



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт
Кафедра

компьютерных наук
автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3

по операционным системам Linux

«Процессы и управление ими в операционной системе Linux»

Студент

ПМ-22-1

подпись, дата

Щелконогов
Е.А.

Руководитель

подпись, дата

Кургасов В.В.

Липецк, 2024 г.

Оглавление

Цель работы.....	3
Ход работы.....	3
Часть I.....	3
Часть II.....	9
Часть III.....	15
Часть IV.....	17
Вывод.....	20
Контрольные вопросы.....	21

Цель работы

Ознакомиться на практике с понятием процесса в операционной системе. Приобрести опыт и навыки управления процессами в операционной системе Linux.

Ход работы

Часть I

1. Войти в систему под пользовательской учётной записью (не root).
2. Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.
3. Посмотреть процессы `ps -f`. Прокомментировать, изучив предварительно справку командой `man ps`.

```
transhumanist@transhumanist:/$ ls -la vmlinuz
lrwxrwxrwx 1 root root 27 окт 14 00:02 vmlinuz -> boot/vmlinuz-6.1.0-26-amd64
transhumanist@transhumanist:/$ ps -f

```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
transhu+	482	379	0	11:43	tty1	00:00:00	-bash
transhu+	495	482	99	11:45	tty1	00:00:00	ps -f

```
transhumanist@transhumanist:/$ _
```

4. Написать с помощью редактора `vi` два сценария `loop` и `loop2`.

```
transhumanist@transhumanist:~$ cat loop loop2
while true; do true; done
while true; do true; echo 'Hello'; done
```

5. Запустить `loop2` на переднем плане: `./loop2`.
6. Остановить, послав сигнал `STOP`.
7. Посмотреть последовательно несколько раз `ps -f`. Записать сообщение, объяснить.

При первом вызове значение `C` у процесса было 28 при втором 1. Процесс проработал довольно долго перед тем как я его остановил.

```

[5]+ Остановлен      ./loop2
transhumanist@transhumanist:~$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
transhu+     477      365   0  11:51 tty1        00:00:00 -bash
transhu+     489      477   0  11:55 tty1        00:00:00 -bash
transhu+     492      477   1  11:55 tty1        00:00:01 -bash
transhu+     496      477   1  11:56 tty1        00:00:00 -bash
transhu+     497      477   1  11:56 tty1        00:00:00 -bash
transhu+     510      477  28  11:57 tty1        00:00:01 -bash
transhu+     511      477  99  11:57 tty1        00:00:00 ps -f

```

8. Убить процесс loop2, пошлав сигнал kill -9 PID. Записать сообщение. Прокомментировать.

После передачи сигнала SIGKILL процесс был завершен.

```

UID          PID     PPID  C  STIME TTY          TIME CMD
transhu+     574      552   1  12:04 tty1        00:00:00 -bash
transhu+     577      574  16  12:04 tty1        00:00:01 -bash
transhu+     578      574  99  12:04 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ kill -9 577
[1]+  Убито          ./loop2
transhumanist@transhumanist:~$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
transhu+     574      552   0  12:04 tty1        00:00:00 -bash
transhu+     581      574  99  12:05 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ _

```

9. Запустить в фоне процесс loop: ./loop&. Не останавливая, посмотреть несколько раз: ps -f. Записать значение, объяснить.

При запуске процесса loop в фоне загрузка процесса C равна 99 все время работы процесса. Это связано с тем, что с момента запуска процесс не прекращал работу.

```

transhumanist@transhumanist:~$ ./loop&
[1] 582
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+      574      552   0 12:04 tty1        00:00:00 -bash
transhu+      582      574  99 12:05 tty1        00:00:23 -bash
transhu+      583      574  66 12:06 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+      574      552   0 12:04 tty1        00:00:00 -bash
transhu+      582      574  99 12:05 tty1        00:00:29 -bash
transhu+      584      574  66 12:06 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$

```

10. Завершить процесс loop командой kill -15 PID. Записать сообщение, прокомментировать.

Процесс 582 (./loop&) был остановлен сигналом SIGTERM(15).

```

transhumanist@transhumanist:~$ kill -15 582
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+      574      552   0 12:04 tty1        00:00:00 -bash
transhu+      587      574  99 12:09 tty1        00:00:00 ps -f
[1]+  Завершено      ./loop

```

11. Третий раз запустить в фоне. Не останавливая, убить командой kill -9 PID.

