

Отчёт по лабораторной работе №6

Управление процессами

Агджабекова Эся Рустамовна

Содержание

1	Цель работы	5
2	Ход выполнения работы	6
2.1	Управление заданиями	6
2.2	Управление процессами	9
2.3	Изменение приоритета процессов	11
2.4	Работа с процессами и приоритетами (команда yes)	12
3	Контрольные вопросы	16
4	Заключение	18

Список иллюстраций

2.1	Запуск и управление заданиями	7
2.2	Работа с заданиями в фоне и переднем плане	8
2.3	Мониторинг процессов в top	9
2.4	Просмотр процессов с помощью ps aux	10
2.5	Отображение иерархии процессов	10
2.6	Изменение приоритета и завершение процессов	12
2.7	Отслеживание процесса yes в top	13
2.8	Завершение процессов yes	14
2.9	Изменение приоритета процессов yes	15

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Ход выполнения работы

2.1 Управление заданиями

1. Получены права администратора с помощью команды `su`.
2. Запущены процессы: `sleep 3600 &`, `dd if=/dev/zero of=/dev/null &`, а также `sleep 7200` без амперсанда в конце. Последний процесс был остановлен комбинацией `Ctrl+Z`.
3. Просмотрен список заданий командой `jobs`, из которого видно, что два первых процесса находятся в состоянии **Running**, а третий — **Stopped** (см. рис. fig. 2.1).

```

eragdzhabekova@eragdzhabekova:~$ su
Password:
root@eragdzhabekova:/home/eragdzhabekova# sleep 3600 &
[1] 3335
root@eragdzhabekova:/home/eragdzhabekova# dd if=/dev/zero of=/dev/null &
[2] 3382
root@eragdzhabekova:/home/eragdzhabekova# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@eragdzhabekova:/home/eragdzhabekova# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@eragdzhabekova:/home/eragdzhabekova# bg 3
[3]+  sleep 7200 &
root@eragdzhabekova:/home/eragdzhabekova# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@eragdzhabekova:/home/eragdzhabekova# fg 1
sleep 3600
^C
root@eragdzhabekova:/home/eragdzhabekova# fg 2
dd if=/dev/zero of=/dev/null
^C158272023+0 records in
158272023+0 records out
81035275776 bytes (81 GB, 75 GiB) copied, 105.137 s, 771 MB/s

root@eragdzhabekova:/home/eragdzhabekova# fg 3
sleep 7200
^C
root@eragdzhabekova:/home/eragdzhabekova# █

```

Рис. 2.1: Запуск и управление заданиями

4. Задание 3 переведено в фоновый режим с помощью команды `bg 3`, после чего его статус изменился на **Running**.
5. Задание 1 возвращено на передний план командой `fg 1`, а затем прервано комбинацией `Ctrl+C`. Аналогично были завершены задания 2 и 3 (см. рис. fig. 2.2).

eragdzhabekova@eragdzhabekova:/home/eragdzhabekova

eragdzhabekova@eragdzhabekova:~ - top

top - 15:22:30 up 7 min, 4 users, load average: 0.56, 0.38, 0.17

Tasks: 232 total, 1 running, 231 sleeping, 0 stopped, 0 zombie

%Cpu(s): 1.8 us, 2.0 sy, 0.0 ni, 95.8 id, 0.0 wa, 0.3 hi, 0.0 si, 0.0 st

Mem Mem : 1961.3 total, 154.6 free, 1199.6 used, 776.1 buff/cache

Mem Swap: 2092.0 total, 2091.7 free, 0.3 used, 761.6 avail Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME	COMMAND
3138	eragdzh+	20	0	1914472	301840	96844	S	3.0	15.0	0:01.92	ptxvis
2039	eragdzh+	20	0	4089512	300676	119780	S	2.3	15.0	0:03.03	gnome-shell
33	root	20	0	0	0	0	I	0.3	0.0	0:00.09	kworker/u10:1-events_unbound
36	root	20	0	0	0	0	I	0.3	0.0	0:00.08	kworker/u9:1-events_unbound
1105	root	20	0	574184	2376	2120	S	0.3	0.1	0:00.16	VBoxDRMClient
3948	eragdzh+	20	0	231612	5544	3368	R	0.3	0.3	0:00.01	top
1	root	20	0	49192	40216	9324	S	0.0	2.0	0:01.50	systemd
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	0:00.04	kworker/0:0-ata_sff
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
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	0:00.03	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	0:00.03	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.07	rcu_preempt
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_exp_par_gp_kthread_worker/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.01	rcu_exp_gp_kthread_worker
21	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	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	cpuhp/0
24	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/1
25	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/1
26	root	rt	0	0	0	0	S	0.0	0.0	0:00.14	migration/1
27	root	20	0	0	0	0	S	0.0	0.0	0:00.03	ksoftirqd/1

