

Отчёт по лабораторной работе №6

Управление процессами

Эзиз Хатамов

Содержание

1	Цель работы	5
2	Отчёт по выполнению работы	6
2.1	Управление заданиями	6
2.2	Управление процессами	9
2.3	Задание 1. Управление приоритетами процессов	10
2.4	Задание 2. Работа с процессами и заданиями	12
3	Контрольные вопросы	17
4	Заключение	19

Список иллюстраций

2.1	Управление заданиями	7
2.2	Фоновый процесс	7
2.3	Отображение фонового процесса dd в top	8
2.4	Завершение работы процесса dd в top	8
2.5	Список процессов dd	9
2.6	Завершение родительской оболочки и дочерних процессов dd . . .	10
2.7	Управление приоритетами и завершение процессов dd	11
2.8	Работа с процессом yes на переднем плане	12
2.9	Запуск процесса yes с nohup	13
2.10	Процессы yes в top	14
2.11	Завершение процессов yes	15
2.12	Изменение приоритетов процессов yes	16

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Отчёт по выполнению работы

2.1 Управление заданиями

1. Пользователь получил полномочия администратора с помощью команды `su -`.
2. В фоновом режиме были запущены команды:
 - `sleep 3600 &`
 - `dd if=/dev/zero of=/dev/null &`Дополнительно была запущена команда `sleep 7200` без амперсанда.
3. Так как команда `sleep 7200` выполнялась на переднем плане, для остановки её работы использовалось сочетание клавиш **Ctrl+Z**.
4. Команда `jobs` показала три активных задания:
 - задание 1 и 2 находились в состоянии **Running**,
 - задание 3 — в состоянии **Stopped**.
5. Для возобновления выполнения задания 3 в фоновом режиме была выполнена команда `bg 3`.
Статус изменился на **Running**.
6. Перемещение задания 1 на передний план осуществлялось командой `fg 1`.
Далее оно было прервано комбинацией **Ctrl+C**.

```

ehatamov@ehatamov:~$ su
Password:
root@ehatamov:/home/ehatamov# sleep 3600 &
[1] 5796
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/null &
[2] 5837
root@ehatamov:/home/ehatamov# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@ehatamov:/home/ehatamov# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@ehatamov:/home/ehatamov# bg 3
[3]+ sleep 7200 &
root@ehatamov:/home/ehatamov# fg 1
sleep 3600
^C
root@ehatamov:/home/ehatamov# fg 2
dd if=/dev/zero of=/dev/null
^C78433936+0 records in
78433935+0 records out
40158174720 bytes (40 GB, 37 GiB) copied, 58.1704 s, 690 MB/s

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

```

Рис. 2.1: Управление заданиями

7. Аналогично задания 2 и 3 были выведены на передний план (fg 2, fg 3) и прерваны комбинацией **Ctrl+C**.

8. В другом терминале под обычным пользователем было запущено `dd if=/dev/zero of=/dev/null &`.

Процесс успешно стартовал в фоне.

```

ehatamov@ehatamov:~$ dd if=/dev/zero of=/dev/zero &
[1] 6169
ehatamov@ehatamov:~$ █

```

Рис. 2.2: Фоновый процесс

9. После выхода из второго терминала команда `top` в другом окне показала,

что процесс **dd** продолжает выполняться.

```
top - 13:04:00 up 24 min, 4 users, load average: 0.43, 0.26, 0.16
Tasks: 264 total, 2 running, 262 sleeping, 0 stopped, 0 zombie
%Cpu(s): 8.7 us, 9.2 sy, 0.0 ni, 81.7 id, 0.0 wa, 0.4 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1425.4 free, 1256.9 used, 1469.6 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2652.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6169	ehatamov	20	0	226848	1896	1896	R	92.0	0.0	0:20.82	dd
5573	ehatamov	20	0	3031136	248160	100228	S	12.0	6.2	0:02.00	ptyxis
2129	ehatamov	20	0	5041252	313580	123204	S	8.0	7.8	0:04.05	gnome-shell
88	root	20	0	0	0	0	I	2.7	0.0	0:00.14	kworker/u18:2-events_unbound
2043	ehatamov	20	0	6612	4108	2188	S	0.3	0.1	0:00.06	dbus-broker
6240	ehatamov	20	0	231608	5388	3212	R	0.3	0.1	0:00.01	top
1	root	20	0	49192	41184	10208	S	0.0	1.0	0:02.56	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
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
12	root	20	0	0	0	0	I	0.0	0.0	0:00.10	kworker/u16:1-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	0:00.01	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.16	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	0:00.05	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

Рис. 2.3: Отображение фонового процесса dd в top

10. Для завершения процесса в top использовалась клавиша **k**, после чего задание **dd** было остановлено.

