Министерство образования Республики Беларусь

Учреждение образования

“Гомельский государственный университет имени Ф. Скорины”

Отчет по лабораторной работе

Лабораторная работа №3. [Команды Linux для работы с процессами](https://learn.dlink.ru/mod/resource/view.php?id=1458)

Выполнил:

Студент группы КФ-17

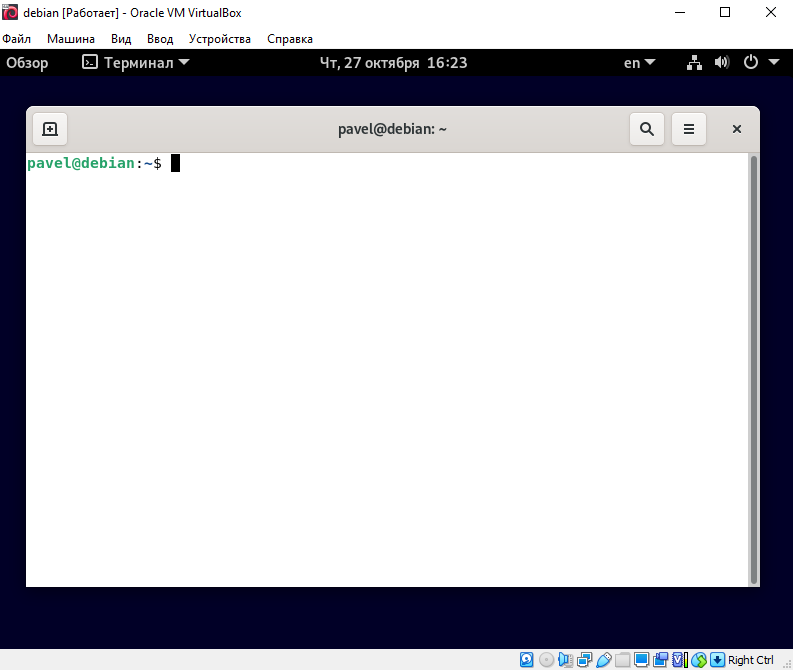
Гуревич П.А.

Гомель 2022

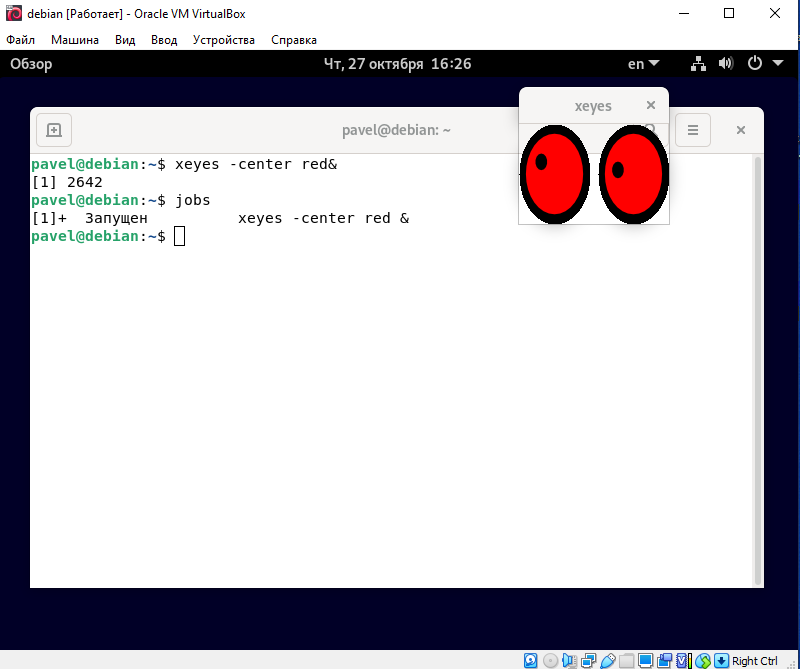
**Цель:** изучение команд Linux для работы с процессами.

ЗАДАНИЕ:

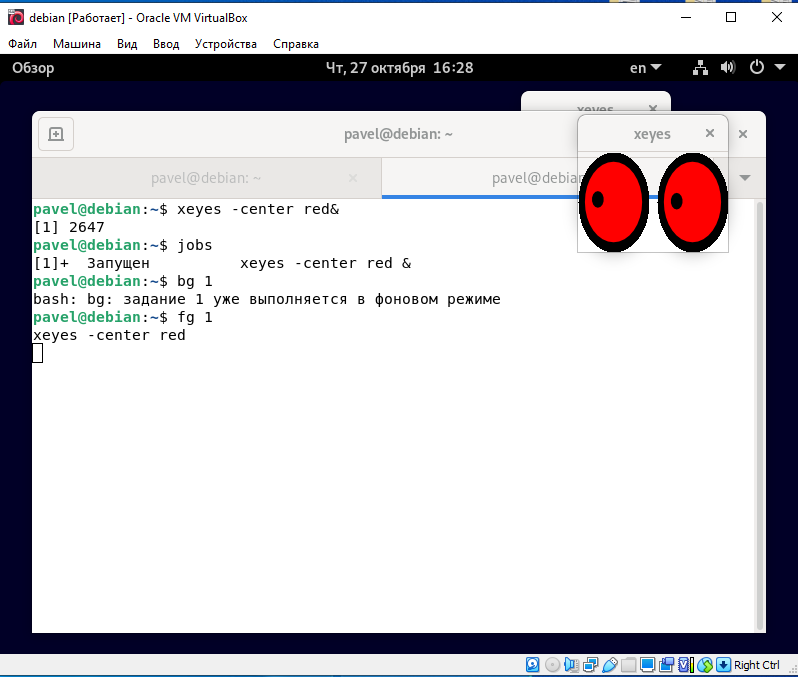
1. Terminal



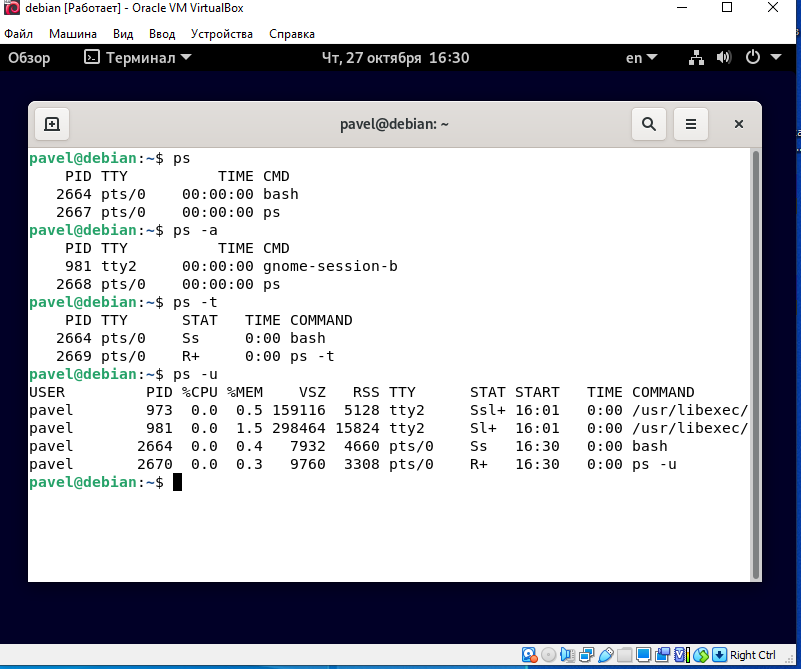
1. Jobs (Команда **top**. Данная команда так же выводит данные о загрузке процессора и памяти компьютера)



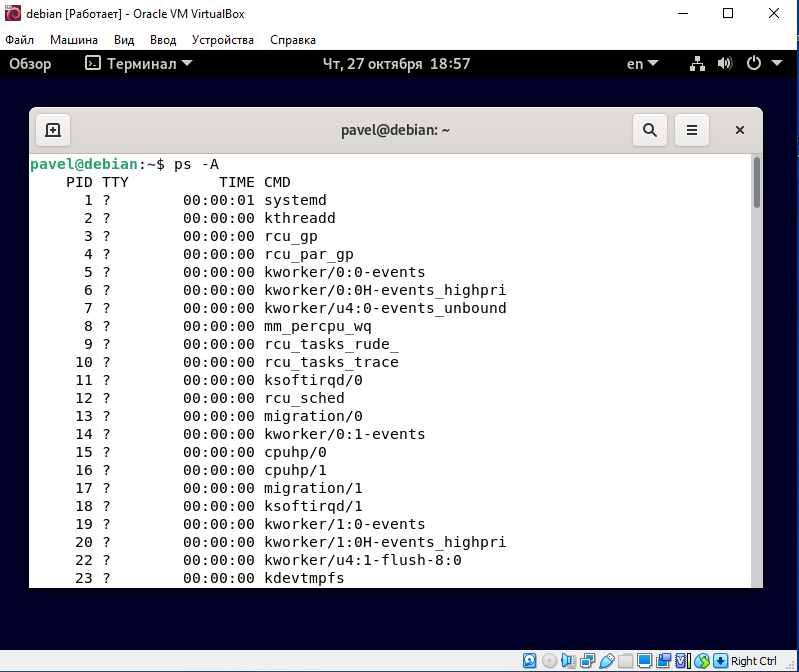
Bg & Fg (Команда **fg** переводит фоновую задачу на передний план. Для просмотра текущих заданий используется команда **jobs**, которая вводится без аргументов.)



