

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**

дисциплина: Основы администрирования операционных систем

Студент: Ко Антон Геннадьевич

Студ. билет № 1132221551

Группа: НПИбд-02-23

**МОСКВА**

2024 г.

## Цель работы:

Целью данной работы является получение навыков управления процессами операционной системы.

## Выполнение работы:

### Управление заданиями:

Для начала получим полномочия администратора **su** – и введём следующие команды:

```
sleep 3600 &
```

```
dd if=/dev/zero of=/dev/null &
```

```
sleep 7200
```

Поскольку мы запустили последнюю команду без **&** после неё, у нас есть 2 часа, прежде чем мы снова получим контроль над оболочкой.

Введём **Ctrl + z** , чтобы остановить процесс. Затем введём **jobs** и увидим три задания, которые мы только что запустили. Первые два имеют состояние **Running**, а последнее задание в настоящее время находится в состоянии **Stopped**. Для продолжения выполнения задания 3 в фоновом режиме введём **bg 3** и с помощью команды **jobs** посмотрим изменения в статусе заданий:

```

[root@agko SenDerMen]# sleep 3600 &
[1] 3263
[root@agko SenDerMen]# dd if=/dev/zero of=/dev/null &
[2] 3266
[root@agko SenDerMen]# sleep 7200
^Z
[3]+  Остановлен      sleep 7200
[root@agko SenDerMen]# jobs
[1]  Запущен          sleep 3600 &
[2]-  Запущен          dd if=/dev/zero of=/dev/null &
[3]+  Остановлен      sleep 7200
[root@agko SenDerMen]# bg 3
[3]+ sleep 7200 &
[root@agko SenDerMen]# █

```

**Рис. 1.** Получение полномочий администратора, ввод трёх команд, остановка процесса, установка выполнения задания 3 в фоновом режиме, просмотр изменений в статусе заданий.

Для перемещения задания 1 на передний план введём **fg 1**, далее введём **Ctrl + c**, чтобы отменить задание 1. С помощью команды **jobs** посмотрим изменения в статусе заданий и сделаем то же самое для отмены заданий 2 и 3:

```

^Z
[3]+ Остановлен    sleep 7200
[root@agko SenDerMen]# jobs
[1]  Запущен        sleep 3600 &
[2]- Запущен        dd if=/dev/zero of=/dev/null &
[3]+ Остановлен    sleep 7200
[root@agko SenDerMen]# bg 3
[3]+ sleep 7200 &
[root@agko SenDerMen]# jobs
[1]  Запущен        sleep 3600 &
[2]- Запущен        dd if=/dev/zero of=/dev/null &
[3]+ Запущен        sleep 7200 &
[root@agko SenDerMen]# fg 1
sleep 3600
^[[A^Z
[1]+ Остановлен    sleep 3600
[root@agko SenDerMen]# fg 1
sleep 3600
^C
[root@agko SenDerMen]# jobs
[2]- Запущен        dd if=/dev/zero of=/dev/null &
[3]+ Запущен        sleep 7200 &
[root@agko SenDerMen]#

```

**Рис. 2.** Перемещение заданий на передний план и их последующая отмена.

Теперь откроем второй терминал и под учётной записью пользователя введём в нём: **dd if=/dev/zero of=/dev/null &**. После введём **exit**, чтобы закрыть второй терминал.

На другом терминале под учётной записью своего пользователя запустим **top**. Мы увидим, что задание dd всё ещё запущено. Для выхода из top используем **q** и вновь запускаем top, в нём используем **k**, чтобы убить задание dd. После этого выйдем из top:

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
<b>3304</b>	<b>SenDerM+</b>	<b>20</b>	<b>0</b>	<b>220988</b>	<b>1792</b>	<b>1792</b>	<b>R</b>	<b>95,0</b>	<b>0,0</b>	<b>0:32.45</b>	<b>dd</b>
1903	SenDerM+	20	0	3596552	354820	122572	S	2,0	9,5	0:25.60	gnome-s+
3172	SenDerM+	20	0	772704	53124	39804	S	0,7	1,4	0:01.82	gnome-t+
1964	SenDerM+	20	0	527124	13692	7040	S	0,3	0,4	0:00.74	ibus-da+
1	root	20	0	107520	16616	10748	S	0,0	0,4	0:01.69	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+
5	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	slub_fl+
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	netns
10	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mm_perc+
12	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tas+
13	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tas+
14	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tas+
15	root	20	0	0	0	0	S	0,0	0,0	0:00.04	ksoftir+
16	root	20	0	0	0	0	S	0,0	0,0	0:00.00	pr/tty0
17	root	20	0	0	0	0	I	0,0	0,0	0:00.09	rcu_pre+

Рис.4. Убийство задания dd в top.