```
top - 13:04:15 up 24 min, 4 users, load average: 0.63, 0.32, 0.18
Tasks: 263 total, 2 running, 261 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.5 us, 6.0 sy, 0.0 ni, 89.2 id, 0.0 wa, 0.3 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1424.2 free, 1257.9 used, 1469.7 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2651.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5573	ehatamov	20	0	3033428	251488	100228	R	3.1	6.3	0:02.28	ptyxis
2129	ehatamov	20	0	5041236	313708	123204	S	0.8	7.8	0:04.31	gnome-shell
1	root	20	0	49192	41184	10208	S	0.0	1.0	0:02.57	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
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
12	root	20	0	0	0	0	I	0.0	0.0	0:00.10	kworker/u16:1-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	0:00.16	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	0:00.05	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

Рис. 2.4: Завершение работы процесса dd в top

2.2 Управление процессами

1. Пользователь получил полномочия администратора с помощью команды `su -`.

2. В фоновом режиме были запущены три процесса:

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

3. Для проверки списка запущенных процессов использовалась команда `ps aux | grep dd`.

В результате были отображены три активных процесса **dd**.

```
root@ehatamov:/home/ehatamov#  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/zero &  
[1] 6594  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/zero &  
[2] 6606  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/zero &  
[3] 6618  
root@ehatamov:/home/ehatamov# ps aux | grep dd  
root      2  0.0  0.0   0   0 ?        S   12:39   0:00 [kthreadd]  
root     91  0.0  0.0   0   0 ?        I<  12:39   0:00 [kworker/R-ipv6_addrconf]  
root    1145  0.0  0.0 512956 2880 ?        Sl  12:39   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh  
ehatamov 2538  0.0  0.6 962676 25584 ?        Ssl 12:40   0:00 /usr/libexec/evolution-addressbook-factory  
root     6594 91.3  0.0 226848 1760 pts/0    R   13:06   0:59 dd if=/dev/zero of=/dev/zero  
root     6606 91.4  0.0 226848 1808 pts/0    R   13:06   0:54 dd if=/dev/zero of=/dev/zero  
root     6618 91.6  0.0 226848 1760 pts/0    R   13:06   0:52 dd if=/dev/zero of=/dev/zero  
root     6736  0.0  0.0 227688 2028 pts/0    S+  13:07   0:00 grep --color=auto dd  
root@ehatamov:/home/ehatamov# renice -n 5 6618  
6618 (process ID) old priority 0, new priority 5  
root@ehatamov:/home/ehatamov#
```

Рис. 2.5: Список процессов dd

4. Приоритет одного из процессов был изменён с помощью команды `renice -n 5 <PID>`.

В отчёте видно, что приоритет процесса с PID 6618 изменился с 0 на 5.

5. Для отображения иерархии процессов была использована команда `ps fax | grep -B5 dd`.

Она показала дерево процессов, включая родительскую оболочку, из которой были запущены все процессы **dd**.

6. Используя найденный PID родительской оболочки, была выполнена команда `kill -9 <PID>`.

