**Instructions:** The purpose of this lab is to gain confidence working in the Bourne Again Shell (BASH).  Many enterprise UNIX/Linux servers do not have a graphical user interface (GUI) installed.  Knowing your away around the command line will pay dividends in your IT career.

Before you continue, you need to have your virtual machines (VMs) installed per the lab environment setup instructions.  If you are having issues setting your first VM, use the discussion forum to seek help, and as a last resort, contact me via e-mail or through [Bookings](https://outlook.office365.com/owa/calendar/TraceyCoon@oregontech.info/bookings/).

As you work through the lab, I will ask for screenshots of your output.  Paste your screenshots into a Word document with a brief explanation of each screenshot.  Make sure you have a cover page with your full name.

**VM:** CentOS 7 Minimal

**Creating a VM Snapshot**

* Go ahead and fire up your CentOS VM from the VirtualBox Manager.
  + **Important:** Before performing any further steps, you need to create a snapshot of your virtual machine. A snapshot saves the VM’s current state, so if you mess something up, you can revert back to your snapshot.
  + **Create a snapshot:** With your CentOS VM running, go back to your VirtualBox Manager window, click the drop-down arrow next to Machine Tools, Click Snapshots. Click Take and in the Snapshot Name textbox enter: Fresh Install 04/12/2020.  Replace date with today’s date.  Click OK.

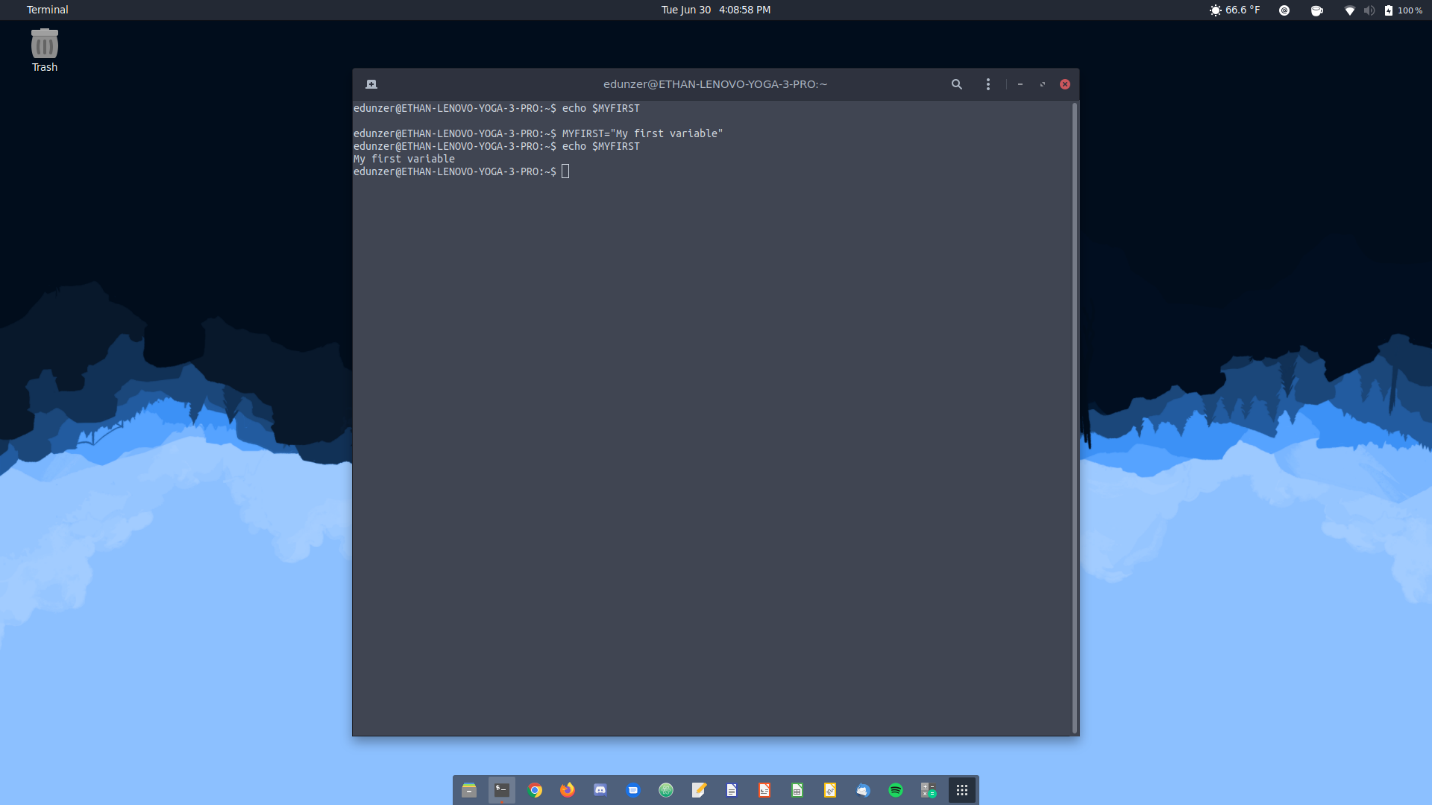
**Give your user admin privileges (sudo)**

* Login to VM with your username and password.
* Switch to the root user with the following command
  + su – (that’s su, space, minus symbol [referred to as tack in Linux])
    - Enter the password of root. Remember, you set this password during installation.  Also, you will notice it doesn’t look like you’re typing anything, this is set by default for security reasons.
  + Add your username to the wheel group. The wheel group is given admin privileges.  Replace <username> with your actual username without the brackets.
    - gpasswd wheel –a <username>
      * Ex: gpasswd wheel –a traceyc
    - Reboot your CentOS VM.