### Управление процессами:

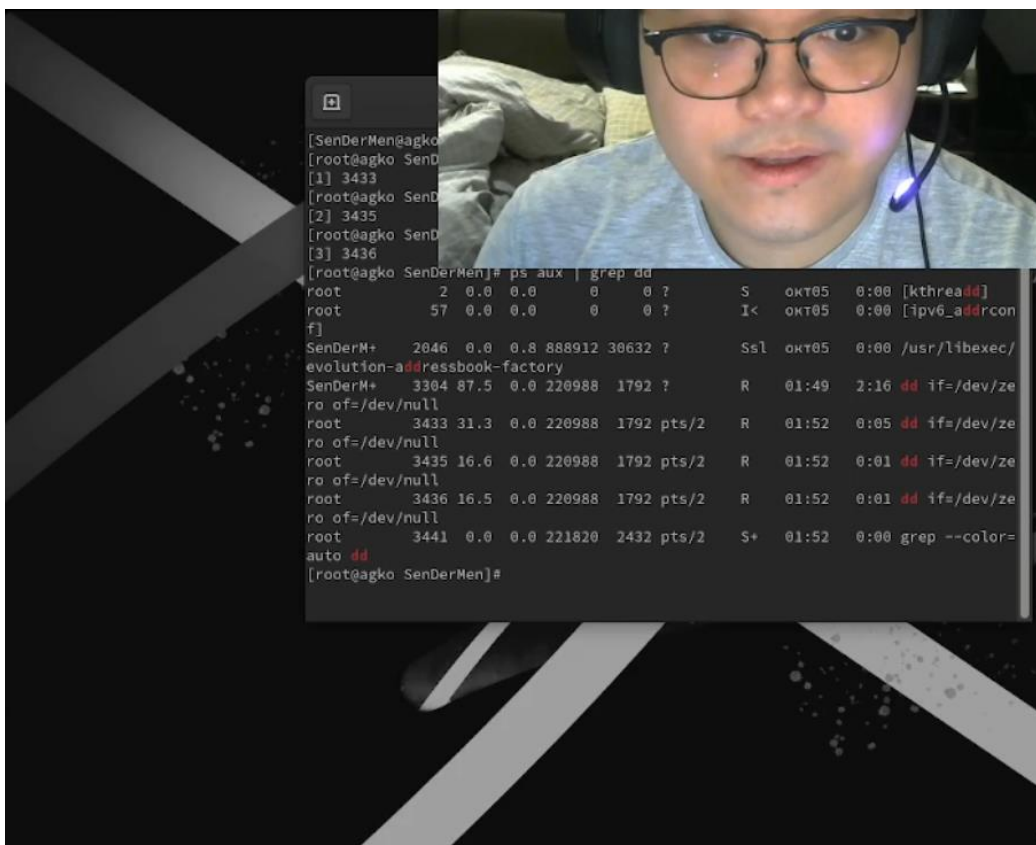
Получим полномочия администратора **su -** и введём следующие команды:

**dd if=/dev/zero of=/dev/null &**

**dd if=/dev/zero of=/dev/null &**

**dd if=/dev/zero of=/dev/null &**

После чего введём **ps aux | grep dd**, которое показывает все строки, в которых есть буквы dd. Запущенные процессы dd идут последними. Используем PID первого процесса dd, чтобы изменить приоритет (**renice -n 5 2682**).



**Рис. 5.** Просмотр всех строк, в которых есть dd. Изменение приоритета.

Введём **ps fax | grep -B5 dd**. Параметр **-B5** показывает соответствующие запросу строки, включая пять строк до этого. Поскольку **ps fax** показывает иерархию отношений между процессами, мы также видим оболочку, из которой были запущены все процессы **dd**, и её PID.

