

# Лабораторная работа №6

Управление процессами

---

Сидорова А.В.

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Сидорова Арина Валерьевна
- студентка НПИбд-02-24
- ст.б. 1132242912
- Российский университет дружбы народов

## Вводная часть

---

Управление процессами — ключевой навык администратора операционных систем, необходимый для эффективного распределения ресурсов, контроля выполнения программ и обеспечения стабильности системы.

## Объект исследования

- Процессы и задания в операционной системе Linux.

## Предмет исследования

- Методы и инструменты управления процессами (команды `jobs`, `nice`, `renice`, `kill`, `nohup` и др.).

1. Получить практические навыки управления процессами и заданиями в операционной системе Linux.
2. Освоить команды для просмотра и управления процессами (ps, top, kill, jobs, fg, bg).
3. Научиться управлять приоритетами процессов (nice, renice).
4. Изучить механизмы фонового выполнения и завершения процессов.
5. Выполнить задания по управлению заданиями и процессами.

## Выполнение лабораторной работы

---





## Получим полномочия администратора

Введем jobs Мы 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 введем bg 3 С помощью команды jobs посмотрим изменения в статусе заданий.

```
avsidorova@avsidorova:~$ sudo -i
[sudo] пароль для avsidorova:
root@avsidorova:~# sleep 3600 &
[1] 4430
root@avsidorova:~# dd if=/dev/zero of=/dev/null &
[2] 4449
root@avsidorova:~# sleep 7200

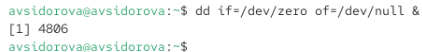
^Z
[3]+  Остановлен    sleep 7200
root@avsidorova:~# jobs
[1]  Запущен        sleep 3600 &
[2]-  Запущен        dd if=/dev/zero of=/dev/null &
[3]+  Остановлен    sleep 7200
root@avsidorova:~# bg 3
[3]+ sleep 7200 &
root@avsidorova:~# fg 1
sleep 3600
^C
root@avsidorova:~# fg 2
dd if=/dev/zero of=/dev/null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

## Для перемещения задания 1 на передний план

Введем `Ctrl + c`, чтобы отменить задание 1. С помощью команды `jobs` посмотрим изменения в статусе заданий. Прделаем то же самое для отмены заданий 2 и 3.

Откроем второй терминал и под учётной записью своего пользователя введем в нём:

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

A terminal window showing the execution of the command `dd if=/dev/zero of=/dev/null &`. The prompt is `avsidorova@avsidorova:~$`. The command is entered and executed, resulting in the prompt `[1] 4806` and then `avsidorova@avsidorova:~$`.

```
avsidorova@avsidorova:~$ dd if=/dev/zero of=/dev/null &  
[1] 4806  
avsidorova@avsidorova:~$
```

Рис. 2: `dd if=/dev/zero of=/dev/null &`

## На другом терминале под учётной записью своего пользователя запустим

мы увидим, что задание dd всё ещё запущено. Для мыхода из top используем q .

```
avsidorova@avsidorova:~$ top
```

```
top - 18:16:16 up 6 min, 4 users, load average: 2,04, 1,62, 0,79
```

```
Tasks: 252 total, 2 running, 250 sleeping, 0 stopped, 0 zombie
```

```
%Cpu(s): 12,0 us, 26,2 sy, 0,0 ni, 54,3 id, 0,0 wa, 7,0 hi, 0,5 si, 0,0 st
```

