

# **Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

## **ЛАБОРАТОРНАЯ РАБОТА №3**

По дисциплине «ОС Linux»

Управление процессами. Планировщик

Студент

Чаплыгин И.С.

Группа ПИ-18

Руководитель

Доцент

Кургасов В.В.

Липецк 2020г

## 1. Цель работы

Закрепить изученный материал. Ознакомится с конвейеризацией и наиболее распространенными командами Linux. Изучить возможность запуска программ по расписанию.

## 2. Задание кафедры

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

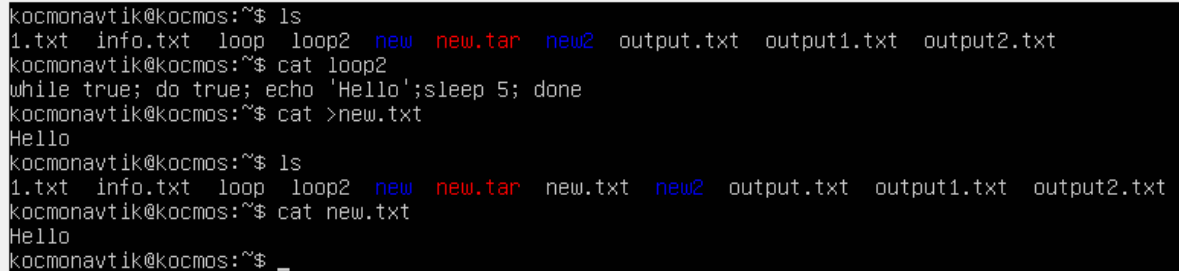
## Оглавление

1. Цель работы .....	2
2. Задание кафедры.....	3
3. Выполнение работы .....	5
3.1 Повторение команд .....	5
3.2 Перенаправление ввода – вывода.....	12
3.3 Процессы .....	17
3.4 Supervisor .....	26
3.5 Планировщик задач .....	29
4. Вывод.....	30

### 3. Выполнение работы

#### 3.1 Повторение команд

Команда “cat” отвечает за просмотр текстовых файлов (команда “cat имя\_файла”, а также их создания (cat > имя\_файла).



```
kosmonavtik@kocmos:~$ ls
1.txt info.txt loop loop2 new new.tar new2 output.txt output1.txt output2.txt
kosmonavtik@kocmos:~$ cat loop2
while true; do true; echo 'Hello';sleep 5; done
kosmonavtik@kocmos:~$ cat >new.txt
Hello
kosmonavtik@kocmos:~$ ls
1.txt info.txt loop loop2 new new.tar new.txt new2 output.txt output1.txt output2.txt
kosmonavtik@kocmos:~$ cat new.txt
Hello
kosmonavtik@kocmos:~$ _
```

Рисунок 1 – Использование команды “cat”

После создания файла, можно сразу же ввести текст и нажать комбинацию клавиш для сохранения “Ctrl+D”, или выйти из режима записи в файл “Ctrl+C” или “Ctrl+Z”.

Так же существует команда “tac”, которая выводит информацию в обратном порядке

Команда “head” так же отвечает за просмотр файлов, но имеет функцию выводить определенное количество строк. По умолчанию команда выводит 10 строк.

```
kocmonavtik@kocmos:~$ head new.txt
.:
1.txt
info.txt
loop
loop2
new
new.tar
new.txt
new2
output.txt
kocmonavtik@kocmos:~$ head -n15 new.txt
.:
1.txt
info.txt
loop
loop2
new
new.tar
new.txt
new2
output.txt
output1.txt
output2.txt

./new:
new2
kocmonavtik@kocmos:~$ _
```

Рисунок 2 – Использование команды “head”

Как видно по рисунку, для вывода определенного количества строк, добавляется опция “-n15”, где 15 – количество строк для вывода.

Команда “tail” схожа с командой head, только выводит заданное количество строк с конца файла.

```
kosmonavtik@kosmos:~$ tail new.txt
test2.txt

./new/new2:

./new2:
new2
test1.txt
test2.txt

./new2/new2:
kosmonavtik@kosmos:~$ tail -n15 new.txt
output2.txt

./new:
new2
test1.txt
test2.txt

./new/new2:

./new2:
new2
test1.txt
test2.txt

./new2/new2:
kosmonavtik@kosmos:~$ _
```

Рисунок 3 – Использование команды “tail”

Команда “more” служит для постраничного просмотра файлов в терминале Linux.

```
/boot:
System.map-5.4.0-48-generic
System.map-5.4.0-52-generic
config-5.4.0-48-generic
config-5.4.0-52-generic
grub
initrd.img
initrd.img-5.4.0-48-generic
initrd.img-5.4.0-52-generic
initrd.img.old
lost+found
vmlinuz
vmlinuz-5.4.0-48-generic
vmlinuz-5.4.0-52-generic
vmlinuz.old

/boot/grub:
fonts
gfxblacklist.txt
grub.cfg
grubenv
i386-pc
unicode.pf2

/boot/grub/fonts:
unicode.pf2

/boot/grub/i386-pc:
915resolution.mod
acpi.mod
adler32.mod
affs.mod
afs.mod
ahci.mod
all_video.mod
asout.mod
--More-- (12%)
```

Рисунок 4 – Использование команды “more”

Команда “less” является более функциональной, по сравнению с “more”. Она так же предназначена для постраничного просмотра больших текстовых файлов, которая позволяет перематывать текст не только вперед, но и назад, осуществлять поиск в обоих направлениях, переходить в конец или в начало файла.

После ввода команды “less new.txt” мы увидим следующее:



```

/boot:
System.map-5.4.0-48-generic
System.map-5.4.0-52-generic
config-5.4.0-48-generic
config-5.4.0-52-generic
grub
initrd.img
initrd.img-5.4.0-48-generic
initrd.img-5.4.0-52-generic
initrd.img.old
lost+found
vmlinuz
vmlinuz-5.4.0-48-generic
vmlinuz-5.4.0-52-generic
vmlinuz.old

/boot/grub:
fonts
gfxblacklist.txt
grub.cfg
grubenv
i386-pc
unicode.pf2

/boot/grub/fonts:
unicode.pf2

/boot/grub/i386-pc:
915resolution.mod
acpi.mod
adler32.mod
affs.mod
afs.mod
ahci.mod
all_video.mod
ebout.mod
:_

```

Рисунок 5 – Использование команды “less”

Данная команда имеет множество различных опций для поиска в тексте, изменения внешнего вида и т.п.

