**Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики.**

**Лабораторная работа №1 по предмету**

**Системное программное обеспечение.**

**Кафедра СУиИ.**

**Группа P3140.**

**Вариант 6.**

**Выполнили: Вахвиянова Полина**

**Мельникова Ольга**

**Проверяющие: Глебова Е.С.**

**Мусаев А.А.**

**Санкт-Петербург.**

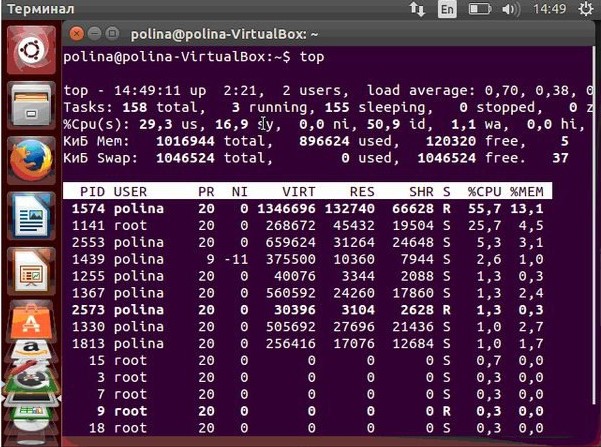
**2016 год.**

**Задание 1.1 Средства просмотра системной информации**

**Команда** **top**

*Команда top* выводит информацию об операционной системе, запущенных процессах, происходящих в операционной системе в реальном времени, о системных ресурсах. Команда *top* позволяет понять причины нестабильной работы операционной системы и обнаружить процессы, потребляющие большинство системных ресурсов.

После запуска команды увидим в терминале вывод, представленный ниже:



**1 строка - общая информация (top)**

Первая строка выводит данные по порядку:

* текущее время (14:49:11)
* время работы системы (up 2:21)
* количество открытых пользовательских сессий (2 users)
* среднюю загрузку системы (load average: 0.70, 0.38, 0.07), три значения соответствуют загрузке в последнюю минуту, пять минут и пятнадцать минут соответственно.

**2 строка - статистика процессов (task)**

Вторая строка выводит следующие данные:

* общее количество процессов в системе (158 total)
* количество работающих в данный момент процессов (3 running)
* количество ожидающих событий процессов (155 sleeping)
* количество остановленных процессов (0 stopped)
* количество процессов, ожидающих родительский процесс для передачи статуса завершения (0 zombie)

**3 строка - статистика использования центрального процессора (cpu)**

В третьей строке приводится информация об использовании центрального процессора. Если вы просуммируете все значения, в результате у вас должно получиться 100%. Давайте рассмотрим предназначение полей этой строки по порядку:

* процент использования центрального процессора пользовательскими процессам (29.3%us)
* процент использования центрального процессора системными процессами (16.9%sy)
* процент использования центрального процессора процессами с приоритетом, повышенным при помощи вызова nice (0.0%ni)
* простой процессора (50,9%id)
* процент использования центрального процессора процессами, ожидающими завершения операций ввода-вывода (1.1%wa)
* процент использования центрального процессора обработчиками аппаратных прерываний (0.0%hi - Hardware IRQ (аппаратные прерывания))
* процент использования центрального процессора обработчиками программных прерываний (0.0%si - Software Interrupts (программные прерывания))
* количество ресурсов центрального процессора "заимствованных" у виртуальной машины гипервизором для других задач (таких, как запуск другой виртуальной машины); это значение будет равно нулю на настольных компьютерах и серверах, не использующих виртуальные машины (0.8%st - Steal Time (заимствованное время)).

**4 и 5 строки - статистика использования памяти (memory usage)**

В четвертой и пятой строке выводится информация об использовании физической оперативной памяти и раздела подкачки соответственно. (SWAP – это специальный раздел на диске или файл, в который операционная система перемещает отдельные блоки оперативной памяти в случае, если оперативной памяти не хватает для работы приложений.) Значения в порядке следования: общее количество памяти (1016944 total), количество используемой памяти (896624 used), количество свободной памяти (120320 free), количество памяти в кэше буферов (5321 buffers).

**Следующие строки - список процессов**

Последним источником информации является список процессов, отсортированный по степени использования центрального процессора (по умолчанию).