Рис. 2.2: Работа с заданиями в фоне и переднем плане

- В отдельном терминале, под своей учётной записью, был запущен процесс `dd if=/dev/zero of=/dev/null &`, затем сессия завершена командой `exit`.
- В другом терминале выполнена команда `top`, которая показала активный процесс `dd` (см. рис. fig. 2.3).
- С помощью встроенной команды `k` в `top` процесс `dd` был завершён, после чего программа `top` закрыта.

top - 15:23:39 up 0 min, 3 users, load average: 0.39, 0.08, 0.03										
Tasks: 247 total, 1 running, 246 sleeping, 0 stopped, 0 zombie										
%Cpu(s): 4.8 us, 4.8 sy, 0.0 ni, 90.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st										
MiB Mem : 1961.3 total, 430.4 free, 1124.0 used, 566.0 buff/cache										
MiB Swap: 2092.0 total, 2092.0 free, 0.0 used, 837.2 avail Mem										
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
2939	eragdzh*	20	0	231616	4900	2852	R	8.3	0.2	0:00.01 top
1	root	20	0	49192	41136	10160	S	0.0	2.0	0:00.75 systemd
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	0:00.02 kworker/0:0-xfs-inodegc/dm-0
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00 kworker/0:1-ata_sff
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/0:0H-events_highpri
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	0:00.00 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	0:00.00 ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.00 rcu_preempt
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00 rcu_exp_par_gp_kthread_worker/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00 rcu_exp_gp_kthread_worker
21	root	rt	0	0	0	0	S	0.0	0.0	0:00.00 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 cpuhp/0
24	root	20	0	0	0	0	S	0.0	0.0	0:00.00 cpuhp/1
25	root	-51	0	0	0	0	S	0.0	0.0	0:00.00 idle_inject/1
26	root	rt	0	0	0	0	S	0.0	0.0	0:00.13 migration/1
27	root	20	0	0	0	0	S	0.0	0.0	0:00.01 ksoftirqd/1
28	root	20	0	0	0	0	I	0.0	0.0	0:00.00 kworker/1:0-cgroup_destroy
29	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/1:0H-events_highpri

Рис. 2.3: Мониторинг процессов в top

2.2 Управление процессами

1. Получены права администратора с помощью команды `su`.
2. Запущены три процесса с использованием команды `dd if=/dev/zero of=/dev/null &`.
3. Для просмотра информации о запущенных процессах применена команда `ps aux | grep dd`, которая вывела список процессов, содержащих подстроку **dd**. В нём видны три запущенных процесса `dd` (см. рис. fig. 2.4).

```

root@eragdzhabekova:/home/eragdzhabekova#
root@eragdzhabekova:/home/eragdzhabekova# dd if=/dev/zero of=/dev/null &
[1] 3058
root@eragdzhabekova:/home/eragdzhabekova# dd if=/dev/zero of=/dev/null &
[2] 3060
root@eragdzhabekova:/home/eragdzhabekova# dd if=/dev/zero of=/dev/null &
[3] 3062
root@eragdzhabekova:/home/eragdzhabekova# ps aux | grep dd
root      2  0.0  0.0      0  0 ?        S   15:23   0:00 [kthreadd]
root     73  0.0  0.0      0  0 ?        I<  15:23   0:00 [kworker/R-ipv6_addrconf]
root    1098  0.0  0.1 512956 3388 ?        Sl  15:23   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
eragdzh+ 2407  0.0  1.2 1036400 25528 ?       Ssl 15:23   0:00 /usr/libexec/evolution-addressbook-factory
root    3058 64.1  0.0 226848 1824 pts/0    R   15:24   0:05 dd if=/dev/zero of=/dev/null
root    3060 59.7  0.0 226848 1900 pts/0    R   15:24   0:04 dd if=/dev/zero of=/dev/null
root    3062 56.3  0.0 226848 1784 pts/0    R   15:24   0:04 dd if=/dev/zero of=/dev/null
root    3087  0.0  0.1 227688 2040 pts/0    S+  15:24   0:00 grep --color=auto dd
root@eragdzhabekova:/home/eragdzhabekova# renice -n 5 3058
3058 (process ID) old priority 0, new priority 5
root@eragdzhabekova:/home/eragdzhabekova#
root@eragdzhabekova:/home/eragdzhabekova# ps fax | grep dd
? ? ?      S      0:00 [kthreadd]
? ? ?      I<    0:00 \_ [kworker/R-ipv6_addrconf]
1098 ?      Sl     0:00 \_ /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
2407 ?      Ssl    0:00 \_ /usr/libexec/evolution-addressbook-factory
3058 pts/0  RN     0:19 |      \_ dd if=/dev/zero of=/dev/null
3060 pts/0  R      0:29 |      \_ dd if=/dev/zero of=/dev/null
3062 pts/0  R      0:27 |      \_ dd if=/dev/zero of=/dev/null
3178 pts/0  S+     0:00 |      \_ grep --color=auto dd
root@eragdzhabekova:/home/eragdzhabekova#

