

Отчет по лабораторной работе №6

Управление процессами

Сидорова Арина Валерьевна

Содержание

1	Цель работы	4
2	выполнение лабораторной работы	5
2.1	Управление заданиями	5
2.2	Управление процессами	8
3	Самостоятельная работа	11
3.1	Задание 1	11
3.2	Задание 2	12
4	Ответы на контрольные вопросы	17
5	выводы	18

Список иллюстраций

2.1	sleep 7200	6
2.2	dd if=/dev/zero of=/dev/null &	6
2.3	top	7
2.4	top 2	8
2.5	x3 dd if=/dev/zero of=/dev/null &	8
2.6	ps aux grep dd	9
2.7	renice -n 5	9
2.8	ps fax grep -B5 dd	9
2.9	kill -9	10
3.1	dd if=/dev/zero of=/dev/null	11
3.2	Увеличение приоритета (5, 15)	12
3.3	Завершаем процессы	12
3.4	Запускаем yes	13
3.5	jobs	13
3.6	bg 2	14
3.7	jobs	14
3.8	nohup yes > /dev/null &	14
3.9	top	15
3.10	x3 yes	15
3.11	kill + ctrl C	15
3.12	killall yes	16
3.13	renice	16

1 Цель работы

Получить навыки управления процессами операционной системы.

2 выполнение лабораторной работы

2.1 Управление заданиями

1. Получим полномочия администратора
2. Введем следующие команды: `sleep 3600 & dd if=/dev/zero of=/dev/null & sleep 7200`
3. 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`.
4. Для продолжения выполнения задания 3 в фоновом режиме введем `bg 3` С помощью команды `jobs` посмотрим изменения в статусе заданий. (рис. 2.1)

```

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

^C244856641+0 records in
244856641+0 records out
125366600192 bytes (125 GB, 117 GiB) copied, 91,4181 s, 1,4 GB/s

root@avsidorova:~# fg 3
sleep 7200
^C
root@avsidorova:~#

```

Рис. 2.1: sleep 7200

6. Для перемещения задания 1 на передний план введем fg 1
7. Введем Ctrl + c , чтобы отменить задание 1. С помощью команды jobs посмотрим изменения в статусе заданий.
8. Прделаем то же самое для отмены заданий 2 и 3.
9. Откроем второй терминал и под учётной записью своего пользователя введем в нём: dd if=/dev/zero of=/dev/null &. (рис. 2.2)

```

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

```

Рис. 2.2: dd if=/dev/zero of=/dev/null &

10. Введем exit, чтобы закрыть второй терминал.
11. На другом терминале под учётной записью своего пользователя запустим top

мы увидим, что задание dd всё ещё запущено. Для мыхода из top используем q
(рис. 2.3)

```
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
 4006 avsidor+  20   0 227308 2140 2140 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 ptxys
  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-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   0:00.15 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
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   0:00.12 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
```

Рис. 2.3: top

12. Вновь запустим top и в нём используем k , чтобы убить задание dd. После этого мыйдем из top. (рис. 2.4)

```

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 ht, 0,5 st, 0,0 sr
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	ptxixis
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	0:00.00	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	ksftirqd/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	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

Рис. 2.4: top 2

2.2 Управление процессами

1. Получим полномочия администратора su -
2. Введем следующие команды: dd if=/dev/zero of=/dev/null & dd if=/dev/zero of=/dev/null & dd if=/dev/zero of=/dev/null & (рис. 2.5)

```

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

3. Введем ps aux | grep dd Это показывает все строки, в которых есть букмы dd. Запущенные процессы dd идут последними. (рис. 2.6)


```

root@avsidorova:~# ps aux | grep dd
root      2  0.0  0.0      0   0 ?        S   18:09   0:00 [kthreadd]
root     76  0.0  0.0      0   0 ?        I<  18:09   0:00 [kworker/R-ipv6_addrconf]
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. 2.6: ps aux | grep dd

- Используем PID одного из процессов dd, чтобы изменить приоритет. Используем renice -n 5 (рис. 2.7)

```

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

```

Рис. 2.7: renice -n 5

- Введем ps fax | grep -B5 dd. 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. (рис. 2.8)

