Purpose

The purpose of this exercise is to review files and configurations associated with user accounts.

Preparation

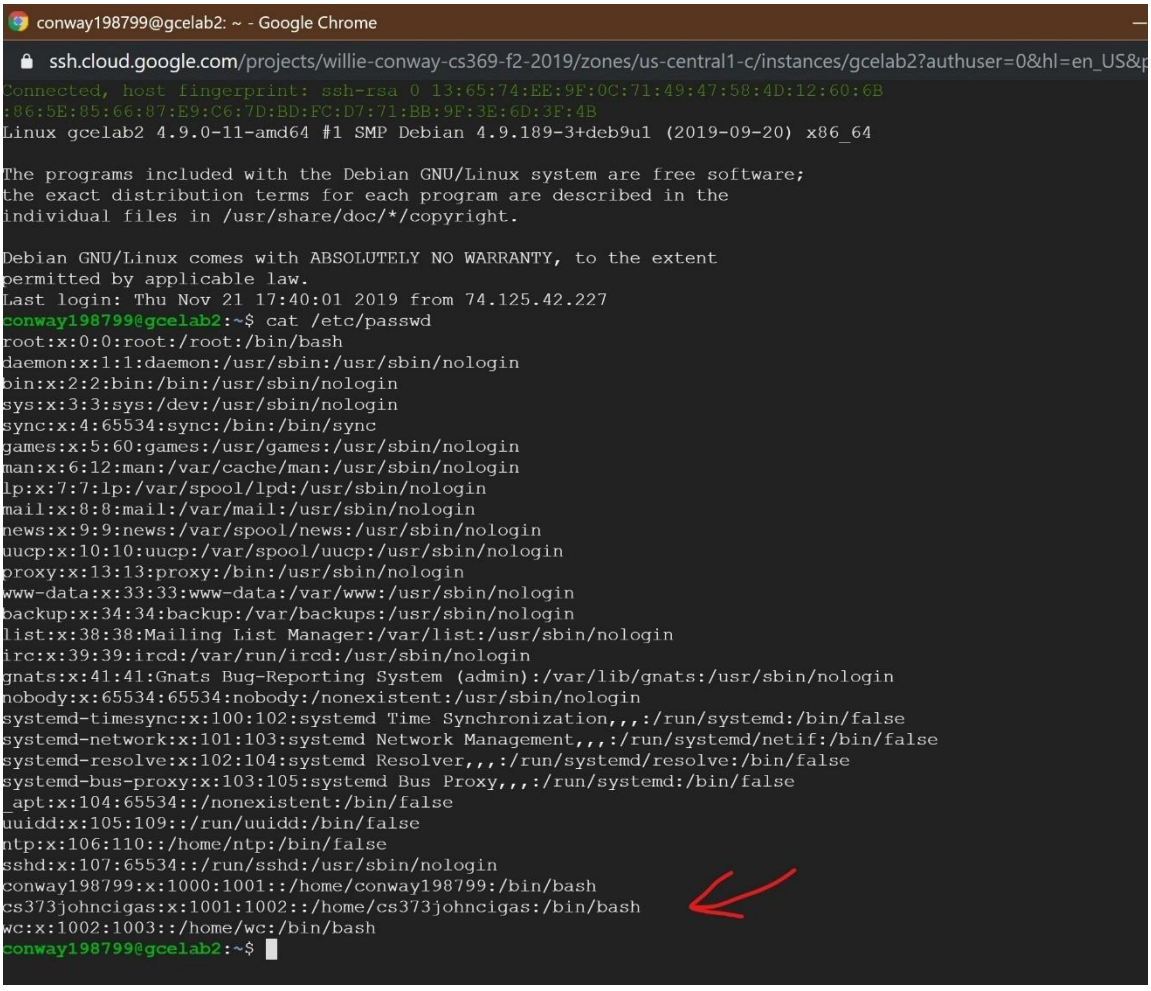
Start your GCP instance and connect to it using the SSH link.

Assignment

Keep a Word document with the answers to the following questions.

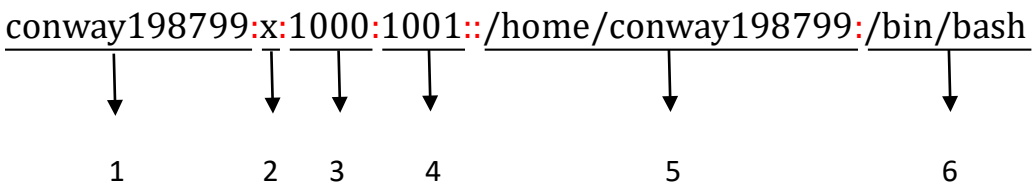
Look at your users (15 points)

Look at the contents of the file /etc/passwd. In your Word document:



(Screenshot output of the file /etc/passwd.)

