

**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

**ЛАБОРАТОРНАЯ РАБОТА №4**

по Операционной системе Linux

Управление процессами в ОС Ubuntu

Студент

Лобов М.Ю.

Группа АИ-18

Руководитель

Кургасов В.В.

Доцент, к.п.н.

Липецк 2020 г.

## **Оглавление**

Цель работы .....	3
Задание кафедры.....	4
Ход работы.....	5
Вывод.....	10
Ответы на контрольные вопросы .....	11

Цель работы

Познакомиться со средствами управления процессами ОС Ubuntu.

## Задание кафедры

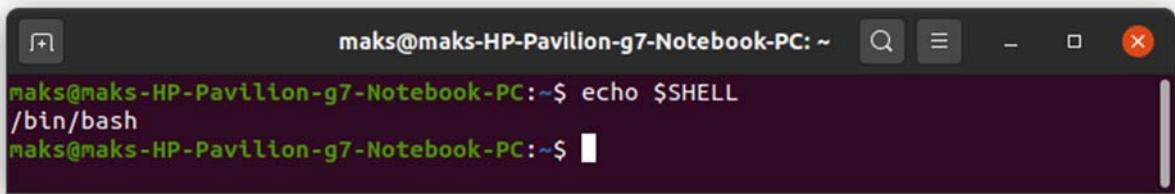
1. Запустить программу виртуализации Oracle VM VirtualBox.
2. Запустить виртуальную машину Ubuntu.
3. Открыть окно интерпретатора команд.
4. Вывести общую информацию о системе.
  - 4.1. Вывести информацию о текущем интерпретаторе команд.
  - 4.2. Вывести информацию о текущем пользователе.
  - 4.3. Вывести информацию о текущем каталоге.
  - 4.4. Вывести информацию об оперативной памяти и области подкачки.
  - 4.5. Вывести информацию о дисковой памяти.
5. Выполнить команды получения информации о процессах.
  - 5.1. Получить идентификатор текущего процесса (PID).
  - 5.2. Получить идентификатор родительского процесса (PPID).
  - 5.3. Получить идентификатор процесса инициализации системы.
  - 5.4. Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд.
  - 5.5. Отобразить все процессы.
6. Выполнить команды управления процессами.
  - 6.1. Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе.
  - 6.2. Определить текущее значение **nice** по умолчанию.
  - 6.3. Запустить интерпретатор **bash** с понижением приоритета.
  - 6.4. Определить PID запущенного интерпретатора.
  - 6.5. Установить приоритет запущенного интерпретатора равным 5.
  - 6.6. Получить информацию о процессах **bash**.

## Ход работы

В связи с тем, что на рабочей машине установлена ОС Linux Ubuntu, а не Linux Ubuntu Server, было принято решение выполнить лабораторную работу в установленном дистрибутиве. Серьёзных отличий не выявлено.

Первым делом выведем информацию о текущем интерпретаторе команд.

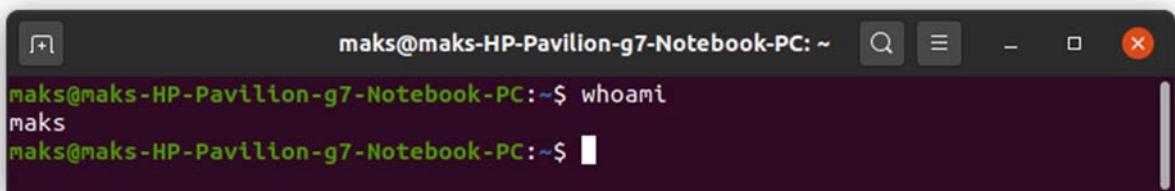
Сделаем это с помощью команды **echo \$SHELL**:



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$ echo $SHELL  
/bin/bash  
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$
```

Рисунок 1 – Информация о текущем интерпретаторе команд

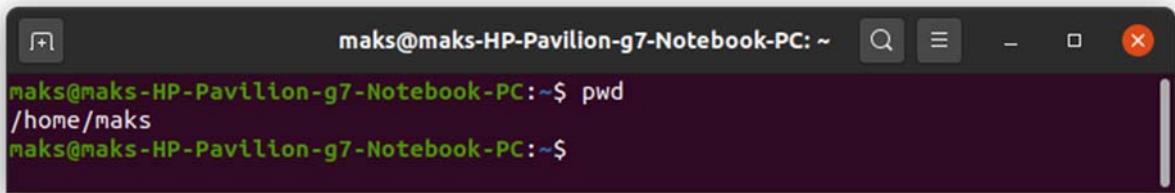
Далее выполним похожее задание: теперь выведем информацию о текущем пользователе, а именно его имя. Сделаем это с помощью команды **whoami**:



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$ whoami  
maks  
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$
```

Рисунок 2 – Информация о текущем пользователе

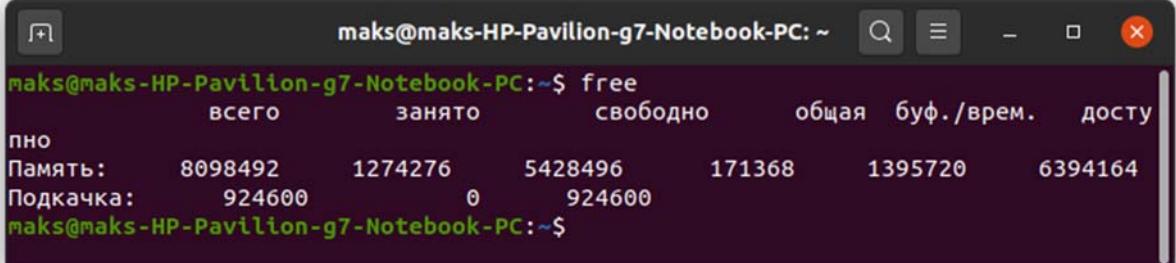
Теперь выведем информацию о текущем каталоге (команда **pwd**):