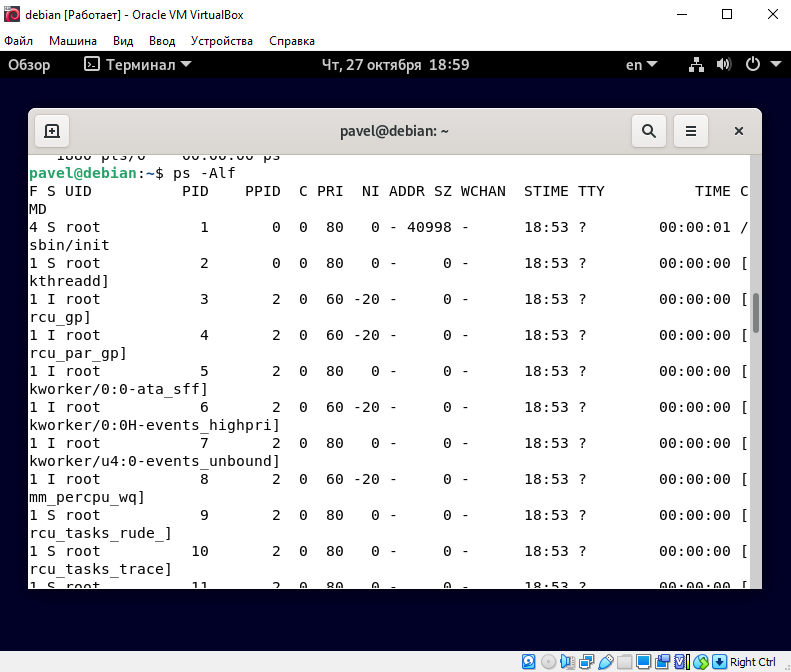
1. Ps (Команда **ps** выводит информацию о запущенных процессах. Если мы хотим просмотреть список всех запущенных процессов, то нужно задать в команде ключ **–A**)