1. List the entire line with your account. What is your home directory? What is your user id? What is your group id? You are a non-system user. Are there any other non-system users? How can you tell?



(Fig.01:/etc/password file format)

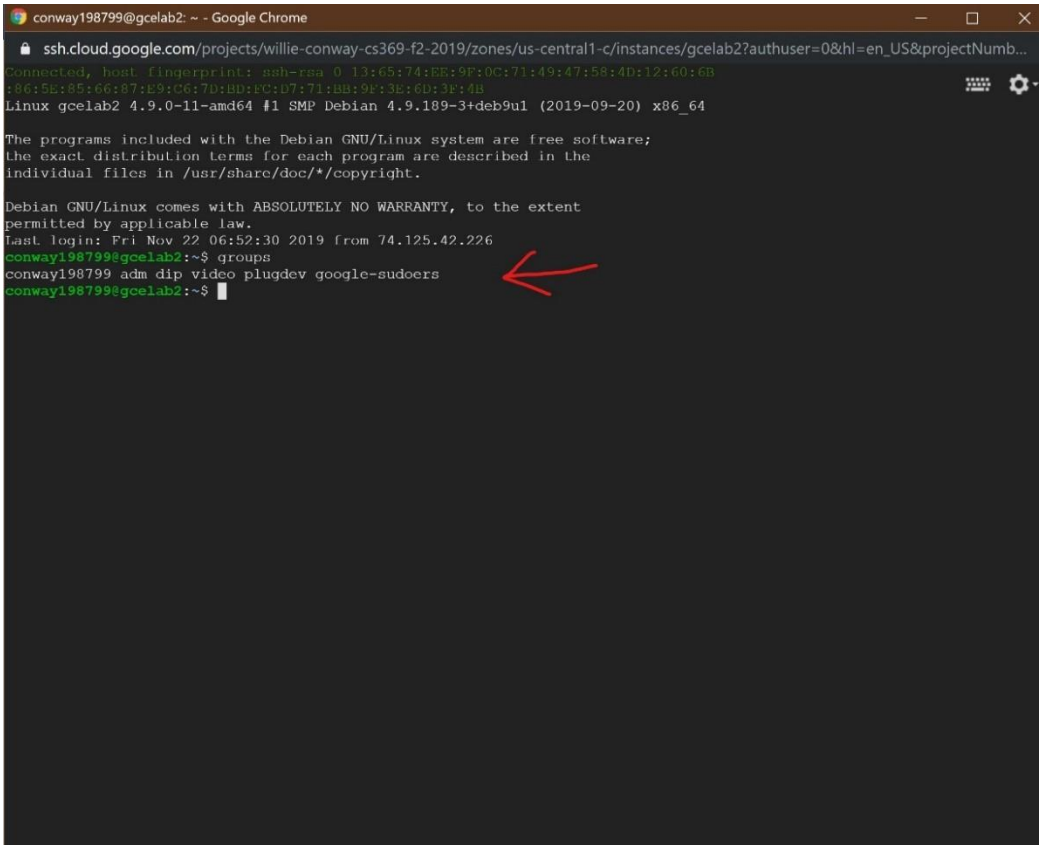
1.Username: First field indicates the name of the user which is used to login.
*2.Password: The X denotes encrypted password which is actually stored inside /shadow file. If the user does not have a password, then the password field will have an *(asterisk).*
3. User ID (UID): Every user must be allotted a user ID (UID). UID 0 (zero) is kept for root user and UIDs 1-99 are kept for further predefined accounts, UID 100-999 are kept by the system for the administrative purpose. UID 1000 is almost always the first non-system user, usually an administrator. If we create a new user on our Ubuntu system, it will be given the UID of 1001.
4. Group ID (GID): It denotes the group of each user; like as UIDs, the first 100 GIDs are usually kept for system use. The GID of 0 relates to the root group and the GID of 1000 usually signifies the users. New groups are generally allotted GIDs begins from 1000.
5. Home directory: Denotes the path of the user's home directory, where all the files and programs are stored. If there is no specified directory, then / becomes user's directory. 6. Command/shell: It denotes the full path of the default shell that executes the command (by the user) and displays the results.

