переднем плане. . . . 7

Остановка процесса через сигнал STOP. . . . 7

Последовательный просмотр процессов через ps –f. . . . 8

Использование сигнала kill -9 PID, чтобы Убить процесс loop2. . . . 8

Запуск в фоне процесса loop: sh loop. . . . 8

Завершение процесса loop командой kill -15 PID. . . . 9

Третий запуск в фоне процесса loop2. Командой kill -9 PID без остановки убить данный процесс. . . . 9

Запуск экземпляра оболочки: bash. . . . 9

Запуск нескольких процессов в фоне. . . . 10

Часть II: . . . 11

Запуск в консоли на выполнение трех задач(две в интерактивном режиме, одна - в фоновом). . . . 11

Перевод одной из задач, выполняющихся в интерактивном режиме, в фоновый режим. . . . 11

Перевод задач из фонового режима в интерактивный и наоборот. . . . 11

Создание именованного канала для архивирования и осуществление передачи в канал. . . . 12

Часть III (вариант 8): . . . 12

Сохранение в файл мгновенного состояния процессов системы указанного пользователя. 12

Послать сигнал SIGINT (по имени и по номеру сигнала) всем процессам, запущенным командой vi. Сообщить, успешно ли был послан сигнал. . . . 13

Измените на 3 единицы приоритеты процессов, владельцем которых является текущий пользователь. . . . 13

Вывод **15**

Список литературы **16**

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 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, послав сигнал STOP.
7) Посмотреть последовательно несколько раз ps –f. Записать сообщение, объяснить.
8) Убить процесс loop2, послав 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 loop 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 режиме, одну - в фоновом.
2) Перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим.
3) Провести эксперименты по переводу задач из фонового режима в интерактивный и наоборот.
4) 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.
5) 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 (Вариант 8):

1) Сохранить в файл мгновенное состояние процессов системы указанного пользователя.

2) Послать сигнал SIGINT (по имени и по номеру сигнала) всем процессам, запущенным командой vi. Сообщить, успешно ли был послан сигнал.

3) Измените на 3 единицы приоритеты процессов, владельцем которых является текущий пользователь.

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.

Выполнение работы

Часть I:

Загрузка пользователем.

Для начала авторизуемся пользователем Artem (рисунок 1).

Рисунок 1.

Поиск файла с образом ядра. Определение по имени файла номера версии Linux.

Файл ядра расположен в директории /boot и называется vmlinuz-<версия> (рисунок 2). Как видно, таких файлов несколько, поэтому узнаем точную версию ядра по средствам команды uname -r.

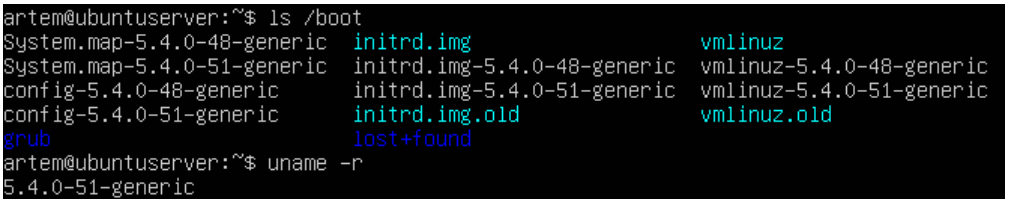

Рисунок 2.

Просмотр процессов ps –f.

Для того, чтобы просмотреть удобный листинг процессов активного пользователя, воспользуемся командой ps -f, результат на рисуноке 3.

Утилита ps выводит снимок процессов. Позволяет найти процессы по имени, пользователю или терминалу.

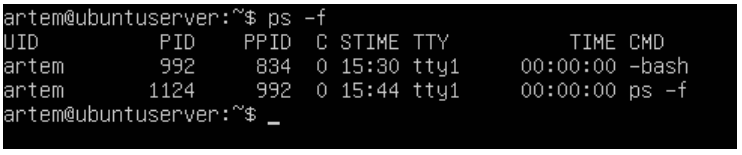

Рисунок 3.

Создание с помощью редактора vi двух сценариев loop и loop2.

С помощью редактора vi создадим два сценария loop и loop2, где сценарий loop содержит (рисунок 4), а loop2 - (рисунок 5). Для создания сценариев используется команда vi <название>(67).