Команда “grep” – одна из самых востребованных команд в терминале Linux. Эта утилита решает множество задач, в основном она используется для поиска строк, соответствующих строке в тексте или содержанию файлов.

```

kocmonavtik@kocmos:~$ ls -R /home |grep kocmonavtik
kocmonavtik
/home/kocmonavtik:
/home/kocmonavtik/new:
/home/kocmonavtik/new/new2:
/home/kocmonavtik/new2:
/home/kocmonavtik/new2/new2:
kocmonavtik@kocmos:~$ grep kocmonavtik /etc/passwd
kocmonavtik:x:1000:1000:Ivan:/home/kocmonavtik:/bin/bash
kocmonavtik@kocmos:~$

```

Рисунок 6 – Использование команды “grep”

Команда “find” предназначена для поиска файлов и каталогов на основе специальных условий. Её можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и т.п.

```
kocmonavtik@kocmos:~$ find -user kocmonavtik
./output1.txt
./.bash_history
./.bashrc
./output2.txt
./config
./config/procps
./profile
./loop2
./new.txt
./viminfo
./new
./new/test2.txt
./new/new2
./new/test1.txt
./output.txt
./new2
./new2/test2.txt
./new2/new2
./new2/test1.txt
./sudo_as_admin_successful
./loop
./cache
./cache/motd.legal-displayed
./bash_logout
./1.txt
./new.tar
./info.txt
./local
./local/share
./local/share/nano
kocmonavtik@kocmos:~$
```

Рисунок 7 – Использование команды “grep”

В данном примере, использовали поиск файлов, владельцем которых является пользователь “kocmonavtik”.

Для просмотра бинарных файлов в виде дампа битов “od”. Имеется возможность указывания системы счисления, в которой будут указываться адреса, данные.

```
kosmonavtik@kocmos:~$ ls
1.txt info.txt loop loop2 new new.tar new.txt new2 output.txt output1.txt output2.txt
kosmonavtik@kocmos:~$ od -t x loop
00000000 6c696877 72742065 203b6575 74206f64
00000020 3b657572 6e6f6420 00000a65
00000032
kosmonavtik@kocmos:~$
```

Рисунок 8 – Использование команды “od”

В данном случае, применялся просмотр данных в шестнадцатеричной системе счисления.

### 3.2 Перенаправление ввода – вывода

В работе с командной строкой Linux есть понятия стандартных устройств ввода, вывода и вывода ошибок.

Stdin – стандартное устройства ввода. Имеет файловый указатель №0. Автоматически открывается всеми процессами.

Stdout – стандартное устройство вывода. Имеет файловый указатель №1. Автоматически открывается всеми процессами.

Stderr – стандартный поток ошибок. Имеет файловый указатель №2. Автоматически открывается всеми процессами.

По умолчанию, практически все команды Linux используют для ввода, вывода перечисленные выше соответственно, если их параметрами не указано обратное.

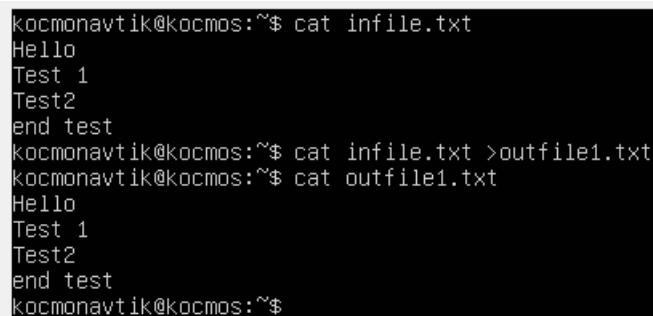
Существуют операторы перенаправления, способные изменять направление ввода и вывода информации, они имеют вид:

“>” – перенаправляет стандартный поток в файл. Если файл существует, то он перезаписывается, в случае отсутствия файла, происходит его создание.

“>>” – перенаправляет стандартный поток в файл. При этом если файл существует, то информация добавляется в конец, если файла не существует, он создается.

“<” – перенаправляет содержимое указанного файла на стандартный ввод программы.

“>&” – перенаправляет стандартные потоки вывода и ошибок друг в друга



```
kosmonavtik@kocmos:~$ cat infile.txt
Hello
Test 1
Test2
end test
kosmonavtik@kocmos:~$ cat infile.txt >outfile1.txt
kosmonavtik@kocmos:~$ cat outfile1.txt
Hello
Test 1
Test2
end test
kosmonavtik@kocmos:~$
```

Рисунок 9 – Пример перенаправления ввода в файл

```
kocmonavtik@kocmos:~$ cat outfile1.txt
Hello
Test 1
Test2
end test
kocmonavtik@kocmos:~$ cat infile.txt >>outfile1.txt
kocmonavtik@kocmos:~$ cat outfile1.txt
Hello
Test 1
Test2
end test
Hello
Test 1
Test2
end test
kocmonavtik@kocmos:~$ _
```

Рисунок 10 – Пример перенаправления ввода в файл

```
kocmonavtik@kocmos:~$ cat nofile.txt >output1.txt 2>&1
kocmonavtik@kocmos:~$ cat output1.txt
cat: nofile.txt: No such file or directory
kocmonavtik@kocmos:~$ _
```

Рисунок 11 – Пример перенаправления вывода ошибок в файл

Канал – программный интерфейс, позволяющий процессами обмениваться данными (односторонний поток). Организацией канала занимается shell. Для управления каналом существует оператор “|”, например команда: “ls -R /home |less”, которая выводит список файлов в текущем каталоге и команда “less”, которая позволяет постранично просматривать вывод команды “ls -R”.