В результате передачи сигнала SIGKILL процессу 593 он был прекращён.

```

transhumanist@transhumanist:~$ ./loop&
[1] 593
transhumanist@transhumanist:~$ kill -9 593
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+      574      552   0 12:04 tty1        00:00:00 -bash
transhu+      594      574  99 12:12 tty1        00:00:00 ps -f
[1]+  Убито          ./loop
transhumanist@transhumanist:~$

```

12. Запустить ещё один экземпляр оболочки: `bash`.
13. Запустить несколько процессов в фоне. Останавливать их и снова запускать. Записать результаты просмотра командой `ps -f`.

```
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+    480      365  0 12:17 tty1        00:00:00 -bash
transhu+    529      480 34 12:18 tty1        00:00:01 sh loop
transhu+    530      480 20 12:18 tty1        00:00:00 sh loop
transhu+    531      480 18 12:18 tty1        00:00:00 sh loop
transhu+    532      480 17 12:18 tty1        00:00:00 sh loop
transhu+    533      480 17 12:18 tty1        00:00:00 sh loop
transhu+    534      480 16 12:18 tty1        00:00:00 sh loop
transhu+    535      480 30 12:18 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ kill -19 529
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+    480      365  0 12:17 tty1        00:00:00 -bash
transhu+    529      480 16 12:18 tty1        00:00:17 sh loop
transhu+    530      480 16 12:18 tty1        00:00:17 sh loop
transhu+    531      480 16 12:18 tty1        00:00:17 sh loop
transhu+    532      480 16 12:18 tty1        00:00:17 sh loop
transhu+    533      480 16 12:18 tty1        00:00:17 sh loop
transhu+    534      480 16 12:18 tty1        00:00:17 sh loop
transhu+    536      480 25 12:20 tty1        00:00:00 ps -f

[1]+  Остановлен    sh loop
transhumanist@transhumanist:~$ kill -18 529
transhumanist@transhumanist:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
transhu+    480      365  0 12:17 tty1        00:00:00 -bash
transhu+    529      480 14 12:18 tty1        00:00:17 sh loop
transhu+    530      480 17 12:18 tty1        00:00:21 sh loop
transhu+    531      480 17 12:18 tty1        00:00:21 sh loop
transhu+    532      480 17 12:18 tty1        00:00:21 sh loop
transhu+    533      480 17 12:18 tty1        00:00:21 sh loop
transhu+    534      480 17 12:18 tty1        00:00:21 sh loop
transhu+    537      480 28 12:20 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ kill -18 529_
```

Рисунок 1: Запуск и остановка нескольких скриптов `loop`

Часть II

1. Запустить в консоли на выполнение три задачи: две в интерактивном режиме, одну - в фоновом.

```
transhumanist@transhumanist:~$ sleep 1000&
[1] 823
transhumanist@transhumanist:~$ sleep 1000
^Z
[2]+  Остановлен      sleep 1000
transhumanist@transhumanist:~$ sleep 800
```

Teleg

2. Перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим.

```
transhumanist@transhumanist:~$ jobs
[1]  Запущен          sleep 1000 &
[2] -  Остановлен     sleep 1000
[3] +  Остановлен     sleep 800
transhumanist@transhumanist:~$ bg 3
[3] + sleep 800 &
transhumanist@transhumanist:~$
```

3. Провести эксперименты по переводу задач из фонового режима в интерактивный и наоборот.

```
transhumanist@transhumanist:~$ fg 3
sleep 800
```

4. Создать именованный канал для архивирования и осуществить передачу в канал:

```
transhumanist@transhumanist:~$ mkfifo pipe
transhumanist@transhumanist:~$ ls
arh7.tar  file2  file_a  loop  pipe  second.txt  Видео  Загрузки  Музыка  'Рабочий стол'
file1    file3  first.txt  loop2  scriptecho.sh  third.txt  Документы  Изображения  Общедоступные  Шаблоны
transhumanist@transhumanist:~$ _
```