```

Рис. 2.4: Просмотр процессов с помощью ps aux

4. Приоритет одного из процессов изменён командой `renice -n 5 <PID>`, где <PID> — идентификатор одного из процессов dd. В результате его приоритет был изменён с 0 на 5.
5. Для отображения иерархии процессов применена команда `ps fax | grep -B5 dd`. На выходе видно дерево процессов, в котором процессы dd являются дочерними по отношению к оболочке root (см. рис. fig. 2.5).

```

1098 ?      Sl     0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
--
2343 ?      Ssl    0:00 \_ /usr/libexec/gvfs-udisks2-volume-monitor
2353 ?      Ssl    0:00 \_ /usr/libexec/goa-identity-service
2357 ?      Ssl    0:00 \_ /usr/bin/gjs -m /usr/share/gnome-shell/org.gnome.ScreenSaver
2378 ?      Ssl    0:00 \_ /usr/libexec/gvfs-mtp-volume-monitor
2404 ?      Ssl    0:00 \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2407 ?      Ssl    0:00 \_ /usr/libexec/evolution-addressbook-factory
--
2846 ?      Ssl    0:01 \_ /usr/bin/ptxix --gapplication-service
2853 ?      Ssl    0:00 | \_ /usr/libexec/ptxix-agent --socket-fd=3
2910 pts/0  Ss     0:00 |      \_ /usr/bin/bash
2988 pts/0  S      0:00 |          \_ su
3021 pts/0  S      0:00 |              \_ bash
3058 pts/0  RN     0:31 |                  \_ dd if=/dev/zero of=/dev/null
3060 pts/0  R      1:03 |                  \_ dd if=/dev/zero of=/dev/null
3062 pts/0  R      1:02 |                  \_ dd if=/dev/zero of=/dev/null
3264 pts/0  R+     0:00 |                      \_ ps fax
3265 pts/0  S+     0:00 |                      \_ grep --color=auto -B5 dd
root@eragdzhabekova:/home/eragdzhabekova# kill -9 2910

```

Рис. 2.5: Отображение иерархии процессов

6. Определён PID родительской оболочки, из которой были запущены процессы dd. Командой

`kill -9 <PID>` оболочка завершена, вследствие чего завершились и все её дочерние процессы. Такой способ позволяет быстро остановить группу связанных процессов.

2.3 Изменение приоритета процессов

1. Запущены три процесса командой

`dd if=/dev/zero of=/dev/null &` в фоновом режиме.

2. Приоритет одного из процессов изменён командой

`renice -n -5 <PID>`, что повысило его приоритет относительно остальных (см. рис. fig. 2.6).

3. Тот же процесс был повторно изменён с использованием команды

`renice -n -15 <PID>`. В отличие от предыдущего изменения, значение `-15` ещё больше повышает приоритет выполнения процесса по сравнению с другими заданиями в системе. Чем **ниже числовое значение `nice`**, тем выше приоритет.