Для каждого объекта в файловой системе Linux существует набор прав доступа, определяющий взаимодействие пользователя с этим объектом. Такими могут быть файлы, каталоги, а также специальные файлы. Так как у каждого объекта в Linux имеется владелец, то права доступа применяются относительно владельца файла. Они состоят из набора 3 групп по три атрибута:

Чтение (r), запись (w), выполнение(x) для владельца;

Чтение, запись, выполнение для группы владельца;

Чтение, запись, выполнение для всех остальных;

Команда “chmod” служит для изменения прав доступа к файлам. Она имеет различные опции:

- 1) Выбор категории пользователей: u – владелец файла g-группа файла o – все остальные пользователи
- 2) Выбор права доступа, которые собираемся дать определенной категории пользователей или отобрать
- 3) Для выбора назначения права доступа или изъятия, используем знаки “+” или “-”

```
kosmonavtik@kosmos:~$ ls -l
total 64
-rw-rw-r-- 1 kosmonavtik kosmonavtik 110 Oct 24 15:13 1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 28 Nov 7 16:04 infile.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 119 Oct 24 15:20 info.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 26 Oct 22 17:24 loop
-rw-rw-r-- 1 kosmonavtik kosmonavtik 48 Oct 23 15:54 loop2
drwxrwxr-x 3 kosmonavtik kosmonavtik 4096 Oct 24 18:29 new
-rw-rw-r-- 1 kosmonavtik kosmonavtik 10240 Oct 27 12:47 new.tar
-rw-rw-r-- 1 kosmonavtik kosmonavtik 3866 Nov 7 13:12 new.txt
drwxrwxr-x 3 kosmonavtik kosmonavtik 4096 Oct 25 17:40 new2
-rw-rw-r-- 1 kosmonavtik kosmonavtik 0 Nov 7 16:37 outfile.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 56 Nov 7 16:14 outfile1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 948 Oct 27 07:38 output.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 43 Nov 7 16:44 output1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 72 Oct 31 05:34 output2.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 43 Nov 7 16:43 result.out
kosmonavtik@kosmos:~$ chmod o+w result.out
kosmonavtik@kosmos:~$ ls -l
total 64
-rw-rw-r-- 1 kosmonavtik kosmonavtik 110 Oct 24 15:13 1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 28 Nov 7 16:04 infile.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 119 Oct 24 15:20 info.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 26 Oct 22 17:24 loop
-rw-rw-r-- 1 kosmonavtik kosmonavtik 48 Oct 23 15:54 loop2
drwxrwxr-x 3 kosmonavtik kosmonavtik 4096 Oct 24 18:29 new
-rw-rw-r-- 1 kosmonavtik kosmonavtik 10240 Oct 27 12:47 new.tar
-rw-rw-r-- 1 kosmonavtik kosmonavtik 3866 Nov 7 13:12 new.txt
drwxrwxr-x 3 kosmonavtik kosmonavtik 4096 Oct 25 17:40 new2
-rw-rw-r-- 1 kosmonavtik kosmonavtik 0 Nov 7 16:37 outfile.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 56 Nov 7 16:14 outfile1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 948 Oct 27 07:38 output.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 43 Nov 7 16:44 output1.txt
-rw-rw-r-- 1 kosmonavtik kosmonavtik 72 Oct 31 05:34 output2.txt
-rw-rw-rw- 1 kosmonavtik kosmonavtik 43 Nov 7 16:43 result.out
kosmonavtik@kosmos:~$
```

Рисунок 12 – Пример работы команды “chmod”

Как можно увидеть, данной командой присвоили остальным пользователям права доступа на запись в файл “result.out”.

Команда “chown” предназначена для изменения владельца файла и группы, к которой относился владелец.

```
root@kocmos:/home/kocmonavtik# ls -l
total 64
-rw-rw-r-- 1 kocmonavtik kocmonavtik 110 Oct 24 15:13 1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 28 Nov 7 16:04 infile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 119 Oct 24 15:20 info.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 26 Oct 22 17:24 loop
-rw-rw-r-- 1 kocmonavtik kocmonavtik 48 Oct 23 15:54 loop2
drwxrwxr-x 3 kocmonavtik kocmonavtik 4096 Oct 24 18:29 new
-rw-rw-r-- 1 kocmonavtik kocmonavtik 10240 Oct 27 12:47 new.tar
-rw-rw-r-- 1 kocmonavtik kocmonavtik 3866 Nov 7 13:12 new.txt
drwxrwxr-x 3 kocmonavtik kocmonavtik 4096 Oct 25 17:40 new2
-rw-rw-r-- 1 kocmonavtik kocmonavtik 0 Nov 7 16:37 outfile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 56 Nov 7 16:14 outfile1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 948 Oct 27 07:38 output.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 43 Nov 7 16:44 output1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 72 Oct 31 05:34 output2.txt
-rw-rw-rw- 1 kocmonavtik kocmonavtik 43 Nov 7 16:43 result.out
root@kocmos:/home/kocmonavtik# chown root ./new2
root@kocmos:/home/kocmonavtik# ls -l
total 64
-rw-rw-r-- 1 kocmonavtik kocmonavtik 110 Oct 24 15:13 1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 28 Nov 7 16:04 infile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 119 Oct 24 15:20 info.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 26 Oct 22 17:24 loop
-rw-rw-r-- 1 kocmonavtik kocmonavtik 48 Oct 23 15:54 loop2
drwxrwxr-x 3 kocmonavtik kocmonavtik 4096 Oct 24 18:29 new
-rw-rw-r-- 1 kocmonavtik kocmonavtik 10240 Oct 27 12:47 new.tar
-rw-rw-r-- 1 kocmonavtik kocmonavtik 3866 Nov 7 13:12 new.txt
drwxrwxr-x 3 root kocmonavtik 4096 Oct 25 17:40 new2
-rw-rw-r-- 1 kocmonavtik kocmonavtik 0 Nov 7 16:37 outfile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 56 Nov 7 16:14 outfile1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 948 Oct 27 07:38 output.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 43 Nov 7 16:44 output1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 72 Oct 31 05:34 output2.txt
-rw-rw-rw- 1 kocmonavtik kocmonavtik 43 Nov 7 16:43 result.out
root@kocmos:/home/kocmonavtik#
```

