

Отчёт по лабораторной работе №6

Управление процессами

Сулейм Гамбердов

Содержание

1	Цель работы	5
2	Ход выполнения	6
2.1	Управление заданиями	6
2.2	Управление процессами	10
2.3	Задание 1. Управление приоритетами процессов	12
2.4	Задание 2. Работа с программой уес и управлением процессами . .	13
3	Контрольные вопросы	18
4	Заключение	20

Список иллюстраций

2.1	Запуск фоновых и приостановленного процесса	7
2.2	Управление заданиями с помощью fg и завершение процессов . .	8
2.3	Запуск dd в фоновом режиме под обычным пользователем	8
2.4	Процесс dd в списке top	9
2.5	Завершение процесса dd из top	10
2.6	Запуск трёх процессов dd	11
2.7	Завершение процессов dd через остановку родительской оболочки	12
2.8	Запуск трёх процессов dd	13
2.9	Запуск и остановка программы yes	14
2.10	Состояния заданий в jobs	14
2.11	Процесс yes в списке top	15
2.12	Массовое завершение процессов yes	16
2.13	Изменение приоритета процессов yes	17

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Ход выполнения

2.1 Управление заданиями

1. С помощью команды *su* получены права суперпользователя.
2. В фоновом режиме запущены два задания:
 - команда **sleep 3600 &** создала процесс ожидания на 1 час,
 - команда **dd if=/dev/zero of=/dev/null &** инициировала бесконечную запись нулевых данных в устройство */dev/null*.

Третья команда **sleep 7200** (ожидание на 2 часа) была запущена без амперсанда, что привело к блокировке терминала.

```

sigamberdov@sigamberdov:~$ su
Password:
root@sigamberdov:/home/sigamberdov# sleep 3600 &
[1] 3461
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[2] 3500
root@sigamberdov:/home/sigamberdov# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@sigamberdov:/home/sigamberdov# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@sigamberdov:/home/sigamberdov# bg 3
[3]+ sleep 7200 &
root@sigamberdov:/home/sigamberdov# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@sigamberdov:/home/sigamberdov# █

```

Рис. 2.1: Запуск фоновых и приостановленного процесса

3. С помощью сочетания **Ctrl+Z** выполнение процесса *sleep 7200* было приостановлено, а команда *jobs* показала три активных задания:
 - два в состоянии *Running*,
 - одно в состоянии *Stopped*.
4. Для возобновления приостановленного задания в фоне использована команда **bg 3**. Повторный вызов *jobs* подтвердил, что все процессы выполняются в фоновом режиме.
5. Перемещение первого задания на передний план выполнено с помощью команды **fg 1**, после чего оно было завершено комбинацией **Ctrl+C**.
6. Аналогичным образом завершены задания 2 и 3 с использованием *fg* и прерывания **Ctrl+C**.

```

root@sigamberdov:/home/sigamberdov# fg 1
sleep 3600
^C
root@sigamberdov:/home/sigamberdov# jobs
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@sigamberdov:/home/sigamberdov# fg 2
dd if=/dev/zero of=/dev/null
^C189679744+0 records in
189679743+0 records out
97116028416 bytes (97 GB, 90 GiB) copied, 124.202 s, 782 MB/s

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

```

Рис. 2.2: Управление заданиями с помощью fg и завершение процессов

7. В новом терминале, под обычным пользователем, был запущен процесс **dd if=/dev/zero of=/dev/null &**.

```

sigamberdov@sigamberdov:~$ dd if=/dev/zero of=/dev/null &
[1] 3933
sigamberdov@sigamberdov:~$ █