4. Все процессы `dd` были завершены командой

`killall dd` (см. рис. fig. 2.7).

```

root@eragdzhbekova:/home/eragdzhbekova#
root@eragdzhbekova:/home/eragdzhbekova# dd if=/dev/zero of=/dev/null &
[1] 3505
root@eragdzhbekova:/home/eragdzhbekova# dd if=/dev/zero of=/dev/null &
[2] 3517
root@eragdzhbekova:/home/eragdzhbekova# dd if=/dev/zero of=/dev/null &
[3] 3519
root@eragdzhbekova:/home/eragdzhbekova# renice -n 5 3517
3517 (process ID) old priority 0, new priority 5
root@eragdzhbekova:/home/eragdzhbekova# renice -n 15 3517
3517 (process ID) old priority 5, new priority 15
root@eragdzhbekova:/home/eragdzhbekova# killall dd
[1] Terminated dd if=/dev/zero of=/dev/null
[2]- Terminated dd if=/dev/zero of=/dev/null
[3]+ Terminated dd if=/dev/zero of=/dev/null
root@eragdzhbekova:/home/eragdzhbekova# █

```

Рис. 2.6: Изменение приоритета и завершение процессов

2.4 Работа с процессами и приоритетами (команда yes)

1. Программа yes была запущена в фоновом режиме с перенаправлением вывода в /dev/null.
2. Та же программа была запущена на переднем плане, затем приостановлена с помощью Ctrl+Z, возвращена в выполнение и остановлена комбинацией Ctrl+C (см. рис. fig. ??).
3. Аналогичные действия были выполнены для варианта запуска без перенаправления потока вывода.
4. С помощью команды jobs проверено состояние заданий.
5. Один из фоновых процессов был возвращён на передний план (fg) и остановлен.
6. Другой процесс был переведён в фоновый режим (bg) и отмечен как **Running** (см. рис. fig. ??).

7. Процесс был запущен с использованием `nohup`, что позволило ему продолжать выполнение даже после выхода из терминала.

```
root@eragdzhabekova:/home/eragdzhabekova# jobs
[1]+  Running                  yes > /dev/null &
root@eragdzhabekova:/home/eragdzhabekova# fg 1
yes > /dev/null
^Z
[1]+  Stopped                  yes > /dev/null
root@eragdzhabekova:/home/eragdzhabekova# jobs
[1]+  Stopped                  yes > /dev/null
root@eragdzhabekova:/home/eragdzhabekova# bg 1
[1]+ yes > /dev/null &
root@eragdzhabekova:/home/eragdzhabekova# jobs
[1]+  Running                  yes > /dev/null &
root@eragdzhabekova:/home/eragdzhabekova# nohup
nohup: ignoring input and redirecting stderr to stdout
root@eragdzhabekova:/home/eragdzhabekova#
```

8. В утилите `top` было подтверждено выполнение процесса `yes`, занимающего значительную долю CPU (см. рис. fig. 2.7).

```
top - 15:34:01 up 10 min, 5 users, load average: 1.19, 0.99, 0.66
Tasks: 234 total, 3 running, 231 sleeping, 0 stopped, 0 zombie
%Cpu(s): 33.3 us, 66.7 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 ht, 0.0 st, 0.0 sr
MiB Mem : 1961.3 total, 227.0 free, 1090.3 used, 810.0 buff/cache
MiB Swap: 2092.0 total, 2091.5 free, 0.5 used, 871.0 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4394	root	20	0	226820	1752	1752	R	90.9	0.1	0:27.82	yes
4105	root	20	0	226820	1796	1796	R	81.8	0.1	2:29.30	yes
1	root	20	0	49192	39024	7920	S	0.0	1.9	0:01.31	systemd
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	0:00.06	kworker/0:0-events
9	root	20	0	0	0	0	I	0.0	0.0	0:00.06	kworker/0:1-cgroup_destroy
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
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	0:00.02	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	0:00.02	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.07	rcu_preempt
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_exp_par_gp_kthread_worker/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.01	rcu_exp_gp_kthread_worker
21	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_exp_gp_kthread_worker

Рис. 2.7: Отслеживание процесса `yes` в `top`

9. В фоновом режиме были запущены ещё три процесса `yes` с перенаправлением вывода. Два из них были завершены разными способами:

- по PID (`kill -9 <PID>`),

- по идентификатору задания (`kill -l <job_id>`).

10. Процесс, запущенный с `nohup`, корректно обработал сигнал `SIGHUP`, продолжая работу, в отличие от обычного процесса, который был завершён (см. рис. fig. 2.8).