Рисунок 13 – Изменение владельца файла

Для изменения не только владельца, но и группы пользователей, необходимо ввести команду “chown root:root ./new2”.

```
root@kocmos:/home/kocmonavtik# chown root:root ./new2
root@kocmos:/home/kocmonavtik# ls -l
total 64
-rw-rw-r-- 1 kocmonavtik kocmonavtik 110 Oct 24 15:13 1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 28 Nov 7 16:04 infile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 119 Oct 24 15:20 info.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 26 Oct 22 17:24 loop
-rw-rw-r-- 1 kocmonavtik kocmonavtik 48 Oct 23 15:54 loop2
drwxrwxr-x 3 kocmonavtik kocmonavtik 4096 Oct 24 18:29 new
-rw-rw-r-- 1 kocmonavtik kocmonavtik 10240 Oct 27 12:47 new.tar
-rw-rw-r-- 1 kocmonavtik kocmonavtik 3866 Nov 7 13:12 new.txt
drwxrwxr-x 3 root root 4096 Oct 25 17:40 new2
-rw-rw-r-- 1 kocmonavtik kocmonavtik 0 Nov 7 16:37 outfile.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 56 Nov 7 16:14 outfile1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 948 Oct 27 07:38 output.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 43 Nov 7 16:44 output1.txt
-rw-rw-r-- 1 kocmonavtik kocmonavtik 72 Oct 31 05:34 output2.txt
-rw-rw-rw- 1 kocmonavtik kocmonavtik 43 Nov 7 16:43 result.out
root@kocmos:/home/kocmonavtik# _
```

Рисунок 14 – Изменение группы пользователей



### 3.3 Процессы

Для просмотра информации о состоянии процессов системы в реальном времени, используется команда “top”.

```
top - 15:54:07 up 42 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 96 total, 1 running, 95 sleeping, 0 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 : 7962.4 total, 7444.5 free, 139.8 used, 378.1 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7584.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1010	root	20	0	0	0	0	I	0.3	0.0	0:00.27	kworker/u2:2-events_power_e+
1	root	20	0	101996	11360	8280	S	0.0	0.1	0:03.90	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	0:00.15	ksoftirqd/0
11	root	20	0	0	0	0	I	0.0	0.0	0:01.14	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

Рисунок 15 – Использование команды “top”

Для выхода из данной утилиты производится нажатием кнопки “q”.

Для выбора столбца для сортировки и добавления, удаления различных характеристик для отображения, используется комбинация клавиш “Shift+F”

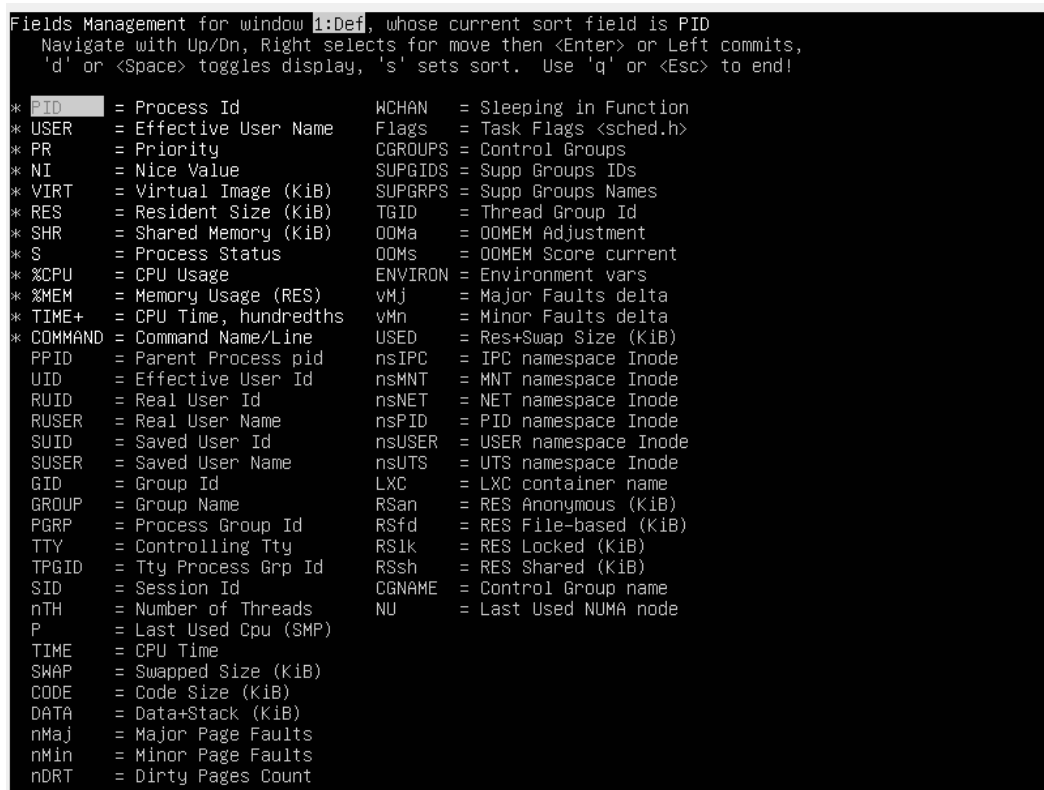


Рисунок 16 – Изменение отображения характеристик и сортировка

Для выбора поля, по которому будет происходить сортировка, необходимо нажать на клавишу “s”. Для добавления/удаления информации в утилите, необходимо нажать клавишу “d” после выбора нужной характеристики.

Для выхода из меню редактирования и сортировки, необходимо нажать клавишу “q”.

Для отображения процессов конкретного пользователя, используется клавиша “u”. После чего необходимо ввести имя пользователя, процессы которого будут отображены.

```
top - 16:14:44 up 1:02, 1 user, load average: 0.00, 0.02, 0.06
Tasks: 93 total, 1 running, 92 sleeping, 0 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 : 7962.4 total, 7192.4 free, 144.5 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.8 avail Mem
Which user (blank for all) kocmonavtik
  PID USER      PR  NI   VIRT   RES    SHR S  %CPU  %MEM    TIME+ COMMAND
    1 root        0   0    1992    196    196 S   0.0   0.0   0:04.89 systemd
```

Рисунок 17 – Пример выполнения

После ввода, необходимо нажать “Enter” для подтверждения или “Esc” для выхода из данной функции утилиты.

```
top - 16:15:41 up 1:03, 1 user, load average: 0.00, 0.01, 0.05
Tasks: 93 total, 2 running, 91 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7192.4 free, 144.5 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.8 avail Mem

  PID USER      PR  NI   VIRT   RES    SHR S  %CPU  %MEM    TIME+ COMMAND
 9486 kocmona+ 20   0   7916   3760   3232 R   0.0   0.0   0:00.01 top
   916 kocmona+ 20   0  18556   9936   8264 S   0.0   0.1   0:00.24 systemd
   918 kocmona+ 20   0 103208   3464    20 S   0.0   0.0   0:00.00 (sd-pam)
   926 kocmona+ 20   0   7068   5084   3376 S   0.0   0.1   0:00.25 bash
```

Рисунок 18 – Отображение процессов определенного пользователя

Для отображения всех процессов, необходимо нажать клавишу “u” и нажать “Enter”, оставляя поле для ввода пользователя пустым.

Для включения цветного вывода, используется кнопка “Z”.