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$ pwd  
/home/maks  
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$
```

Рисунок 3 – Информация о текущем пользователе

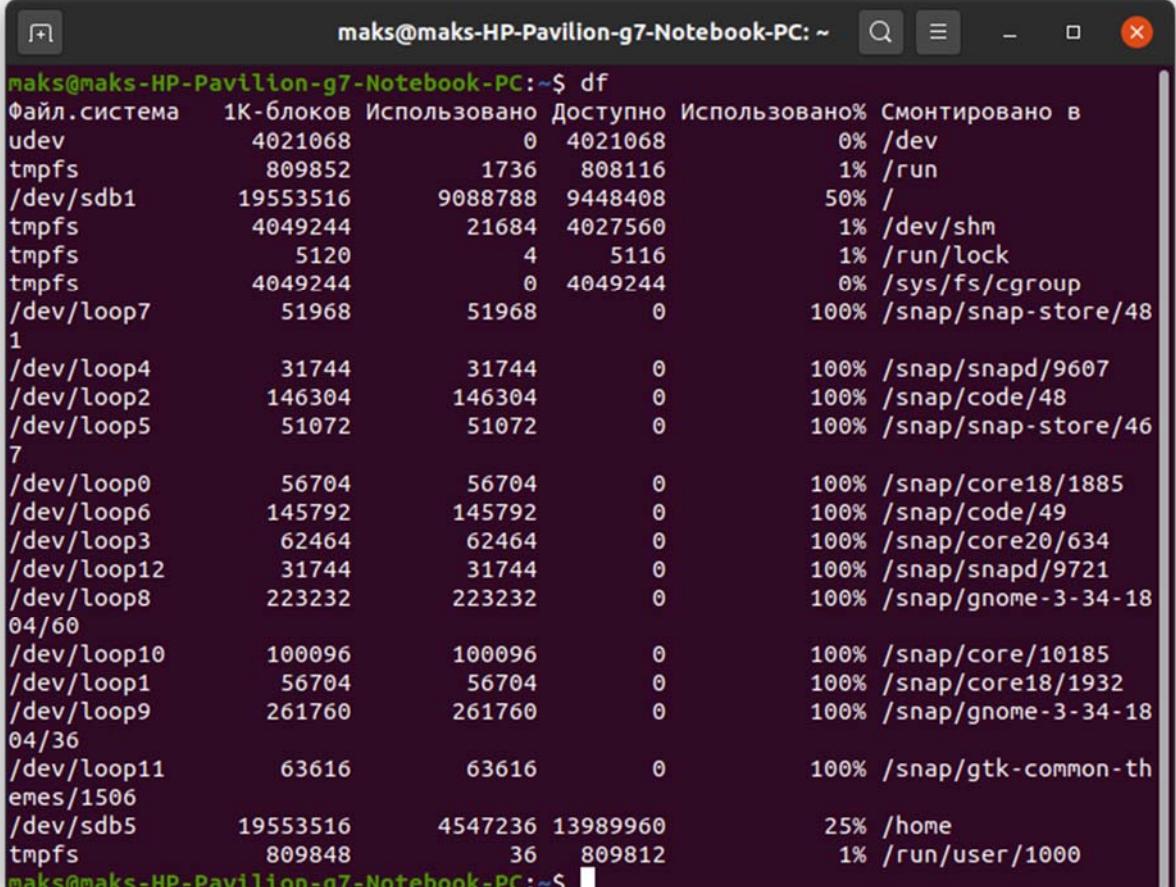
Следующим шагом будет вывод информации об оперативной памяти и области подкачки. В этом нам поможет команда **free**:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ free
              всего      занято      свободно      общая  буф./врем.    доступ
пно
Память:     8098492      1274276      5428496      171368      1395720      6394164
Подкачка:    924600          0      924600
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 4 – Информация об оперативной памяти и области подкачки

Последним заданием в разделе будет вывод информации о дисковой памяти. Для этого задействуем команду **df**:

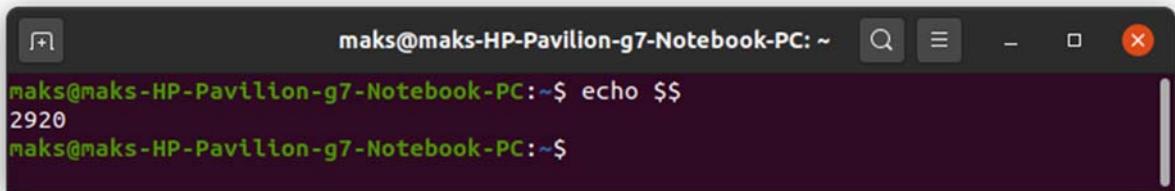


```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ df
Файл.система 1К-блоков Использовано Доступно Использовано% Смонтировано в
udev           4021068          0   4021068          0% /dev
tmpfs          809852        1736   808116          1% /run
/dev/sdb1      19553516    9088788  9448408         50% /
tmpfs          4049244       21684  4027560          1% /dev/shm
tmpfs          5120            4     5116          1% /run/lock
tmpfs          4049244          0  4049244          0% /sys/fs/cgroup
/dev/loop7      51968        51968          0        100% /snap/snap-store/48
1
/dev/loop4      31744        31744          0        100% /snap/snapd/9607
/dev/loop2      146304       146304          0        100% /snap/code/48
/dev/loop5      51072        51072          0        100% /snap/snap-store/46
7
/dev/loop0      56704        56704          0        100% /snap/core18/1885
/dev/loop6      145792       145792          0        100% /snap/code/49
/dev/loop3      62464         62464          0        100% /snap/core20/634
/dev/loop12     31744        31744          0        100% /snap/snapd/9721
/dev/loop8      223232       223232          0        100% /snap/gnome-3-34-18
04/60
/dev/loop10     100096       100096          0        100% /snap/core/10185
/dev/loop1      56704        56704          0        100% /snap/core18/1932
/dev/loop9      261760       261760          0        100% /snap/gnome-3-34-18
04/36
/dev/loop11     63616         63616          0        100% /snap/gtk-common-themes/1506
/dev/sdb5      19553516    4547236  13989960         25% /home
tmpfs          809848          36   809812          1% /run/user/1000
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 5 – Информация о дисковой памяти

Теперь перейдём к выполнению заданий из раздела вывода информации о процессах.