**Getting familiar with BASH (Bourne Again Shell)**

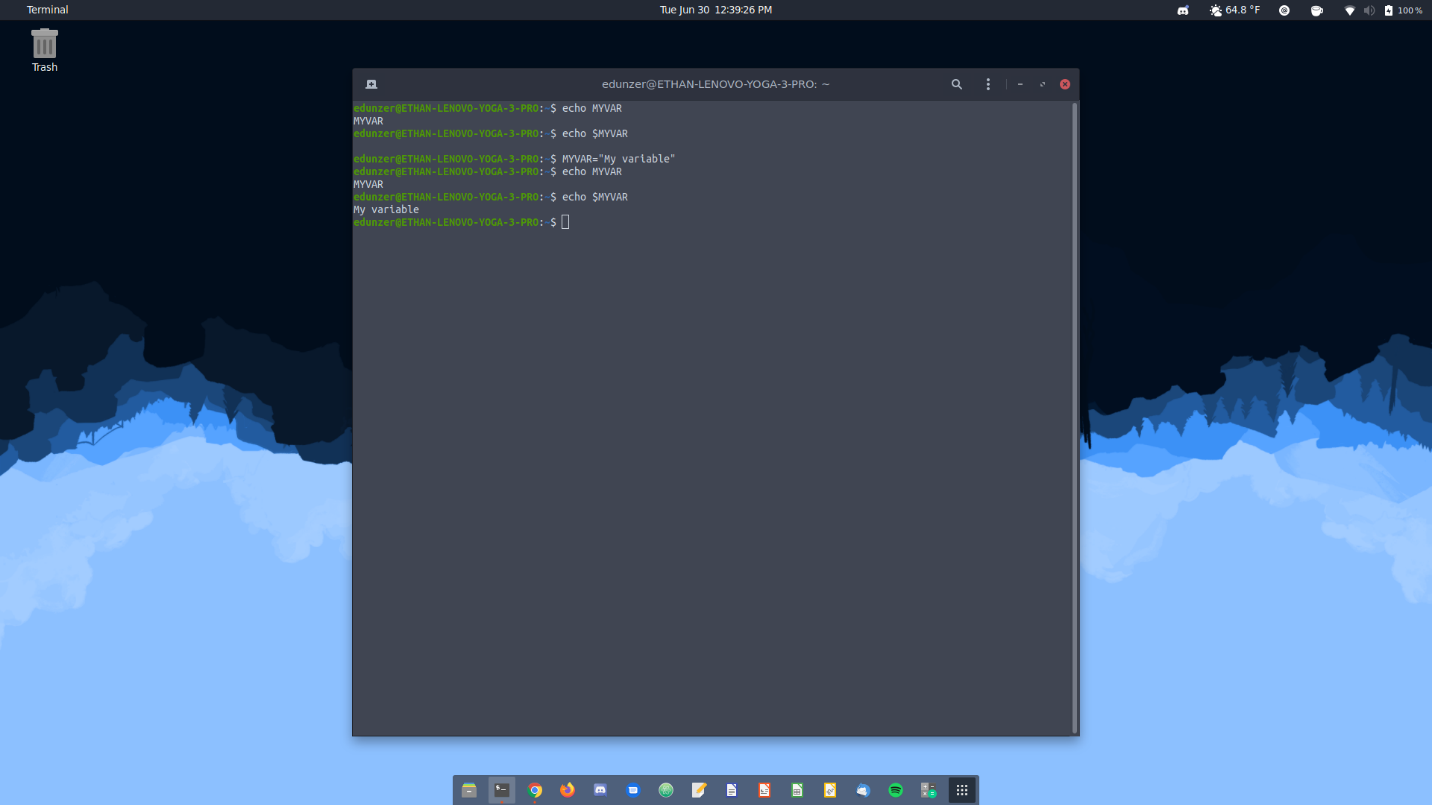
* Variables, like in programming, are used throughout the Linux OS and are used by the shell, as well as other programs.
  + Display all environment variables: set | more
  + Press the space bar to view more variables and when you’re done, press q to quit.
* Use the following commands to create, view, and delete variables within a shell
  + echo $MYFIRST (no output)
  + MYFIRST=”My first variable”
  + echo $MYFIRST (shows your variable)

**\*\*\*Screenshot #1**



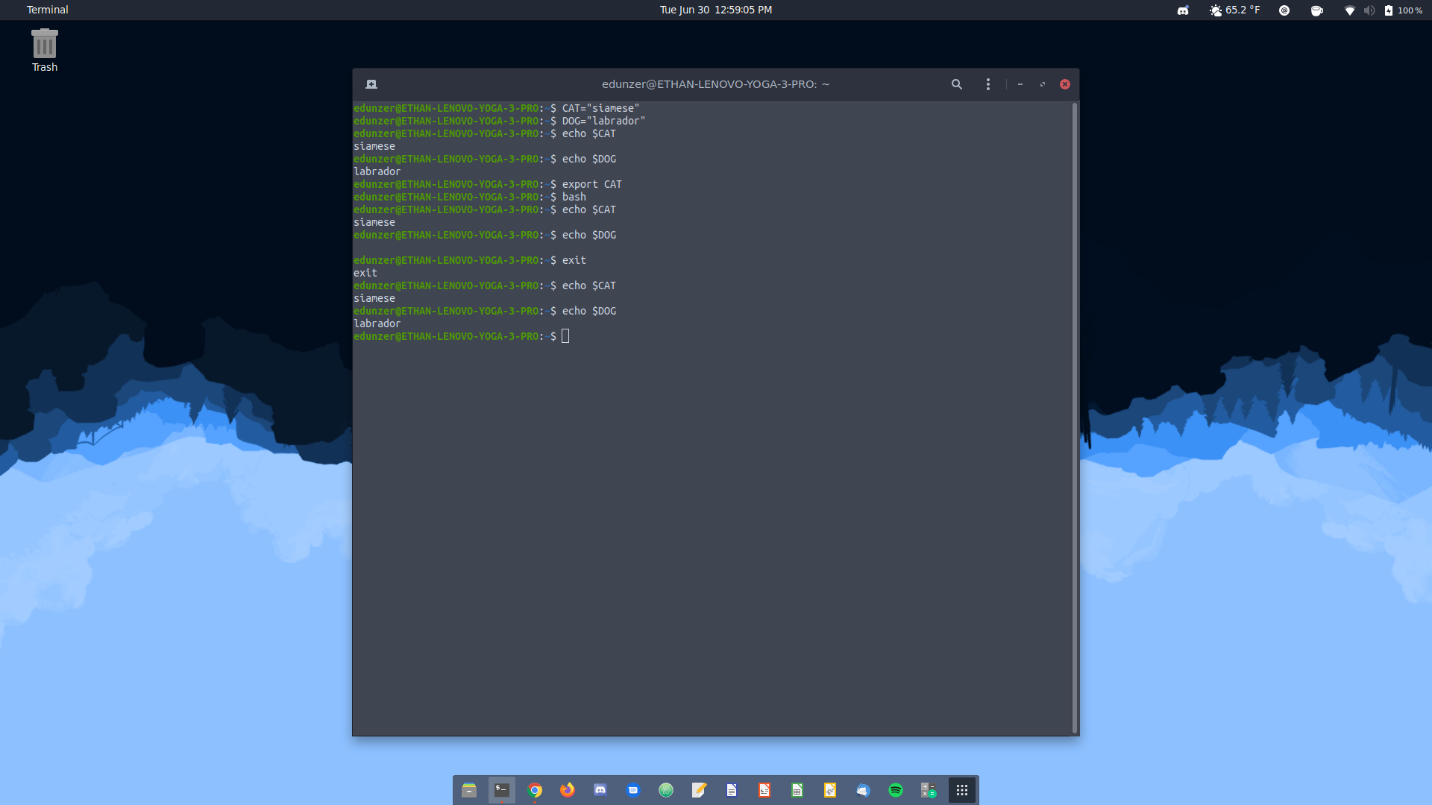
* Notice the importance of placing a $ in front of your variable when you want to reference it. Otherwise, the echo command will simply print all characters immediately following the command.  Enter the following commands pressing the Enter key after each bullet:
  + echo MYVAR
  + echo $MYVAR
  + MYVAR=”My variable”
  + echo MYVAR
  + echo $MYVAR