```
top - 16:29:12 up 1:17, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 93 total, 1 running, 92 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 4.8 sy, 0.0 ni, 95.2 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7191.9 free, 145.0 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1068	root	20	0	0	0	0	I	4.5	0.0	0:02.89	kworker/0:1-events
1	root	20	0	103920	12696	8280	S	0.0	0.2	0:04.91	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	0:00.41	ksoftirqd/0
11	root	20	0	0	0	0	I	0.0	0.0	0:01.82	rcu_sched
12	root	rt	0	0	0	0	S	0.0	0.0	0:00.05	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

Рисунок 19 – Применение цветного вывода

У каждого окна есть своя цветовая схема, чтобы настроить под себя их цвет, необходимо нажать клавишу “Z”.

```
Help for color mapping - "Current Window" = 1:Def
color - 04:25:44 up 8 days, 50 min, 7 users, load average:
Tasks: 64 total, 2 running, 62 sleeping, 0 stopped,
%Cpu(s): 76.5 user, 11.2 system, 0.0 nice, 12.3 idle
Nasty Message! -or- Input Prompt
```

PID	TTY	PR	NI	%CPU	TIME+	VIRT	SWAP	S	COMMAND
17284	pts/2	8	0	0.0	0:00.75	1380	0	S	/bin/bash
8601	pts/1	7	-10	0.4	0:00.03	916	0	R	color -b -z
11005	?	9	0	0.0	0:02.50	2852	1008	S	amor -sessi

```
available toggles: B =disable bold globally (Off),
z =color/mono (On), b =tasks "bold"/reverse (On)

1) Select a target as an upper case letter, current target is T:
S = Summary Data, M = Messages/Prompts,
H = Column Heads, T = Task Information
2) Select a color as a number or use the up/down arrow keys
to raise/lower the 8 colors value, current color is 1:
0 = black, 1 = red, 2 = green, 3 = yellow,
4 = blue, 5 = magenta, 6 = cyan, 7 = white
3) Then use these keys when finished:
'q' or <Esc> to abort changes to window '1:Def'
'a' or 'w' to commit & change another, <Enter> to commit and end
```

Рисунок 20 – Изменение цвета окон

Для применения всех изменений, необходимо нажать “Enter”, в противном случае “Esc”.

```
top - 16:36:48 up 1:24, 1 user, load average: 0.00, 0.01, 0.00
Tasks: 93 total, 1 running, 92 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.2 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7191.9 free, 145.0 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1068	root	20	0	0	0	0	I	0.2	0.0	0:03.83	kworker/0:1-events
9486	kocmona+	20	0	7916	3760	3232	R	0.2	0.0	0:07.11	top
11	root	20	0	0	0	0	I	0.0	0.0	0:01.95	rcu_sched
191	root	-51	0	0	0	0	S	0.0	0.0	0:00.83	irq/18-vmwgfx
508	root	rt	0	280288	18096	8184	S	0.0	0.2	0:00.68	multipathd
9493	root	20	0	0	0	0	I	0.0	0.0	0:00.11	kworker/u2:1-events_power_e+
9494	root	20	0	0	0	0	I	0.0	0.0	0:00.10	kworker/u2:0-events_unbound
1	root	20	0	103320	12696	8280	S	0.0	0.2	0:04.92	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	0:00.42	ksoftirqd/0
12	root	rt	0	0	0	0	S	0.0	0.0	0:00.06	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	bicg_punt_bio

Рисунок 21 – Измененная цветовая схема окон

Для отображения абсолютного пути запущенных процессов, необходимо нажать клавишу “с”.

```
top - 16:38:16 up 1:26, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 92 total, 1 running, 91 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.3 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7191.6 free, 145.3 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
525	root	20	0	0	0	0	S	0.0	0.0	0:00.00	[jbd2/sda2-8]
526	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	[ext4-rsv-conver]
527	root	0	-20	0	0	0	S	0.0	0.0	0:00.01	[loop3]
528	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	[loop4]
529	root	0	-20	0	0	0	S	0.0	0.0	0:00.08	[loop5]
542	systemd+	20	0	90388	6404	5536	S	0.0	0.1	0:00.20	/lib/systemd/systemd-timesy+
588	systemd+	20	0	26740	7844	6884	S	0.0	0.1	0:00.14	/lib/systemd/systemd-networ+
593	systemd+	20	0	24044	12116	8112	S	0.0	0.1	0:00.22	/lib/systemd/systemd-resolv+
618	root	20	0	5568	2892	2672	S	0.0	0.0	0:00.01	/usr/sbin/cron -f
619	message+	20	0	7700	4788	4008	S	0.0	0.1	0:00.47	/usr/bin/dbus-daemon --syst+
627	root	20	0	26284	17848	10200	S	0.0	0.2	0:00.18	/usr/bin/python3 /usr/bin/n+
631	syslog	20	0	224324	4880	3844	S	0.0	0.1	0:00.03	/usr/sbin/rsyslogd -n -iNONE
633	root	20	0	636144	28152	15768	S	0.0	0.3	0:02.72	/usr/lib/snapd/snapd
641	root	20	0	16808	7868	6896	S	0.0	0.1	0:00.26	/lib/systemd/systemd-logind
644	daemon	20	0	3792	2436	2260	S	0.0	0.0	0:00.00	/usr/sbin/atd -f
656	root	20	0	5972	3996	3228	S	0.0	0.0	0:00.04	/bin/login -p --
670	root	20	0	105100	20936	13288	S	0.0	0.3	0:00.17	/usr/bin/python3 /usr/share+
671	root	20	0	12160	6956	6116	S	0.0	0.1	0:00.01	sshd: /usr/sbin/sshd -D [li+
701	root	20	0	236292	8932	8016	S	0.0	0.1	0:00.04	/usr/lib/policykit-1/polkit+
916	kocmona+	20	0	18556	9936	8264	S	0.0	0.1	0:00.24	/lib/systemd/systemd --user
918	kocmona+	20	0	103208	3464	20	S	0.0	0.0	0:00.00	(sd-pam)
926	kocmona+	20	0	7068	5084	3376	S	0.0	0.1	0:00.25	-bash
1068	root	20	0	0	0	0	I	0.0	0.0	0:03.94	[kworker/0:1-events]
2278	root	20	0	238060	9324	8348	S	0.0	0.1	0:00.11	/usr/lib/accountsservice/ac+
9478	root	20	0	0	0	0	I	0.0	0.0	0:00.00	[kworker/0:2]
9493	root	20	0	0	0	0	I	0.0	0.0	0:00.11	[kworker/u2:1-events_power_+
9494	root	20	0	0	0	0	I	0.0	0.0	0:00.12	[kworker/u2:0-events_power_+