В результате завершилась корневая оболочка и автоматически остановились все дочерние процессы **dd**.

```

--
 85 ?      I<    0:00  \_ [kworker/R-acpi_thermal_pm]
 86 ?      I<    0:00  \_ [kworker/R-kmpath_rdacd]
 87 ?      I<    0:00  \_ [kworker/R-kalud]
 88 ?      I    0:00  \_ [kworker/u18:2-events_unbound]
 90 ?      I<    0:00  \_ [kworker/R-mld]
 91 ?      I<    0:00  \_ [kworker/R-ipv6_addrconf]

--
908 ?      S    0:00  /usr/sbin/chrond -F 2
926 ?      SNs   0:00  /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --initfile=/lib/alsa/init/00m
ain rdaemon
951 ?      Ssl   0:00  /usr/sbin/ModemManager
954 ?      Ssl   0:00  /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid
1143 ?     SI    0:00  /usr/bin/VBoxDRMClient
1145 ?     SI    0:00  /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh

--
2456 ?     Ssl   0:00  \_ /usr/libexec/ibus-portal
2468 ?     Ssl   0:00  \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2491 ?     Ssl   0:00  \_ /usr/libexec/evolution-calendar-factory
2494 ?     Ssl   0:00  \_ /usr/libexec/gvfs-goa-volume-monitor
2497 ?     Ssl   0:00  \_ /usr/libexec/goa-identity-service
2538 ?     Ssl   0:00  \_ /usr/libexec/evolution-addressbook-factory

--
5573 ?     Rsl   0:04  \_ /usr/bin/ptxis --gapapplication-service
5581 ?     Ssl   0:00  | \_ /usr/libexec/ptxis-agent --socket-fd=3
5650 pts/0  Ss    0:00  | \_ /usr/bin/bash
5693 pts/0  S    0:00  | | \_ su
5725 pts/0  S    0:00  | | \_ bash
6594 pts/0  R    1:45  | | \_ dd if=/dev/zero of=/dev/zero
6606 pts/0  R    1:39  | | \_ dd if=/dev/zero of=/dev/zero
6618 pts/0  RN    1:38  | | \_ dd if=/dev/zero of=/dev/zero
6843 pts/0  R+   0:00  | | \_ ps fax
6844 pts/0  S+   0:00  | | \_ grep --color=auto -B5 dd
root@ehatamov:/home/ehatamov# kill -9 5650
Hangup

```

Рис. 2.6: Завершение родительской оболочки и дочерних процессов dd

2.3 Задание 1. Управление приоритетами процессов

1. Пользователь получил права администратора с помощью команды `su`.
2. В фоновом режиме были запущены три процесса:

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

Каждый процесс получил собственный PID.

3. Приоритет одного из процессов был изменён командой `renice -n 5 <PID>`.
В результате его приоритет изменился с **0** на **5**.

4. Далее приоритет того же процесса был снова изменён:

- команда `renice -n 15 <PID>` повысила значение приоритета до **15**.

Разница заключается в том, что:

- **меньшее значение nice (например, -5)** повышает приоритет процесса, предоставляя ему больше процессорного времени;
- **большее значение nice (например, 15)** понижает приоритет, снижая доступное процессорное время.

5. Для завершения всех запущенных процессов **dd** использовалась команда `killall dd`.

Все три процесса были успешно остановлены.

```
enatamov@enatamov:~$  
ehatamov@ehatamov:~$ su  
Password:  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/null &  
[1] 7138  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/null &  
[2] 7140  
root@ehatamov:/home/ehatamov# dd if=/dev/zero of=/dev/null &  
[3] 7142  
root@ehatamov:/home/ehatamov# renice -n 5 7138  
7138 (process ID) old priority 0, new priority 5  
root@ehatamov:/home/ehatamov# renice -n 15 7138  
7138 (process ID) old priority 5, new priority 15  
root@ehatamov:/home/ehatamov# killall dd  
[2]- Terminated          dd if=/dev/zero of=/dev/null  
[1]- Terminated          dd if=/dev/zero of=/dev/null  
[3]+ Terminated          dd if=/dev/zero of=/dev/null  
root@ehatamov:/home/ehatamov#
```

Рис. 2.7: Управление приоритетами и завершение процессов dd

2.4 Задание 2. Работа с процессами и заданиями

1. В фоновом режиме была запущена программа `yes` с подавлением потока вывода:

- `yes > /dev/null &`

2. Программа `yes` была запущена на переднем плане с перенаправлением вывода.

С помощью **Ctrl+Z** выполнение было приостановлено, затем процесс возобновлён и завершён.