**\*\*\*Screenshot #2**



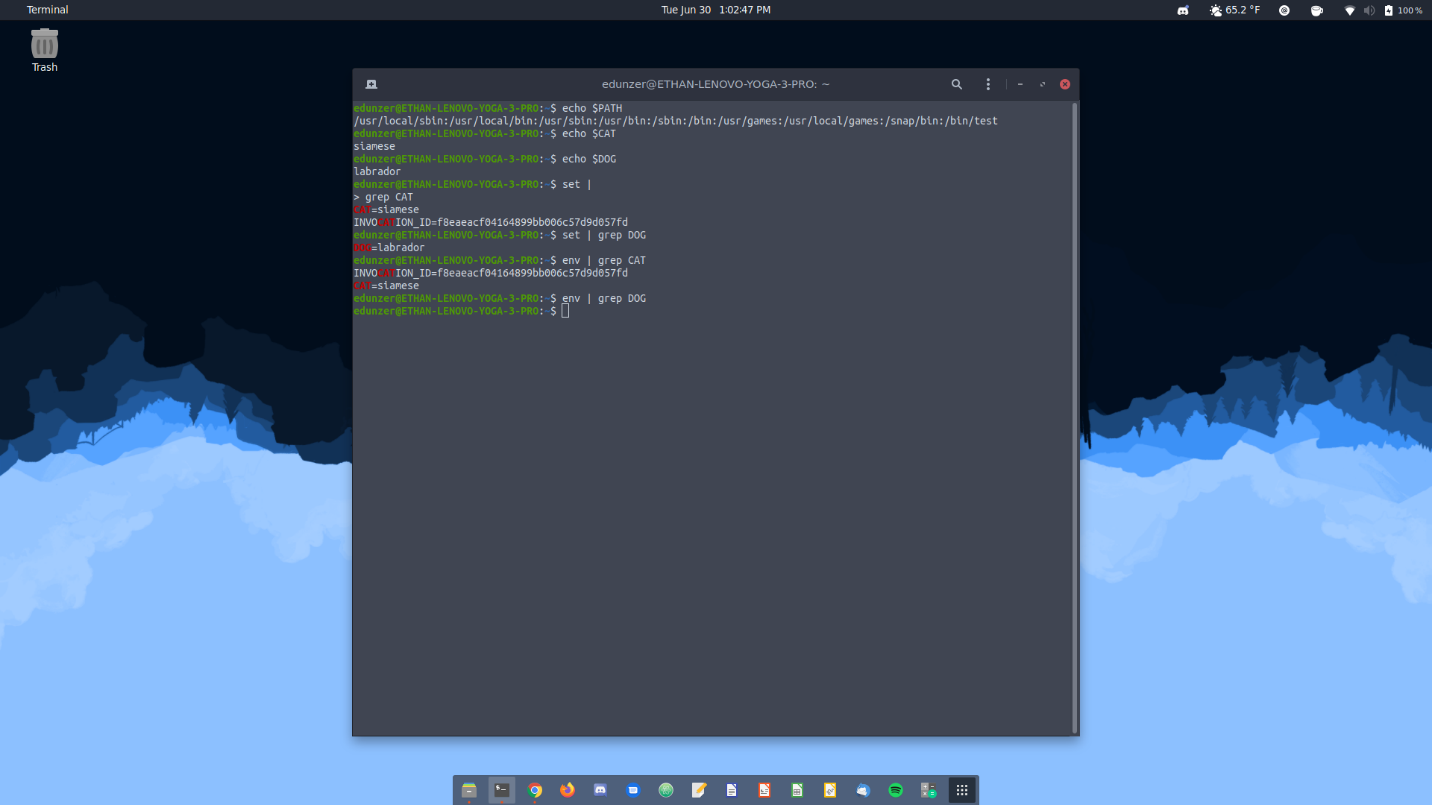
* Once you create a variable, you can only access that variable via your current shell. To make it available system-wide, you’ll need to export your variable.  Perform the following commands to view the importance of variable export.
  + Create 2 variables, CAT with a value of siamese and DOG with a value of labrador
  + echo $CAT (shows output)
  + echo $DOG (shows output)
  + export CAT
  + bash
  + echo $CAT (shows output)
  + echo $DOG (NO output)
  + exit
  + echo $CAT (shows output)
  + echo $DOG (shows output)