- а) списка файлов домашнего каталога вместе с подкаталогами (ключ -R);


```

transhumanist@transhumanist:~$ cat pipe > file_a&
[1] 834
transhumanist@transhumanist:~$ ls -R > pipe
[1]+  Завершён      cat pipe > file_a
transhumanist@transhumanist:~$ ls
arh7.tar  file2  file_a  loop  pipe  second.txt  Видео  Загрузки  Музыка  'Рабочий стол'
file1     file3  first.txt  loop2  scriptecho.sh  third.txt  Документы  Изображения  Общедоступные  Шаблоны
transhumanist@transhumanist:~$ cat file_a
.:
arh7.tar
file1
file2
file3
file_a
first.txt
loop
loop2
pipe
scriptecho.sh
second.txt
third.txt
Видео
Документы
Загрузки
Изображения
Музыка
Общедоступные
Рабочий стол
Шаблоны

./Видео:

./Документы:

./Загрузки:

./Изображения:

./Музыка:

./Общедоступные:

./Рабочий стол:

./Шаблоны:
transhumanist@transhumanist:~$ _

```

b) одного каталога вместе с файлами и подкаталогами.

```
transhumanist@transhumanist:~$ cat ./
cat: ./: Это каталог
transhumanist@transhumanist:~$ ls -R Загрузки > pipe
[1]+  Завершён      cat pipe > file_b
transhumanist@transhumanist:~$ cat file_b
Загрузки:
transhumanist@transhumanist:~$ _
```

Музык

Часть III

Вариант 7

1. Вывести информацию о состоянии процессов системы в реальном режиме с обновлением один раз в 5 секунд. Отсортировать вывод по идентификатору пользователя по возрастанию и убыванию

```
Tasks: 67 total, 1 running, 66 sleeping, 0 stopped, 0 zombie
%CPU(s): 0.0 us, 2.4 sy, 0.0 ni, 97.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 960.7 total, 655.9 free, 218.8 used, 223.8 buff/cache
MiB Swap: 975.0 total, 975.0 free, 0.0 used, 741.9 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
  1 root        20   0 101960 11920 9072 S   0.0  1.2   0:02.65 systemd
  2 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kthreadd
  3 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 rcu_gp
  4 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 rcu_par_gp
  5 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 slub_flushq
  6 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 netns
  8 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kworker/0:0H-kblockd
  9 root        20   0      0      0    0 I   0.0  0.0   0:01.48 kworker/u2:0-events_unbound
 10 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 mm_percpu_wq
 11 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_kthread
 12 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_rude_kthread
 13 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_trace_kthread
 14 root        20   0      0      0    0 S   0.0  0.0   0:00.25 ksoftirqd/0
 15 root        20   0      0      0    0 I   0.0  0.0   0:00.51 rcu_preempt
 16 root        rt    0      0      0    0 S   0.0  0.0   0:00.09 migration/0
 18 root        20   0      0      0    0 S   0.0  0.0   0:00.00 cpuhp/0
 20 root        20   0      0      0    0 S   0.0  0.0   0:00.01 kdevtmpfs
 21 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 inet_frag_wq
 22 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kauditd
 23 root        20   0      0      0    0 S   0.0  0.0   0:00.00 khungtaskd
 25 root        20   0      0      0    0 S   0.0  0.0   0:00.00 pom_reaper
 26 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 writeback
 28 root        20   0      0      0    0 S   0.0  0.0   0:00.69 kcompactd0
 29 root        25   5      0      0    0 S   0.0  0.0   0:00.00 ksm
 30 root        39  19      0      0    0 S   0.0  0.0   0:00.09 khugepaged
 31 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kintegrityd
 32 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kblockd
 33 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 blkcg_punt_bio
 34 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 tpm_dev_wq
 35 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 edac-poller
 36 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 devfreq_wq
 38 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kswapd0
 44 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kthrotld
 46 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 acpi_thermal_pm
 48 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 mld
 49 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 ipv6_addrconf
 54 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kstrp
 59 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 zswap-shrink
 60 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kworker/u3:0
 116 root       0 -20      0      0    0 I   0.0  0.0   0:00.00 ata_sff
 117 root        20   0      0      0    0 S   0.0  0.0   0:00.00 scsi_eh_0
 118 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 scsi_tmf_0
 119 root        20   0      0      0    0 S   0.0  0.0   0:00.01 scsi_eh_1
transhumanist@transhumanist:~$ top -d 5 -o UID_
```

```
The nmhpcmc included with the Debian GNU/Linux system are free software:
Tasks: 67 total, 1 running, 66 sleeping, 0 stopped, 0 zombie
%CPU(s): 0.0 us,100.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 960.7 total, 655.9 free, 218.8 used, 223.8 buff/cache
MiB Swap: 975.0 total, 975.0 free, 0.0 used, 741.9 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 462 transhu+  20   0 18912 10540 8812 S   0.0  1.1   0:00.38 systemd
 463 transhu+  20   0 108020 2380    0 S   0.0  0.3   0:00.00 (sd-pam)
 478 transhu+  20   0 8280 4904 3440 S   0.0  0.5   0:01.11 bash
 592 transhu+  20   0 11644 4972 3072 R  11.8  0.5   0:00.05 top
 256 systemd+  20   0 90880 6636 5752 S   0.0  0.7   0:01.01 systemd-timesyn
 342 message+  20   0 9120 4808 4236 S   0.0  0.5   0:00.25 dbus-daemon
 381 www-data  20   0 753748 11932 3236 S   0.0  1.1   0:00.01 apache2
 382 www-data  20   0 753748 8988 3292 S   0.0  0.9   0:00.01 apache2
  1 root        20   0 101960 11920 9072 S   0.0  1.2   0:02.67 systemd
  2 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kthreadd
  3 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 rcu_gp
  4 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 rcu_par_gp
  5 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 slub_flushq
  6 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 netns
  8 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kworker/0:0H-kblockd
  9 root        20   0      0      0    0 I   0.0  0.0   0:01.48 kworker/u2:0-events_unbound
 10 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 mm_percpu_wq
 11 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_kthread
 12 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_rude_kthread
 13 root        20   0      0      0    0 I   0.0  0.0   0:00.00 rcu_tasks_trace_kthread
 14 root        20   0      0      0    0 S   0.0  0.0   0:00.25 ksoftirqd/0
 15 root        20   0      0      0    0 I   0.0  0.0   0:00.51 rcu_preempt
 16 root        rt    0      0      0    0 S   0.0  0.0   0:00.10 migration/0
 18 root        20   0      0      0    0 S   0.0  0.0   0:00.00 cpuhp/0
 20 root        20   0      0      0    0 S   0.0  0.0   0:00.01 kdevtmpfs
 21 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 inet_frag_wq
 22 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kauditd
 23 root        20   0      0      0    0 S   0.0  0.0   0:00.00 khungtaskd
 25 root        20   0      0      0    0 S   0.0  0.0   0:00.00 pom_reaper
 26 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 writeback
 28 root        20   0      0      0    0 S   0.0  0.0   0:00.75 kcompactd0
 29 root        25   5      0      0    0 S   0.0  0.0   0:00.00 ksm
 30 root        39  19      0      0    0 S   0.0  0.0   0:00.09 khugepaged
 31 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kintegrityd
 32 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kblockd
 33 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 blkcg_punt_bio
 34 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 tpm_dev_wq
 35 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 edac-poller
 36 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 devfreq_wq
 38 root        20   0      0      0    0 S   0.0  0.0   0:00.00 kswapd0
 44 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 kthrotld
 46 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 acpi_thermal_pm
 48 root        0 -20      0      0    0 I   0.0  0.0   0:00.00 mld
transhumanist@transhumanist:~$ top -d 5 -o UID_
```