Рисунок 22 – Отображение абсолютного пути

Для изменения обновления информации о процессах, необходимо нажать клавишу “d” и ввести новое значение для изменения интервала обновления.

```
top - 16:42:12 up 1:30, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 93 total, 1 running, 92 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.3 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7191.6 free, 145.3 used, 625.5 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.1 avail Mem
Change delay from 3.0 to 2
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
  525 root        20   0       0        0        0 S   0.0   0.0   0:00.00 [kswapd0]
```

Рисунок 23 – Изменения интервала обновления

Как видно по рисунку, интервал обновления информации будет изменен после нажатия клавиши “Enter”.

Для того чтобы убить процесс по его PID, необходимо нажать клавишу “k”.

```
top - 16:52:11 up 1:40, 1 user, load average: 0.06, 0.03, 0.00
Tasks: 94 total, 1 running, 93 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.5 us, 1.0 sy, 0.0 ni, 94.0 id, 4.5 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7192.1 free, 144.8 used, 625.6 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.6 avail Mem
PID to signal/kill [default pid = 916] 9499
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
  916 koscmona+  20   0   18556    9936   8264 S   0.0   0.1   0:00.24 /lib/systemd/systemd --user
  918 koscmona+  20   0  103208    3464     20 S   0.0   0.0   0:00.00 (sd-pam)
  926 koscmona+  20   0    7068    5084   3376 S   0.0   0.1   0:00.26 -bash
  9499 koscmona+  20   0    2608    1636    1540 S   0.0   0.0   0:00.00 sh loop2
  9505 koscmona+  20   0    7916    3752    3220 R   0.5   0.0   0:00.87 top
  9511 koscmona+  20   0     4260     528     464 S   0.0   0.0   0:00.00 sleep 5
   644 daemon    20   0    3792    2436    2260 S   0.0   0.0   0:00.00 /usr/sbin/atd -f
```

Рисунок 24 – Завершение процесса сигналом “kill”

После ввода, необходимо нажать клавишу “Enter”.

Для сортировки процессов по загрузке ЦП, необходимо нажать комбинацию клавиш «Shift+P»

```
top - 17:17:52 up 2:05, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 93 total, 1 running, 92 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.5 us, 0.0 sy, 0.0 ni, 99.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7191.9 free, 145.0 used, 625.6 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
9505	kocmon+	20	0	7916	3752	3220	R	0.5	0.0	0:08.29	top
1	root	20	0	103320	12696	8280	S	0.0	0.2	0:04.94	/sbin/init maybe-ubiquity
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	0:00.61	[ksoftirqd/0]
11	root	20	0	0	0	0	I	0.0	0.0	0:02.18	[rcu_sched]
12	root	rt	0	0	0	0	S	0.0	0.0	0:00.08	[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]

Рисунок 25 – Сортировка по загрузке ЦП

Для изменения приоритета процесса, необходимо нажать клавишу “r”, после чего ввести PID процесса, которому необходимо изменить приоритет.

```
top - 17:32:42 up 2:20, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 94 total, 1 running, 93 sleeping, 0 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 : 7962.4 total, 7188.6 free, 144.9 used, 628.9 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.1 avail Mem
PID to renice [default pid = 641] 9694
```

Рисунок 26 – Ввод PID процесса для изменения приоритета

```
top - 17:32:42 up 2:20, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 94 total, 1 running, 93 sleeping, 0 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 : 7962.4 total, 7188.6 free, 144.9 used, 628.9 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.1 avail Mem
Renice PID 9694 to value 10
```

Рисунок 27 – Изменение приоритета

Задать приоритет возможно от -20 до 19, где -20 имеет самый высокий приоритет.

Для сохранения вывода результатов команды “top”, необходимо написать команду “top -n кол-во\_итераций -b >имя\_файла”.

```
kosmonavtik@kosmos:~$ top -n 5 -b >top-output.txt
kosmonavtik@kosmos:~$
```

Рисунок 28 – Ввод команды сохранения отчета в файл

Как видно, в файл “top-output.txt” помещается отчет о процессах с 5 обновлениями информации.

Напишем команду “nano top-output.txt” для просмотра содержимого.

```
GNU nano 4.8 top-output.txt
top - 17:43:09 up 2:31, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 94 total, 1 running, 93 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.9 us, 5.9 sy, 0.0 ni, 88.2 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7962.4 total, 7188.8 free, 144.5 used, 629.2 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used, 7569.5 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR S  %CPU  %MEM    TIME+  COMMAND
 542 systemd+ 20   0   90388   6404   5536 S   0.0   0.1   0:00.22 /lib/systemd/systemd-timesy+
 593 systemd+ 20   0  24044  12116  8112 S   0.0   0.1   0:00.23 /lib/systemd/systemd-resolv+
 588 systemd+ 20   0  26740   7844  6884 S   0.0   0.1   0:00.15 /lib/systemd/systemd-networ+
 631 syslog   20   0  224324  4880   3844 S   0.0   0.1   0:00.04 /usr/sbin/rsyslogd -n -iNONE
    1 root      20   0  103320  12700  8280 S   0.0   0.2   0:05.03 /sbin/init maybe-ubiquity
    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   6.7   0.0   0:00.79 [ksoftirqd/0]
   11 root      20   0      0      0      0 I   0.0   0.0   0:02.50 [rcu_sched]
   12 root      rt    0      0      0      0 S   0.0   0.0   0:00.10 [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]

[ Soft wrapping of overlong lines enabled ]
^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste Text ^T To Spell  ^_ Go To Line M-E Redo
```

Рисунок 29 – Просмотр содержимого файла



Для получения справки по основным командам утилиты “top”, необходимо нажать клавишу “h”.

