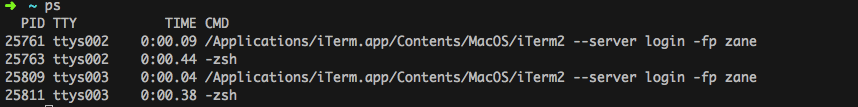
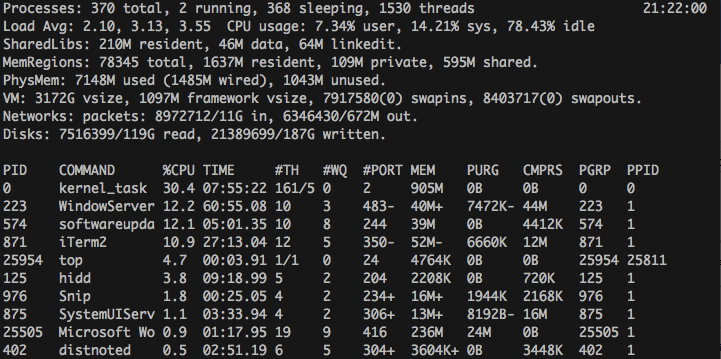
**Experiment 1**: **(using your computer) 1) Run *top* and see what processes the system is using. Try to understand what processes are running and interpret the information displayed for each process, i.e. interpret the columns.**

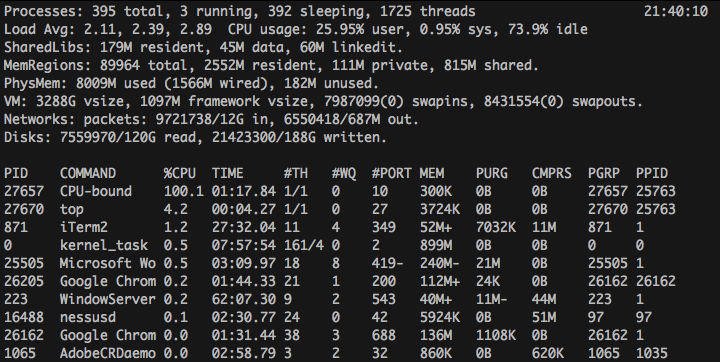


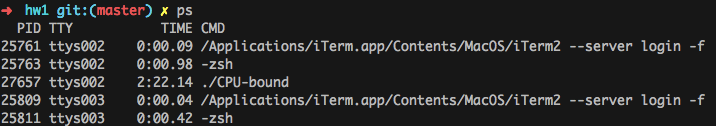


The columns of the top command show information about each process. The PID is the Process ID of the shown process, this is a unique for each process. The Command is the command that is running, such as the name of the file that is running. The %CPU is the percentage of the CPU that is being used by that process. The Time is how long that process has been running for. The MEM is the amount of memory usage that process takes up. The PPID is the Parent Process ID, which tells which process is the parent of the process.

2) **Now start executing a single instance of your *CPU-bound* process in the background. Recall that a process can be executed in the background by using the "&", e.g. *CPU- bound &*. Find out how to start a process, stop it, and start it again, run it in the foreground and send it to the background.**

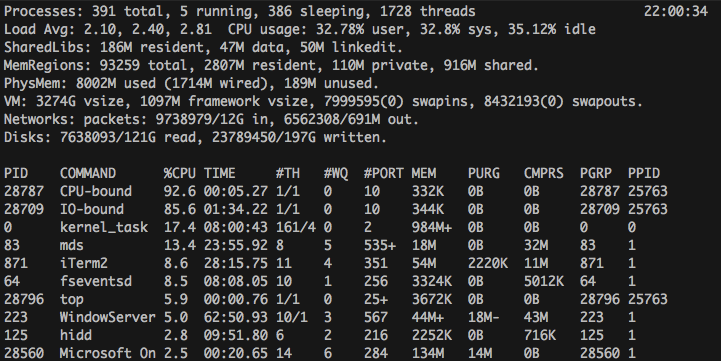
**3) Next investigate the impact of the *CPU-bound* program using ps and top.**

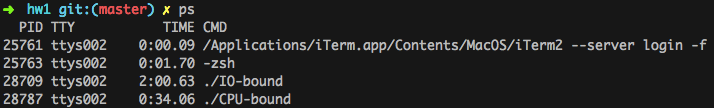
****

****

** **

**4) Now start the *IO-bound* in the background and again check using *ps* and *top* its impact. At this point you should have a single instance of both programs executing. What do you notice about the processes’ CPU utilization?**





I found that the IO-bound process used less CPU than the IO-bound process. However, it doesn’t appear that the IO-bound was running as low as expected. I would expect the reason for this difference is due to my solid-state drive which is able to return from IO commands.   
To support this hypothesis I ran the same IO-bound process on the cs-course60.cs.uidaho.edu machine to verify that the code worked as expect.

