

Отчёт по лабораторной работе №6

Управление процессами

Максат Хемраев

Содержание

1	Цель работы	5
2	Отчёт по выполнению работы	6
2.1	Управление заданиями в Linux	6
2.2	Управление процессами в Linux	9
2.3	Задание 1. Управление приоритетами и завершением процессов .	11
2.4	Задание 2. Управление процессами с использованием программы <i>yesh</i>	12
3	Контрольные вопросы	16
3.0.1	1. Какая команда даёт обзор всех текущих заданий оболочки?	16
3.0.2	2. Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?	16
3.0.3	3. Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?	16
3.0.4	4. Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой в данный момент работает пользователь, невозможен. Что можно сделать, чтобы отменить задание? .	17
3.0.5	5. Какая команда используется для отображения отношений между родительскими и дочерними процессами?	17
3.0.6	6. Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?	17
3.0.7	7. В системе в настоящее время запущено 20 процессов <i>dd</i> . Как проще всего остановить их все сразу?	17
3.0.8	8. Какая команда позволяет остановить команду с именем <i>mysommand</i> ?	18
3.0.9	9. Какая команда используется в <i>top</i> , чтобы убить процесс? .	18
3.0.10	10. Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?	18
4	Заключение	19

Список иллюстраций

2.1	Запуск и приостановка процессов	7
2.2	Процесс dd в списке top	8
2.3	Завершение процесса dd в top	9
2.4	Список процессов dd	10
2.5	Иерархия процессов dd	10
2.6	Управление приоритетами и завершение процессов	12
2.7	Запуск и остановка yes	12
2.8	Состояния заданий	13
2.9	Работа nohup-процесса	14
2.10	Процесс yes в top	14
2.11	Завершение процессов yes	15

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Отчёт по выполнению работы

2.1 Управление заданиями в Linux

1. Вошёл в систему с полномочиями администратора с помощью команды **su**.

2. Запустил три задания:

- **sleep 3600** в фоновом режиме;
- **dd if=/dev/zero of=/dev/null** в фоновом режиме;
- **sleep 7200** — без указания **&**, поэтому оболочка ожидала завершения процесса.

Для приостановки последней команды применил комбинацию **Ctrl+Z**.

```

mhemraev@mhemraev:~$ su
Password:
root@mhemraev:/home/mhemraev# sleep 3600 &
[1] 3113
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[2] 3173
root@mhemraev:/home/mhemraev# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@mhemraev:/home/mhemraev# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@mhemraev:/home/mhemraev# bg 3
[3]+ sleep 7200 &
root@mhemraev:/home/mhemraev# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@mhemraev:/home/mhemraev# fg 1
sleep 3600
^C
root@mhemraev:/home/mhemraev# fg 2
dd if=/dev/zero of=/dev/null
^C106259624+0 records in
106259623+0 records out
54404926976 bytes (54 GB, 51 GiB) copied, 73.5491 s, 740 MB/s
root@mhemraev:/home/mhemraev# fg 3
sleep 7200
^C
root@mhemraev:/home/mhemraev# █

```

Рис. 2.1: Запуск и приостановка процессов

3. С помощью команды **jobs** проверил список заданий. Первые два процесса находились в состоянии *Running*, а третий — в состоянии *Stopped*.
4. Перевёл задание 3 в фоновый режим с помощью команды **bg 3**.
После повторной проверки с помощью **jobs** убедился, что все три задания работают в фоне.
5. Перенёс задание 1 на передний план командой **fg 1**, затем завершил его с помощью **Ctrl+C**. Аналогично были завершены задания 2 и 3.
6. В новом терминале под обычным пользователем запустил команду **dd if=/dev/zero of=/dev/null &**.
После этого закрыл терминал командой **exit**.
7. В другом терминале под своей учётной записью запустил мониторинг процессов с помощью утилиты **top**. В списке был виден процесс **dd**, который

