

**Липецкий государственный технический университет**

**Факультет автоматизации и информатики**

**Кафедра автоматизированных систем управления**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**по дисциплине «Операционная система Linux»**

**Управление процессами в Linux**

Студент

Осипов А.А.

Группа АИ-18

Руководитель

Кургасов В.В.

Липецк 2020 г.

## Оглавление

Цель работы .....	3
Задание кафедры.....	4
Ход работы .....	5
Вывод .....	18

Цель работы

Приобрести опыт и навыки управления процессами в операционной системе Linux.

### Задание кафедры

1. Повторить команды cat, head, tail, more, less, grep, find
2. Разобраться с понятиями конвейер, перенаправление ввода-вывода(1).
3. Ознакомиться с информацией из рекомендованных источников(2) и других про конвейеризации(3).
4. Повторить назначение прав доступа (4). Команды chmod, chown
5. Ознакомиться с информацией(5) по теме процессы, посмотреть и опробовать примеры наиболее распространенных(6) команд, изучить возможность запуска процессов в supervisor (7).
6. Изучить возможность автоматического запуска программ по расписанию

## Ход работы

Повторим команды `cat` и `head`. Создадим с помощью команды `cat` файл `file_1` и выведем строки файла.

Команда `cat` - читает данные из файла или стандартного ввода и выводит их на экран.

- b - нумеровать только непустые строки;
- E - показывать символ \$ в конце каждой строки;
- n - нумеровать все строки;
- s - удалять пустые повторяющиеся строки;
- T - отображать табуляции в виде ^I;
- h - отобразить справку;
- v - версия утилиты.

Команда `head` - самый простой способ использования команды с указанием имени файла, но без опций. В таком случае будут выведены на экран первые 10 строк.

-c (--bytes) — позволяет задавать количество текста не в строках, а в байтах. При записи в виде `--bytes=[-]NUM` выводит на экран все содержимое файла, кроме NUM байт, расположенных в конце документа.

-n (--lines) — показывает заданное количество строк вместо 10, которые выводятся по умолчанию. Если записать эту опцию в виде `--lines=[-]NUM`, будет показан весь текст кроме последних NUM строк.

-q (--quiet, --silent) — выводит только текст, не добавляя к нему название файла.

-v (--verbose) — перед текстом выводит название файла.

-z (--zero-terminated) — символы перехода на новую строку заменяет символами завершения строк.

```
artem@artemserver:~$ cat > file_1
qwe
asd
zxc
^Z
[2]+  Stopped                  cat > file_1
artem@artemserver:~$ cat file_1
qwe
asd
zxc
artem@artemserver:~$ head file_1
qwe
asd
zxc
artem@artemserver:~$ _
```

Рисунок 1 – Использование команды cat и head

Повторим команду tail. Выведем последнюю строку из файла file\_1.

Команда tail - позволяет выводить заданное количество строк с конца файла, а также выводить новые строки в интерактивном режиме.

- с - выводить указанное количество байт с конца файла;
- f - обновлять информацию по мере появления новых строк в файле;
- n - выводить указанное количество строк из конца файла;
- pid - используется с опцией -f, позволяет завершить работу утилиты, когда завершится указанный процесс;
- q - не выводить имена файлов;
- retry - повторять попытки открыть файл, если он недоступен;
- v - выводить подробную информацию о файле;

```
artem@artemserver:~$ tail -n 1 file_1
zxc
artem@artemserver:~$
```

Рисунок 2 – Использование команды tail

Далее откроем файл two с помощью команд more и less

Команда more - предназначена для постраничного просмотра файлов в терминале Linux.

-d — вывод информации в конце страницы о клавишах, использующихся для продолжения работы, завершения её или получения инструкций;

-l — игнорирование в тексте символа разрыва страницы;

-f — подсчёт числа логических строк вместо экранных;

-p — очистка экрана терминала для того, чтобы пользователю не пришлось пользоваться прокруткой перед выводом следующей порции текста;

-c — устранение потребности в прокрутке (как и -p) — отображение текста, начиная с верха экрана, и стирание при этом предыдущего вывода построчно;

-s — замена нескольких пустых строк, расположенных подряд, одной пустой строкой;

-u — удаление подчёркивания;

-n — отображение n-го количества строк;

+n — отображение текста, начиная со строки с номером n;

+/строка — поиск в файле указанной строки и начало вывода текста именно с неё;

--help — вызов справки;

-v (--version) — вывод на экран текущей версии утилиты.