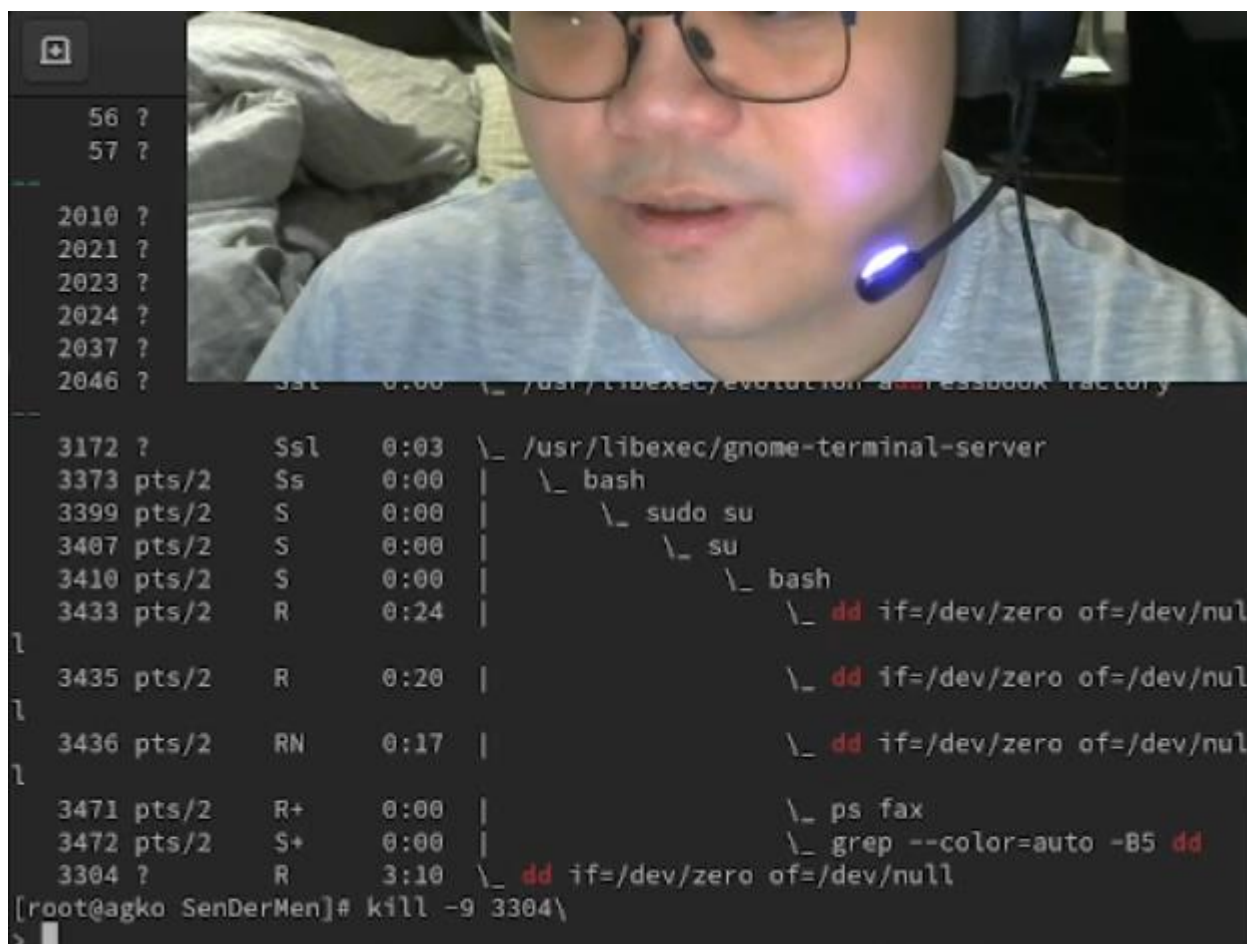


Рис. 6. Просмотр иерархии отношений между процессами.

Теперь найдём PID корневой оболочки, из которой были запущены процессы `dd`, и введём `kill -9` (указав PID оболочки). Мы увидим, что наша корневая оболочка закрылась, а вместе с ней и все процессы `dd` (остановка родительского процесса — простой и удобный способ остановить все его дочерние процессы).

```
[root@agko SenDerMen]# kill -9 3304
```

**Рис. 7.** Закрытие корневой оболочки.

### Самостоятельная работа (задание 1):

Получим полномочия администратора `su` – и запустим команду `dd if=/dev/zero of=/dev/null &` трижды как фоновое задание. Затем увеличим приоритет первой команды, используя значение приоритета `-5`, после чего изменим приоритет того же процесса ещё раз, но используем на этот раз значение `-15` (мы можем менять приоритет команды от `-20` (самый высокий приоритет) до `19` (самый низкий приоритет)). Завершим все процессы `dd`, которые мы запустили командой: `killall dd` (Рис. 3).

```
[SenDerMen@agko ~]$ dd if=/dev/zero of=/dev/null &  
[3] 3510  
[SenDerMen@agko ~]$ dd if=/dev/zero of=/dev/null &  
[4] 3515  
[SenDerMen@agko ~]$ dd if=/dev/zero of=/dev/null &  
[5] 3520
```

**Рис. 8.** Запуск команды трижды как фоновое задание.



```

[SenDerMen@agko ~]$ renice -n 5 3510
3510 (process ID) old priority 0, new priority 5
[SenDerMen@agko ~]$ renice -n 15 3510
3510 (process ID) old priority 5, new priority 15
[SenDerMen@agko ~]$ kill -9 3510
[SenDerMen@agko ~]$ kill -9 3515
[3]  Убито          dd if=/dev/zero of=/dev/null
[4]  Убито          dd if=/dev/zero of=/dev/null
[SenDerMen@agko ~]$ kill -9 3520
[5]  Убито          dd if=/dev/zero of=/dev/null
[SenDerMen@agko ~]$

```

Рис. 9. Увеличение приоритета первой команды, завершение всех процессов.