3. Программа `yes` была запущена на переднем плане без подавления потока вывода.

После нескольких выводов символа **y** её выполнение было приостановлено (**Ctrl+Z**), возобновлено и завершено (**Ctrl+C**).

```
root@enatamov:/home/enatamov#
root@ehatamov:/home/ehatamov#
root@ehatamov:/home/ehatamov# yes > /dev/null &
[1] 7536
root@ehatamov:/home/ehatamov# yes > /dev/null
^Z
[2]+  Stopped                  yes > /dev/null
root@ehatamov:/home/ehatamov# yes > /dev/null
^C
root@ehatamov:/home/ehatamov# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@ehatamov:/home/ehatamov# █
```

Рис. 2.8: Работа с процессом `yes` на переднем плане

4. Проверка состояния заданий с помощью команды `jobs` показала активные и приостановленные процессы.
5. Один из процессов, выполнявшийся в фоне, был переведён на передний план командой `fg` и остановлен.

6. Процесс с подавлением вывода был переведён обратно в фоновый режим командой `bg`.
7. Повторная команда `jobs` подтвердила, что процесс находится в состоянии **Running** в фоне.
8. Для запуска процесса, продолжающего работу после выхода из терминала, использовалась команда:

- `nohup yes > /dev/null &`

```
y
y
y
y
y
y^C
root@ehatamov:/home/ehatamov#
root@ehatamov:/home/ehatamov# nohup yes > /dev/null &
[3] 7813
nohup: ignoring input and redirecting stderr to stdout
root@ehatamov:/home/ehatamov# jobs
[1]  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
[3]-  Running                  nohup yes > /dev/null &
root@ehatamov:/home/ehatamov# █
```

Рис. 2.9: Запуск процесса `yes` с `nohup`

9. После закрытия окна терминала процесс, запущенный через `nohup`, продолжил работу.
10. Для просмотра состояния процессов использовалась утилита `top`, где были видны процессы **yes**, нагружающие процессор.

ehatamov@ehatamov:/home/ehatamov - top										
top - 13:16:49 up 37 min, 5 users, load average: 1.66, 1.45, 0.99										
Tasks: 265 total, 3 running, 262 sleeping, 0 stopped, 0 zombie										
%Cpu(s): 27.3 us, 31.8 sy, 0.0 ni, 40.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st										
MiB Mem : 3909.0 total, 1279.5 free, 1392.4 used, 1480.2 buff/cache										
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2516.6 avail Mem										
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
7536	root	20	0	226820	1804	1804	R	90.9	0.0	3:17.48 yes
7813	root	20	0	226820	1708	1708	R	90.9	0.0	1:07.53 yes
2129	ehatamov	20	0	5110620	313980	123476	S	9.1	7.8	0:09.15 gnome-shell
5573	ehatamov	20	0	3036752	365340	100384	S	9.1	9.1	0:07.26 ptyxis
1	root	20	0	49192	41184	10208	S	0.0	1.0	0:03.18 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
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00 kworker/0:0H-events_highpri
12	root	20	0	0	0	0	I	0.0	0.0	0:00.14 kworker/u16:1-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	0:00.22 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	0:00.09 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	0:00.18 migration/1

Рис. 2.10: Процессы yes в top

11. Дополнительно были запущены три программы yes в фоновом режиме.

12. Два процесса были завершены разными способами:

- по PID с помощью команды `kill <PID>`;
- по идентификатору задания с помощью команды `kill %<номер_задания>`.

13. Была предпринята попытка отправить сигнал **SIGHUP (1)** процессу, запущенному с помощью `nohup`, и обычному процессу.