```
MiB Mem : 3653,5 total, 847,1 free, 2056,0 used, 1070,0 buff/cache
```

```
MiB Swap: 4036,0 total, 4036,0 free, 0,0 used. 1597,5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4806	avsidor+	20	0	227308	2148	2148	R	88,4	0,1	0:18.02	dd
2437	avsidor+	20	0	4189436	400536	137000	S	14,2	10,7	1:05.32	gnome-shell
3974	avsidor+	20	0	1925540	361212	98840	S	12,9	9,7	0:06.09	ptxis
30	root	20	0	0	0	0	I	3,0	0,0	0:00.37	kworker/u9:0-events_unbound
128	root	20	0	0	0	0	I	1,3	0,0	0:01.84	kworker/u10:4-xfs-cil/dm-0
1	root	20	0	49196	41220	10252	S	0,3	1,1	0:02.28	systemd
18	root	20	0	0	0	0	I	0,3	0,0	0:00.27	rcu_preempt
43	root	39	19	0	0	0	S	0,3	0,0	0:00.77	khugepaged
55	root	0	-20	0	0	0	I	0,3	0,0	0:00.11	kworker/1:1H-xfs-log/dm-0
658	root	20	0	50016	17468	15804	S	0,3	0,5	0:00.94	systemd-journal
3795	avsidor+	20	0	2983716	213304	108196	S	0,3	5,7	0:14.84	Isolated Web Co
4181	avsidor+	20	0	2659340	79952	65104	S	0,3	2,1	0:00.20	Web Content
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-rcu_gp
5	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-sync_wq
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-slab_flushwq
7	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-netns
8	root	20	0	0	0	0	I	0,0	0,0	0:00.15	kworker/0:0-ata_sff
10	root	0	-20	0	0	0	I	0,0	0,0	0:00.08	kworker/0:0H-kblockd
11	root	20	0	0	0	0	I	0,0	0,0	0:00.00	kworker/u8:0-events_unbound
12	root	20	0	0	0	0	I	0,0	0,0	0:00.06	kworker/u8:1-netns
13	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-mm_percpu_wq
14	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_kthread
15	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_rude_kthread
16	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_trace_kthread
17	root	20	0	0	0	0	S	0,0	0,0	0:00.11	ksoftirqd/0

Вновь запустим top и в нём используем k , чтобы убить задание dd. После этого мыйдем из top.

```
top - 18:17:22 up 7 min, 4 users, load average: 2,12, 1,74, 0,89
Tasks: 250 total, 2 running, 247 sleeping, 1 stopped, 0 zombie
%Cpu(s): 11,2 us, 27,3 sy, 0,0 ni, 54,3 id, 0,0 wa, 6,7 hi, 0,5 si, 0,0 st
MiB Mem : 3653,5 total, 830,8 free, 2071,2 used, 1071,1 buff/cache
MiB Swap: 4036,0 total, 4036,0 free, 0,0 used. 1582,3 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4806	avsidor+	20	0	227308	2148	2148	R	93,0	0,1	1:16.52	dd
2437	avsidor+	20	0	4196156	400536	137000	S	9,6	10,7	1:15.22	gnome-shell
3974	avsidor+	20	0	1923420	360432	98840	S	6,3	9,6	0:11.27	ptxix
128	root	20	0	0	0	0	I	3,0	0,0	0:02.41	kworker/u10:4-events_unbound
1	root	20	0	49196	41220	10252	S	0,7	1,1	0:02.43	systemd
20	root	20	0	0	0	0	S	0,3	0,0	0:00.15	rcu_exp_gp_kthread_worker
55	root	0	-20	0	0	0	I	0,3	0,0	0:00.13	kworker/1:1H-xfs-log/dm-0
117	root	20	0	0	0	0	I	0,3	0,0	0:00.28	kworker/u10:3-events_unbound
658	root	20	0	50016	17468	15804	S	0,3	0,5	0:01.06	systemd-journal
3464	avsidor+	20	0	3649560	470260	201768	S	0,3	12,6	0:57.52	firefox
5009	avsidor+	20	0	231612	5412	3236	R	0,3	0,1	0:00.04	top
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-rcu_gp
5	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-sync_wq
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-slub_flushwq
7	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-netns
8	root	20	0	0	0	0	I	0,0	0,0	0:00.20	kworker/0:0-ata_sff
10	root	0	-20	0	0	0	I	0,0	0,0	0:00.08	kworker/0:0H-kblockd
11	root	20	0	0	0	0	I	0,0	0,0	0:00.00	kworker/u8:0-events_unbound
12	root	20	0	0	0	0	I	0,0	0,0	0:00.07	kworker/u8:1-netns
13	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-mm_percpu_wq
14	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_kthread
15	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_rude_kthread
16	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_trace_kthread
17	root	20	0	0	0	0	S	0,0	0,0	0:00.11	ksoftirqd/0
18	root	20	0	0	0	0	I	0,0	0,0	0:00.33	rcu_preempt
19	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_exp_par_gp_kthread_worker/0
21	root	rt	0	0	0	0	S	0,0	0,0	0:00.01	migration/0
22	root	-51	0	0	0	0	S	0,0	0,0	0:00.00	idle_inject/0
23	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuha/0



## Введем следующие команды:

```
root@avsidorova:~# dd if=/dev/zero of=/dev/null &  
[1] 5133  
root@avsidorova:~# dd if=/dev/zero of=/dev/null &  
[2] 5164  
root@avsidorova:~# dd if=/dev/zero of=/dev/null &  
[3] 5202
```

Рис. 5: x3 dd if=/dev/zero of=/dev/null &



Это показывает все строки, в которых есть буквы dd.

Запущенные процессы dd идут последними.

```
root@avsidorova:~# ps aux | grep dd
root          2  0.0  0.0      0   0 ?        S   18:09   0:00 [kthreaddd]
root         76  0.0  0.0      0   0 ?        I<  18:09   0:00 [kworker/R-ipv6_aaddrconf]
root        1113  0.0  0.0 512956 3160 ?        Sl   18:09   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
avsidor+    2869  0.0  0.7 963332 26260 ?        Ssl  18:10   0:00 /usr/libexec/evolution-addressbook-factory
avsidor+    3687  0.0  0.9 260704 34732 ?        Sl   18:11   0:00 /usr/lib64/firefox/firefox -contentproc -parentBuildID 20250918202509 -prefsHandle 0:35109 -prefMapHandle 1:271057 -sandboxReporter 2 -chrootClient 3 -ipcHandle 4 -initialChannelId {cc74303f-7d36-4002-9ecc-5567deda1b6b} -parentPid 3464 -appDir /usr/lib64/firefox/browser 3 rdd
avsidor+    4806 83.6  0.0 227308 2148 ?        R   18:15   2:09 dd if=/dev/zero of=/dev/null
root        5133 56.7  0.0 227308 2088 pts/1    R   18:17   0:22 dd if=/dev/zero of=/dev/null
root        5164 34.6  0.0 227308 1972 pts/1    R   18:18   0:09 dd if=/dev/zero of=/dev/null
root        5202 31.3  0.0 227308 2008 pts/1    R   18:18   0:04 dd if=/dev/zero of=/dev/null
root        5239  0.0  0.0 227712 2284 pts/1    S+  18:18   0:00 grep --color=auto dd
root@avsidorova:~#
```

Рис. 6: ps aux | grep dd

Используем PID одного из процессов dd, чтобы изменить приоритет.

Используем renice -n 5

```
root@avsidorova:~# renice -n 5 2869
2869 (process ID) old priority 0, new priority 5
root@avsidorova:~#
```

Рис. 7: renice -n 5

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