Команда less - позволяет перематывать текст не только вперёд, но и назад, осуществлять поиск в обоих направлениях, переходить сразу в конец или в начало файла.

-a, --search-skip-screen — не осуществлять поиск в тексте, который в данный момент отображен на экране;

-bn, --buffers=n — задать размер буфера памяти;

-c, --clear-screen — листать текст, полностью стирая содержимое экрана (построчная прокрутка работать не будет);

-Dxcolor, --color=xcolor — задать цвет отображаемого текста;

-E, --QUIT-AT-EOF — выйти, когда утилита достигнет конца файла;

-e, --quit-at-eof — выйти, когда утилита второй раз достигнет конца файла;

-F, --quit-if-one-screen — выйти, если содержимое файла помещается на одном экране;

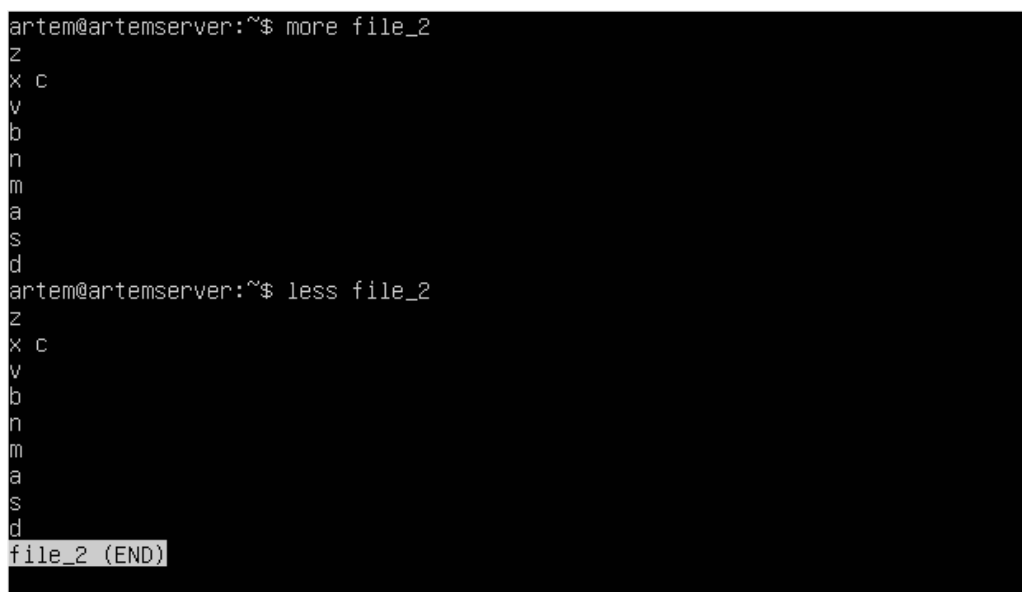
-f, --force — открыть специальный файл;

-hn, --max-back-scroll=n — задать максимальное количество строк для прокрутки назад;

-yn, --max-forw-scroll=n — задать максимальное количество строк для прокрутки вперёд;

-i, --ignore-case — игнорировать регистр;

-I, --IGNORE-CASE — игнорировать регистр, даже если паттерн для поиска содержит заглавные буквы;



```
artem@artemserver:~$ more file_2
z
x c
v
b
n
m
a
s
d
artem@artemserver:~$ less file_2
z
x c
v
b
n
m
a
s
d
file_2 (END)
```

Рисунок 3 – Использование команд more и less

С помощью команды `grep` найдём в файле `two` элемент `q`, с помощью команды `find` найдём файл `two`.

Команда `grep` - с её помощью можно искать не только строки в файлах, но и фильтровать вывод команд, и много чего ещё.

- b - показывать номер блока перед строкой;
- c - подсчитать количество вхождений шаблона;



- h - не выводить имя файла в результатах поиска внутри файлов Linux;
- i - не учитывать регистр;
- l - отобразить только имена файлов, в которых найден шаблон;
- n - показывать номер строки в файле;
- s - не показывать сообщения об ошибках;
- v - инвертировать поиск, выдавать все строки кроме тех, что содержат шаблон;

- w - искать шаблон как слово, окружённое пробелами;
- e - использовать регулярные выражения при поиске;
- An - показать вхождение и n строк до него;
- Bn - показать вхождение и n строк после него;
- Cn - показать n строк до и после вхождения;

Команда `find` - команда для поиска файлов и каталогов на основе специальных условий. Ее можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и другим подобным критериям.