Рисунок 4.

Рисунок 5.

Рисунок 6.

Рисунок 7.

После создания, сценарии остаются в корневой папке пользователя, создавшего их.

Запустить loop2 на переднем плане.

Запустим сценарий loop2 на исполнение. Воспользуемся командой sh <имя$_{>(8)}$.

Рисунок 8.

Остановка процесса через сигнал STOP.

Сигнал STOP приостанавливает процесс: в таком состоянии процесс не удаляется из таблицы процессов, но и не выполняется. Приостановить процесс можно комбинацией клавиш "Ctrl + Z"(рисунок 9).

Рисунок 9.

Последовательный просмотр процессов через ps –f.

Выполнив команду ps -f несколько раз, можем обнаружить(рисунок 10), что процесс до сих пор отображается в списке процессов, однако время выполнения программы остаётся неизменным.

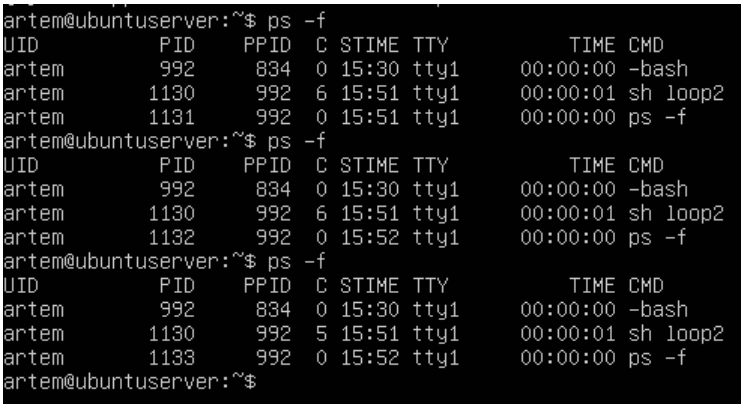

Рисунок 10.

Использование сигнала kill -9 PID, чтобы Убить процесс loop2.

На рисунке 10 можно заметить, что в таблице процессов содержится процесс loop2 с PID 1130. Воспользовавшись командой kill -9 1130 мы сможем завершить процесс.

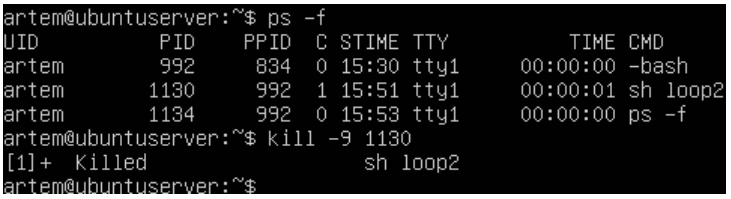

Рисунок 11.

Так же, из рисунка видно, что консоль сообщает нам о том, что процесс был завершён.

Запуск в фоне процесса loop: sh loop.

Для того, чтобы запустить процесс сценария параллельно, достаточно добавить в конец командной строки символ "": sh имя$_{(12)}$.

```
artem@ubuntuserver:~$ sh loop&
[1] 1135
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1135     992 92 15:54 tty1     00:00:03 sh loop
artem       1136     992  0 15:54 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1135     992 99 15:54 tty1     00:00:05 sh loop
artem       1137     992  0 15:54 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1135     992 95 15:54 tty1     00:00:07 sh loop
artem       1138     992  0 15:54 tty1     00:00:00 ps -f
artem@ubuntuserver:~$
```

Рисунок 12.

Завершение процесса loop командой kill -15 PID.

На рисунке 13 приведена таблица процессов, где присутствует исполняемый в фоновом режиме процесс loop.

```
artem@ubuntuserver:~$ kill -15 1135
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1139     992  0 15:56 tty1     00:00:00 ps -f
[1]+  Terminated              sh loop
artem@ubuntuserver:~$ _
```

Рисунок 13.

Сигнал номер 15 (TERM) служит для прерывания работы задания.

Третий запуск в фоне процесса loop2. Командой kill -9 PID без остановки убить данный процесс.