2. Завершить выполнение процесса, владельцем которого является текущий пользователь, с помощью сигнала SIGQUIT двумя способами: задав имя сигнала и используя комбинацию клавиш.

```
transhumanist@transhumanist:~$ kill -s SIGQUIT 595
[1]+  Выход          sleep 1000
transhumanist@transhumanist:~$ sleep 1000
^\\Выход
```

3. Измените на 2 единицы приоритет процесса, запущенного из командного интерпретатора.

```
transhumanist@transhumanist:~$ sleep 1000&
[1] 636
transhumanist@transhumanist:~$ renice -n 2 -p 636
636 (process ID) old priority 0, new priority 2
transhumanist@transhumanist:~$
```

Часть IV

1. Открыть окно интерпретатора команд.
2. Вывести общую информацию о системе:
 - a) вывести информацию о текущем интерпретаторе команд;
 - b) вывести информацию о текущем пользователе;
 - c) вывести информацию о текущем каталоге;
 - d) вывести информацию об оперативной памяти и области подкачки;
 - e) вывести информацию о дисковой памяти.

```
transhumanist@transhumanist:~$ echo $SHELL
/bin/bash
transhumanist@transhumanist:~$ echo $USER
transhumanist
transhumanist@transhumanist:~$ echo $PWD
/home/transhumanist
transhumanist@transhumanist:~$ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	960Mi	222Mi	649Mi	756Ki	226Mi	738Mi
Swap:	974Mi	0B	974Mi			