продолжал выполняться.

```
mhemraev@mhemraev:~ - top
mhemraev@mhemraev:/home/mhemraev
top - 10:32:25 up 4 min, 4 users, load average: 0.38, 0.30, 0.12
Tasks: 239 total, 2 running, 237 sleeping, 0 stopped, 0 zombie
%Cpu(s): 19.0 us, 33.3 sy, 0.0 ni, 47.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1961.3 total, 73.4 free, 1251.6 used, 802.8 buff/cache
MiB Swap: 2092.0 total, 2091.7 free, 0.3 used, 709.7 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 3558 mhemraev   20   0 226848 1908 1908 R 83.3   0.1   0:11.29 dd
    1 root        20   0 49192 39232 8208 S  0.0   2.0   0:00.98 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   0:00.03 kworker/0:0-cgroup_destroy
    9 root        20   0      0      0      0 I  0.0   0.0   0:00.00 kworker/0:1-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/u8:0-events_unbound
   12 root        20   0      0      0      0 I  0.0   0.0   0:00.01 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   0:00.01 ksoftirqd/0
   18 root        20   0      0      0      0 I  0.0   0.0   0:00.02 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
```

Рис. 2.2: Процесс dd в списке top

- Повторно запустил **top** и завершил задание **dd** с помощью горячей клавиши **k**, указав PID процесса. После завершения процесса покинул утилиту командой **q**.


```

mhemraev@mhemraev:~ - top
mhemraev@mhemraev/home/mhemraev
mhemraev@mhemraev:~ - top

top - 10:32:53 up 5 min, 4 users, load average: 0.63, 0.37, 0.15
Tasks: 239 total, 2 running, 237 sleeping, 0 stopped, 0 zombie
%Cpu(s): 9.3 us, 15.2 sy, 0.0 ni, 75.3 id, 0.0 wa, 0.3 hi, 0.0 si, 0.0 st
MiB Mem : 1961.3 total, 90.2 free, 1270.3 used, 767.4 buff/cache
MiB Swap: 2092.0 total, 2091.7 free, 0.3 used, 691.0 avail Mem
PID to signal/kill [default pid = 3558] 3558

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3558	mhemraev	20	0	226848	1908	1908	R	99.7	0.1	0:39.56	dd
2879	mhemraev	20	0	1919212	243876	97824	S	0.7	12.1	0:02.25	ptxixis
2084	mhemraev	20	0	4027528	308348	119928	S	0.3	15.4	0:02.82	gnome-shell
3647	mhemraev	20	0	231596	5268	3220	R	0.3	0.3	0:00.01	top
1	root	20	0	49192	39232	8208	S	0.0	2.0	0:01.00	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	0:00.03	kworker/0:0-cgroup_destroy
9	root	20	0	0	0	0	I	0.0	0.0	0:00.01	kworker/0:1-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/u8:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.01	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	0:00.01	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.03	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

Рис. 2.3: Завершение процесса dd в top

2.2 Управление процессами в Linux

1. Получил полномочия администратора с помощью команды **su**.
2. Запустил три ресурсоёмких процесса в фоне:
 - **dd if=/dev/zero of=/dev/null &** (трижды).
3. Проверил список активных процессов, отфильтровав только те, что содержат **dd**, с помощью команды **ps aux | grep dd**. В выводе отобразились PID всех запущенных процессов **dd**.

```

root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[1] 3735
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[2] 3742
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[3] 3744
root@mhemraev:/home/mhemraev# ps aux | grep dd
root      2  0.0  0.0      0   0 ?        S    10:27   0:00 [kthreadd]
root     73  0.0  0.0      0   0 ?        I<   10:27   0:00 [kworker/R-ipv6_addrconf]
root    1109  0.0  0.1 512956 2496 ?        Sl   10:27   0:00 /usr/sbin/VBoxService --pidfile /var/run/vb
oxadd-service.sh
mhemraev 2499  0.0  1.1 962676 22108 ?        Ssl  10:28   0:00 /usr/libexec/evolution-addressbook-factory
root    3735 70.1  0.0 226848 1808 pts/0    R    10:33   0:08 dd if=/dev/zero of=/dev/null
root    3742 65.2  0.0 226848 1700 pts/0    R    10:33   0:07 dd if=/dev/zero of=/dev/null
root    3744 64.4  0.0 226848 1780 pts/0    R    10:33   0:06 dd if=/dev/zero of=/dev/null
root    3768  0.0  0.1 227688 2072 pts/0    S+   10:33   0:00 grep --color=auto dd
root@mhemraev:/home/mhemraev# renice -n 5 3744
3744 (process ID) old priority 0, new priority 5
root@mhemraev:/home/mhemraev#

```

Рис. 2.4: Список процессов dd

4. Изменил приоритет одного из процессов с помощью команды **renice -n 5 <PID>**, что позволило задать новый приоритет.
5. Выполнил команду **ps fax | grep -B5 dd**, чтобы просмотреть дерево процессов. В выводе отобразилась иерархия, где процессы **dd** были запущены из родительской оболочки **bash**, запущенной от имени суперпользователя.