**5) Repeat the experiments with multiple instances of the processes:**

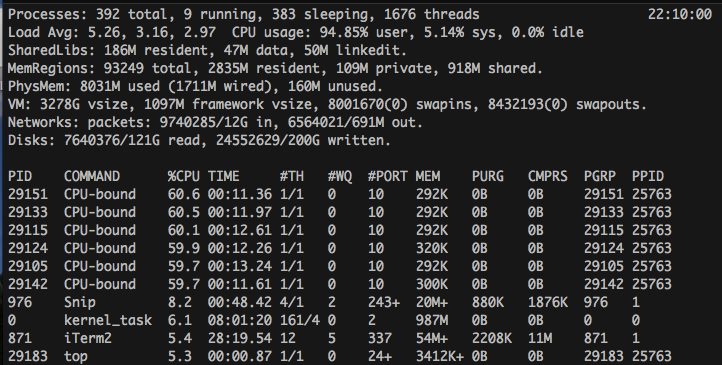
1. **Start multiple instances of your *CPU-bound* program to get the cores working.** 
   1. **What do you notice now about your *CPU-bound* processes?**

The Processes don’t use as much CPU as a single CPU-bound process does, However the Total CPU usage is much higher than when only a single process was running.

When running multiple CPU-bound processes each process only used around 60% of the CPU, but when I only ran a single process it was able to use around 90% of the CPU

* 1. **How many instances of your CPU program did you run and what CPU  utilization did you observe?**

I ran 6 instances of the CPU-bound processes. I observed around 99% CPU utilization in total.



1. **Start multiple instances of your *IO-bound* program.** 
   1. **What do you notice now about your mix of processes?**

I found that the IO-bound processes each used drastically less CPU than when the program was running by itself.  
Each process used about 38% of the CPU, which is much less than when running a single IO bound process.

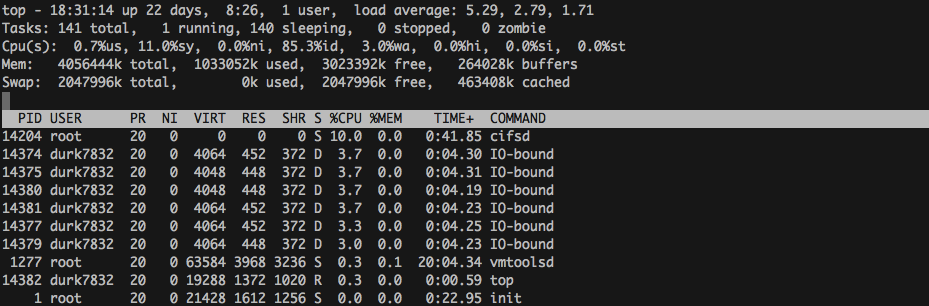
* 1. **How many instances of your IO program should you (did you) run and  what CPU utilization did you observe?**

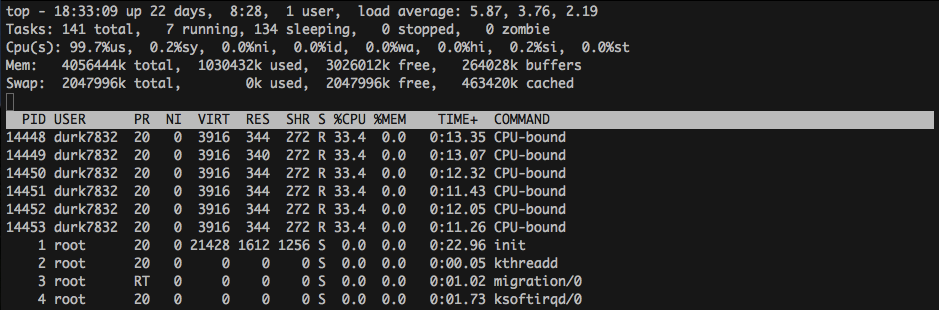
I ran 6 instances of my IO program and I found that the total CPU utilization was around 80% in total.

**6) Once you are sure you can document your findings, kill the processes. How do you find out which processes you should kill? By now you should have figured out how to get the process ID (e.g., using top, ps) and use the man pages on the *kill* command. IMPORTANT: Verify that you actually killed your programs!**

You can find which process that needs to be killed by using the ps or top command to get the process ID of the process you wish to kill. Once you have the PID you can use kill to kill the process with the given PID.

**Experiment 2: Do this only after you are done with Experiment 1, unless you do not own a computer. Gather the information from the experiment on the CS computers. Document differences between the two systems, i.e., did you observe any measurable differences between the CPU and I/O utilization between the target systems?**

  
When running 6 IO-bound processes the CPU usage is much lower (around 3% per process) than when I ran the code on my device (around 80%). The total CPU usage of the 6 processes on the cs-course60.cs.uidaho.edu was around 15%. It appears that my device is able to return from the I/O process much quicker than the remote machine.   
  
Next I ran the CPU-bound tests.

I found that the cs-course60.cs.uidaho.edu server used about half as much CPU percentage per program that my personal machine did. However, the total CPU usage was about the same on the server as it was on my own device.