```

Рис. 2.3: Запуск dd в фоновом режиме под обычным пользователем

8. После выхода из терминала процесс продолжил выполняться в системе. Проверка с помощью команды **top** показала активность процесса **dd**, полностью загружающего процессор.


```

top - 10:01:36 up 6 min, 4 users, load average: 0.63, 0.43, 0.19
Tasks: 261 total, 2 running, 259 sleeping, 0 stopped, 0 zombie
%Cpu(s): 8.1 us, 8.1 sy, 0.0 ni, 83.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1313.8 free, 1416.9 used, 1416.0 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2492.0 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3933	sigambe+	20	0	226848	1664	1664	R	100.0	0.0	0:13.40	dd
1	root	20	0	49192	41048	10080	S	0.0	1.0	0:01.60	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	0:00.00	kworker/0:0-rcu_gp
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/u16:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.02	kworker/u16:1-ipv6_addrconf
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	0:00.05	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	0:00.02	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	0:00.16	migration/1
27	root	20	0	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/1

Рис. 2.4: Процесс dd в списке top

9. Повторный запуск **top** и использование встроенной команды *k* позволили завершить процесс *dd*. После этого список процессов обновился, и задание исчезло.

```
top - 10:01:56 up 6 min, 4 users, load average: 0.73, 0.47, 0.21
Tasks: 259 total, 1 running, 258 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.4 us, 5.6 sy, 0.1 ni, 89.6 id, 0.0 wa, 0.4 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1285.3 free, 1445.1 used, 1416.4 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2463.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3240	sigambe+	20	0	3031132	314200	100024	S	2.6	7.8	0:03.38	ptxixis
2081	sigambe+	20	0	5042472	307396	122568	S	1.8	7.7	0:04.02	gnome-shell
586	root	20	0	0	0	0	S	0.6	0.0	0:00.12	xfsaild/dm-0
44	root	20	0	0	0	0	D	0.4	0.0	0:00.13	kworker/u19:0+events_unbound
3932	root	20	0	0	0	0	I	0.2	0.0	0:00.03	kworker/u19:4+events_unbound
1	root	20	0	49192	41048	10080	S	0.0	1.0	0:01.62	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	0:00.00	kworker/0:0-mm_percpu_wq
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/u16:0+events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.02	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	0:00.00	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.05	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	0:00.03	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

Рис. 2.5: Завершение процесса dd из top

2.2 Управление процессами

1. С помощью команды *su* получены права администратора.
2. В фоновом режиме запущены три ресурсоёмких процесса:

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

Каждый процесс выполняет бесконечную запись нулевых данных в устройство */dev/null*.

```
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[1] 4379
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[2] 4381
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[3] 4383
root@sigamberdov:/home/sigamberdov# ps aux | grep dd
root      2  0.0  0.0      0  0 ?        S   09:55   0:00 [kthreadd]
root     92  0.0  0.0      0  0 ?        I<  09:55   0:00 [kworker/R-ipv6_addrconf]
root    1135  0.0  0.0 512956 3084 ?        Sl   09:55   0:00 /usr/sbin/VBoxService --pidfile /var/run/
vboxadd-service.sh
sigambe+ 2478  0.0  0.6 1036400 24952 ?        Ssl  09:55   0:00 /usr/libexec/evolution-addressbook-factor
y
root     4379 99.5  0.0 226848 1664 pts/0    R   10:04   0:12 dd if=/dev/zero of=/dev/null
root     4381 99.2  0.0 226848 1780 pts/0    R   10:04   0:11 dd if=/dev/zero of=/dev/null
root     4383 98.8  0.0 226848 1900 pts/0    R   10:04   0:10 dd if=/dev/zero of=/dev/null
root     4414  0.0  0.0 227688 2036 pts/0    S+  10:04   0:00 grep --color=auto dd
root@sigamberdov:/home/sigamberdov# renice -n 5 4379
4379 (process ID) old priority 0, new priority 5
root@sigamberdov:/home/sigamberdov# ps fax | grep -B5 dd
  PID TTY          STAT       TIME COMMAND
    2 ?                S          0:00 [kthreadd]
  --
   87 ?                I<          0:00 \_ [kworker/R-kmpath_rdacd]
   88 ?                I<          0:00 \_ [kworker/R-kaluald]
   89 ?                I<          0:00 \_ [kworker/R-mld]
   90 ?                I          0:00 \_ [kworker/2:2-mm_percpu_wq]
   91 ?                I<          0:00 \_ [kworker/2:1H-xfs-log/dm-0]
   92 ?                I<          0:00 \_ [kworker/R-ipv6_addrconf]
```