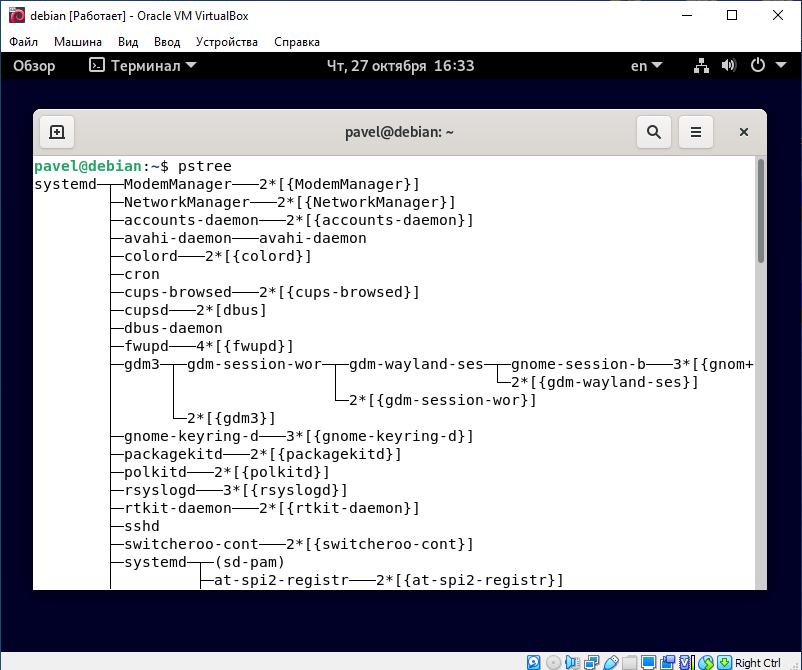
4.1 ps -A



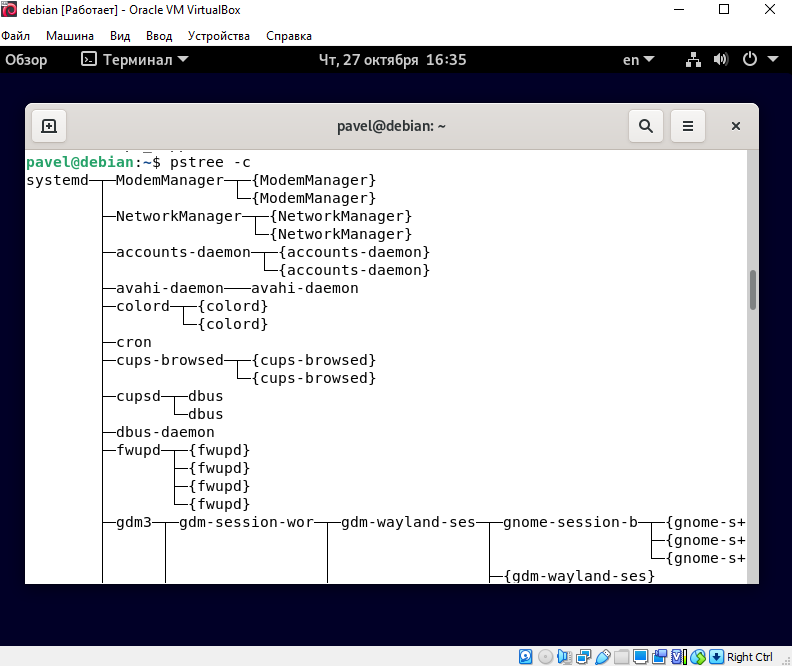
4.2 ps -Alf



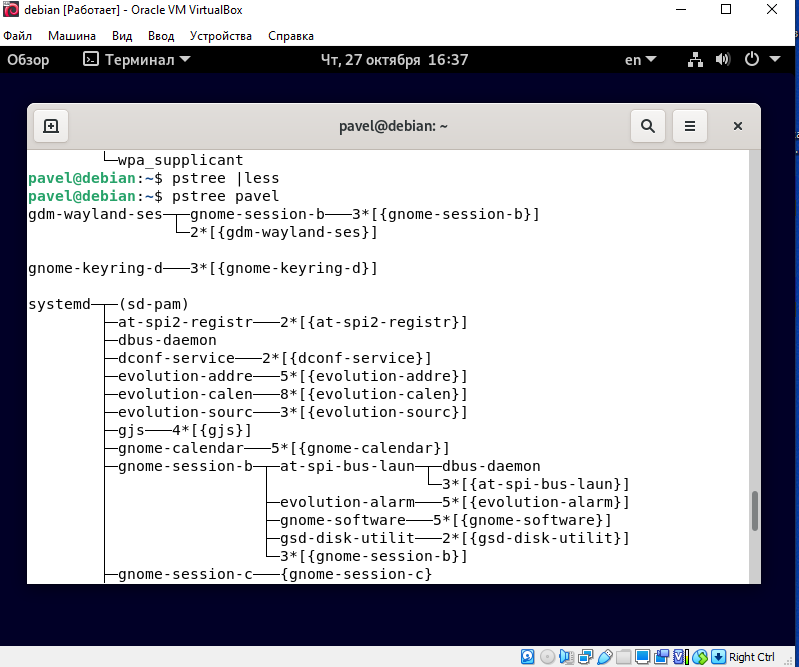
1. pstree



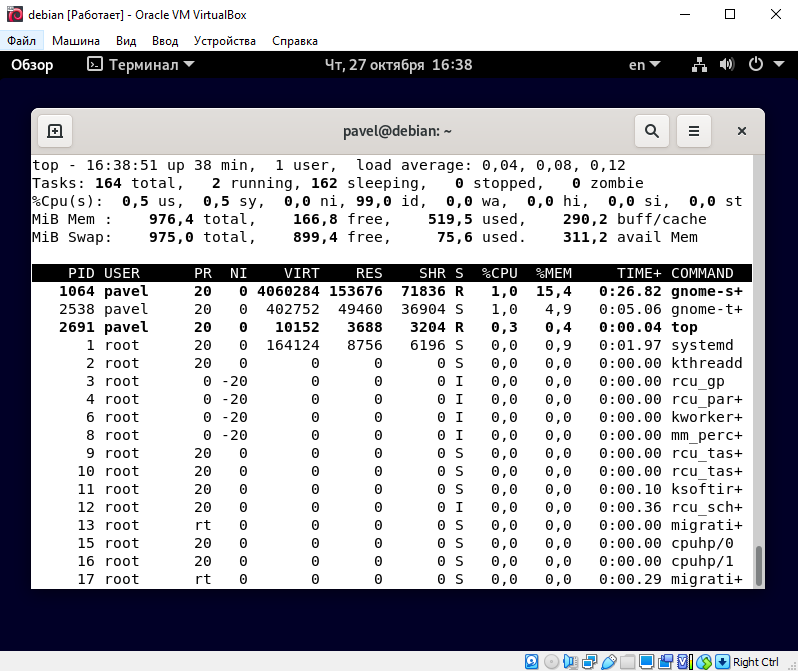