* PID - идентификатор процесса (4522)
* USER - имя пользователя, который является владельцем процесса (root)
* PR - приоритет процесса (15)
* NI - значение "NICE", влияющие на приоритет процесса (0)
* VIRT - объем виртуальной памяти, используемый процессом (132m)
* RES - объем физической памяти, используемый процессом (14m)
* SHR - объем разделяемой памяти процесса (3204)
* S - указывает на статус процесса: S=sleep (ожидает событий) R=running (работает) Z=zombie (ожидает родительский процесс) (S)
* %CPU - процент использования центрального процессора данным процессом (0.3)
* %MEM - процент использования оперативной памяти данным процессом (0.7)
* TIME+ - общее время активности процесса (0:17.75)
* COMMAND - имя процесса (bb\_monitor.pl)

Содержимое окна обновляется каждые 5 секунд. Чтобы выйти из программы top, нужно нажать клавишу [q].

Полезные интерактивные команды, которые можно использовать в top:

* [1] Отобразить всю статистику по всем ядрам.
* [c] Абсолютный путь расположения модуля команды и её аргументы.
* [h] Вывести справку о программе.
* [k] Уничтожить процесс. Программа запрашивает у вас код процесса и сигнал, который будет ему послан.
* [M] Сортировать по объёму используемой памяти.
* [n] Изменить число отображаемых процессов. Вам предлагается ввести число.
* [P] Сортировать по загрузке процессора.
* [r] Изменить приоритет процесса.
* [u] Сортировать по имени пользователя.
* [Z] Выбрать цвет подсветки.
* [z] Подсветить работающие процессы.
* [Пробел] Немедленно обновить содержимое экрана.

Список процессов может быть отсортирован по используемому времени ЦПУ (по умолчанию), по использованию памяти, по PID, по времени исполнения. Переключать режимы отображения можно с помощью следующих команд:

<Shift>+<N> — сортировка по PID;

<Shift>+<A> — сортировать процессы по возрасту;

<Shift>+<P> — сортировать процессы по использованию ЦПУ (центрального процессорного устройства);

<Shift>+<M> — сортировать процессы по использованию памяти;

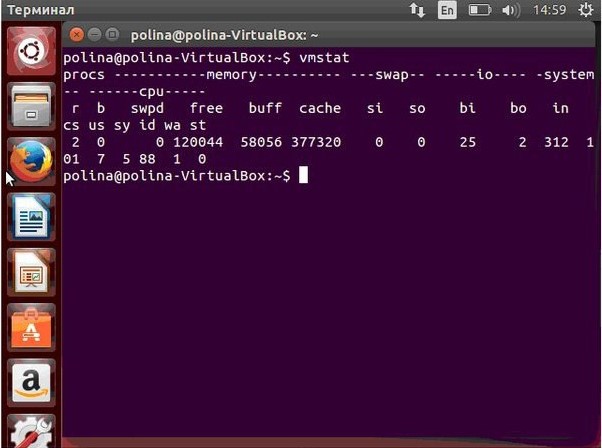
<Shift>+<T> — сортировка по времени выполнения.

Кроме команд, определяющих режим сортировки, команда top воспринимает еще ряд команд, которые позволяют управлять процессами в интерактивном режиме. С помощью команды <K> можно завершить некоторый процесс (его PID будет запрошен), а с помощью команды <R> можно переопределить значение nice для некоторого процесса.

**Команда vmstat (раздел memory, подраздел swpd)**

Команда vmstat позволяет вывести информацию об использовании памяти, дисков, процессора.

Запуск команды будет выглядеть так:



Результат работы команды разбит на шесть колонок – procs (процессы), memory (память), swap(раздел подкачки), io (диск I/O), system (система/ядро), CPU (процессор).

В разделе «Memory» swpd – количество блоков(совокупностей файлов), перемещённых в раздел swap. Swap – это специальный раздел на диске или файл, в который операционная система перемещает отдельные блоки оперативной памяти в случае, если оперативной памяти не хватает для работы приложений; один из механизмов виртуальной памяти. Таким образом, подраздел swpd позволяет отобразить объем используемой виртуальной памяти.

Опции vmstat:

-a – переключение колонки memory в active/inactive значение вместо buff/cache (active – память, реально используемая в данный момент определённым процессом, inactive – память, выделенная ранее какому-то процессу, которого уже нет, buff – объем памяти, занятой под дисковые буферы, cache - объем памяти, занятой под кэш системы);

-f – информация о количестве системных вызовов fork(2), vfork(2) и rfork(2), выполненных с момента старта системы, и количество страниц виртуальной памяти, задействованных каждым из них;

-m – использование динамической памяти ядра, выделенной с использованием malloc(9);

-n – задаёт интервал выполнения в секундах;

-s – изменение вида отображение результатов (в две колонки);

-d – статистика использования дисков;