Рис. 2.6: Запуск трёх процессов dd

3. Для отображения информации о запущенных процессах использована команда **ps aux | grep dd**.
В результате показаны строки, содержащие *dd*, среди которых последними идут активные процессы.
4. Для изменения приоритета одного из процессов применена команда **renice -n 5 <PID>**, где <PID> — идентификатор одного из процессов *dd*. Приоритет был изменён с 0 на 5.
5. С помощью команды **ps fax | grep -B5 dd** получено дерево процессов.
Так как *ps fax* выводит иерархию, стало видно, что все процессы *dd* являются дочерними по отношению к оболочке *bash*, запущенной от имени root.
6. Для завершения сразу всех процессов *dd* был найден PID родительской оболочки и выполнена команда **kill -9 <PID>**. После этого оболочка root закрылась, а вместе с ней завершились и все дочерние процессы *dd*.

```
Process Exited from Signal 9 Restart

92 ? I< 0:00 \_ [kworker/R-ipv6_addrconf]
--
898 ? S 0:00 /usr/sbin/chrond -F 2
909 ? SNs 0:00 /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --init
file=/lib/alsa/init/00main rdaemon
940 ? Ssl 0:00 /usr/sbin/ModemManager
943 ? Ssl 0:00 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid
1133 ? Sl 0:00 /usr/bin/VBoxDRMClient
1135 ? Sl 0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxaddr-service.sh
--
2402 ? Ssl 0:00 \_ /usr/libexec/evolution-calendar-factory
2404 ? Ssl 0:00 \_ /usr/libexec/gvfs-udisks2-volume-monitor
2407 ? Ssl 0:00 \_ /usr/libexec/goa-identity-service
2441 ? Ssl 0:00 \_ /usr/libexec/gvfs-mtp-volume-monitor
2463 ? Ssl 0:00 \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2478 ? Ssl 0:00 \_ /usr/libexec/evolution-addressbook-factory
--
3240 ? Ssl 0:05 \_ /usr/bin/ptxis --gapplication-service
3248 ? Ssl 0:00 | \_ /usr/libexec/ptxis-agent --socket-fd=3
3318 pts/0 Ss 0:00 | \_ /usr/bin/bash
3372 pts/0 S 0:00 | \_ su
3407 pts/0 S 0:00 | \_ bash
4379 pts/0 RN 0:37 | \_ dd if=/dev/zero of=/dev/null
4381 pts/0 R 0:36 | \_ dd if=/dev/zero of=/dev/null
4383 pts/0 R 0:36 | \_ dd if=/dev/zero of=/dev/null
4475 pts/0 R+ 0:00 | \_ ps fax
4476 pts/0 S+ 0:00 | \_ grep --color=auto -B5 dd

root@sigamberdov:/home/sigamberdov# kill -9 3318
root@sigamberdov:/home/sigamberdov#
Hangup
```

Рис. 2.7: Завершение процессов dd через остановку родительской оболочки

2.3 Задание 1. Управление приоритетами процессов

1. С помощью *su* получены права суперпользователя.
2. В фоновом режиме трижды запущена команда **dd if=/dev/zero of=/dev/null &**.

Каждый процесс выполняет бесконечную запись нулевых данных в устройство */dev/null*.

```

sigamberdov@sigamberdov:~$ su
Password:
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[1] 5226
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[2] 5228
root@sigamberdov:/home/sigamberdov# dd if=/dev/zero of=/dev/null &
[3] 5240
root@sigamberdov:/home/sigamberdov# renice -n 5 5226
5226 (process ID) old priority 0, new priority 5
root@sigamberdov:/home/sigamberdov# renice -n 15 5226
5226 (process ID) old priority 5, new priority 15
root@sigamberdov:/home/sigamberdov# killall dd
[2]-  Terminated          dd if=/dev/zero of=/dev/null
[3]+  Terminated          dd if=/dev/zero of=/dev/null
[1]+  Terminated          dd if=/dev/zero of=/dev/null
root@sigamberdov:/home/sigamberdov# █

```

Рис. 2.8: Запуск трёх процессов dd

3. Для одного из процессов изменён приоритет с 0 на 5 с помощью команды **renice -n 5 <PID>**.
4. При повторном изменении приоритета того же процесса команда **renice -n 15 <PID>** задала новое значение — 15.
Разница заключается в том, что большее положительное значение приоритета означает меньший приоритет выполнения.
Процесс с *nice=15* будет получать меньше процессорного времени по сравнению с процессами с более низким значением.
5. Для завершения всех запущенных процессов *dd* использована команда **killall dd**. Все процессы успешно остановлены.