```

--
2410 ?      Ssl      0:00 \_ /usr/libexec/gvfs-mtp-volume-monitor
2436 ?      Ssl      0:00 \_ /usr/libexec/goa-identity-service
2438 ?      Ssl      0:00 \_ /usr/libexec/evolution-calendar-factory
2441 ?      Ssl      0:00 \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2458 ?      Ssl      0:00 \_ /usr/libexec/gvfs-goa-volume-monitor
2499 ?      Ssl      0:00 \_ /usr/libexec/evolution-addressbook-factory
--
2879 ?      Ssl      0:03 \_ /usr/bin/ptxis --gaplication-service
2886 ?      Ssl      0:00 | \_ /usr/libexec/ptxis-agent --socket-fd=3
2940 pts/0    Ss       0:00 | \_ /usr/bin/bash
3007 pts/0    S        0:00 | | \_ su
3037 pts/0    S        0:00 | | \_ bash
3735 pts/0    R        1:18 | | \_ dd if=/dev/zero of=/dev/null
3742 pts/0    R        1:18 | | \_ dd if=/dev/zero of=/dev/null
3744 pts/0    RN       1:05 | | \_ dd if=/dev/zero of=/dev/null
3979 pts/0    R+       0:00 | | \_ ps fax
3980 pts/0    S+       0:00 | | \_ grep --color=auto -B5 dd
root@mhemraev:/home/mhemraev#

```

Рис. 2.5: Иерархия процессов dd

6. Определил PID корневой оболочки и завершил её вместе с дочерними процессами с помощью команды **kill -9 <PID>**. В результате оболочка закрылась, а все процессы **dd** были автоматически завершены.

2.3 Задание 1. Управление приоритетами и завершением процессов

1. Получил полномочия администратора с помощью команды **su**.
2. Запустил три фоновых процесса командой **dd if=/dev/zero of=/dev/null &**. Каждый процесс получил свой PID.
3. Изменил приоритет одного из процессов:
 - сначала задал значение **-5** с помощью команды **renice -n 5 <PID>**;
 - затем изменил тот же процесс на приоритет **-15**.

Разница заключается в том, что чем меньше (и более отрицательное) значение `nice`, тем выше приоритет выполнения процесса. При **-5** процесс получает больше ресурсов процессора по сравнению с обычными задачами, а при **-15** его приоритет становится ещё выше, что может повлиять на производительность остальных процессов системы.

4. Завершил все запущенные процессы **dd**, используя команду **kill <PID>** для каждого из них.

```

mhemraev@mhemraev:~$ su
Password:
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[1] 4396
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[2] 4408
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[3] 4410
root@mhemraev:/home/mhemraev# renice -n 5 4408
4408 (process ID) old priority 0, new priority 5
root@mhemraev:/home/mhemraev# renice -n 15 4408
4408 (process ID) old priority 5, new priority 15
root@mhemraev:/home/mhemraev# kill 4396
root@mhemraev:/home/mhemraev#
[1] Terminated dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev# kill 4408
[2]- Terminated dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# kill 4410
[3]+ Terminated dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# █

```

Рис. 2.6: Управление приоритетами и завершение процессов

2.4 Задание 2. Управление процессами с использованием программы **yes**

1. Запустил программу **yes** в фоновом режиме с перенаправлением вывода в **/dev/null**.
2. Запустил **yes** на переднем плане с подавлением вывода. Остановил выполнение комбинацией **Ctrl+Z**, затем вновь запустил и завершил процесс.