- P никогда не открывать символические ссылки

- L - получает информацию о файлах по символическим ссылкам. Важно для дальнейшей обработки, чтобы обрабатывалась не ссылка, а сам файл.

- maxdepth - максимальная глубина поиска по подкаталогам, для поиска только в текущем каталоге установите 1.

- depth - искать сначала в текущем каталоге, а потом в подкаталогах

- mount искать файлы только в этой файловой системе.

- version - показать версию утилиты `find`

- print - выводить полные имена файлов

- type f - искать только файлы

- type d - поиск папки в Linuxашт

```
artem@artemserver:~$ grep -n -i qwe file_1
1:qwe
artem@artemserver:~$ find file_2
file_2
artem@artemserver:~$ _
```

Рисунок 4 – Использование команд grep и find

Следующим заданием нужно разобраться с понятием конвейера.

Конвейер или канал в linux перенаправляет вывод одной команды в ввод другой, при этом не показывая промежуточный результат выполнения команды, а сразу показывает в терминале отфильтрованный вывод. В данном случае мы хотим направить stdout команды ls на stdin команды sort.

```
artem@artemserver:~$ ls | sort -r
ress
res
new
loop.sh
loop2.sh
loop
file_2
file_1
file
channel
artem@artemserver:~$
```

Рисунок 5 – Использование состыкованных команд (конвейер)

Затем нам нужно разобраться с перенаправлением ввода-вывода.

Стандартные потоки ввода и вывода в linux являются одним из наиболее распространенных средств для обмена информацией процессов, а перенаправление `>`, `>>` и `|` является одной из самых популярных конструкций командного интерпретатора bash.

Операторы перенаправления:

`>` - перезаписывает информацию из файла, если там уже что-то есть, если файла не существует – файл создается.

`>>` - перенаправляет стандартный поток в файл. При этом если файл существует, то информация добавляется в конец, если не существует – файл создается.

`<` - считывание данных из файла.

> & - перенаправляет стандартные потоки вывода и ошибок друг в друга.

```
artem@artemserver:~$ cat file_1
qwe
asd
zxc
artem@artemserver:~$ cat file_2
q
w
e
r
t
y
u
i
artem@artemserver:~$ sort <file_1> file_2
artem@artemserver:~$ cat file_2
asd
qwe
zxc
artem@artemserver:~$ cat file_1
qwe
asd
zxc
artem@artemserver:~$
```

Рисунок 6 – Перенаправление ввода-вывода

Выполним следующее задание, используем команды `chmod`, `chown`.

Команда `chmod` используется для изменения прав доступа к файлам или каталогам.

r - чтение;

w - запись;

x - выполнение;

s - выполнение от имени суперпользователя (дополнительный);

Также есть три категории пользователей, для которых вы можете установить эти права на файл linux:

u - владелец файла;

g - группа файла;

o - все остальные пользователи;

```

artem@artemserver:~$ ls -l
total 36
-rw-rw-r-- 1 artem artem 0 Nov 6 10:09 channel
-rw-rw-r-- 1 artem artem 12 Nov 9 13:06 file
-rw-rw-r-- 1 artem artem 12 Nov 9 17:20 file_1
-rw-rw-r-- 1 artem artem 12 Nov 9 17:21 file_2
-rw-rw-r-- 1 artem artem 27 Nov 5 11:00 loop
-rwxrwxr-x 1 artem artem 40 Nov 6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem 26 Nov 6 09:02 loop.sh
drwxrwxr-x 3 artem artem 4096 Nov 6 10:02 new
-rw-rw-r-- 1 artem artem 75 Nov 6 10:06 res
-rw-rw-r-- 1 artem artem 95 Nov 6 10:09 ress
artem@artemserver:~$ chmod 777 file
artem@artemserver:~$ ls -l
total 36
-rw-rw-r-- 1 artem artem 0 Nov 6 10:09 channel
-rwxrwxrwx 1 artem artem 12 Nov 9 13:06 file
-rw-rw-r-- 1 artem artem 12 Nov 9 17:20 file_1
-rw-rw-r-- 1 artem artem 12 Nov 9 17:21 file_2
-rw-rw-r-- 1 artem artem 27 Nov 5 11:00 loop
-rwxrwxr-x 1 artem artem 40 Nov 6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem 26 Nov 6 09:02 loop.sh
drwxrwxr-x 3 artem artem 4096 Nov 6 10:02 new
-rw-rw-r-- 1 artem artem 75 Nov 6 10:06 res
-rw-rw-r-- 1 artem artem 95 Nov 6 10:09 ress
artem@artemserver:~$ _

```

Рисунок 7 – Использование команды chmod

Команда `chown` позволяет использовать соответствующую утилиту для изменения владельца файла или директории.