2.4 Задание 2. Работа с программой **yes** и управлением процессами

1. В фоновом режиме запущена программа **yes** с перенаправлением вывода в */dev/null*.

2. Программа **yes** запущена на переднем плане с подавлением вывода. Выполнение было приостановлено (**Ctrl+Z**), затем процесс переведён в фоновый режим и завершён.

```
root@sigamberdov:/home/sigamberdov#
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[1] 5532
root@sigamberdov:/home/sigamberdov# yes > /dev/null
^Z
[2]+  Stopped                  yes > /dev/null
root@sigamberdov:/home/sigamberdov# yes > /dev/null
^C
root@sigamberdov:/home/sigamberdov# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@sigamberdov:/home/sigamberdov#
```

Рис. 2.9: Запуск и остановка программы yes

3. Программа **yes** запущена на переднем плане без подавления вывода. После приостановки и последующего возобновления процесс был завершён.
4. С помощью команды **jobs** проверены состояния заданий: одно выполняется в фоне, другое — остановлено.

```
root@sigamberdov:/home/sigamberdov#
root@sigamberdov:/home/sigamberdov# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@sigamberdov:/home/sigamberdov# fg 1
yes > /dev/null
^C
root@sigamberdov:/home/sigamberdov# bg 2
[2]+  yes > /dev/null &
root@sigamberdov:/home/sigamberdov# jobs
[2]+  Running                  yes > /dev/null &
root@sigamberdov:/home/sigamberdov# nohup yes > /dev/null &
[3] 5692
nohup: ignoring input and redirecting stderr to stdout
root@sigamberdov:/home/sigamberdov# jobs
[2]-  Running                  yes > /dev/null &
[3]+  Running                  nohup yes > /dev/null &
root@sigamberdov:/home/sigamberdov#
```

Рис. 2.10: Состояния заданий в jobs

5. Один из фоновых процессов был переведён на передний план командой **fg**,

после чего завершён (**Ctrl+C**).

6. Остановленный процесс был возобновлён в фоне с помощью команды **bg**. Теперь он выполняется в состоянии *Running*.
7. Для запуска процесса, который продолжит выполняться после выхода из терминала, использована команда **nohup yes > /dev/null &**.
8. После выхода и повторного входа в консоль проверка с помощью **top** показала, что процесс *yes* продолжает работать.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5544	root	20	0	226820	1676	1676	R	100.0	0.0	1:05.90	yes
5692	root	20	0	226820	1700	1700	R	90.9	0.0	0:42.86	yes
1	root	20	0	49192	41048	10080	S	0.0	1.0	0:02.37	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	0:00.05	kworker/0:0-events
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/u16:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.07	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	0:00.14	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
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	0:00.00	migration/1

Рис. 2.11: Процесс *yes* в списке **top**

9. Запущено ещё три процесса **yes > /dev/null &**. Два из них были завершены: один по PID командой **kill**, другой по номеру задания через **kill -<номер>**.
10. Для эксперимента с сигналами отправлен сигнал **SIGHUP**:
 - процессу, запущенному через *nohup* — он продолжил работу,

- обычному процессу — он завершился.