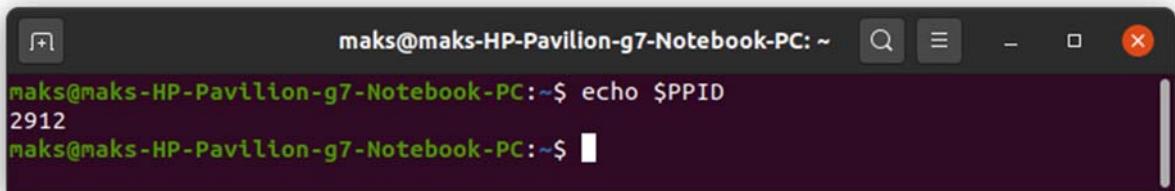
Первым заданием будет получение идентификатора текущего процесса (PID). Сделаем это с помощью команды **echo \$\$**:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ echo $$  
2920  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 6 – Идентификатор текущего процесса

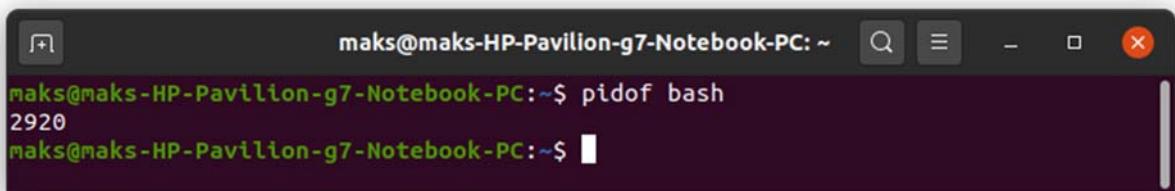
Следующим шагом получим идентификатор родительского процесса (PPID). Используем команду **echo \$PPID**:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ echo $PPID  
2912  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 7 – Идентификатор родительского процесса

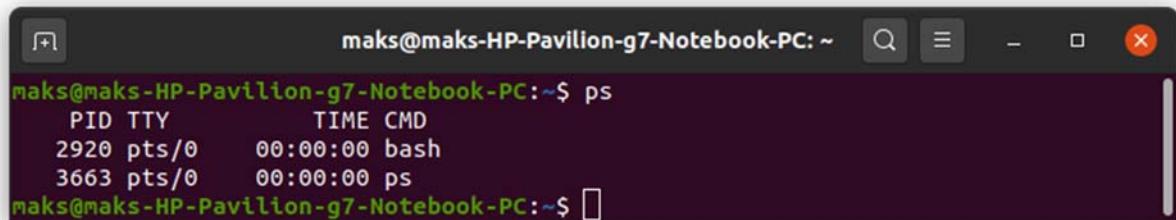
Для выполнения следующего задания, а именно получения идентификатора процесса инициализации системы, нам понадобится команда **pidof**, которая выводит идентификатор процесса по названию процесса:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ pidof bash  
2920  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 8 – Идентификатор процесса инициализации системы