**\*\*\*Screenshot #3**



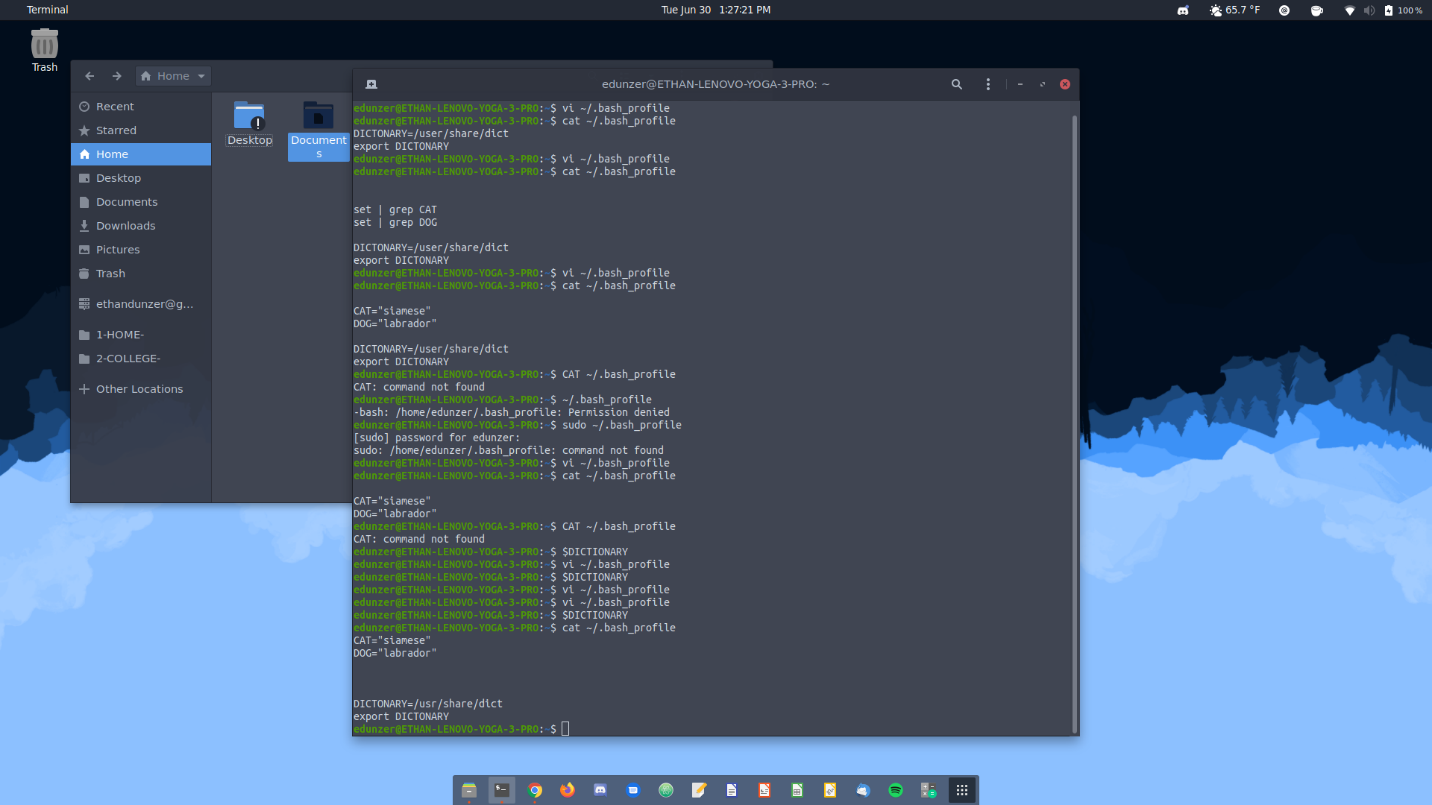
* For a command to be available in your shell, the shell needs to know what directories to look in for that command. To add directories, use the following commands.  Adding directories allows you to add more commands without having to copy into existing directories.
  + echo $PATH (shows current path)
  + PATH=”$PATH:/bin/test”
  + echo $PATH (you should see your new directory at the end)
* The set command will show all shell and environment variables, the env command only shows variables that have been set. Enter these commands:
  + echo $CAT
  + echo $DOG
  + set | grep CAT
  + set | grep DOG
  + env | grep CAT
  + env | grep DOG You shouldn’t get an output here