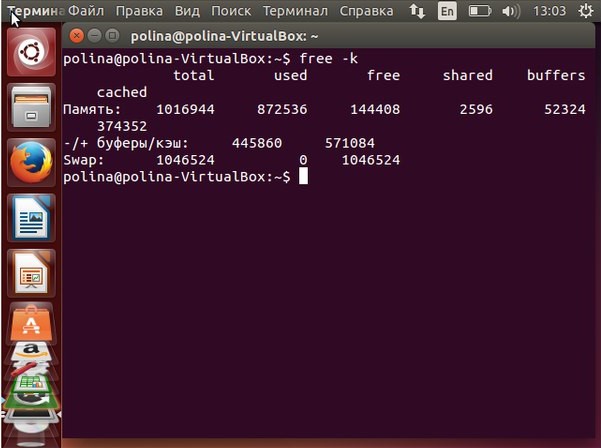
-w – увеличивает визуальный размер поля для больших объемов;

-p [partition name] – информация по конкретном разделу с указанием имени раздела;

-S [k | K | m | M] – с указанием, в чем выводить информацию (килобиты, килобайты, мегабиты, мегабайты соответственно);

-V – версия vmstat.

**Команда *free –k***



*Команда free с опцией –k* позволяет отобразить объем занятой оперативной памяти в килобайтах. Также данная команда предоставляет информацию о разделе подкачки swap.

После запуска команды увидим в терминале вывод, представленный ниже:

(скрин)

Строки обозначают следующее:

* Mem – физическая память;
* Swap – виртуальная память;

Колонки обозначают следующее:

* total – общий размер ОЗУ(оперативного запоминающего устройства);
* used – реально использующаяся в данный момент и зарезервированная системой память;
* free – свободная память (free = total - used);
* shared – разделяемая память;
* buffers – буферы в памяти – страницы памяти, зарезервированные системой для выделения их процессам, когда им это потребуется;
* cached – файлы, которые недавно были использованы системой/процессами и хранящиеся в памяти на случай, если они снова потребуются.

Ключи запуска команды free:

-b – вывод памяти в байтах;

-с - указывает количество раз, доступных для обновления информации, инициируемой командой -s. (“free -s 5 -c 2” отобразит 2 вывода информации о памяти с 5-секундным интервалом)

-g - вывод памяти в гигабайтах;

-k - вывод памяти в килобайтах;

-m - вывод памяти в мегабайтах;

-s - для указания времени обновления информации(free –s “time,sec”);

-t – полная память total (включая swap);

**Задание 1.2 Команды для работы с каталогами, папками и файлами**

**Цель работы**: Получить навыки работы с каталогами, папками и файлами в ОС Linux Ubuntu.

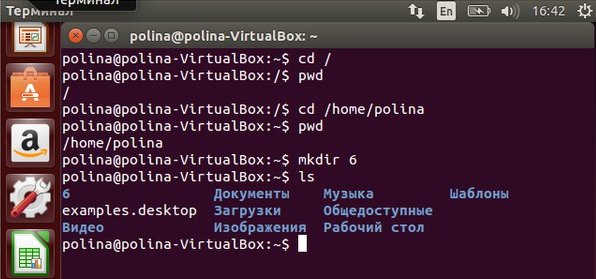
**Порядок выполнения:**

1. Откроем терминал.

2. Вслух, с выражением причитаем задание лабораторной работы и приступим к выполнению.

3. Перейдём в корневой каталог командой cd /. Отобразим текущее положение с помощью команды pwd.

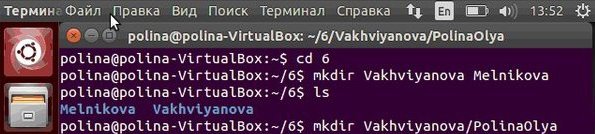
4. Создадим директорию dir1 = номер варианта «6» с помощью команды mkdir. Для этого перейдём в папку /home/polina/, используя абсолютный путь. Выведем содержимое директории, используя команду ls.



Перейдём в директорию 6.

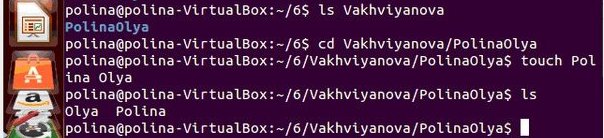
5. Создаём одновременно в dir1 директории dir2, dir4 – фамилии Вахвиянова и Мельникова участников группы. mkdir (..) (..)

6. Не переходя в dir2(Vakhviyanova) создать папку dir3 = PolinaOlya.