-c, --changes - подробный вывод всех выполняемых изменений;

-f, --silent, --quiet - минимум информации, скрыть сообщения об ошибках;

--dereference - изменять права для файла к которому ведет символическая ссылка вместо самой ссылки (поведение по умолчанию);

-h, --no-dereference - изменять права символических ссылок и не трогать файлы, к которым они ведут;

--from - изменять пользователя только для тех файлов, владельцем которых является указанный пользователь и группа;

-R, --recursive - рекурсивная обработка всех подкаталогов;

-H - если передана символическая ссылка на директорию - перейти по ней;

-L - переходить по всем символическим ссылкам на директории;

-P - не переходить по символическим ссылкам на директории (по умолчанию).

744 - разрешить все для владельца, а остальным только чтение;  
755 - все для владельца, остальным только чтение и выполнение;  
764 - все для владельца, чтение и запись для группы, и только чтение для остальных;

777 - всем разрешено все.

```
artem@artemserver:~$ ls -l
total 36
prw-rw-r-- 1 artem artem    0 Nov  6 10:09 channel
-rwxrwxrwx 1 artem artem   12 Nov  9 13:06 file
-rw-rw-r-- 1 artem artem   12 Nov  9 17:20 file_1
-rw-rw-r-- 1 artem artem   12 Nov  9 17:21 file_2
-rw-rw-r-- 1 artem artem   27 Nov  5 11:00 loop
-rwxrwxr-x 1 artem artem   40 Nov  6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem   26 Nov  6 09:02 loop.sh
drwxrwxr-x 3 artem artem 4096 Nov  6 10:02 new
-rw-rw-r-- 1 artem artem   75 Nov  6 10:06 res
-rw-rw-r-- 1 artem artem   95 Nov  6 10:09 ress
artem@artemserver:~$ sudo chown root file_2
[sudo] password for artem:
artem@artemserver:~$ ls -l
total 36
prw-rw-r-- 1 artem artem    0 Nov  6 10:09 channel
-rwxrwxrwx 1 artem artem   12 Nov  9 13:06 file
-rw-rw-r-- 1 artem artem   12 Nov  9 17:20 file_1
-rw-rw-r-- 1 root  artem   12 Nov  9 17:21 file_2
-rw-rw-r-- 1 artem artem   27 Nov  5 11:00 loop
-rwxrwxr-x 1 artem artem   40 Nov  6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem   26 Nov  6 09:02 loop.sh
drwxrwxr-x 3 artem artem 4096 Nov  6 10:02 new
-rw-rw-r-- 1 artem artem   75 Nov  6 10:06 res
-rw-rw-r-- 1 artem artem   95 Nov  6 10:09 ress
artem@artemserver:~$
```

Рисунок 8 – Использование команды chown

Посмотрим и опробуем примеры наиболее распространенных команд.

Команда top - команда предназначена для вывода списка работающих процессов в системе и информацию о них.

- v - вывести версию программы;
- b - режим только для вывода данных, программа не воспринимает интерактивных команд и выполняется пока не будет завершена вручную;
- c - отображать полный путь к исполняемым файлам команд;
- d - интервал обновления информации;
- H - включает вывод потоков процессов;
- i - не отображать процессы, которые не используют ресурсы процессора;

-n - количество циклов обновления данных, после которых надо закрыть программу;

-o - поле, по которому надо выполнять сортировку;

-O - вывести все доступные поля для сортировки;

-p - отслеживать только указанные по PID процессы, можно указать несколько PID;

-u - выводить только процессы, запущенные от имени указанного пользователя.