**\*\*\*Screenshot #4**



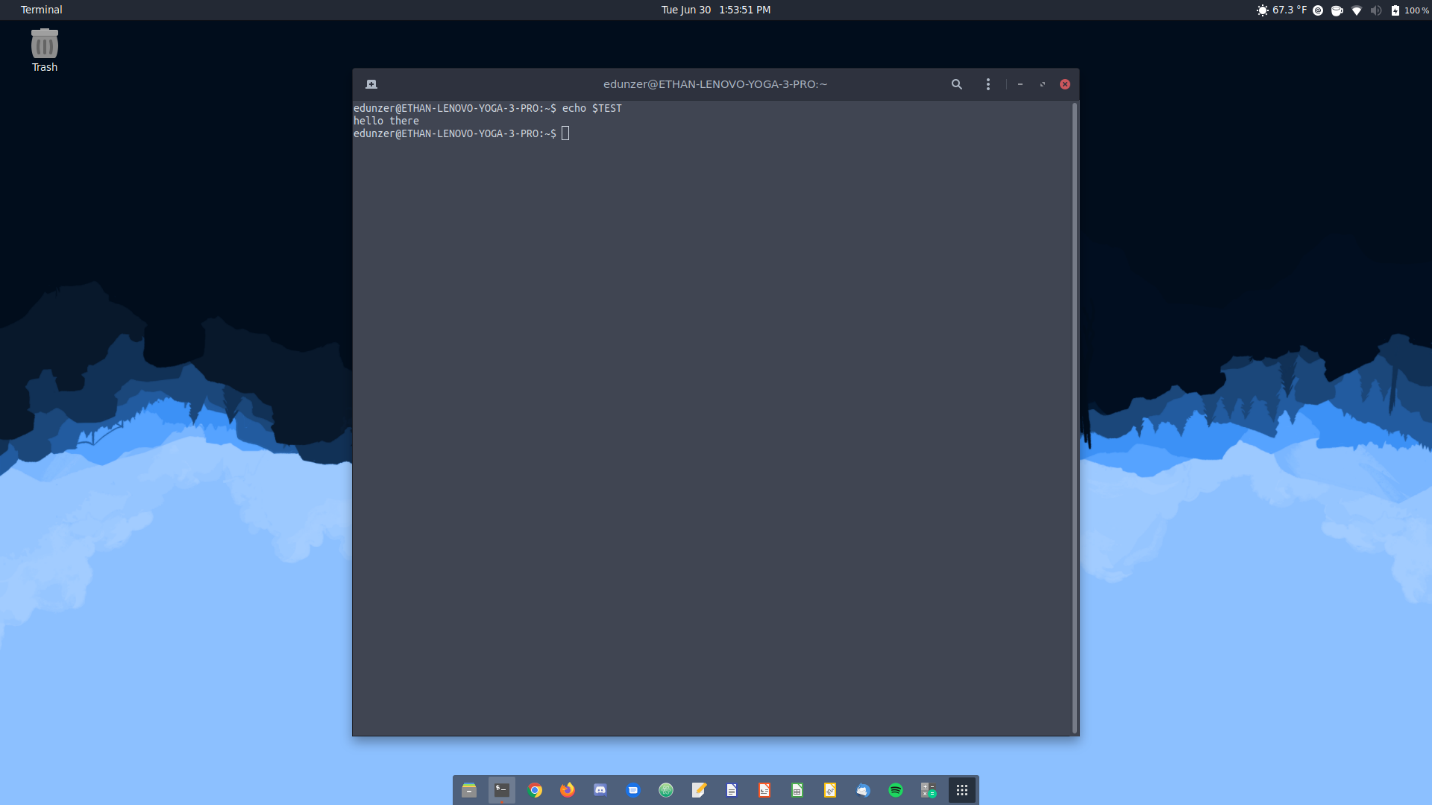
* To make environment shell variables stick permanently, you’ll need to place them in your profile, meaning they’ll be available each time you log on (or reboot)
  + vi ~/.bash\_profile
    - **Note: vi is a text editor and it can be difficult to learn. You might need to Google vi and its commands to get familiar with it.**
  + At the end of the file, enter the following:
    - DICTIONARY=/usr/share/dict
    - export DICTIONARY
  + Save the file and exit vi
  + cat ~/.bash\_profile you should see your new entries at the bottom of the profile
  + Logout of CentOS, and re-login. Open a terminal and type echo $DICTIONARY.  You should see the directory pointing to your new variable.

**\*\*\*Screenshot #5**



* Sometimes you only need variables in the local shell. To permanently set local shell variables, you’ll place them in the ~/.bashrc file.
  + vi ./bashrc
  + Go to the bottom of the file by pressing the o key and enter color=red
  + Save and close vi.
  + cat ~./bashrc you should see your new variable at the bottom of the output
  + echo $color no output
  + bash
  + echo $color with output
  + exit
* This time, you’ll create an environment variable that is available to ALL users. To do this, you’ll need to edit the profile found in the /etc/ directory.
  + sudo /bin/bash
  + vi /etc/profile
  + At the end of the document enter the following
    - TEST=”hello there”
    - export TEST
  + Save and quit vi
  + tail /etc/profile, you should see your entry at the bottom of the output
  + Logout and log back in to CentOS
  + echo $TEST, you should see the proper output