```

root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 4699
root@mhemraev:/home/mhemraev# yes > /dev/null
^Z
[2]+ Stopped yes > /dev/null
root@mhemraev:/home/mhemraev# yes > /dev/null
^C
root@mhemraev:/home/mhemraev# █

```

Рис. 2.7: Запуск и остановка **yes**

3. Запустил **yes** на переднем плане без подавления вывода. Приостановил

выполнение, затем продолжил и завершил процесс.

4. Проверил состояния процессов с помощью команды **jobs** — отобразились все текущие задания и их статусы (*Running* или *Stopped*).

```
root@mhemraev:/home/mhemraev# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@mhemraev:/home/mhemraev# fg 1
yes > /dev/null
^C
root@mhemraev:/home/mhemraev# jobs
[2]+  Stopped                  yes > /dev/null
root@mhemraev:/home/mhemraev# bg 2
[2]+  yes > /dev/null &
root@mhemraev:/home/mhemraev# jobs
[2]+  Running                  yes > /dev/null &
root@mhemraev:/home/mhemraev# nohup yes > /dev/null &
[3] 4944
nohup: ignoring input and redirecting stderr to stdout
root@mhemraev:/home/mhemraev# jobs
[2]-  Running                  yes > /dev/null &
[3]+  Running                  nohup yes > /dev/null &
root@mhemraev:/home/mhemraev# █
```

Рис. 2.8: Состояния заданий

5. Перевёл один из процессов на передний план (**fg**), после чего остановил его.
6. Перевёл другой процесс с подавлением вывода в фоновый режим командой **bg**, что изменило его статус на *Running*.
7. Запустил процесс в фоне с помощью **nohup yes > /dev/null &**, что позволило сохранить его выполнение даже после выхода из терминала. Проверка показала, что процесс продолжил работу.

```

root@mhemraev:/home/mhemraev# ps aux | grep yes
root      4944  97.2  0.0 226820 1772 ?        R   10:42   0:38 yes
root      5097   0.0  0.1 227688 2152 pts/1    S+  10:43   0:00 grep --color=auto yes
root@mhemraev:/home/mhemraev# yop
bash: yop: command not found...
root@mhemraev:/home/mhemraev# top

top - 10:43:20 up 15 min,  5 users,  load average: 0.96, 0.94, 0.64
Tasks: 231 total,  2 running, 229 sleeping,  0 stopped,  0 zombie
%Cpu(s):  7.7 us, 15.4 sy,  0.0 ni, 76.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  1961.3 total,  174.8 free,  1267.5 used,  690.5 buff/cache
MiB Swap:  2092.0 total,  2091.7 free,  0.3 used,  693.8 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 4944 root        20   0 226820 1772 1772 R  90.9   0.1   0:53.55 yes
     1 root         0   0  49192 39400 8376 S   0.0   2.0   0:01.58 systemd
     2 root         0   0      0      0      0 S   0.0   0.0   0:00.00 kthreadd
     3 root         0   0      0      0      0 S   0.0   0.0   0:00.00 pool_workqueue_release
     4 root        -20  0      0      0      0 I   0.0   0.0   0:00.00 kworker/R-rcu_gp
     5 root        -20  0      0      0      0 I   0.0   0.0   0:00.00 kworker/R-sync_wq
     6 root        -20  0      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
     9 root         0   0      0      0      0 I   0.0   0.0   0:00.05 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

```

Рис. 2.9: Работа nohup-процесса

- Получил информацию о процессах с помощью утилиты **top**, где был виден работающий процесс **yes**.