5.1



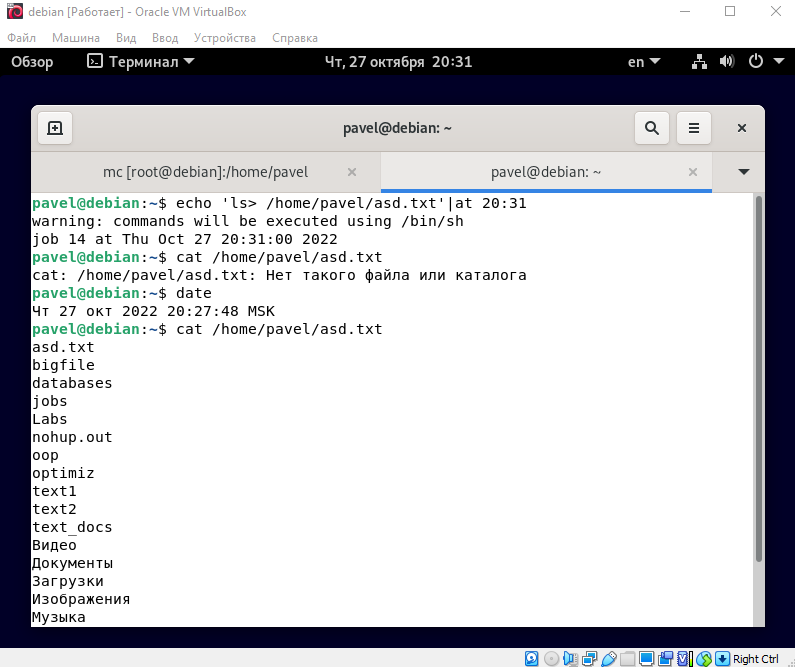
5.2



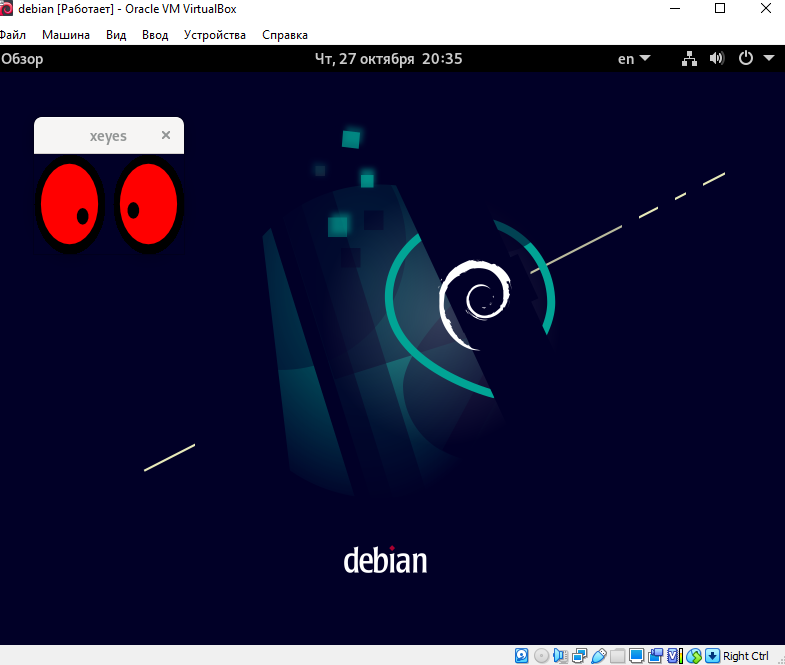
1. Top (Команда **top**. Данная команда так же выводит данные о загрузке процессора и памяти компьютера)