```
Help for Interactive Commands - procs-ng UNKNOWN
Window 1:Def: Cumulative mode Off. System: Delay 2.0 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 30 – Вызов справки

Так же имеется возможно вызвать утилиту “top” на определенное количество обновлений информации командой “top -n кол-во\_итераций”. После определенного количества обновлений информации о процессах, утилита автоматически остановится.

### 3.4 Supervisor

Supervisor – это система клиент/сервер, при помощи которой пользователь (администратор) может контролировать подключенные процессы в системах типа UNIX. Инструмент создает процессы в виде под-процессов от своего имени, поэтому имеет полный контроль над ними.

Перед началом работы, необходимо установить supervisor. Для этого напишем в консоли данную команду: “apt-get install supervisor”.

```
root@kocmos:/home/kocmonavtik# apt-get install supervisor
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  supervisor-doc
The following NEW packages will be installed:
  supervisor
0 upgraded, 1 newly installed, 0 to remove and 77 not upgraded.
Need to get 281 kB of archives.
After this operation, 1682 kB of additional disk space will be used.
Get:1 http://ru.archive.ubuntu.com/ubuntu focal/universe amd64 supervisor all 4.1.0-1ubuntu1 [281 kB]
Fetched 281 kB in 1s (484 kB/s)
Selecting previously unselected package supervisor.
(Reading database ... 107348 files and directories currently installed.)
Preparing to unpack .../supervisor_4.1.0-1ubuntu1_all.deb ...
Unpacking supervisor (4.1.0-1ubuntu1) ...
Setting up supervisor (4.1.0-1ubuntu1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/supervisor.service → /lib/systemd/system/supervisor.service.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.2) ...
root@kocmos:/home/kocmonavtik# _
```

Рисунок 31 – Установка supervisor

После установки, нужно сконфигурировать и добавить программы/процессы, которыми будет управлять supervisor. В файл конфигурации “supervisord.conf”, который используется по умолчанию, возможно дополнить новым процессом. Так же возможно создать файл запуска процесса в каталоге “/etc/supervisor/conf.d”

Приведём пример:

Создадим файл запуска процесса “supervisorScript”

В данном файле написан скрипт: “while true; do true; echo ‘Hello’; sleep 5; done”

```
root@kocmos:/home/kocmonavtik# ls
1.txt      loop2      new2       output1.txt  test.err.log  testCron3.txt
infile.txt new        outfile.txt output2.txt  test.out.log  top-output.txt
info.txt   new.tar    outfile1.txt result.out   testCron      visor.txt
loop       new.txt    output.txt  supervisorScript testCron2.txt
root@kocmos:/home/kocmonavtik# cd /etc/supervisor/conf.d
root@kocmos:/etc/supervisor/conf.d# cat /home/kocmonavtik/supervisorScript
while true; do true; echo 'Hello';sleep 5; done
root@kocmos:/etc/supervisor/conf.d# nano script1.conf
```

Рисунок 32 – Создание файла запуска

Напишем в файле:

**[program:test]** – название воркера;

**command= /bin/sh /home/kocmonavtik/supervisorScript >>**

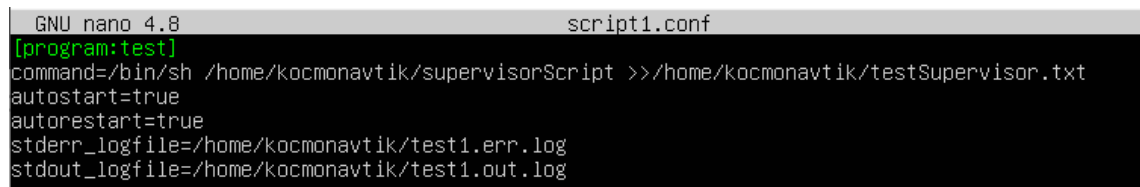
**/home/kocmonavtik/testSupervisor.txt** – команда на запуск скрипта;

**autostart=true** – запуск процесса при старте системы

**autorestart = true** – запуск процесса при его остановке.

**stderr\_logfile=/home/kocmonavtik/test1.err.log** – файл с выводом лога ошибки

**stdout\_logfile=/home/kocmonavtik/test1.out.log** – файл с выводом лога процесса при работе

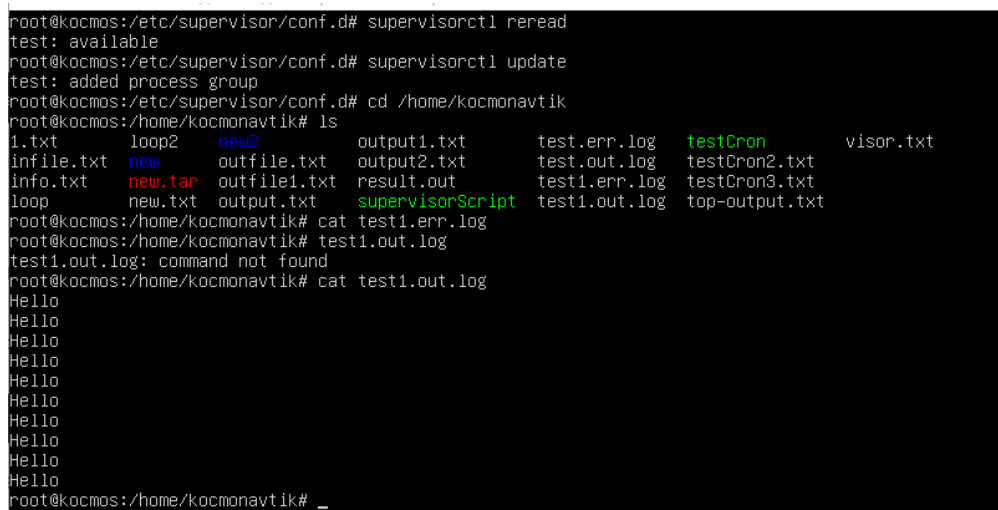


```
GNU nano 4.8 script1.conf
[program:test]
command=/bin/sh /home/kocmonavtik/supervisorScript >>/home/kocmonavtik/testSupervisor.txt
autostart=true
autorestart=true
stderr_logfile=/home/kocmonavtik/test1.err.log
stdout_logfile=/home/kocmonavtik/test1.out.log
```

Рисунок 33 – Ввод и настройка нового процесса

После ввода, сохраняем и выходим из текстового редактора.