```
transhumanist@transhumanist:~$ df -h
```

Файловая система	Размер	Использовано	Дост	Использовано%	Смонтировано в
udev	462M	0	462M	0%	/dev
tmpfs	97M	416K	96M	1%	/run
/dev/sda1	31G	3,5G	26G	12%	/
tmpfs	481M	0	481M	0%	/dev/shm
tmpfs	5,0M	0	5,0M	0%	/run/lock
tmpfs	97M	0	97M	0%	/run/user/1000

```
transhumanist@transhumanist:~$
```

3. Выполнить команды получения информации о процессах:
 - a) получить идентификатор текущего процесса(PID);
 - b) получить идентификатор родительского процесса(PPID);
 - c) получить идентификатор процесса инициализации системы;
 - d) получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд;
 - e) отобразить все процессы;

```

transhumanist@transhumanist:~$ echo $$
478
transhumanist@transhumanist:~$ echo ${PPID}
363
transhumanist@transhumanist:~$ ps h -eo pid | head -1
1
transhumanist@transhumanist:~$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
transhu+      478       363  0  13:01 tty1        00:00:02 -bash
transhu+      636       478  0  13:35 tty1        00:00:00 sleep 1000
transhu+      650       478 99  13:42 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ ps -e_

```

4. Выполнить команды управления процессами:

- a) получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе - `ps -f`;
- b) определить текущее значение nice по умолчанию - `nice` без аргументов;
- c) запустить интерпретатор bash с понижением приоритета - `nice -n 10 bash`;
- d) определить PID запущенного интерпретатора - `echo \$\$`;
- e) установить приоритет запущенного интерпретатора равным 5
su -c „renice -n 5 <PID>“
- f) получить информацию о процессах bash - ps lax | grep bash.