(According to the above figure, my home directory is the absolute path /home/conway198799. My User ID (UID) is 1000 and the Group ID (GID) is 1001. According to the screenshot, there is a total of three non-system users (conway198799, cs373johncigas and wc). UID 1000 is almost always the first non-system user, usually an administrator. I'm the first non-system user of my SSH, username conway198799. When I authorized to share my SSH with username cs373johncigas through GCP, they were given the User ID (UID) 1001. When I used a third-party tool to access my SSH instance with putty using the username wc, I was given the User ID (UID) 1002.)

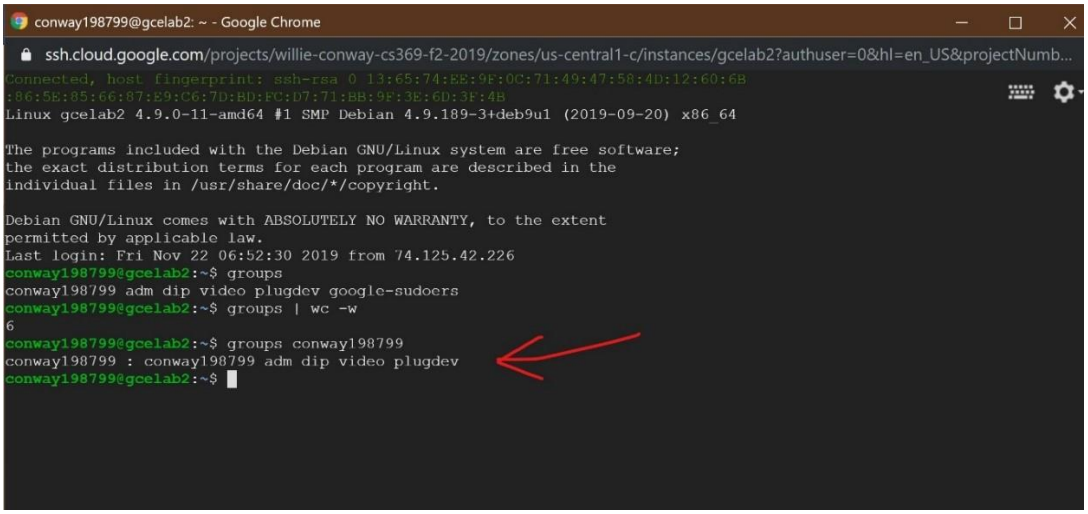
(Each of these users are part of a user group. The GID of 0 relates to the root group and the GID of 1000 usually signifies the users. New groups are generally allotted GIDs begins from 1000. The username conway198799 is the first member of the group with a Group ID (GID) of 1001. The username cs373johncigas is the second member of the group with

a Group ID (GID) of 1002. The username wc is the third member of the group with a Group ID (GID) of 1003.)