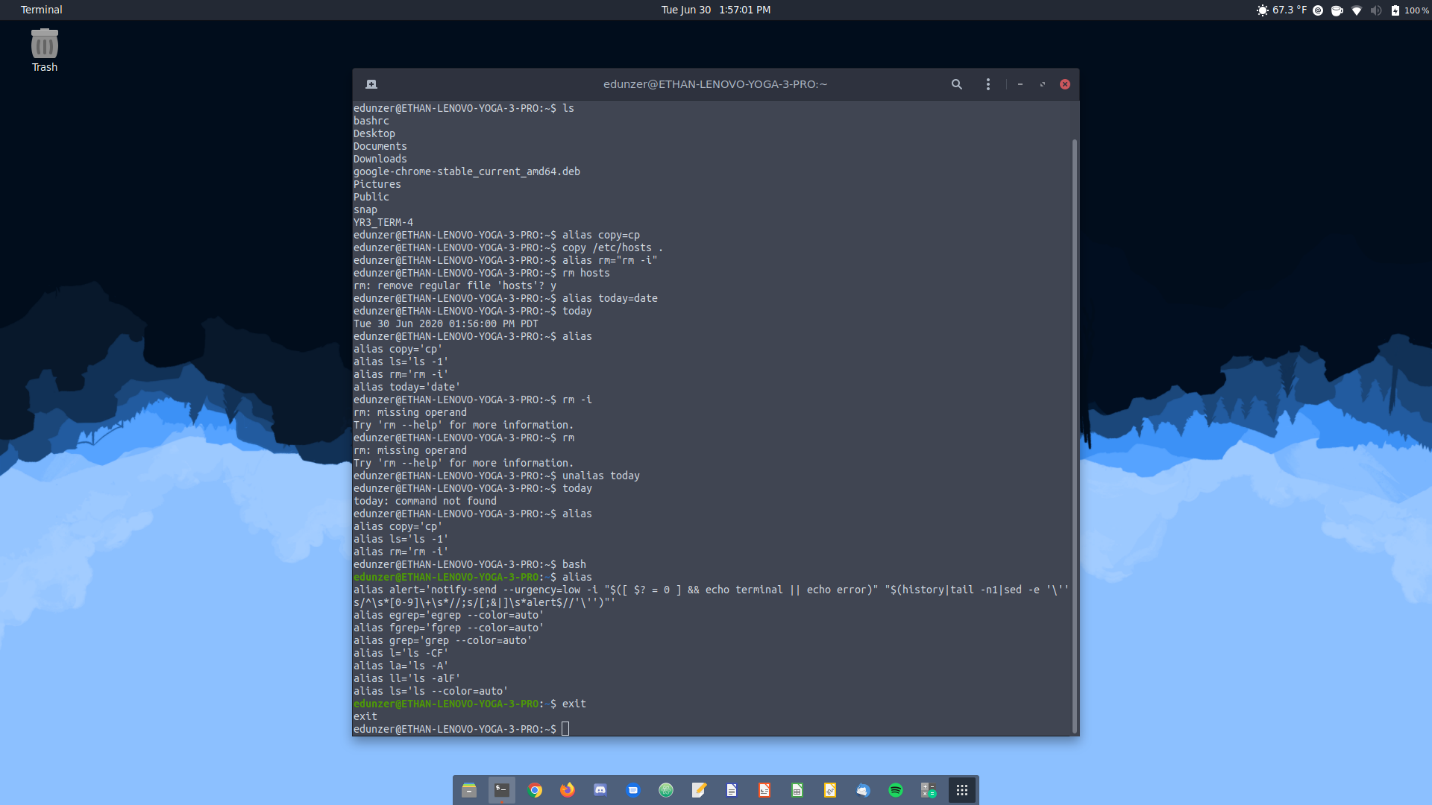
**\*\*\*Screenshot #6**



**Aliases and Functions**

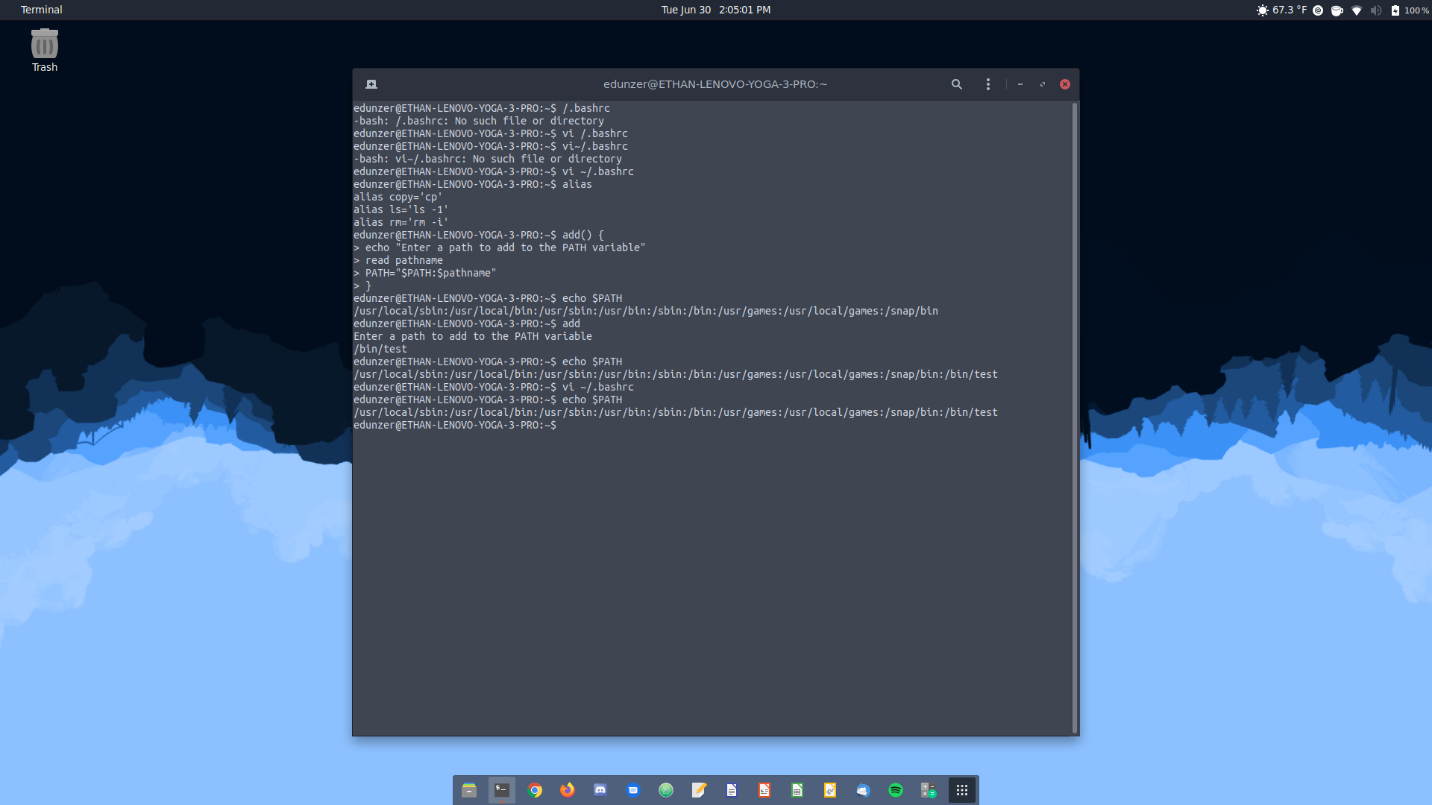
* Run the list (ls) command, take note of the output
* Create an alias for ls
  + alias ls=”ls -l”
* Run the list command again, you should see a different output
* Create the following
  + alias copy=cp
  + copy /etc/hosts . take note of the space and period after /etc/hosts
  + alias rm=”rm -i"
  + rm hosts
    - y
  + alias today=date
  + today
* To see all of your aliases, type alias
* To remove the today alias:
  + unalias today
  + today
* Each shell has its own aliases, see this by doing the following
  + alias
  + bash
  + alias
  + exit

**\*\*\*Screenshot #7**



* Like variables, aliases can be saved in your shell profile. You’ll need to edit the /.bashrc file and enter the alias at the bottom of the file.  For practice, edit the /.bashrc file and enter the alias copy=cp
  + Save and exit your text editor, then use the cat command to see your changes.
  + Verify by using the alias command to see copy=cp
* Aliases help us create functions that are easy to read/create, perform the following (you’ll need to hit enter after each bulleted item)
  + add() {
  + echo “Enter a path to add to the PATH variable”
  + read pathname
  + PATH=”$PATH:$pathname”
  + }
  + echo $PATH
  + add
  + /bin/test
  + echo $PATH
* Finally, the above function is only saved in your current shell. Like everything else, you’ll need to permanently save it in your ~/.bashrc file.  Try performing this action without any assistance.