Для того, чтобы supervisor считал обновленные настройки, напомним “supervisorctl reread”. После чего, необходимо дать команду запуска всех сконфигурированных процессов, для этого вводим “supervisorctl update”.



```
root@kocmos:/etc/supervisor/conf.d# supervisorctl reread
test: available
root@kocmos:/etc/supervisor/conf.d# supervisorctl update
test: added process group
root@kocmos:/etc/supervisor/conf.d# cd /home/kocmonavtik
root@kocmos:/home/kocmonavtik# ls
1.txt      loop2      new2       output1.txt  test.err.log  testCron      visor.txt
infile.txt new        outfile.txt output2.txt  test.out.log  testCron2.txt
info.txt   new.tar    outfile1.txt result.out   test1.err.log  testCron3.txt
loop       new.txt    output.txt  supervisorScript  test1.out.log  top-output.txt
root@kocmos:/home/kocmonavtik# cat test1.err.log
root@kocmos:/home/kocmonavtik# test1.out.log
test1.out.log: command not found
root@kocmos:/home/kocmonavtik# cat test1.out.log
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
root@kocmos:/home/kocmonavtik# _
```

Рисунок 33 – Пример работоспособности скрипта

Как можно увидеть, в логах ошибок ничего нету т.к. ошибок в работе процесса не обнаружено. А в логах вывода показано что сделал процесс.

Для проверки статусов процессов, которые запущены через supervisor, необходимо написать команду “supervisorctl”.

```
root@kocmos:/home/kocmonavtik# supervisorctl
test                                RUNNING    pid 1243, uptime 0:06:02
supervisor> _
```

Рисунок 34 – Вызов менеджера процесса supervisor

В данном менеджере существуют команды: start, stop, restart – это основные команды.

```
root@kocmos:/home/kocmonavtik# supervisorctl
test                                RUNNING    pid 1243, uptime 0:06:02
supervisor> stop test
test: stopped
supervisor> _
```

Рисунок 35 – Остановка процесса “test”

### 3.5 Планировщик задач

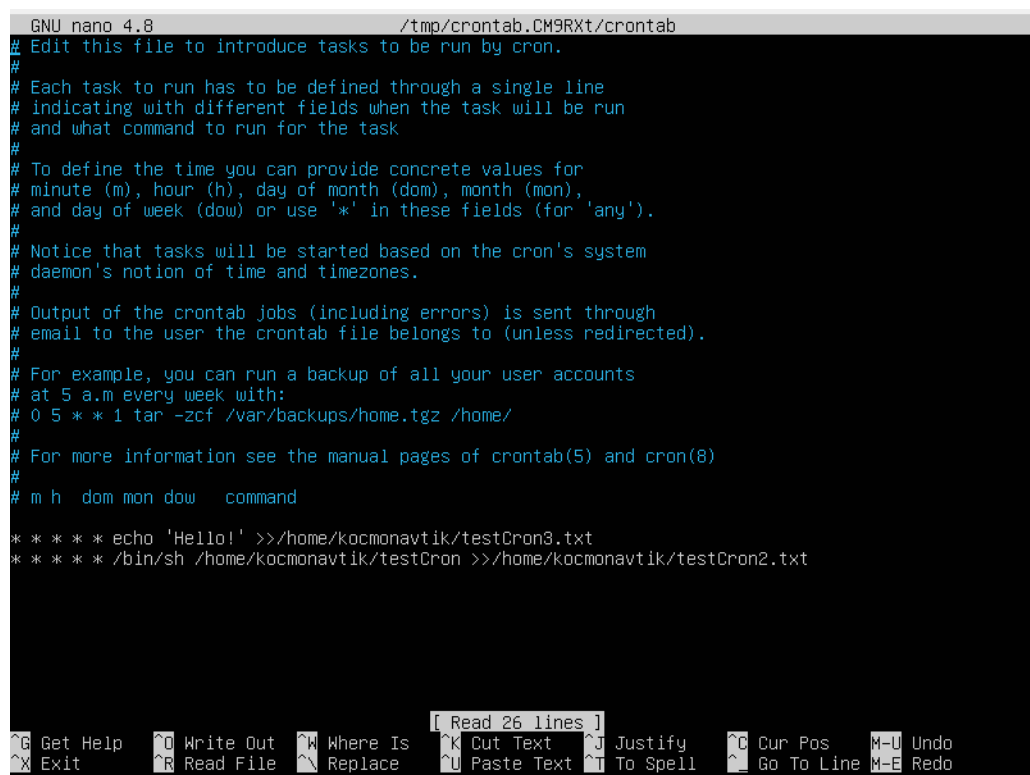
Для планирования задач и их запуска в определенное время или в промежутки времени, существует программа – демон “cron”.

Для редактирования файла расписания, необходимо написать команду “crontab -e”.

Синтаксис настройки одной задачи выглядит следующим образом:

Минута час день месяц день\_недели путь\_к исполняемому\_файлу

Например “\* \* \* \* \*” означает что запуск будет производится каждую минуту, а “0 0 1 \* \*” означает, что запуск будет производится в первый день каждого месяца



```
GNU nano 4.8 /tmp/crontab.CM9RXt/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * echo 'Hello!' >>/home/kocmonavtik/testCron3.txt
* * * * * /bin/sh /home/kocmonavtik/testCron >>/home/kocmonavtik/testCron2.txt
```

Рисунок 36 – Запись процессов для автозапуска

Как можно увидеть по рисунку, добавили процесс, который вводит слово “Hello” в текстовый файл и так же добавили процесс, который выполняет скрипт и отправляет вывод в файл.

Для проверки существующих процессов, которые находятся в списке автозапуска, необходимо ввести команду “crontab -l”. Для удаления всех существующих задач, необходимо ввести команду “crontab -r”.

#### 4. Вывод

Выполнив лабораторную работу, закрепили изученный материал. Ознакомились с контейнеризацией и наиболее распространенными командами Linux. Изучили возможность запуска программ по расписанию.