```

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-kalud]
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 daemon
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 {cle072e8-fda0-4fa2-918c-e0624351a73f} -parentPid 3464 -greomni /usr/lib64/firefox/omni.ja -
appomni /usr/lib64/firefox/browser/omni.ja -appDir /usr/lib64/firefox/browser 2 tab
3687 ? Sl 0:00 | \_ /usr/lib64/firefox/firefox -contentproc -parentBuildID 20250918202509 -prefs
Handle 0:35109 -prefMapHandle 1:271057 -sandboxReporter 2 -chrootClient 3 -ipcHandle 4 -initialChannelId {cc74303f-7d
36-4002-9ecc-5567deda1b6b} -parentPid 3682 -appDir /usr/lib64/firefox/browser 3 rdd

```

Рис. 2.8: ps fax | grep -B5 dd

6. Найдем PID корневой оболочки, из которой были запущены процессы dd, и введем kill -9 (заменив на значение PID оболочки). 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. (рис. 2.9)

```
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. 2.9: kill -9

3 Самостоятельная работа

3.1 Задание 1

1. Запустим команду `dd if=/dev/zero of=/dev/null` трижды как фоновое задание.
(рис. 3.1)

```
^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:~# █
```

Рис. 3.1: `dd if=/dev/zero of=/dev/null`

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 (рис. 3.2)

```

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. 3.2: Увеличение приоритета (5, 15)

4. Завершим все процессы dd, которые мы запустили. (рис. 3.3)

```

[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
91089951+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
85803602+0 records out
43931444224 bytes (44 GB, 41 GiB) copied, 382,937 s, 115 MB/s

root@avsidorova:~# jobs
root@avsidorova:~# █

```

Рис. 3.3: Завершаем процессы

3.2 Задание 2

1. Запустим программу ues в фоновом режиме с подавлением потока вывода. (рис. 3.4)

```

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. 3.4: Запускаем yes

2. Запустим программу yes на переднем плане с подавлением потока вывода. При- остановим выполнение программы. Заново запустим программу yes с теми же параметрами, затем завершим её выполнение.
3. Запустим программу yes на переднем плане без подавления потока вывода. При- остановим выполнение программы. Заново запустим программу yes с теми же параметрами, затем завершим её выполнение.
4. Проверим состояния заданий, воспользовавшись командой jobs. (рис. 3.5)

```

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. 3.5: 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. (рис. 3.6)

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

Рис. 3.6: bg 2

7. Проверим состояния заданий, воспользовавшись командой jobs. Обратим внимание, что процесс стал выполняющимся (Running) в фоновом режиме. (рис. 3.7)

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

Рис. 3.7: jobs

8. Запустим процесс в фоновом режиме таким образом, чтобы он продолжил свою работу даже после отключения от терминала. (рис. 3.8)

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

Рис. 3.8: nohup yes > /dev/null &

9. Закроем окно и заново запустим консоль. Убедимся, что процесс продолжил свою работу.
10. Получим информацию о запущенных в операционной системе процессах с помощью утилиты top. (рис. 3.9)

```

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. 3.9: top

11. Запустим ещё три программы yes в фоновом режиме с подавлением потока вывода. (рис. 3.10)

```

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. 3.10: x3 yes

12. Убеем два процесса: для одного используем его PID, а для другого — его идентификатор конкретного задания. (рис. 3.11)

```

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. 3.11: kill + ctrl C

13. Попробуем послать сигнал 1 (SIGHUP) процессу, запущенному с помощью nohup, и обычному процессу.

14. Запустим ещё несколько программ `yes` в фоновом режиме с подавлением потока вывода.
15. Завершим их работу одновременно, используя команду `killall`. (рис. 3.12)

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

Рис. 3.12: `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.
17. Используя утилиту `renice`, изменим приоритет у одного из потоков `yes` таким образом, чтобы у обоих потоков приоритеты были равны. (рис. 3.13)

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

Рис. 3.13: `renice`

4 Ответы на контрольные вопросы

1. jobs
2. Ctrl+Z, затем bg
3. Ctrl+C
4. Использовать kill или killall из другой сессии.
5. ps fax или pstree
6. renice -n -5 1234 (требуются права суперпользователя)
7. killall dd
8. killall mycommand
9. Нажать k, ввести PID процесса.
10. nice -n -10 command (с учетом прав доступа).

5 выводы

Получили намыки управления процессами операционной системы.