```

root@eragdzhabekova:/home/eragdzhabekova# jobs
[3]+  Running                  yes > /dev/null &
root@eragdzhabekova:/home/eragdzhabekova# kill -1 4618
[3]+  Hangup                   yes > /dev/null
root@eragdzhabekova:/home/eragdzhabekova# kill -1 4394
root@eragdzhabekova:/home/eragdzhabekova# kill -1 4105
root@eragdzhabekova:/home/eragdzhabekova# yes > /dev/null &
[1] 4753
root@eragdzhabekova:/home/eragdzhabekova# yes > /dev/null &
[2] 4755
root@eragdzhabekova:/home/eragdzhabekova# yes > /dev/null &
[3] 4757
root@eragdzhabekova:/home/eragdzhabekova# killall yes
[1] Terminated                yes > /dev/null
[3]+ Terminated                yes > /dev/null
[2]+ Terminated                yes > /dev/null
root@eragdzhabekova:/home/eragdzhabekova# █

```

Рис. 2.8: Завершение процессов `yes`

11. Все запущенные процессы `yes` были завершены одновременно с помощью команды `killall yes`.
12. Для сравнения приоритетов одна программа `yes` была запущена в фоне с обычным приоритетом, а вторая — с помощью команды `nice -n 5`, что задало ей более низкий приоритет. Проверка с помощью `ps -l` показала различие значений **NI** и **PRI**.
13. Командой `renice` приоритет одного процесса был изменён, в результате чего оба процесса получили одинаковый приоритет (см. рис. fig. 2.9).

```

root@eragdzhabekova:/home/eragdzhabekova# yes > /dev/null &
[1] 4834
root@eragdzhabekova:/home/eragdzhabekova# nice -n 5 yes > /dev/null &
[2] 4856
root@eragdzhabekova:/home/eragdzhabekova# ps -l
F S  UID      PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        4522    4486  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        4533    4522  0  80   0 - 57543 do_wai pts/0      00:00:00 bash
4 R   0        4834    4533  98  80   0 - 56705 -      pts/0      00:00:10 yes
4 R   0        4856    4533  95  85   5 - 56705 -      pts/0      00:00:03 yes
4 R   0        4858    4533  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@eragdzhabekova:/home/eragdzhabekova# renice -n 5 4834
4834 (process ID) old priority 0, new priority 5
root@eragdzhabekova:/home/eragdzhabekova# ps -l
F S  UID      PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        4522    4486  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        4533    4522  0  80   0 - 57543 do_wai pts/0      00:00:00 bash
4 R   0        4834    4533  98  85   5 - 56705 -      pts/0      00:00:29 yes
4 R   0        4856    4533  97  85   5 - 56705 -      pts/0      00:00:22 yes
4 R   0        4903    4533  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@eragdzhabekova:/home/eragdzhabekova# killall yes
[2]+  Terminated                  nice -n 5 yes > /dev/null
[1]+  Terminated                  yes > /dev/null
root@eragdzhabekova:/home/eragdzhabekova#

```

Рис. 2.9: Изменение приоритета процессов yes

3 Контрольные вопросы

1. **Какая команда даёт обзор всех текущих заданий оболочки?**

Команда: `jobs`.

2. **Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?**

Комбинация клавиш `Ctrl+Z` останавливает выполнение, а затем командой `bg` можно продолжить работу в фоне.

3. **Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?**

Комбинация: `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?**

Использовать команду `kill <PID>` или `killall <имя_процесса>` в другой оболочке/терминале.

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

Команда: `ps fax`.

6. **Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?**

Команда: `renice -n -5 -p 1234`. (чем меньше значение `nice`, тем выше приоритет).

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

Команда: `killall dd`.

8. **Какая команда позволяет остановить команду с именем `mycommand`?**

Команда: `killall mycommand`.

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

В `top` используется клавиша `k`, затем вводится PID процесса.

10. **Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?**

Использовать команду `nice` с положительным значением:

`nice -n 10 <команда>` — это снижает приоритет процесса и уменьшает нагрузку на систему.

4 Заключение

В ходе лабораторной работы были изучены основные приёмы управления заданиями и процессами в Linux. Освоены способы запуска процессов в фоновом и переднем режиме, их приостановки, возобновления и завершения. Получены практические навыки использования команд `jobs`, `fg`, `bg`, `kill`, `killall`, `ps`, `top`, а также изменения приоритетов процессов с помощью `nice` и `renice`. Закреплены умения отслеживать взаимосвязь родительских и дочерних процессов и контролировать использование системных ресурсов.