```
2869 (process ID) old priority 0, new priority 5
root@avsidorova:~# ps fax | grep -B5 dd
  PID TTY          STAT       TIME COMMAND
    2 ?            S          0:00 [kthreadd]
--
   69 ?            I          0:00 \_ [kworker/1:2-events]
   71 ?            I<         0:00 \_ [kworker/R-acpi_thermal_pm]
   72 ?            I<         0:00 \_ [kworker/R-kmpath_rdacd]
   73 ?            I<         0:00 \_ [kworker/R-kalua]
   75 ?            I<         0:00 \_ [kworker/R-mld]
   76 ?            I<         0:00 \_ [kworker/R-ipv6_addrconf]
--
  873 ?            Sns        0:00 /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --initfile=/lib
/alsa/init/00main rdaemon
  875 ?            S          0:00 /usr/sbin/chronyd -F 2
  918 ?            Ssl        0:00 /usr/sbin/ModemManager
  919 ?            Ssl        0:00 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid
 1110 ?            Sl         0:00 /usr/bin/VBoxDRMClient
 1113 ?            Sl         0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
--
 2830 ?            S          0:00 | \_ /usr/bin/Xwayland :0 -rootless -noreset -accessx -core -auth /run/user/1000/.mut
ter-Xwaylandauth.S8N6D3 -listenfd 4 -listenfd 5 -displayfd 6 -initfd 7 -byteswappedclients -enable-ei-portal
 3116 ?            Sl         0:00 | \_ /usr/libexec/mutter-x11-frames
 3464 ?            Sl         1:07 | \_ /usr/lib64/firefox/firefox
 3641 ?            Sl         0:00 | \_ /usr/lib64/firefox/firefox -contentproc -parentBuildID 20250918202509 -prefs
Handle 0:34968 -prefMapHandle 1:271057 -sandboxReporter 2 -chrootClient 3 -ipcHandle 4 -initialChannelId {5ded0f5c-b8
56-4ad3-8fe0-593d306e5340} -parentPid 3464 -appDir /usr/lib64/firefox/browser 1 socket
 3682 ?            Sl         0:01 | \_ /usr/lib64/firefox/firefox -contentproc -isForBrowser -prefsHandle 0:35109 -
prefMapHandle 1:271057 -jsInitHandle 2:242716 -parentBuildID 20250918202509 -sandboxReporter 3 -chrootClient 4 -ipcHa
ndle 5 -initialChannelId {c1e072e8-fda0-4fa2-918c-e0624351a73f} -parentPid 3464 -greomni /usr/lib64/firefox/omni.ja -
```

## Найдем PID корневой оболочки, из которой были запущены процессы dd,

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.

```
4040 pts/0 Ss 0:00 | \_ /usr/bin/bash
4336 pts/0 S+ 0:00 | \_ sudo -i
4379 pts/1 Ss 0:00 | \_ sudo -i
4380 pts/1 S 0:00 | \_ -bash
5133 pts/1 R 1:06 | \_ dd if=/dev/zero of=/dev/null
5164 pts/1 R 0:57 | \_ dd if=/dev/zero of=/dev/null
5202 pts/1 R 0:47 | \_ dd if=/dev/zero of=/dev/null
5552 pts/1 R+ 0:00 | \_ ps fax
5553 pts/1 S+ 0:00 | \_ grep --color=auto -B5 dd
3998 ? S 0:00 \_ catatonit -P
4806 ? R 4:02 \_ dd if=/dev/zero of=/dev/null
root@avsidorova:~# kill -9 4380
Убито
avsidorova@avsidorova:~$
```

Рис. 9: kill -9

## Самостоятельная работа

---



## 1. Запустим команду

dd if=/dev/zero of=/dev/null трижды как фоновое задание.

```
^L
[2]+  Остановлен    dd if=/dev/zero of=/dev/null
root@avsidorova:~# dd if=/dev/zero of=/dev/null &
[3] 6653
root@avsidorova:~#
root@avsidorova:~#
root@avsidorova:~# dd if=/dev/zero of=/dev/nul
dd: запись в '/dev/nul': На устройстве не осталось свободного места
3193+0 records in
3192+0 records out
1194304 bytes (4,2 MB, 4,0 MiB) copied, 0,791666 s, 5,3 MB/s
root@avsidorova:~# dd if=/dev/zero of=/dev/null &
[4] 6680
root@avsidorova:~#

root@avsidorova:~#
root@avsidorova:~# dd if=/dev/zero of=/dev/null &
[5] 6701
root@avsidorova:~#
root@avsidorova:~# █
```

2. Увеличим приоритет одной из этих команд, используя значение приоритета -5.



### 3. Изменим приоритет того же процесса ещё раз, но используем на этот раз значение

-15. Разница в значении приоритетов

```
root@avsidorova:~# renice -n -5 6653
6653 (process ID) old priority 0, new priority -5
root@avsidorova:~# renice -n -15 6653
6653 (process ID) old priority -5, new priority -15
root@avsidorova:~#
```

Рис. 11: Увеличение приоритета (5, 15)

## 4. Завершим все процессы dd, которые мы запустили.