2. Show the result of the groups command.

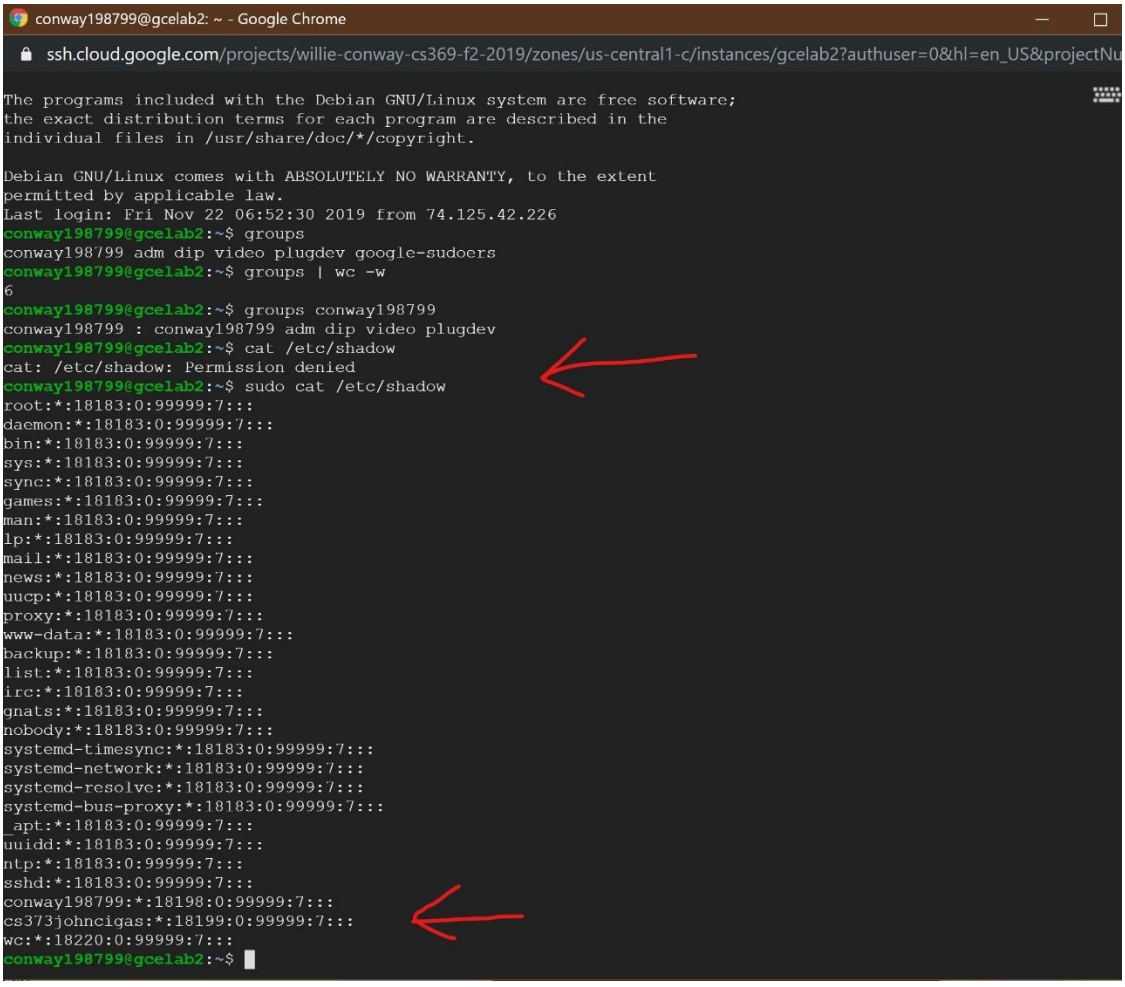


(The Linux groups command prints the names of the primary and any supplementary groups for each given username, or the current process if no names are given. If more than one name is given, the name of each user is printed before the list of that user's groups and the username is separated from the group list by a colon. See screenshot below.)



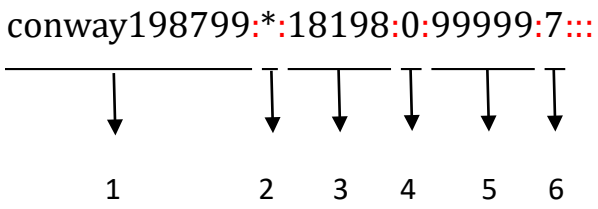
(Using the Linux groups command with username conway198799.)

Since you have an account on your instance, you have an entry in /etc/passwd, and you also have an entry in /etc/shadow. View this file.



(Using Linux command sudo to bypass permissions, then using Linux cat command to open and display file /etc/shadow.)

3. List the line for your account. What does the password field contain? What does this mean? Explain why this seems like a contradiction.



(Fig.02:/etc/shadow file format)

1.Username: It is your login name.

2.Password: It is your encrypted password. The password should be minimum 8-12 characters long including special characters, digits, lower case alphabetic and more.

3. Last password change (lastchanged): *The date of the last password change, expressed as the number of days since Jan 1, 1970.*

4. Minimum: The minimum number of days required between password changes i.e. the number of days left before the user is allowed to change their password

5. Maximum: The maximum number of days the password is valid (after that user is forced to change their password)

6. Warn: The number of days before password is to expire that user is warned that their password must be changed.

*(The password file contains an asterisk(*).If the user does not have a password, then the password field will have an asterisk(*). If the password field contains some string that is not a valid result of crypt(3), for instance ! or *, the user will not be able to use a Unix password to log in (but the user may log in the system by other means). This field may be empty, in which case no passwords are required to authenticate as the specified login name. However, some applications which read the /etc/shadow file may decide not to permit any access at all if the password field is empty.)*