Через команду sh loop вновь запускается в фоновом режиме процесс loop. Для удаления процесса в этот раз используем команду kill -9 <PID> (рисунок 14).

```
artem@ubuntuserver:~$ sh loop&
[1] 1140
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1140     992 96 15:58 tty1     00:00:04 sh loop
artem       1141     992  0 15:58 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ kill -9 1140
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1142     992  0 15:58 tty1     00:00:00 ps -f
[1]+  Killed                  sh loop
```

Рисунок 14.

Запуск экземпляра оболочки: bash.

Чтобы запустить новый экземпляр оболочки bash, используем команду bash (рисунок 15).

```
artem@ubuntuserver:~$ bash
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1154    1148  0 16:00 tty1     00:00:00 ps -f
```

Рисунок 15.

Запуск нескольких процессов в фоне.

На рисунке 16 осуществляется запуск в фоновом режиме трёх процессов loop в фоновом режиме. Через команду kill -19 <PID> процесс приостанавливается, что видно в полноформатном листинге ps -f (рисунок 16).

```
artem@ubuntuserver:~$ sh loop&
[1] 1155
artem@ubuntuserver:~$ sh loop&
[2] 1156
artem@ubuntuserver:~$ sh loop&
[3] 1157
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1155    1148 47 16:01 tty1     00:00:03 sh loop
artem       1156    1148 38 16:01 tty1     00:00:02 sh loop
artem       1157    1148 32 16:01 tty1     00:00:01 sh loop
artem       1158    1148  0 16:01 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ kill -19 1155
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1155    1148 31 16:01 tty1     00:00:09 sh loop
artem       1156    1148 36 16:01 tty1     00:00:10 sh loop
artem       1157    1148 35 16:01 tty1     00:00:09 sh loop
artem       1159    1148  0 16:01 tty1     00:00:00 ps -f

[1]+  Stopped                 sh loop
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1155    1148 26 16:01 tty1     00:00:09 sh loop
artem       1156    1148 38 16:01 tty1     00:00:12 sh loop
artem       1157    1148 37 16:01 tty1     00:00:12 sh loop
artem       1160    1148  0 16:01 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ _
```

Рисунок 16.

Далее для возобнавления работы используем kill -18 <PID>. Листинг, подтверждающий это, приведён на рисунке 17.

```
artem@ubuntuserver:~$ kill -18 1155
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1155    1148 16 16:01 tty1     00:00:10 sh loop
artem       1156    1148 42 16:01 tty1     00:00:28 sh loop
artem       1157    1148 42 16:01 tty1     00:00:27 sh loop
artem       1161    1148  0 16:02 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        992     834  0 15:30 tty1     00:00:00 -bash
artem       1148     992  0 16:00 tty1     00:00:00 bash
artem       1155    1148 17 16:01 tty1     00:00:12 sh loop
artem       1156    1148 42 16:01 tty1     00:00:29 sh loop
artem       1157    1148 41 16:01 tty1     00:00:29 sh loop
artem       1162    1148  0 16:02 tty1     00:00:00 ps -f
artem@ubuntuserver:~$
```

Рисунок 17.

Часть II:

Запуск в консоли на выполнение трех задач(две в интерактивном режиме, одна - в фоновом).

Для начала создадим 2 интерактивных процесса. Используем sh loop и будем приостанавливать каждый из них с помошью "Ctrl + Z в резуль. Так же запустим один процесс в фоне sh loop. Результат приведён на рисунке 18.

```
artem@ubuntuserver:~$ sh loop
^Z
[1]+  Stopped                 sh loop
artem@ubuntuserver:~$ sh loop
^Z
[2]+  Stopped                 sh loop
artem@ubuntuserver:~$ sh loop&
[3] 933
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        922     641  0 19:04 tty1     00:00:00 -bash
artem        931     922 13 19:04 tty1     00:00:03 sh loop
artem        932     922  7 19:04 tty1     00:00:00 sh loop
artem        933     922 93 19:04 tty1     00:00:02 sh loop
artem        934     922  0 19:05 tty1     00:00:00 ps -f
```

Рисунок 18.

Перевод одной из задач, выполняющихся в интерактивном режиме, в фоновый режим.

Используя команду bg переведём процессы 1 и 2 (которые в данный момент стоят на паузе) в фоновый режим (рисунок 19).

```
artem@ubuntuserver:~$ bg 1 2
[1]- sh loop &
[2]+ sh loop &
```

Рисунок 19.