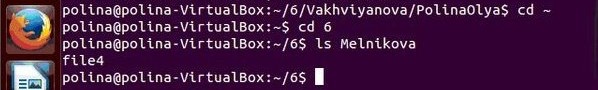
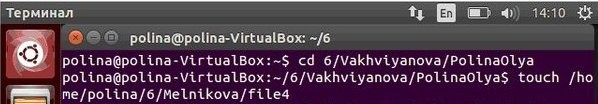


(Подтверждение создания на следующем снимке).

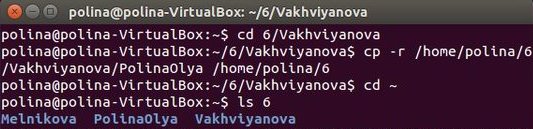
7. Перейдём в папку dir3(PolinaOlya), используя относительный путь. Одновременно создаём два файла file1 и file2 = Polina и Olya, используя команду touch.



8. Создаём file4, не переходя в dir4(Melnikova).



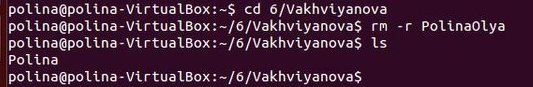
9. Скопируем dir3(PolinaOlya) в dir1(6) со всем содержимым при помощи команды cp с ключом -r.



10. Из dir3(PolinaOlya, той, что в dir2(Vakhviyanova)) скопируем файл с именем Polina, соответствующий фамилии dir2 в папку dir2(Vakhviyanova).

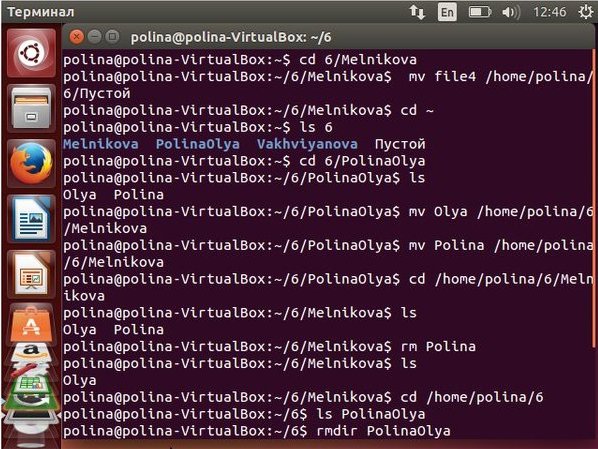


Удалим папку dir3(PolinaOlya) со всем содержимым, для этого испольхуем команду rm и ключ -r.

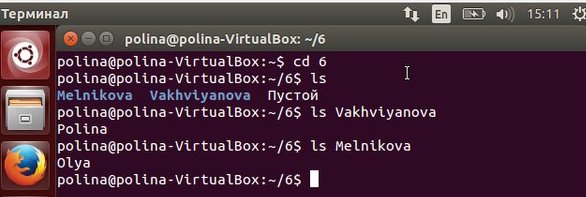


11. Одновременно переименуем file4 в «Пустой» и переместим его в dir1(6). Для этого используем команду mv.

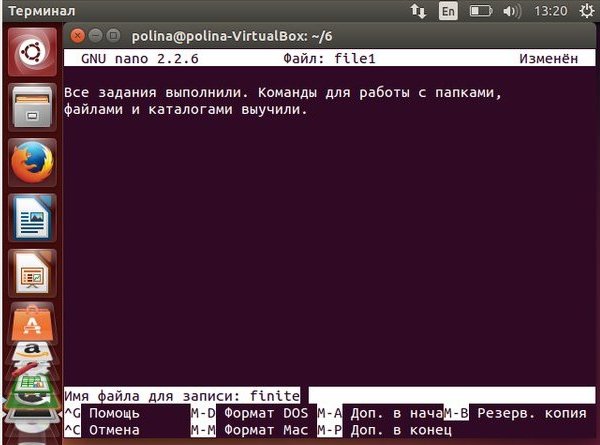
12. Из dir3(PolinaOlya, которая теперь находится в dir1(6)) переместим файлы с именами в dir4(Melnikova). Оставим в папке файл, соответствующий фамилии(Olya), другой файл удалим. Удалим пустую папку dir3(PolinaOlya), использовав команду для удаления пустой папки rmdir.



Просмотрим содержимое папок dir1, dir2,dir4.



13. Перейдём из текущей папки в папку dir1(6), используя специальные символы( cd -). Создаём текстовый документ file1, запустив консольный текстовый редактор командой nano, и запишем в него текст: «Все задания выполнили. Команды для работы с папками, файлами и каталогами выучили». Сохраняем файл под названием finite.(подтверждение на следующем скрине). Прикладываем скрин текстового редактора с введенным текстом.



14. Выводим содержимое файла finite в терминале.