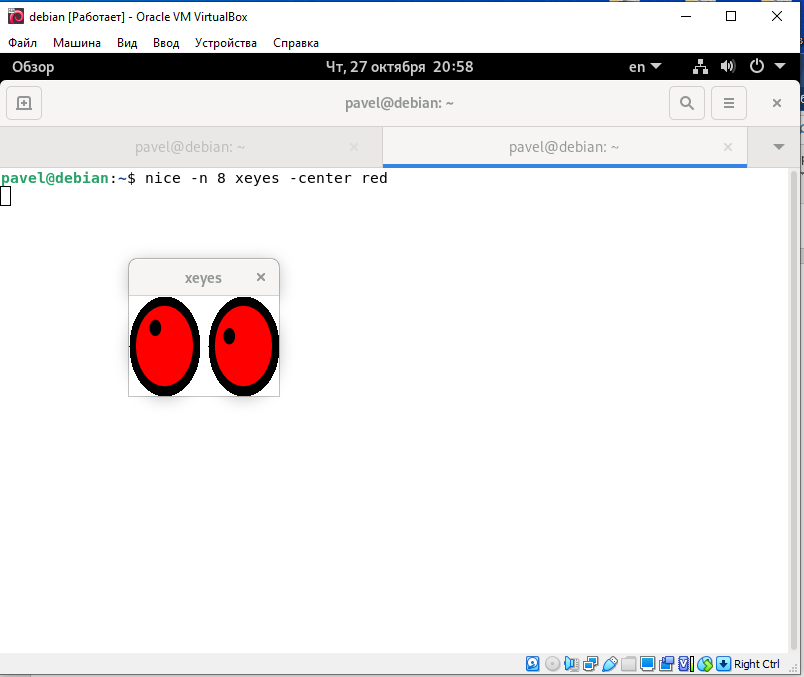


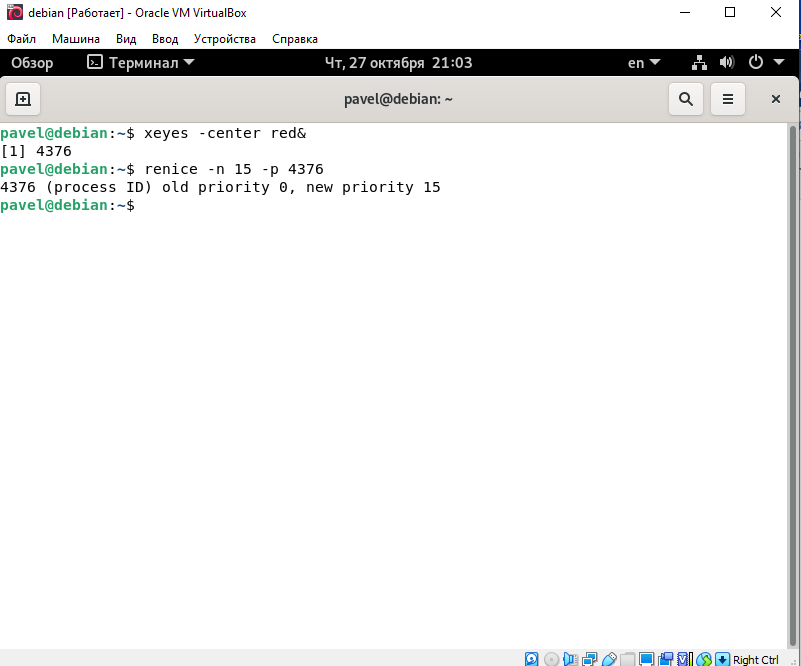
1. At (At – запуск команды по расписанию)



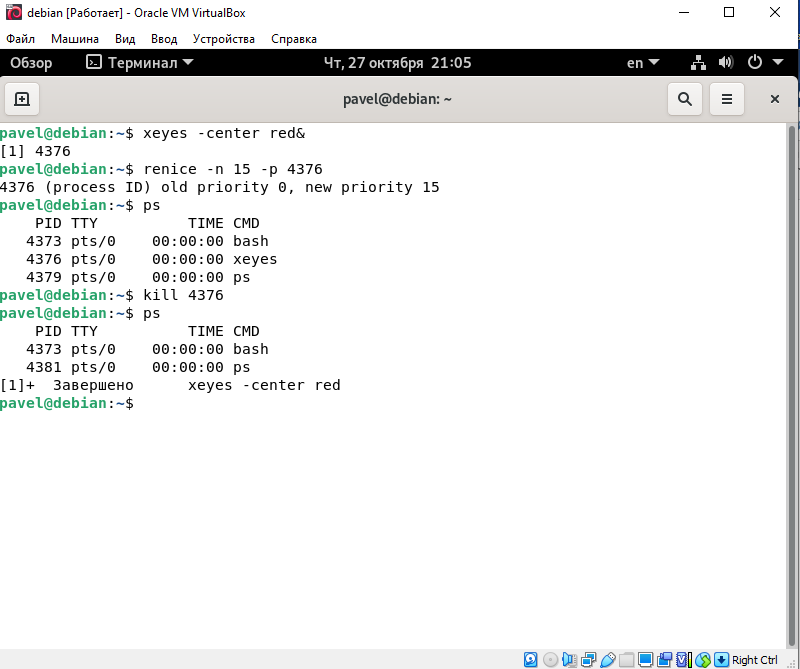
1. Nohup. **(Nohup -** команда благодаря которой программа продолжит выполняться после завершения командной оболочки.)