Перевод задач из фонового режима в интерактивный и наоборот.

В случае перевода процесса из фонового в интерактивный режим, программа сразу начинает взаимодействие с пользователем, в связи с чем её приходится ставит на паузу (рисунок 20).

```
artem@ubuntuserver:~$ fg 3
sh loop
^Z
[3]+  Stopped                 sh loop
artem@ubuntuserver:~$ bg
[3]+ sh loop &
artem@ubuntuserver:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
artem        922     641  0 19:04 tty1     00:00:00 -bash
artem        931     922 22 19:04 tty1     00:00:17 sh loop
artem        932     922 22 19:04 tty1     00:00:15 sh loop
artem        933     922 50 19:04 tty1     00:00:29 sh loop
artem        936     922  0 19:05 tty1     00:00:00 ps -f
artem@ubuntuserver:~$ _
```

Рисунок 20.

Создание именованного канала для архивирования и осуществление передачи в канал.

Для начала создадим поток pipe используя команду mkfifo. После передадив в неё список файлов домашнего каталога с подкаталогами, используя ключ -R (рисунок 21)

```
artem@ubuntuserver:~$ ls
1000  loop  loop2  result.txt  snap  test.sh  text.txt
artem@ubuntuserver:~$ mkfifo pipe
artem@ubuntuserver:~$ ls
1000  loop  loop2  pipe  result.txt  snap  test.sh  text.txt
artem@ubuntuserver:~$ ls -R > pipe
```

Рисунок 21.

А теперь передадим туда лишь файлы того каталога, в котором находимся, используем ls -l > pipe (рисунок 22).

```
artem@ubuntuserver:~$ ls -l > pipe
```

Рисунок 22.

Часть III (вариант 8):

Сохранение в файл мгновенного состояния процессов системы указанного пользователя.

Для умпрощения сохранения в файл состояния процессов системы указанного пользователя был написан следующий сценарий (рисунок 23).

```
read n_
(rm text.txt)
(ps -f -u $n) >> text.txt
~
~
~
```

Рисунок 23.

При его запуске интерактивная программа требует от вас ввода имени пользователя, информацию о процессах которого вы хотите сохранить. После ввода имени пользователя, программа создаёт txt документ с именем text (рисунок 24).

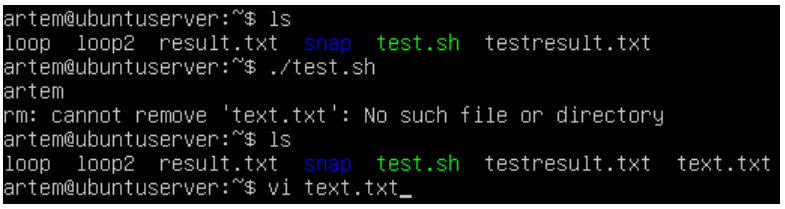

Рисунок 24.

Открыв текстовый файл мы увидим довольно подробный листинг (ps -f) для заданного пользователя, это видно на рисунке 25.

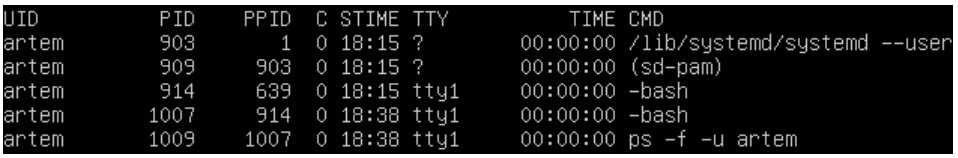

Рисунок 25.

Послать сигнал SIGINT (по имени и по номеру сигнала) всем процессам, запущенным командой vi. Сообщить, успешно ли был послан сигнал.

У нас имеются два активных процесса vi (рисунок 26).

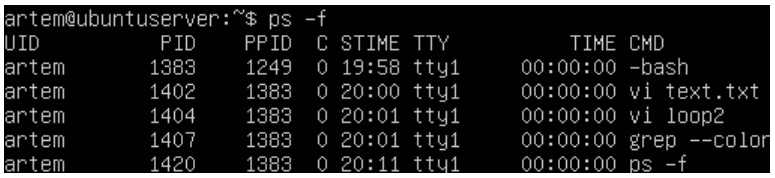

Рисунок 26.