```

root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 5192
root@mhemraev:/home/mhemraev# yes > /dev/null &
[2] 5194
root@mhemraev:/home/mhemraev# yes > /dev/null &
[3] 5196
root@mhemraev:/home/mhemraev# kill 5192
[1] Terminated yes > /dev/null
root@mhemraev:/home/mhemraev# fg 2
yes > /dev/null
^C
root@mhemraev:/home/mhemraev# kill -1 5196
[3]+ Hangup yes > /dev/null
root@mhemraev:/home/mhemraev# kill -1 4944
root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 5414
root@mhemraev:/home/mhemraev# yes > /dev/null &
[2] 5416
root@mhemraev:/home/mhemraev# yes > /dev/null &
[3] 5418
root@mhemraev:/home/mhemraev# killall yes
[1] Terminated yes > /dev/null
[2]- Terminated yes > /dev/null
[3]+ Terminated yes > /dev/null
root@mhemraev:/home/mhemraev#

```

Рис. 2.10: Процесс yes в top

- Запустил ещё три процесса **yes** в фоне с подавлением вывода. Завершил

их разными способами: один через **kill <PID>**, другой через **kill -<номер задания>**, а оставшиеся командой **killall yes**.

```

root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 5678
root@mhemraev:/home/mhemraev# nice -n 5 yes > /dev/null &
[2] 5691
root@mhemraev:/home/mhemraev# ps -l
 F S   UID     PID   PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
 4 S    0      5016    4993  0  80   0 - 58153 do_wai pts/1    00:00:00 su
 4 S    0      5028    5016  0  80   0 - 57543 do_wai pts/1    00:00:00 bash
 4 R    0      5678    5028 98  80   0 - 56705 -      pts/1    00:00:15 yes
 4 R    0      5691    5028 97  85   5 - 56705 -      pts/1    00:00:07 yes
 4 R    0      5713    5028  0  80   0 - 57682 -      pts/1    00:00:00 ps
root@mhemraev:/home/mhemraev# renice -n 5 5678
5678 (process ID) old priority 0, new priority 5
root@mhemraev:/home/mhemraev# ps -l
 F S   UID     PID   PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
 4 S    0      5016    4993  0  80   0 - 58153 do_wai pts/1    00:00:00 su
 4 S    0      5028    5016  0  80   0 - 57543 do_wai pts/1    00:00:00 bash
 4 R    0      5678    5028 98  85   5 - 56705 -      pts/1    00:00:40 yes
 4 R    0      5691    5028 97  85   5 - 56705 -      pts/1    00:00:32 yes
 4 R    0      5771    5028  0  80   0 - 57682 -      pts/1    00:00:00 ps
root@mhemraev:/home/mhemraev#

```

Рис. 2.11: Завершение процессов yes

10. Запустил два процесса **yes**: один с обычным приоритетом, второй с повышенным приоритетом с помощью **nice -n 5**. Сравнил приоритеты с помощью **ps -l**: значения поля **NI** (nice value) оказались разными.
11. Изменил приоритет одного из процессов с помощью команды **renice**, чтобы оба процесса имели одинаковое значение приоритета.

3 Контрольные вопросы

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

- Команда **jobs** отображает список всех запущенных заданий в текущей оболочке с указанием их состояния (*Running*, *Stopped* и т.д.).
-

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

- Для остановки используется комбинация **Ctrl+Z**.
 - Чтобы возобновить выполнение задания в фоне — команда **bg <номер_задания>**.
-

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

- Комбинация **Ctrl+C** завершает текущее активное задание.
-

3.0.4 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>** из другой оболочки.
-

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

- Команда **ps fax** (или **pstree**) показывает дерево процессов с иерархией.
-

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

- Используется команда **renice** с отрицательным значением `nice` (чем меньше число, тем выше приоритет).
-

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

- Использовать команду **killall dd**, которая завершает все процессы с именем **dd**.
-

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

- Для этого используется команда **`killall mycommand`**.
-

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

- В `top` для завершения процесса используется клавиша **`k`**, после чего указывается PID процесса.
-

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

- Для запуска с пониженным приоритетом (что гарантирует ресурс другим процессам) используется команда **`nice`** с положительным значением.
-

4 Заключение

В ходе работы были изучены основные приёмы управления процессами и заданиями в Linux. Рассмотрены способы запуска процессов в фоновом и переднем планах, их приостановка, возобновление и завершение. Освоены инструменты изменения приоритета процессов с помощью утилит **nice** и **renice**, а также методы массового завершения процессов через **killall**. Дополнительно были изучены средства мониторинга системных ресурсов с помощью **top** и команды для анализа иерархии процессов. Полученные знания позволяют эффективно контролировать работу процессов и управлять их ресурсами в многозадачной среде.