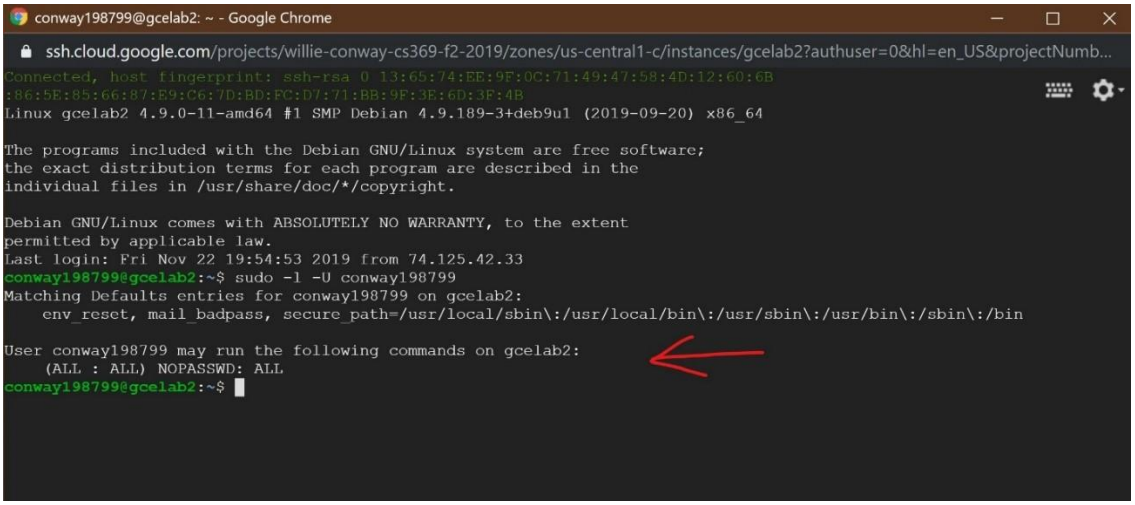
(This happens to be a contradiction, because we have a server which is accessible via SSH. There is a user in /etc/passwd who has a valid login shell. The same user has an asterisk(*) in the second field of the /etc/shadow file. The asterisk(*) in the location of the hash password, in the /etc/shadow file effectively disables all password based logins as no user input will ever result in

a hash value of (*). However, the user can still login with their SSH keys. The reason is to set an invalid shell is to prevent interactive login sessions from working, block sudo users and similar from working.)

We'll explore this topic in more detail in the next assignment.

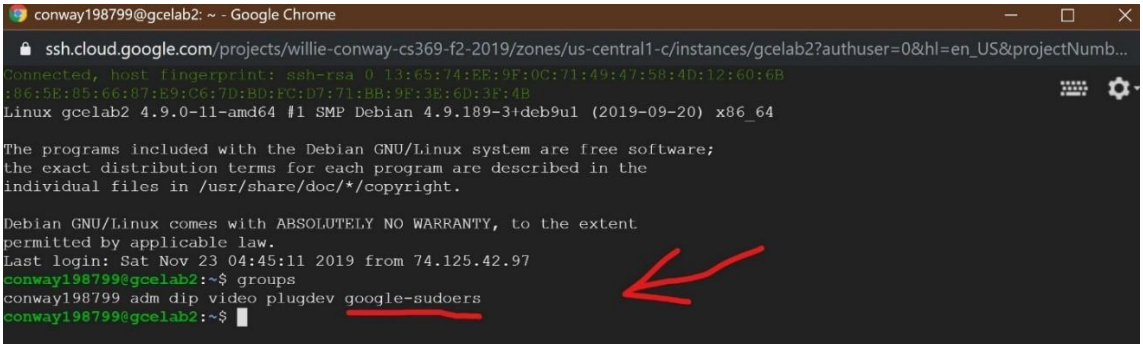
4. What command did you have to issue before you could read the file /etc/shadow? What does this command let you do in general? Not all users on a system will be able to issue this command. From what you've seen so far in this lab, what gives you the permission to issue the command?

   The command I used to access the /etc/shadow file was the Linux sudo command. The sudo command allows you to run programs with the security privileges of another user (by default, as the superuser). It prompts you for your personal password and confirms your request to execute a command by checking a file, called sudoers , which the system administrator configures. All users can read the /etc/passwd and /etc/shadow files, however they should not be able to change the fields. The owner of the /etc/shadow file is usually the user root. The group is often set to an administrative group, like shadow. Other users are not allowed to read the file directly, to prevent them from gathering hashes passwords of others.



   (I used the Linux sudo -l -U command to check the sudo access for the user conway198799. If the user can run a few or all commands with sudo, you should be able to see the following output. As you can see, the user conway198799 is granted all privileges to use sudo. The line (ALL : ALL) NOPASSWD: ALL means that the user can uses all sudo privileges and that no password is required to use sudo for the user.)