- процесс без `nohup` завершился;
- процесс с `nohup` продолжил работу.

```

-----
root@ehatamov:/home/ehatamov# yes > /dev/null &
[1] 8096
root@ehatamov:/home/ehatamov# yes > /dev/null &
[2] 8101
root@ehatamov:/home/ehatamov# yes > /dev/null &
[3] 8103
root@ehatamov:/home/ehatamov# kill 8103
root@ehatamov:/home/ehatamov# fg 1
yes > /dev/null
^C
[3] Terminated yes > /dev/null
root@ehatamov:/home/ehatamov#
root@ehatamov:/home/ehatamov# kill -1 8101
[2]+ Hangup yes > /dev/null
root@ehatamov:/home/ehatamov# kill -1 7813
root@ehatamov:/home/ehatamov# kill -1 7536
root@ehatamov:/home/ehatamov# yes > /dev/null &
[1] 8235
root@ehatamov:/home/ehatamov# yes > /dev/null &
[2] 8237
root@ehatamov:/home/ehatamov# yes > /dev/null &
[3] 8239
root@ehatamov:/home/ehatamov# killall yes
[2]- Terminated yes > /dev/null
[3]+ Terminated yes > /dev/null
[1]+ Terminated yes > /dev/null
root@ehatamov:/home/ehatamov# █

```

Рис. 2.11: Завершение процессов yes

14. Были запущены новые процессы yes в фоне.

15. Для их одновременного завершения использовалась команда:

- `killall yes`

16. Программа yes была запущена с разным приоритетом:

- `yes > /dev/null &` (обычный приоритет 0),
- `nice -n 5 yes > /dev/null &` (пониженный приоритет 5).

Сравнение в выводе `ps -l` показало различие в значениях **NI** (nice).

17. Приоритет одного из потоков был изменён командой `renice`, чтобы у обоих процессов он стал одинаковым.

```
root@ehatamov:/home/ehatamov#
root@ehatamov:/home/ehatamov# yes > /dev/null &
[1] 8411
root@ehatamov:/home/ehatamov# nice -n 5 yes > /dev/null &
[2] 8445
root@ehatamov:/home/ehatamov# ps -l | grep yes
4 R    0    8411    7977 99  80    0 - 56705 -    pts/2    00:00:17  yes
4 R    0    8445    7977 99  85    5 - 56705 -    pts/2    00:00:06  yes
root@ehatamov:/home/ehatamov# renice -n 5 8411
8411 (process ID) old priority 0, new priority 5
root@ehatamov:/home/ehatamov# ps -l | grep yes
4 R    0    8411    7977 99  85    5 - 56705 -    pts/2    00:00:30  yes
4 R    0    8445    7977 99  85    5 - 56705 -    pts/2    00:00:19  yes
root@ehatamov:/home/ehatamov# killall yes
[1]-  Terminated                  yes > /dev/null
[2]+  Terminated                  nice -n 5 yes > /dev/null
root@ehatamov:/home/ehatamov#
```

Рис. 2.12: Изменение приоритетов процессов `yes`

3 Контрольные вопросы

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

- `jobs`
- Пример: `jobs` — выводит список всех фоновых и приостановленных заданий.

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

- `Ctrl+Z` — приостановка выполнения.
- `bg` — продолжение выполнения в фоне.

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

- `Ctrl+C` — завершает текущее задание.

4. **Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой работает пользователь, невозможен. Что можно сделать?**

- Использовать команду `kill <PID>` в другой оболочке.
- Пример: `kill 1234` — завершает процесс с PID **1234**.

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

- `ps fax`
 - Пример: `ps fax | grep dd` — покажет дерево процессов с фильтрацией по **dd**.
6. **Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?**
- `renice -n -5 -p 1234` — повысить приоритет процесса.
7. **В системе в настоящее время запущено 20 процессов dd. Как проще всего остановить их все сразу?**
- `killall dd` — завершает все процессы с именем **dd**.
8. **Какая команда позволяет остановить команду с именем mycommand?**
- `killall mycommand`
9. **Какая команда используется в top, чтобы убить процесс?**
- В `top` нажать клавишу `k`, затем указать **PID** процесса.
10. **Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?**
- Использовать `nice` с положительным значением.
 - Пример: `nice -n 10 ./script.sh` — запускает команду с пониженным приоритетом, оставляя ресурсы другим процессам.

4 Заключение

В ходе работы были изучены команды для управления заданиями и процессами в Linux: `jobs`, `bg`, `fg`, `kill`, `killall`, `ps`, а также утилиты `top`, `nice` и `renice`. Эти инструменты позволяют контролировать выполнение процессов, управлять их приоритетами и корректно завершать работу в различных ситуациях.