С опциями запуска всё, теперь давайте поговорим про интерактивные команды, которые вы можете выполнять во время работы программы.

h - вывод справки по утилите;

q или Esc - выход из top;

A - выбор цветовой схемы;

d или s - изменить интервал обновления информации;

H - выводить потоки процессов;

k - послать сигнал завершения процессу;

W - записать текущие настройки программы в конфигурационный файл;

Y - посмотреть дополнительные сведения о процессе, открытые файлы, порты, логи и т.д.;

Z - изменить цветовую схему;

l - скрыть или вывести информацию о средней нагрузке на систему;

m - выключить или переключить режим отображения информации о памяти;

x - выделять жирным колонку, по которой выполняется сортировка;

y - выделять жирным процессы, которые выполняются в данный момент;

z - переключение между цветным и одноцветным режимами;

c - переключение режима вывода команды, доступен полный путь и только команда;

F - настройка полей с информацией о процессах;

o - фильтрация процессов по произвольному условию;

- u - фильтрация процессов по имени пользователя;
- V - отображение процессов в виде дерева;
- i - переключение режима отображения процессов, которые сейчас не используют ресурсы процессора;
- n - максимальное количество процессов, для отображения в программе;
- L - поиск по слову;
- <> - перемещение поля сортировки вправо и влево;

```
top - 18:10:06 up 1:23, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 94 total, 1 running, 91 sleeping, 2 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.7 id, 0.3 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 981.3 total, 572.0 free, 127.9 used, 281.3 buff/cache
MiB Swap: 1781.0 total, 1781.0 free, 0.0 used, 706.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	101968	11348	8256	S	0.0	1.1	0:01.18	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-kblockd
9	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
10	root	20	0	0	0	0	S	0.0	0.0	0:00.12	ksoftirqd/0
11	root	20	0	0	0	0	I	0.0	0.0	0:01.43	rcu_sched
12	root	rt	0	0	0	0	S	0.0	0.0	0:00.03	migration/0
13	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
14	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
15	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
16	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
17	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_kthre
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kauditd
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khungtaskd
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper
21	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeback
22	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kcompactd0
23	root	25	5	0	0	0	S	0.0	0.0	0:00.00	ksmd
24	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
70	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd
71	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kblockd
72	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	blkcg_punt_bio
73	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	tpm_dev_wq
74	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	ata_sff
75	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	md
76	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	edac-poller
77	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	devfreq_wq
78	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	watchdogd

Рисунок 9 – использование команды top

```

top - 18:16:45 up 1:30, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 93 total, 1 running, 90 sleeping, 2 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 981.3 total, 572.0 free, 127.9 used, 281.3 buff/cache
MiB Swap: 1781.0 total, 1781.0 free, 0.0 used, 706.1 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0   101968  11348   8256 S   0.0   1.1   0:01.18 systemd
    2 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kthreadd
    3 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 rcu_gp
    4 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 rcu_par_gp
    6 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kworker/0:0H-kblockd
    9 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
   10 root        20   0         0         0      0 S   0.0   0.0   0:00.13 ksoftirqd/0
   11 root        20   0         0         0      0 I   0.0   0.0   0:01.48 rcu_sched
   12 root        rt    0         0         0      0 S   0.0   0.0   0:00.03 migration/0
   13 root       -51   0         0         0      0 S   0.0   0.0   0:00.00 idle_inject/0
   14 root        20   0         0         0      0 S   0.0   0.0   0:00.00 cpuhp/0
   15 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kdevtmpfs
   16 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 netns
   17 root        20   0         0         0      0 S   0.0   0.0   0:00.00 rcu_tasks_kthre
   18 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kauditd
   19 root        20   0         0         0      0 S   0.0   0.0   0:00.00 khungtaskd
   20 root        20   0         0         0      0 S   0.0   0.0   0:00.00 oom_reaper
   21 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 writeback
   22 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kcompactd0
   23 root        25   5         0         0      0 S   0.0   0.0   0:00.00 ksm
   24 root       39  19         0         0      0 S   0.0   0.0   0:00.00 khugepaged
   70 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kintegrityd
   71 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kblockd
   72 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 blkcg_punt_bio
   73 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 tpm_dev_wq
   74 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 ata_sff
   75 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 md
   76 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 edac-poller
   77 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 devfreq_wq
   78 root        rt    0         0         0      0 S   0.0   0.0   0:00.00 watchdogd

```

Рисунок 10 – Использование команды top, пример нажатия клавиши z  
Изменим кнопкой d интервал обновления экрана с 3.0 секунд до 1.5 секунд.