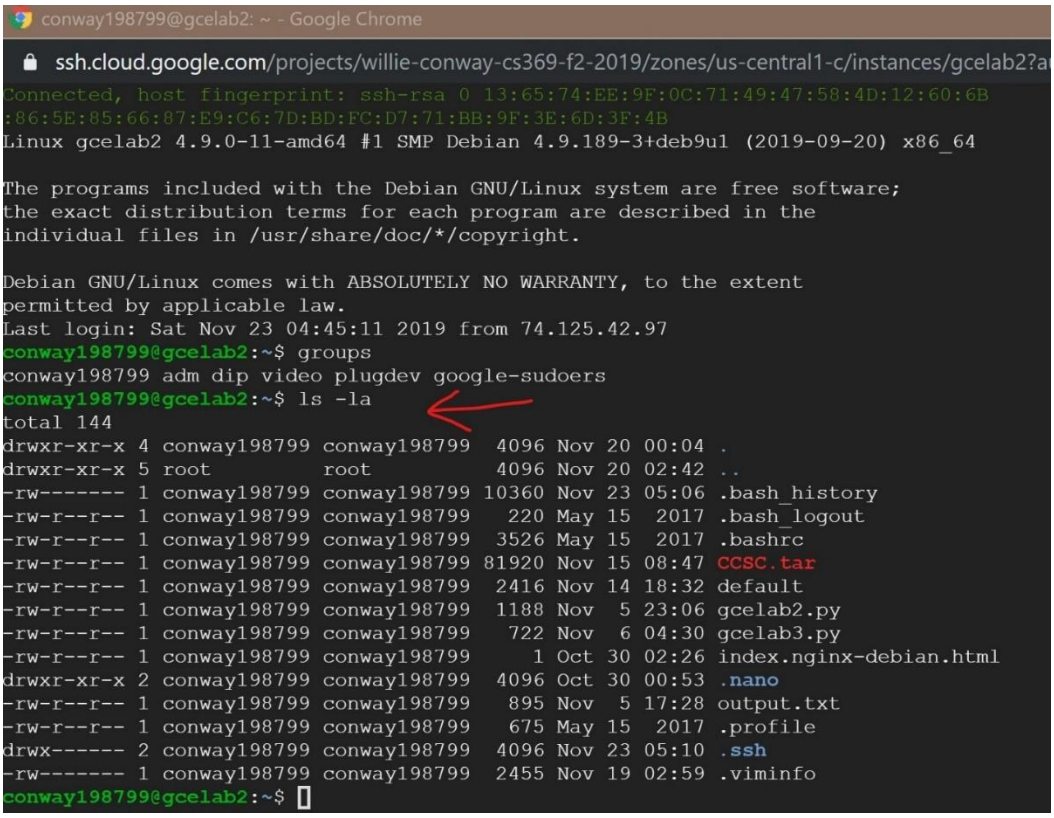
   The reason conway198799 has the permission to use the Linux sudo command, is because the user was granted administrative privileges through root. Since conway198799 was added as a new user to the system, root had to decide if the user should be able to perform administrative tasks through sudo. If the user you created will be your primary user on the system, you usually want to enable sudo privileges so that you can do routine configuration and maintenance. To add these privileges to our new user, we need to add the new user to the sudo group. By default, users who belong to the sudo group are allowed to use the sudo command.



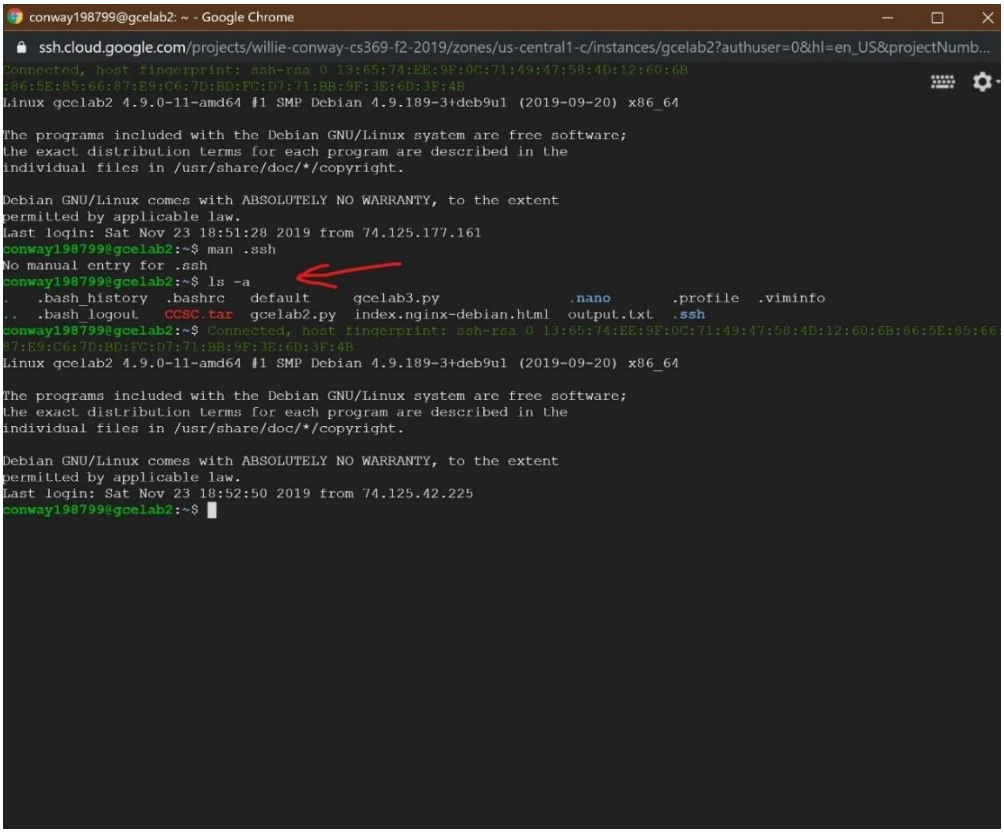
(In the screenshot, when we use the Linux groups command, we can see that user conway198799 is a user of the google-sudoers group.)