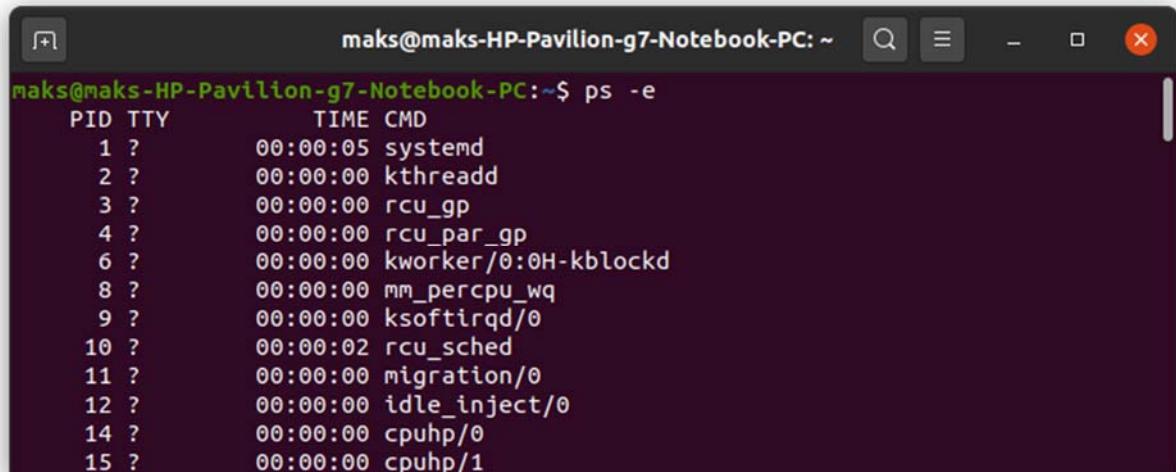
Теперь выведем информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд. Для этого нам нужно исполнить команду **ps** без аргументов:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ps
  PID TTY      TIME CMD
 2920 pts/0    00:00:00 bash
 3663 pts/0    00:00:00 ps
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 9 – Процессы текущего пользователя в текущем интерпретаторе

Если же нам нужно вывести вообще все процессы, то нам нужно выполнить команду **ps** с аргументом **-e**:

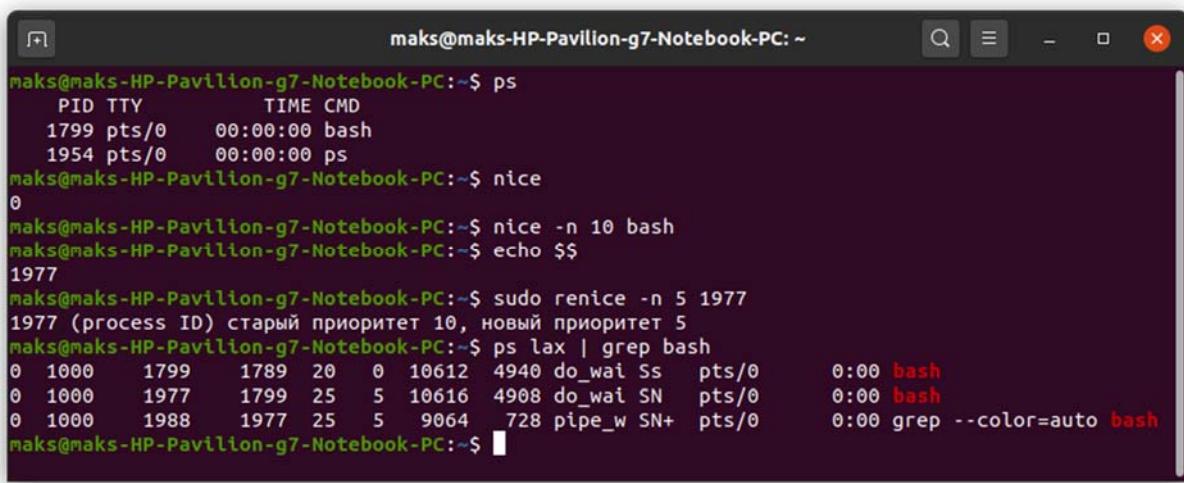


```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ps -e
  PID TTY      TIME CMD
    1 ?        00:00:05 systemd
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 rcu_gp
    4 ?        00:00:00 rcu_par_gp
    6 ?        00:00:00 kworker/0:0H-kblockd
    8 ?        00:00:00 mm_percpu_wq
    9 ?        00:00:00 ksoftirqd/0
   10 ?       00:00:02 rcu_sched
   11 ?       00:00:00 migration/0
   12 ?       00:00:00 idle_inject/0
   14 ?       00:00:00 cpuhp/0
   15 ?       00:00:00 cpuhp/1
```

Рисунок 10 – Все процессы

В последнем разделе нам придётся познакомиться с командами управления процессами.