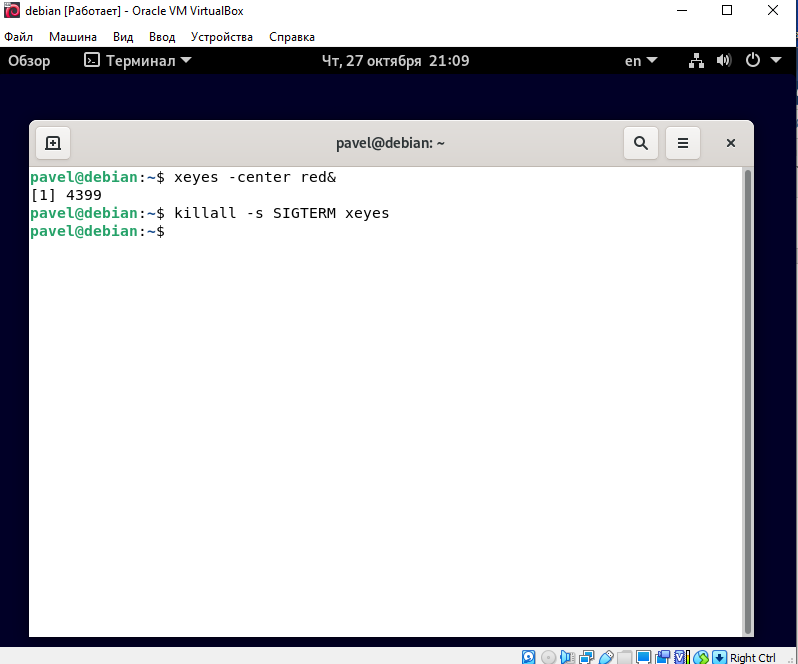
1. Nice (**nice** -задать относительный приоритет некоторой команды.)
2. Renice Команда**(renice**) позволяет изменить относительный приоритет уже работающего процесса.



1. Kill (завершить процессы)



1. Killall(завершить все процессы запущенные одной командой)



**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Процесс ­- это **программа, находящаяся в состоянии исполнения**.

Linux - является **многозадачной** операционной системой, т.е. позволяет выполнять много процессов одновременно. Реализована четкая **иерархия** процессов в системе. Каждый процесс в системе имеет всего одного родителя и может иметь один или более порожденных процессов. Процесс **init** имеет идентификатор равный 1. От него порождаются все остальные процессы. Процесс**init** никогда не завершается.

1. **Идентификатор процесса**(**P**rocess **ID**entifier — **PID**). Каждый процесс в системе имеет уникальный числовой идентификатор. Обычно процессы, запущенные позднее, имеют большее значение PID. Значение PID используется в ряде команд для задания процесса.

**Идентификатор родительского процесса** (**P**arent **PID**— **PPID**). Данный атрибут процесс получает во время своего запуска. Он используется для получения статуса родительского процесса.

**Реальный и эффективный идентификаторы пользователя** (**UID**, **EUID**) и **группы** (**GID**, **EGID**). Данные атрибуты процесса говорят о его принадлежности к конкретному пользователю и группе пользователей.

Процесс может в текущий момент находиться в одном из следующих **состояний**:

* *Runable*(R) — процесс выполняется или готов к выполнению.
* *Sleeping*(S) — процесс находится в состоянии ожидания некоторого внешнего события (например, завершения операции ввода-вывода) и не выполняется. При получении сигнала [1] процесс временно прерывает ожидание для обработки сигнала. Поэтому такие процессы можно завершить до окончания операции ввода-вывода.
* *Uninterruptible*(D) — состояние аналогично предыдущему с той разницей, что ожидание не прерывается для обработки сигналов.Такой процесс нельзя завершить до окончания операции ввода-вывода.
* *Stopped*(T) — процесс остановлен. Процесс переходит в это состояние когда получает соответствующий сигнал. Так же в этом состоянии находится процесс, если производится его отладка.
* *Zombie*(Z) — процесс-«зомби». Этот процесс завершил свое выполнение и почти полностью выгружен из памяти. Единственная информация, которая хранится о данном процессе — *код его завершения*[2], значение которого может потребоваться родительскому процессу. После запроса кода завершения родительским процессом, процесс-«зомби» полностью удаляется из памяти.

**PID – идентификатор процесса, используется в ряде команд для задания процесса**

1. Когда мы запускаем программы обычным способом, они запускаются как программы **переднего плана** (**foreground**). Это значит, что после запуска такой программы мы будем видеть на терминале данные, которые она выводит, и можем вводить с клавиатуры данные, запрашиваемые программой.