11. Вновь запущено несколько процессов **yes > /dev/null &**, после чего все они были завершены одной командой **killall yes**.

```
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[1] 6168
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[2] 6180
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[3] 6183
root@sigamberdov:/home/sigamberdov# kill 6168
[1] Terminated yes > /dev/null
root@sigamberdov:/home/sigamberdov# fg 2
yes > /dev/null
^C
root@sigamberdov:/home/sigamberdov# kill -1 6183
[3]+ Hangup yes > /dev/null
root@sigamberdov:/home/sigamberdov# kill -1 5544
root@sigamberdov:/home/sigamberdov# kill -1 5692
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[1] 6283
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[2] 6285
root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[3] 6287
root@sigamberdov:/home/sigamberdov# killall yes
[2]- Terminated yes > /dev/null
[1]- Terminated yes > /dev/null
[3]+ Terminated yes > /dev/null
root@sigamberdov:/home/sigamberdov#
```

Рис. 2.12: Массовое завершение процессов yes

12. Для сравнения приоритетов запущены два процесса:

- обычный **yes > /dev/null &**,
- с приоритетом **nice -n 5 yes > /dev/null &**.

С помощью команды **ps -l | grep yes** проверены их приоритеты (*PR* и *NI*). Видно, что второй процесс получил более низкий приоритет.

13. Командой **renice -n 5 <PID>** приоритет одного из процессов изменён так, чтобы оба процесса имели одинаковые значения.


```

root@sigamberdov:/home/sigamberdov# yes > /dev/null &
[1] 6377
root@sigamberdov:/home/sigamberdov# nice -n 5 yes > /dev/null &
[2] 6400
root@sigamberdov:/home/sigamberdov# ps -l | grep yes
4 R    0    6377    5841 99   80    0 - 56705 -      pts/1    00:00:23 yes
4 R    0    6400    5841 99   85    5 - 56705 -      pts/1    00:00:13 yes
root@sigamberdov:/home/sigamberdov# renice -n 5 6377
6377 (process ID) old priority 0, new priority 5
root@sigamberdov:/home/sigamberdov# ps -l | grep yes
4 R    0    6377    5841 99   85    5 - 56705 -      pts/1    00:00:40 yes
4 R    0    6400    5841 99   85    5 - 56705 -      pts/1    00:00:30 yes
root@sigamberdov:/home/sigamberdov# killall yes
[1]-  Terminated                  yes > /dev/null
[2]+  Terminated                  nice -n 5 yes > /dev/null
root@sigamberdov:/home/sigamberdov# █

```

Рис. 2.13: Изменение приоритета процессов yes

3 Контрольные вопросы

1. Для получения обзора всех текущих заданий оболочки используется команда **jobs**.
Она показывает список процессов, запущенных в рамках текущей сессии, их состояния (*Running*, *Stopped*) и номера.
2. Чтобы остановить текущее задание и продолжить его выполнение в фоне, применяют комбинацию **Ctrl+Z**, а затем команду **bg <номер_задания>**.
3. Для отмены (прерывания) текущего задания используется комбинация **Ctrl+C**.
4. Если доступ к оболочке, в которой работает пользователь, невозможен, завершить задание можно через другой терминал, используя **kill <PID>** или **killall <имя_процесса>**.
5. Для отображения отношений между родительскими и дочерними процессами используется команда **ps fax** или её аналог **pstree**.
6. Чтобы изменить приоритет процесса с идентификатором 1234 на более высокий, используется команда:
renice -n -5 -p 1234
(отрицательное значение повышает приоритет процесса).
7. Если в системе запущено 20 процессов **dd**, проще всего завершить их всех одной командой:
killall dd

8. Чтобы остановить процесс с именем *mycommand*, применяется команда:
killall mycommand
9. В утилите **top** для завершения процесса используется клавиша **k**, после чего вводится PID процесса, который нужно завершить.
10. Чтобы запустить команду с высоким приоритетом, но при этом не перегружать систему, можно использовать утилиту **nice**, задав умеренно отрицательный приоритет. Например:
nice -n -5 команда
Это позволит процессу работать быстрее, но не лишит остальные процессы процессорного времени.

4 Заключение

В ходе выполнения лабораторной работы были освоены приёмы управления заданиями и процессами в Linux. Изучены способы перевода процессов между фоном и передним планом, завершения их работы разными методами, использования сигналов и изменения приоритетов с помощью *nice* и *renice*. Освоенные приёмы позволяют эффективно администрировать систему и контролировать ресурсы, обеспечивая стабильную работу даже при большом количестве процессов.