```
[5] Запущен      dd if=/dev/zero of=/dev/null &
root@avsidorova:~# fg 1
dd if=/dev/zero of=/dev/null
^Z
[1]+  Остановлен  dd if=/dev/zero of=/dev/null
root@avsidorova:~# ^C
root@avsidorova:~# fg 2
dd if=/dev/zero of=/dev/null
^C52658564+0 records in
52658564+0 records out
26961184768 bytes (27 GB, 25 GiB) copied, 276,272 s, 97,6 MB/s

root@avsidorova:~# fg 3
dd if=/dev/zero of=/dev/null
^C219674843+0 records in
219674843+0 records out
112473519616 bytes (112 GB, 105 GiB) copied, 240,295 s, 468 MB/s

root@avsidorova:~# ^C
root@avsidorova:~# ^C
root@avsidorova:~# fg 4
dd if=/dev/zero of=/dev/null
^C91089952+0 records in
91089952+0 records out
46638054912 bytes (47 GB, 43 GiB) copied, 236,1 s, 198 MB/s

root@avsidorova:~# fg 5
dd if=/dev/zero of=/dev/null
^C84135265+0 records in
84135265+0 records out
43077255680 bytes (43 GB, 40 GiB) copied, 240,464 s, 179 MB/s

root@avsidorova:~# jobs
[1]+  Остановлен  dd if=/dev/zero of=/dev/null
root@avsidorova:~# fg 1
dd if=/dev/zero of=/dev/null
^C85803603+0 records in
85803603+0 records out
43931444224 bytes (44 GB, 41 GiB) copied, 382,937 s, 115 MB/s

root@avsidorova:~# jobs
root@avsidorova:~#
```



## 1. Запустим программу yes в фоновом режиме с подавлением потока вывода.

```
root@avsidorova:~# yes > /dev/null &
[1] 7465
root@avsidorova:~# yes > /dev/null
^Z
[2]+  Остановлен      yes > /dev/null
root@avsidorova:~# yes > /dev/null &
[3] 7517
root@avsidorova:~# yes > /dev/null
^C
root@avsidorova:~# ^C
root@avsidorova:~# jobs
[1]   Запущен          yes > /dev/null &
[2]+  Остановлен      yes > /dev/null
[3]-  Запущен          yes > /dev/null &
root@avsidorova:~# █
```

Рис. 13: Запускаем yes

## 2. Запустим программу `ues` на переднем плане с подавлением потока вывода.

Приостановим выполнение программы. Заново запустим программу `ues` с теми же параметрами, затем завершим её выполнение.

##3. Запустим программу `ues` на переднем плане без подавления потока вывода. Приостановим выполнение программы. Заново запустим программу `ues` с теми же параметрами, затем завершим её выполнение.

#### 4. Проверим состояния заданий, воспользовавшись командой jobs.

```
root@avsidorova:~# jobs
[1]  Запущен          yes > /dev/null &
[2]+ Остановлен      yes > /dev/null
[3]- Запущен          yes > /dev/null &
root@avsidorova:~# fg 3
yes > /dev/null
^C
root@avsidorova:~# fg 1
yes > /dev/null
^Z
[1]+  Остановлен      yes > /dev/null
root@avsidorova:~#
```

Рис. 14: jobs

5. Переведем процесс, который у нас выполняется в фоновом режиме, на передний план, затем остановим его.

## 6. Переведем любой наш процесс с подавлением потока вывода в фоновый режим.