```

transhumanist@transhumanist:~$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
transhu+      674       669  1  13:46 tty1        00:00:00 -bash
transhu+      677       674 99  13:46 tty1        00:00:00 ps -f
transhumanist@transhumanist:~$ nice
0
transhumanist@transhumanist:~$ nice -n 10 bash
transhumanist@transhumanist:~$ echo $$
679
transhumanist@transhumanist:~$ su -c "renice -n 5 679"
Пароль:
679 (process ID) old priority 10, new priority 5
transhumanist@transhumanist:~$ ps lax | grep bash
4 1000      674       669  20   0   7976  4760 do_wai S    tty1        0:00 -bash
0 1000      679       674  25   5   8004  4644 do_wai SN   tty1        0:00 bash
0 1000      684       679  25   5   6356  2196 pipe_r SN+  tty1        0:00 grep bash
transhumanist@transhumanist:~$

```

Вывод

В результате выполнения данной лабораторной работы я научился взаимодействовать и управлять процессами в операционной системе Linux. Я научился посылать сигналы прерывания процессами и переводить их из интерактивного режима в фоновый и обратно в зависимости от того, какие функции они должны выполнять в системе.

Контрольные вопросы

1. Перечислите состояния задачи в ОС Linux.

- Runnable (R) — процесс выполняется/готов к выполнению;
- Sleeping (S) — процесс находится в ожидании события;
- Uninterruptible (D) — аналогичен предыдущему, однако не прерывается для обработки сигналов;
- Stopped (T) — процесс остановлен;
- Zombie (Z) — процесс завершил выполнение и почти полностью выгружен из памяти, хранит только код завершения.

2. Как создаются задачи в ОС Linux?

При помощи команды `fork()` дублируется текущий процесс, после чего в памяти размещаются данные о новом, дочернем для текущего, процессе.

При вызове новой команды в фоновом режиме процесс командной оболочки дублируется с помощью `fork`, и дочерний процесс с новым PID исполняет заданную команду.

3. Назовите классы потоков ОС Linux.

Пользовательские потоки — реализуются через специальные библиотеки потоков

Потоки на уровне ядра — реализуются через системные вызовы

4. Как используется приоритет планирования при запуске задачи?

В зависимости от приоритета задачи ей выделяется разное количество квантов времени. На выполнение идут в первую очередь задачи с более высоким приоритетом.

5. Объясните, что произойдёт, если запустить программу в фоновом режиме без подавления потока вывода.

Она продолжит отправлять сообщения в стандартный поток вывода.

6. Объясните разницу между действием сочетаний клавиш Ctrl^Z и Ctrl^C.

Ctrl+Z посылает SIGTSTP (20), приостанавливающий процесс *временно*.

Ctrl+C посылает SIGINT(2), вызывающий *завершение работы* процесса.

7. Опишите, что значит каждое поле вывода команды jobs.

В общем виде вывод команды jobs можно представить следующей конструкцией:

[<Id>] [+|-] <status> <command>

- <ID> - номер задачи в списке задач;
- [+|-] - обозначает «текущую» задачу — знак + означает, что задача выбрана будет параметром по умолчанию для команд fg, bg. Знак „-“ означает, что задача является следующим параметром по умолчанию после изменения текущего параметра;
- Status — текущий статус задачи (Running/Interrupted/...);
- command — команда, запущенная в виде отдельной задачи.

8. Назовите главное отличие утилиты top от ps.

`ps` выводит список процессов в момент вызова команды (CLI инструмент).

`top` выводит интерактивный список процессов (TUI инструмент).

9. В чем отличие результата выполнения команд top и htop?

`top` предустановлена с большинстве UNIX-систем и предоставляет базовый мониторинг процессов.

`htop` (как и atop, btop) является сторонней TUI утилитой и устанавливается отдельно, но взамен предлагает более удобный и понятный интерфейс.

10. Какую комбинацию клавиш нужно использовать для принудительного завершения задания, запущенного в интерактивном режиме?

Ctrl+C, посылает SIGINT

11. Какую комбинацию клавиш нужно использовать для приостановки задания, запущенного в интерактивном режиме?

Ctrl+Z, посылает SIGTSTP

12. Какая команда позволяет послать сигнал конкретному процессу?

``kill -SIG PID``, где SIG — сигнал, PID — id процесса.

13. Какая команда позволяет поменять поправку к приоритету уже запущенного процесса?

`renice -n <новый_приоритет> PID`

14. Какая команда позволяет запустить задание с пониженным приоритетом?

`nice -n <новая_программа> <команда>`

15. Какая команда позволяет запустить задание с защитой от прерывания при выходе из системы пользователя?

`nohup <команда> [ключи] &`

16. Какой процесс всегда присутствует в системе и является предком всех процессов?

Процесс init (PID 1).

17. Каким образом можно запустить задание в фоновом режиме?

Добавить знак амперсанд (&) в конце команды.

18. Каким образом задание, запущенное в фоновом режиме, можно перевести в интерактивный режим?

С помощью `fg <номер>`, где <номер> отвечает за номер задачи в списке задач jobs.

19. Каким образом приостановленное задание можно перевести в интерактивный режим?

С помощью `fg <номер>`, где `<номер>` отвечает за номер задачи в списке задач `jobs`. При переводе в интерактивный режим приостановленная задача возобновляется.

20. Что произойдёт с заданием, выполняющимся в фоновом режиме, если оно попытается обратиться к терминалу?

Задание приостановится, так как не сможет считать данные с потока ввода (см. рисунок 12).

21. Сколько терминалов может быть открыто в одной системе? Как перемещаться между терминалами (какие комбинации клавиш необходимо использовать)?

Шесть терминалов (`tty1-6`). Перемещаться между ними можно с помощью сочетаний клавиш `Ctrl+Alt+[F1-F6]`.

22. В чем отличие идентификаторов PID и PPID? При каких условиях возможна ситуация, когда PPID равен нулю или отсутствует?

PID — Уникальный идентификатор, выделяемый каждому процессу.

PPID — идентификатор процесса, породившего данный процесс. Если у процесса «нет родителя» или он преждевременно завершился, что в качестве родителя указывается `init`, `PPID=1`.

23. Поясните, от чего зависит максимальное значение PID.

По умолчанию `~32000`, т. к. значение лежит в целочисленном типе `int`. Может быть изменено на величину до 4 млн.

24. В каком случае, при создании нового процесса, его идентификатор (PID) будет меньше, чем у процесса, запущенного ранее?

В качестве PID выдаётся первое свободное значение идентификатора, начиная с последнего выданного PID-а. Если счетчик последнего PID переполнится, то поиск снова начнётся с 1 и может оказаться меньше PPID.