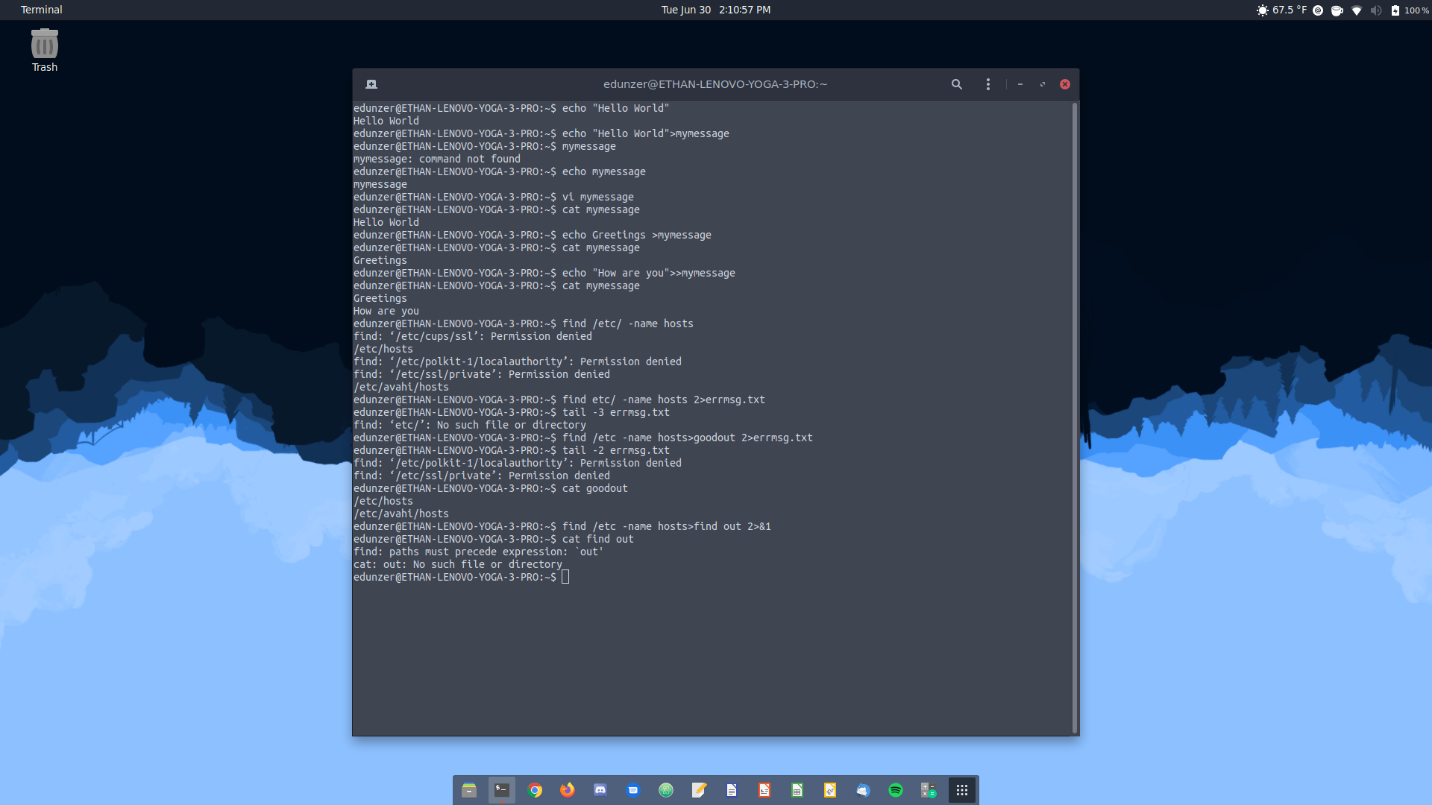
**\*\*\*Screenshot #8**



**More BASH practice**

* Once you’re logged in, open up a terminal and type echo “Hello World” and press enter
  + Your output should be exactly what you put into quotes
  + Now, use the same echo command and message, but this time append > mymessage to the end of your command (echo “Hello World” > mymessage)
  + This time, you’ll notice you have no output. Use the concatenate command to view the contents of mymessage.  This time you should see the same output, Hello World, that you got from your first echo command.
  + This task forced Linux to save your output into a file called mymessage, which is saved in your current working directory. This allows you to go back and view the message anytime you want.
* Here’s some more practice with saving your output
  + use the concatenate command to view mymessage
  + Enter the command echo Greetings > mymessage
  + use the concatenate command to view mymessage
    - notice your output is different and overwrote your original mymessage
  + This time, use two redirection symbols, echo “How are you” >> mymessage
  + concatenate mymessage
    - notice, you added to mymessage and did not overwrite the original contents
  + Find is a useful command when looking for specific files on you Linux system. Let’s practice with the find command and continue our practice of saving outputs to files
    - Enter find /etc/ -name hosts
      * you should see a couple of directories with hosts and a lot of Permission denied
    - enter find /etc -name hosts 2> errmsg.txt
    - Now enter tail -3 errmsg.txt
    - Enter find /etc -name hosts >goodout 2>errmsg.txt
    - tail -2 errmsg.txt
    - cat goodout
    - Enter find /etc -name hosts >find.out 2>&1
      * If you cat the find.out file, you should see both errors and standard outputs in the same file

**\*\*\*Screenshot #9**



* Now we’re going to use the translate command (tr). This command is useful when you want to enter a bunch of commands via a file.  However, sometimes our files aren’t perfect, so the tr command will translate uppercase to lowercase, vice versa, so all inputs will be entered correctly in the shell.
  + enter tr a-z A-Z
    - you’ll have a blank prompt, so enter:
    - this is interesting
    - how do I stop this?
    - Notice all of your text is converted to upper case
    - press Ctrl C to get out of the blank prompt
  + enter the change directory (cd) command to make sure you’re in your home directory
  + enter tr A-Z a-z > myfile
    - enter the following 2 lines:
      * Wow, I SEE now
      * This works!
    - This time Ctrl D to save and exit
    - concatenate myfile
      * notice all of your uppercase text was converted to lowercase
    - enter tr a-z A-Z < myfile
      * notice this time all of the text is converted to uppercase
    - The tee command gives you your output to your screen and an output file
      * enter cal 12 2015 | tee mycal
      * cat mycal
        + notice, you get the same output you just saw on your screen
      * Now, let’s use another command, xargs, that takes a command’s output and uses the outputs as arguments. This exercise will use another command, cut, which is not like the cut option in Windows.  Let’s see how we can filter and edit our output to what we want.
        + cat /etc/passwd

the passwd file is a list of usernames, not passwords. /etc/shadow holds password hashes

Notice the data is kind of a mess and we’d need to clean it up to accurately capture what usernames are configured on our Linux box.