### Самостоятельная работа (задание 2):

Получим полномочия администратора **su** – и запустим программу **yes** в фоновом режиме с подавлением потока вывода (**yes > /dev/null &**), далее запустим программу **yes** на переднем плане с подавлением потока вывода и приостановим выполнение программы. Заново запустим программу **yes** с теми же параметрами, затем завершим её выполнение. Повторим действия, но уже запустим программу **yes** на переднем плане без подавления потока вывода (**yes > /dev/null**). Также приостановим выполнение программы и заново запустим программу **yes** с теми же параметрами, затем завершим её выполнение. Проверим состояния заданий, воспользовавшись командой **jobs**. Далее переведём процесс, который у нас выполняется в фоновом режиме, на передний план, затем остановим его (**fg 1, после чего Ctrl+c**). Переведём 3 процесс с подавлением потока вывода в фоновый режим (**bg 3**) и проверим состояния

заданий, воспользовавшись командой **jobs**. Обратим внимание, что процесс стал выполняющимся (Running) в фоновом режиме. Запустим процесс в фоновом режиме таким образом, чтобы он продолжил свою работу даже после отключения от терминала (**nohup yes > /dev/null &**). Закроем окно и заново запустим консоль. Убедимся, что процесс продолжил свою работу.

```
[SenDerMen@agko ~]$ jobs
[1]+  Запущен          yes > /dev/null &
[SenDerMen@agko ~]$ fg
yes > /dev/null
^Z
[1]+  Остановлен      yes > /dev/null
[SenDerMen@agko ~]$ bg
[1]+ yes > /dev/null &
[SenDerMen@agko ~]$ jobs
[1]+  Запущен          yes > /dev/null &
[SenDerMen@agko ~]$ nohup yes
nohup: ввод игнорируется, вывод добавляется в 'nohup.out'
[SenDerMen@agko ~]$
```

**Рис. 10.** Запуск программы `yes` в фоновом режиме с подавлением потока вывода. Запуск программы `yes` на переднем плане без подавления потока вывода. Перевод процесса на передний план и его остановка. Перевод процесса в фоновый режим. Проверка состояния заданий. Запуск процесса в фоновом режиме с условиями.

Сейчас получим информацию о запущенных в операционной системе процессах с помощью утилиты **top**.

```

top - 02:43:41 up 8:11, 2 users, load average: 0,93, 1,93, 2,03
Tasks: 177 total, 2 running, 175 sleeping, 0 stopped, 0 zombie
%Cpu(s): 58,8 us, 41,2 sy, 0,0 ni, 0,0 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
MiB Mem : 3660,0 total, 233,4 free, 1222,2 used, 2501,5 buff/cache
MiB Swap: 2048,0 total, 2044,2 free, 3,8 used. 2437,8 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  4723 SenDerM+  20   0   220948   1792   1792 R   76,5   0,0   0:08.60 yes
  1903 SenDerM+  20   0 3609436 360788 120916 S    5,9   9,6   1:15.53 gnome-s+
    1 root      20   0   107520   18152   10748 S    0,0   0,5   0:01.74 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+
    5 root       0 -20         0         0         0 I    0,0   0,0   0:00.00 slub_fl+
    6 root       0 -20         0         0         0 I    0,0   0,0   0:00.00 netns
   10 root       0 -20         0         0         0 I    0,0   0,0   0:00.00 mm_perc+
   12 root      20   0         0         0         0 I    0,0   0,0   0:00.00 rcu_tas+
   13 root      20   0         0         0         0 I    0,0   0,0   0:00.00 rcu_tas+
   14 root      20   0         0         0         0 I    0,0   0,0   0:00.00 rcu_tas+
   15 root      20   0         0         0         0 S    0,0   0,0   0:00.05 ksoftir+
   16 root      20   0         0         0         0 S    0,0   0,0   0:00.01 pr/tty0
   17 root      20   0         0         0         0 I    0,0   0,0   0:00.12 rcu_pre+
   18 root      rt    0         0         0         0 S    0,0   0,0   0:00.03 migrati+
   19 root     -51   0         0         0         0 S    0,0   0,0   0:00.00 idle_in+
  