```
^Z
[2]+  Остановлен    yes > /dev/null
root@avsidorova:~# bg 2
[2]+ yes > /dev/null &
root@avsidorova:~#
```

Рис. 15: bg 2



## 7. Проверим состояния заданий, воспользовавшись командой jobs.

Обратим внимание, что процесс стал выполняющимся (Running) в фоновом режиме.

```
[2]+ yes > /dev/null &  
root@avsidorova:~# jobs  
[1]+  Остановлен      yes > /dev/null  
[2]-  Запущен         yes > /dev/null &  
root@avsidorova:~#
```

Рис. 16: jobs

8. Запустим процесс в фоновом режиме таким образом, чтобы он продолжил свою работу даже после отключения от терминала.

```
[4]- Запущен yes /dev/null &  
root@avsidorova:~# nohup yes > /dev/null &  
[3] 8229
```

Рис. 17: nohup yes > /dev/null &

## 10. Получим информацию о запущенных в операционной системе процессах с помощью утилиты top.

avsidorova@avsidorova:~\$ top

top - 18:47:30 up 34 min, 5 users, load average: 7,53, 6,85, 5,94

Tasks: **255** total, **7** running, **248** 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 : **3653,5** total, **793,1** free, **2085,8** used, **1052,6** buff/cache

MiB Swap: **4036,0** total, **4036,0** free, **0,0** used. **1567,7** avail Mem

Рис. 18: top

## 11. Запустим ещё три программы yes в фоновом режиме с подавлением потока вывода.

```
40 root      0 -20      0      0      0 1      0,0      0,  
[1]+  Остановлен   top  
avsidorova@avsidorova:~$ yes > /dev/null &  
[2] 8659  
avsidorova@avsidorova:~$ yes > /dev/null &  
[3] 8683  
avsidorova@avsidorova:~$ yes > /dev/null &  
[4] 8694  
avsidorova@avsidorova:~$ █
```

Рис. 19: x3 yes

12. Убьём два процесса: для одного используем его PID, а для другого — его идентификатор конкретного задания.

```
avsidorova@avsidorova:~$ yes > /dev/null &
[1] 9073
avsidorova@avsidorova:~$ yes > /dev/null &
[2] 9083
avsidorova@avsidorova:~$ yes > /dev/null &
[3] 9097
avsidorova@avsidorova:~$ fg 1
yes > /dev/null
^C
avsidorova@avsidorova:~$ kill -9 9083
[2]-  Убито                yes > /dev/null
avsidorova@avsidorova:~$
```

Рис. 20: kill + ctrl C

14. Запустим ещё несколько программ yes в фоновом режиме с подавлением потока вывода.

15. Завершим их работу одновременно, используя команду killall.

```
root@avsidorova:~# killall yes
root@avsidorova:~# jobs
[1]-  Завершено      yes > /dev/null
[2]+  Завершено      yes > /dev/null
```

Рис. 21: killall yes

## 16. Запустим программу `yes` в фоновом режиме с подавлением потока вывода.

Используя утилиту `nice`, запустим программу `yes` с теми же параметрами и с приоритетом, большим на 5. Сравним абсолютные и относительные приоритеты у этих двух процессов. Используя утилиту `renice`, изменим приоритет у одного из потоков `yes` таким образом, чтобы у обоих потоков приоритеты были равны.

```
root@avsidorova:~# killall yes
root@avsidorova:~# jobs
[1]-  Завершено      yes > /dev/null
[2]+  Завершено      yes > /dev/null
```

Рис. 22: `renice`

## Результаты

---



- Освоены команды `ps`, `jobs`, `fg`, `bg`, `kill`, `nice`, `renice`, `nohup`.
- Выполнено управление заданиями: запуск, остановка, перевод между фоном и передним планом.
- Изменены приоритеты процессов, завершены процессы через PID и имя.
- Запущены фоновые процессы, устойчивые к разрыву сессии (`nohup`).

```
REVEALJS_THEME = beige
```

```
...
```