Если же программа выполняет какие-либо операции, не требующие реакции пользователя в течение длительного времени, например, сложные расчеты или операции обмена данными с диском, то имело бы смысл позволить пользователю во время работы таких программ вводить другие команды. Это можно сделать, запустив команду в **фоновом режиме** **background**). Для запуска команды в фоновом режиме в конце команды нужно указать знак **«&».**

Kоманда **bg** продолжает выполнение приостановленной задачи в фоновом режиме. Команда **fg** переводит фоновую задачу на передний план. Для просмотра текущих заданий используется команда **jobs**, которая вводится без аргументов.

1. Команда **ps** выводит информацию о запущенных процессах. Если мы хотим просмотреть список всех запущенных процессов, то нужно задать в команде ключ **-A** или **-e** (в листинге приведены только начало и конец выдачи). Более полный вывод можно получить, выполнив команду **ps -Al**: Наконец, указав ключ **forest** мы получим отображение иерархии процессов.
2. Команда **top**. Данная команда так же выводит данные о загрузке процессора и памяти компьютера. Команда при запуске создает собственную интерактивную среду с возможностью ввода команд, в которой информация периодически обновляется.

В утилите **top** можно вводить команды с клавиатуры в интерактивном режиме. Наиболее полезные команды:

* h — отображение справки по работе с программой;
* k — отправка сигнала процессу. При этом требуется дополнительно ввести идентификатор процесса, а затем номер или название отправляемого сигнала (по умолчанию используется TERM или 15);
* M — сортировать процессы по объёму занятой ими памяти (поле %MEM). Сверху отображаются процессы, занимающие больше памяти;
* P — сортировать процессы по занятому ими времени CPU (поле %CPU). Этот метод используется для сортировки по умолчанию. Сверху отображаются процессы, занимающие больше процессорного времени;
* u — вывести процессы заданного пользователя, имя которого будет запрошено командой. Необходимо ввести имя пользователя, а не его UID. Если не будет введено никакого имени, будут показаны все процессы;
* i — вывести информацию только о работающих в данный момент процессах (по умолчанию выводятся все процессы, даже спящие). Повторное использование этой команды вернёт вас назад к списку всех процессов;
* r — эта команда используется для изменения динамического приоритета выбранного процесса. При выполнении команды требуется ввести значение идентификатора процесса и новое значение динамического приоритета процесса

Встроенных команд утилиты top

* **h** - вывод справки по утилите;
* **q** или **Esc** - выход из top;
* **A** - выбор цветовой схемы;
* **d** или**s** - изменить интервал обновления информации;
* **H** - выводить потоки процессов;
* **k** - послать сигнал завершения процессу;
* **W** - записать текущие настройки программы в конфигурационный файл;
* **Y** - посмотреть дополнительные сведения о процессе, открытые файлы, порты, логи и т д;
* **Z** - изменить цветовую схему;
* **l** - скрыть или вывести информацию о средней нагрузке на систему;
* **m** - выключить или переключить режим отображения информации о памяти;
* **x** - выделять жирным колонку, по которой выполняется сортировка;
* **y** - выделять жирным процессы, которые выполняются в данный момент;
* **z** - переключение между цветным и одноцветным режимами;
* **c** - переключение режима вывода команды, доступен полный путь и только команда;
* **F** - настройка полей с информацией о процессах;
* **o** - фильтрация процессов по произвольному условию;
* **u** - фильтрация процессов по имени пользователя;
* **V** - отображение процессов в виде дерева;
* **i** - переключение режима отображения процессов, которые сейчас не используют ресурсы процессора;
* **n** - максимальное количество процессов, для отображения в программе;
* **L** - поиск по слову;
* **<>** - перемещение поля сортировки вправо и влево

1. At – запуск команды по расписанию. При использовании команды **at** следует помнить, что команды будут выполняться демоном **cron**, поэтому не будут связаны ни с каким терминалом и не будут привязаны к графической среде. Не нужно ждать появления каких-либо сообщений на терминале или запуска программ XWindow. Факт запуска команд проще всего отследить по изменениям в файловой системе. **Nohup-** команда благодаря которой программа продолжит выполняться после завершения командной оболочки.
2. С помощью задания относительного приоритета можно повысить или понизить приоритет некоторого процесса относительно других процессов. Относительный приоритет может задаваться целым числом от -20 (повышает приоритет процесса) до 19 (понижает приоритет процесса).

**nice** -задать относительный приоритет некоторой команды. Команда**(renice**) позволяет изменить относительный приоритет уже работающего процесса.

1. **Сигнал -** механизм оповещения процессов об асинхронных событиях. Основной командой, с помощью которой можно отправить сигнал процессу, является команда **kill**. Как видно из названия, в большинстве случаев она используется для завершения процесса.  То же самое можно выполнить с помощью одной команды**killall**. Она позволяет завершить все процессы (в общем случае – послать им сигнал), запущенные одной командой