* + - * + cut -d: -f1 < /etc/passwd

notice how much cleaner things already are…we basically cut everything after the first colon :.

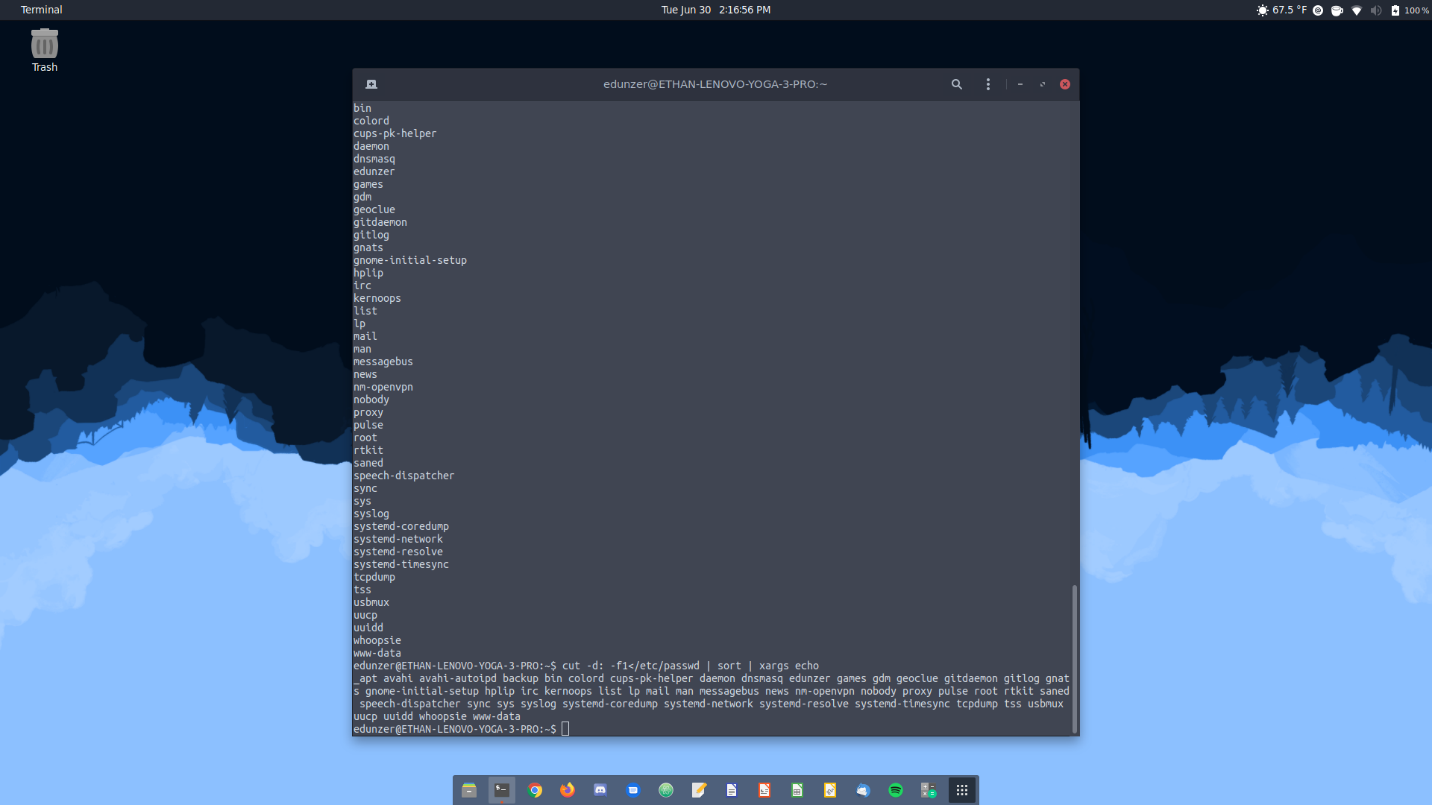
* + - * + cut -d: -f1 < /etc/passwd | sort

even better, now all of our usernames are alphabetically sorted

* + - * + cut -d: -f1 < /etc/passwd |sort | xargs echo

now you can see all user names, sorted, and only over a few lines (instead of scrolling off the screen).

**\*\*\*Screenshot #10**



**Find versus Locate**

* Find is a useful tool, but it works like the Windows search function without indexing. In other words, it must physically look in all directories for your search criteria.  There is a search command that runs faster, locate, which only looks in a database.
* The database that locate uses gets updated daily, at system startup, or after 24 hours of systems runtime. So, if you’re searching for newly created files, locate might not find them.  Luckily, we can update the database manually and continue to use the locate command.
* switch to root using the su – root command
* Let’s practice with find
  + find / -name desktop
  + find /home/<username>/Music -mmin -5
    - Also, you’ll need to replace <username> with you real username
  + touch /home/<username>/Music/mysong
  + find /home/<username>/Music -mmin -5
  + find /usr -size +20M
  + find /usr -size +20M | xargs ls -lh
  + find /var/spool/at -type d
  + find /var/spool/at -type d -ls
  + cd
  + touch mydata my4Gfile myhello
  + find /root -user root | grep my
* Now let’s practice with locate
  + touch myvideo
  + locate myvideo
    - you shouldn’t have an output here, or, if you do, it should be an error
  + updatedb
  + locate myvideo
  + locate inittab
  + locate -c inittab

**\*\*\*Screenshot #11**