```

top - 18:12:25 up 1:25, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 94 total, 1 running, 91 sleeping, 2 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 981.3 total, 572.0 free, 127.9 used, 281.3 buff/cache
MiB Swap: 1781.0 total, 1781.0 free, 0.0 used, 706.1 avail Mem
Change delay from 3.0 to 1.5

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
  912 root        20   0         0         0      0 I   2.4   0.0   0:04.60 kworker/0:1-events
  976 artem      20   0    9124    3768   3244 R   2.4   0.4   0:00.31 top
    1 root        20   0   101968  11348   8256 S   0.0   1.1   0:01.18 systemd
    2 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kthreadd
    3 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 rcu_gp
    4 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 rcu_par_gp
    6 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kworker/0:0H-kblockd
    9 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
   10 root        20   0         0         0      0 S   0.0   0.0   0:00.12 ksoftirqd/0
   11 root        20   0         0         0      0 I   0.0   0.0   0:01.44 rcu_sched
   12 root        rt    0         0         0      0 S   0.0   0.0   0:00.03 migration/0
   13 root       -51   0         0         0      0 S   0.0   0.0   0:00.00 idle_inject/0
   14 root        20   0         0         0      0 S   0.0   0.0   0:00.00 cpuhp/0
   15 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kdevtmpfs
   16 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 netns
   17 root        20   0         0         0      0 S   0.0   0.0   0:00.00 rcu_tasks_kthre
   18 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kauditd
   19 root        20   0         0         0      0 S   0.0   0.0   0:00.00 khungtaskd
   20 root        20   0         0         0      0 S   0.0   0.0   0:00.00 oom_reaper
   21 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 writeback
   22 root        20   0         0         0      0 S   0.0   0.0   0:00.00 kcompactd0
   23 root        25   5         0         0      0 S   0.0   0.0   0:00.00 ksm
   24 root       39  19         0         0      0 S   0.0   0.0   0:00.00 khugepaged
   70 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kintegrityd
   71 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 kblockd
   72 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 blkcg_punt_bio
   73 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 tpm_dev_wq
   74 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 ata_sff
   75 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 md
   76 root         0 -20         0         0      0 I   0.0   0.0   0:00.00 edac-poller

```

Рисунок 11 - Использование команды top, пример нажатия клавиши d



При нажатии на клавишу h получим справку по верхней команде.

```
Help for Interactive Commands - procpss-ng UNKNOWN
Window 1:Def: Cumulative mode Off. System: Delay 1.5 secs; Secure mode Off.

Z,B,E,e Global: 'Z' colors; 'B' bold; 'E'/'e' summary/task memory scale
l,t,m Toggle Summary: 'l' load avg; 't' task/cpu stats; 'm' memory info
0,1,2,3,I Toggle: '0' zeros; '1/2/3' cpus or numa node views; 'I' Irix mode
f,F,X Fields: 'f'/'F' add/remove/order/sort; 'X' increase fixed-width

L,&,<,> . Locate: 'L'/'&' find/again; Move sort column: '<'/'>' left/right
R,H,J,C . Toggle: 'R' Sort; 'H' Threads; 'J' Num justify; 'C' Coordinates
c,i,S,j . Toggle: 'c' Cmd name/line; 'i' Idle; 'S' Time; 'j' Str justify
x,y . Toggle highlights: 'x' sort field; 'y' running tasks
z,b . Toggle: 'z' color/mono; 'b' bold/reverse (only if 'x' or 'y')
u,U,o,O . Filter by: 'u'/'U' effective/any user; 'o'/'O' other criteria
n,#,^O . Set: 'n'/'#' max tasks displayed; Show: Ctrl+'O' other filter(s)
V,v . Toggle: 'V' forest view; 'v' hide/show forest view children

k,r Manipulate tasks: 'k' kill; 'r' renice
d or s Set update interval
W,Y Write configuration file 'W'; Inspect other output 'Y'
q Quit
( commands shown with '.' require a visible task display window )
Press 'h' or '?' for help with Windows,
Type 'q' or <Esc> to continue █
```

Рисунок 12 – Использование команды top, пример нажатия клавиши h

Изучим возможность запуска процессов в supervisor.

Supervisor – это менеджер процессов, который существенно упрощает управление долго работающими программами, предоставляя простой и понятный интерфейс. Помимо этого, он способен обеспечить бесперебойную работу веб-сервиса.

Изучим возможность автоматического запуска программ по расписанию.

Cron - планировщик, который позволяет выполнять нужные вам скрипты раз в час, раз в день, неделю или месяц, а также в любое заданное вами время или через любой интервал.

## Вывод

В ходе выполнения лабораторной работы я ознакомился на практике с понятием процесса в операционной системе, приобрел опыт и навыки управления процессами в операционной системе Linux.