Look at your configuration files (10 points)

5. List all the files in your home directory, including the hidden ones. Show the command you used and the output.



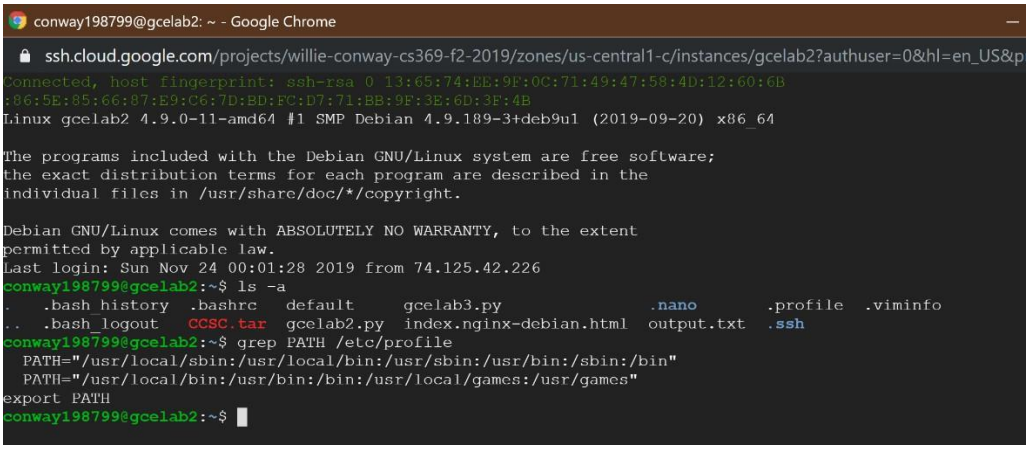
(Using Linux ls -la command to list long format including hidden files for my /home/conway198799 directory.)



(Using Linux ls -a command to list all files including hidden files for my /home/conway198799 directory.)

6. Compare the *contents* of the files you found with the *contents* of the files described in the text. Briefly describe any additions or deletions.

(Linux Fundamentals Chapter 30 user profiles briefs information of Logged on users who have a number of preset aliases, variables, and functions. The shell uses a number of startup files that are executed, whenever the shell is invoked.) **system profile**

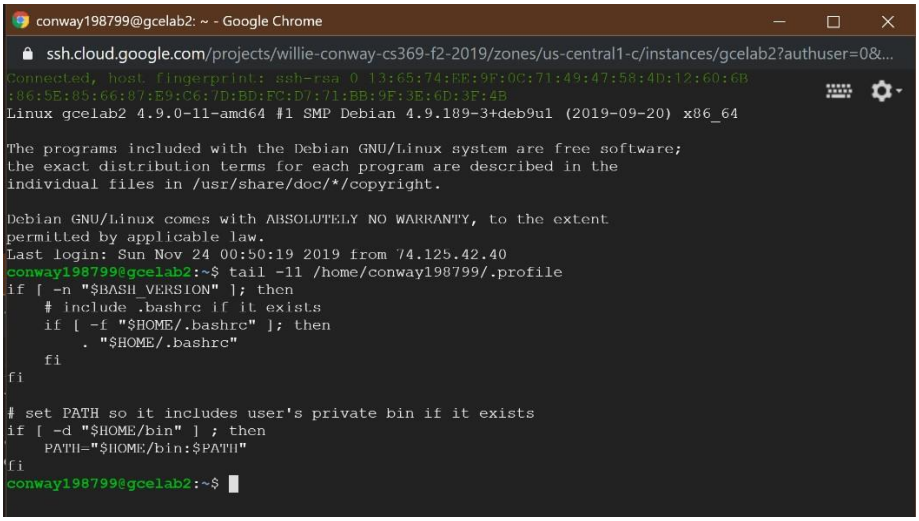


(This screenshot shows the use of the Linux grep command to show PATH manipulation in /etc/profile on Debian. The root user can use this script to set aliases, functions, and variables for every user on the system. The bash shell will verify the existence of /etc/profile and source it if it exists. This appears to be the same as the text.)