```

**Рис. 11.** Получение информации о запущенных в операционной системе процессах.

Запустим ещё три программы **yes** в фоновом режиме с подавлением потока вывода (**yes > /dev/null &**). Убьём два процесса: для одного используем его PID (**kill -9 3098**), а для другого — его идентификатор конкретного задания (**fg 2** и **Ctrl+c**). Попробуем послать сигнал 1 (SIGHUP) процессу, запущенному с помощью **nohup** (**kill -1 3100**), и обычному процессу (**kill -1 2993**).

```

[SenDerMen@agko ~]$ kill -9 %2
[2]- Убито yes > /dev/null
[SenDerMen@agko ~]$ jobs
[3]+ Запущен yes > /dev/null &
[SenDerMen@agko ~]$ kill -s SIGHUP 4723
[SenDerMen@agko ~]$ kill -s SIGHUP 4840
[3]+ Обрыв терминальной линии yes > /dev/null
[SenDerMen@agko ~]$ kill -s SIGHUP 4723
[SenDerMen@agko ~]$ kill -s SIGHUP 4723 %1
bash: kill: %1: нет такого задания
[SenDerMen@agko ~]$ kill -s SIGHUP 4723
[SenDerMen@agko ~]$ kill -s SIGHUP 4723_pohup
bash: kill: 4723_pohup: аргументами должны быть идентификаторы процессов или за-
ний
[SenDerMen@agko ~]$

```

**Рис. 12.** Запуск трёх программ `yes` в фоновом режиме с подавлением потока вывода, убийство двух процессов, попытка послать сигнал 1 (SIGHUP).

Запустим ещё несколько программ `yes` в фоновом режиме с подавлением потока вывода (`yes > /dev/null &`) и завершим их работу одновременно, используя команду `killall yes`.

```

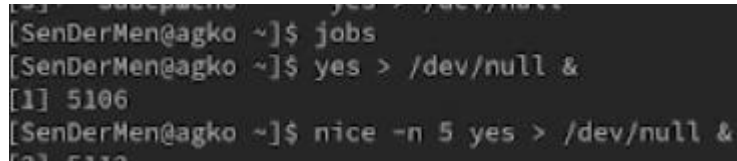
[SenDerMen@agko ~]$ killall yes
[1] Завершено yes > /dev/null
[2]- Завершено yes > /dev/null
[3]+ Завершено yes > /dev/null

```

**Рис. 13.** Запуск программ `yes` в фоновом режиме с подавлением потока вывода и одновременное завершение их работы.

После чего запустим программу `yes` в фоновом режиме с подавлением потока вывода (`yes > /dev/null &`). Используя утилиту `nice` (`nice -n 15 yes`), запустим программу `yes` с теми же параметрами и с приоритетом, большим на

5. Сравним абсолютные и относительные приоритеты у этих двух процессов (**ps -l | grep yes**). Используя утилиту **renice**, изменим приоритет у одного из потоков **yes** таким образом, чтобы у обоих потоков приоритеты были равны (**renice -n 15 3109**).



```
[SenDerMen@agko ~]$ jobs
[SenDerMen@agko ~]$ yes > /dev/null &
[1] 5106
[SenDerMen@agko ~]$ nice -n 5 yes > /dev/null &
[2] 5113
```

**Рис. 14.** Запуск программы **yes** в фоновом режиме с подавлением потока вывода. Запуск программы **yes** с теми же параметрами и с приоритетом, большим на 5. Сравнение абсолютных и относительных приоритетов, изменение приоритета.

Ответы на контрольные вопросы:

1. Какая команда даёт обзор всех текущих заданий оболочки? **jobs**.
2. Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме? **bg номер\_задания**.

3. Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки? Ctrl+c.

4. Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой в данный момент работает пользователь, невозможен. Что можно сделать, чтобы отменить задание? Внутри top использовать k, чтобы убить задание.

5. Какая команда используется для отображения отношений между родительскими и дочерними процессами? ps fax.

6. Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий? renice -n приоритет\_процесса <PID>.

7. В системе в настоящее время запущено 20 процессов dd. Как проще всего остановить их все сразу? killall dd.

8. Какая команда позволяет остановить команду с именем mycommand? Сначала узнаем PID процесса mycommand -ps aux | grep mycommand далее команда kill -9 <PID>.

9. Какая команда используется в top, чтобы убить процесс? k.

10. Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов? Запустить команду в фоновом режиме.

Вывод:

В ходе выполнения лабораторной работы были получены навыки управления процессами операционной системы.