Попробуем послать сигнал SIGINT двумя способами(по номеру и имени сигнала) на два этих процесса разом. Из рисунка 27 видно, что данный сигнал не оказывает никакого влияния на процессы vi text.txt и vi loop2.

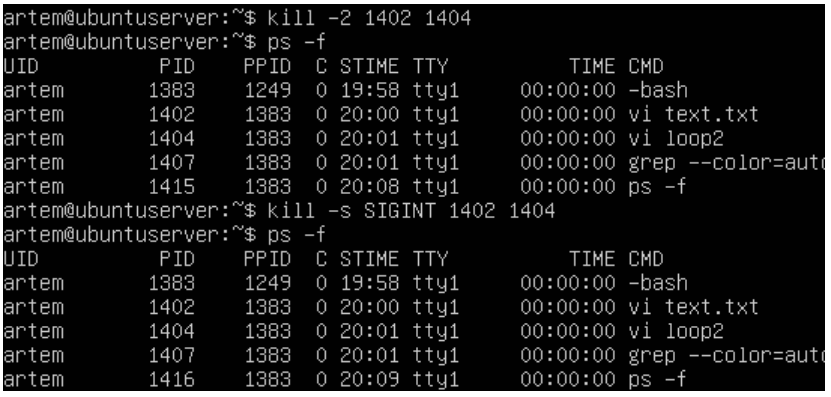

Рисунок 27.

Измените на 3 единицы приоритеты процессов, владельцем которых является текущий пользователь.

На рисунке 28 представлен листинг процессов пользователя artem с указанием их приоритетов.

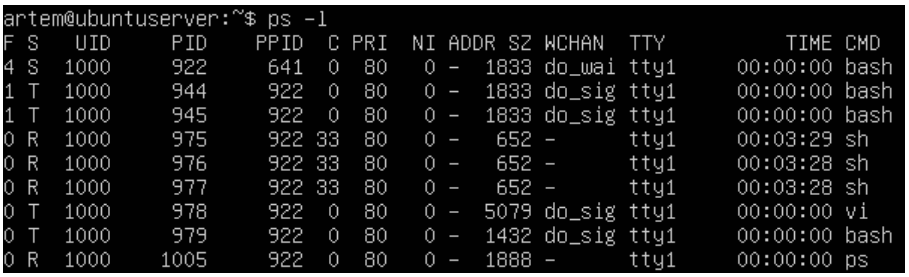

Рисунок 28.

Используя команду renice увеличим значение приоритета (NI) на 3 для всех процессов пользователя artem. Результат представлен на рисунке 29.

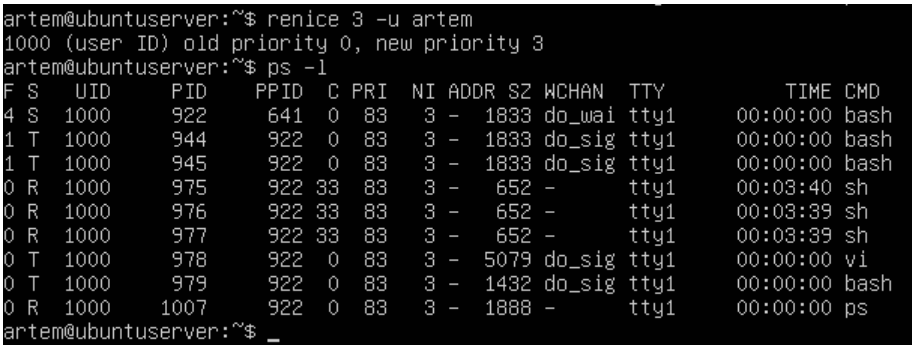

Рисунок 29.

Вывод

В ходе лабораторной работы были усвоены понятия процесса в операционной системе и приобретены навыки управления процессами в операционной системе Linux.

Список литературы

[1] Львовский, С.М. Набор и верстка в системе LaTeX [Текст] / С.М. Львовский. М.: МЦНМО, 2006. — 448 с.

[2] SEDICOMM. 30 полезных команд «ps» для мониторинга процессов Linux: https://blog.sedicomm.com/2018/05/28/30-poleznyh-komand-ps-dlya-monitoringa-protsessov-linux/ (дата обращения: 29.10.2020). - Текст: электронный.