Выполним следующие действия: выведем информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд (**ps**) и определим текущее значение **nice** по умолчанию (**nice**). Затем запустим новый экземпляр интерпретатора **bash** с понижением приоритета (**nice -n 10 bash**) и узнаем его идентификатор (**echo \$\$**). Затем установим приоритет запущенного интерпретатора равным 5 (**renice -n 5 1977**) и получим информацию о процессах **bash** (**ps lax | grep bash**). Сделаем это:



```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ps
  PID TTY      TIME CMD
 1799 pts/0    00:00:00 bash
 1954 pts/0    00:00:00 ps
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ nice
0
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ nice -n 10 bash
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ echo $$
1977
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ sudo renice -n 5 1977
1977 (process ID) старый приоритет 10, новый приоритет 5
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ps lax | grep bash
 0 1000    1799    1789  20   0 10612 4940 do_wai Ss    pts/0      0:00 bash
 0 1000    1977    1799  25   5 10616 4908 do_wai SN    pts/0      0:00 bash
 0 1000    1988    1977  25   5  9064   728 pipe_w SN+   pts/0      0:00 grep --color=auto bash
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 11 – Операции с созданным интерпретатором

## Вывод

В ходе выполнения лабораторной работы мы познакомились со средствами управления процессами в ОС Ubuntu и изучили некоторые команды, способные помочь в работе с процессами в Linux.

## Ответы на контрольные вопросы

1. Перечислите состояния задачи в ОС Ubuntu.

- 1) Running (выполнение) – после выделения ей процесса;
- 2) Sleeping (спячки) – состояние блокировки;
- 3) Stopped (останов) – остановка работы;
- 4) Zombie (зомби) – выполнение задачи прекратилась, однако она не была удалена;
- 5) Dead (смерть) – задача может быть удалена из системы;
- 6) Active (активный) и inspired (неактивный) – используются при планировании выполнения процесса.

2. Как создаются задачи в ОС Ubuntu?

В системе Linux и процессы, и потоки называют задачами (**task**).

Изнутри они представляют собой единую структуру данных.

Планировщик процессов хранит список всех задач в виде двух структур данных.

Первая структура представляет собой кольцевой список, каждая запись которого содержит указатели на предыдущую и последующую задачу. Обращение к этой структуре происходит в том случае, когда ядру необходимо проанализировать все задачи, которые должны быть выполнены в системе.

Второй структурой является хэш-таблица. При создании задачи ей присваивается уникальный идентификатор процесса (**process identifier, PID**). Идентификаторы процессов передаются хэш-функции для определения местоположения процесса в таблице процессов. Хэш-метод обеспечивает быстрый доступ к специфическим структурам данных задачи, если ядру известен ее PID.

Каждая задача таблицы процессов представляется в виде структуры **task\_struct**, служащей в роли дескриптора процесса (т.е. блока управления процессором (PCB)). В структуре **task\_struct** хранятся переменные и вложенные структуры, описывающие процесс.

3. Назовите классы потоков в ОС Ubuntu.

- 1) Потоки реального времени, обслуживаемые по алгоритму FIFO;
- 2) Потоки реального времени, обслуживаемые в порядке циклической очереди;
- 3) Потоки разделения времени.

4. Как используется приоритет планирования при запуске задачи?

Планировщик использует приоритет и квант следующим образом. Сначала он вычисляет называемую в системе Linux «добротелью» (**goodness**) величину каждого готового процесса по следующему алгоритму:

```
if (class == real_time) goodness = 1000 + priority;  
if (class == timesharing & quantum > 0) goodness = quantum + priority;  
if (class == timesharing && quantum == 0) goodness = 0;
```

В остальном алгоритм планирования очень прост: когда нужно принять решение, выбирается поток с максимальным значением «добротели». Во время работы процесса его квант (переменная **quantum**) уменьшается на единицу на каждом тике. Центральный процессор отнимается у потока при выполнении одного из следующих условий:

1. Квант потока уменьшился до 0.
2. Поток блокируется на операции ввода-вывода и т.д.
3. В состояние готовности перешел ранее заблокированный поток с более высокой «добротелью».

5. Как можно изменить приоритет для выполняющейся задачи?

С этим может помочь команда `renice -n <новое значение приоритета> <PID запущенного процесса>`