~/.bash_profile and ~/.bash_login

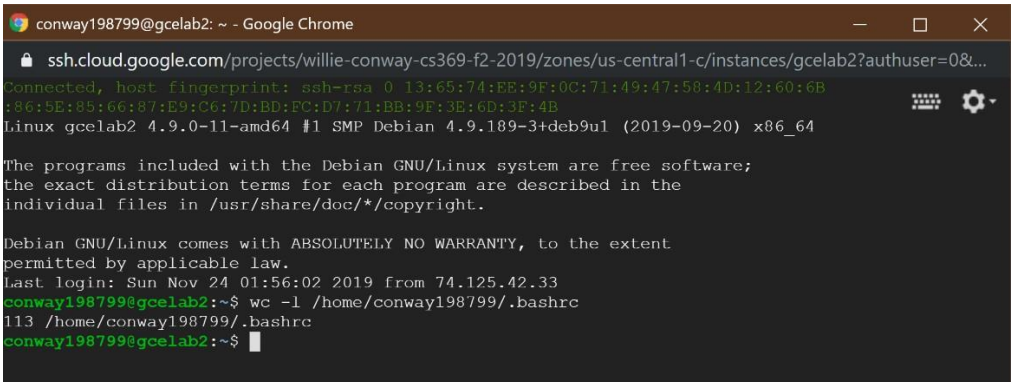
(The ~/.bash_profile is missing. When this file exists in the home directory, then bash will source it. On Debian Linux 9 (stretch) this file does not exist by default. When .bash_profile does not exist, then bash will check for ~/.bash_login and source it. The ~/.bash_login file is missing. Debian does not have this file by default. As you can see, neither of these files are located in my /home/conway198799 directory.)

~/.profile



(When neither ~/.bash_profile and ~/.bash_login exist, then bash will verify the existence of ~/.profile and execute it. On Debian this script can execute ~/.bashrc and will add $HOME/bin to the $PATH variable. The content appears as the same in the text.)

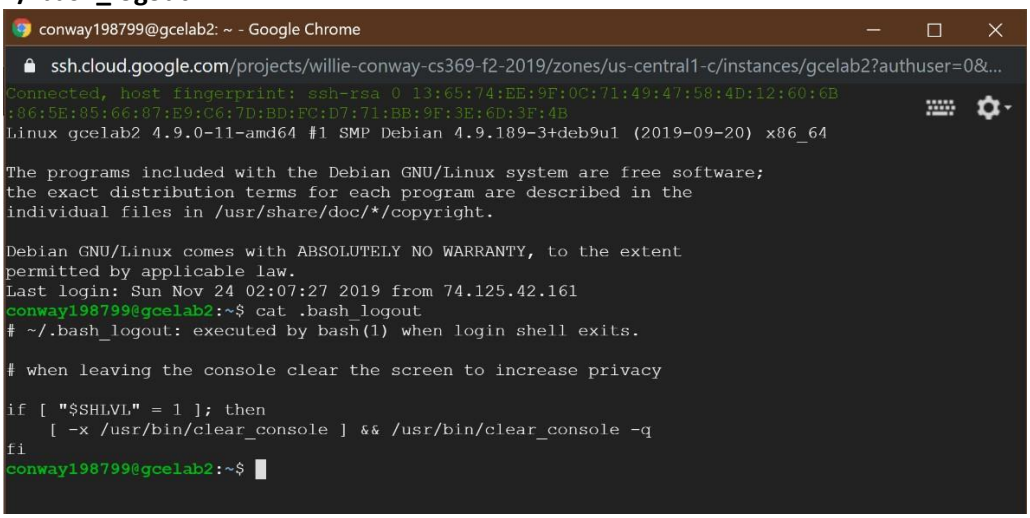
~/.bashrc



(The ~/.bashrc script is often sourced by other scripts. Let us take a look at what it does by default. On Debian this script is quite a bit longer and configures $PS1, some history variables

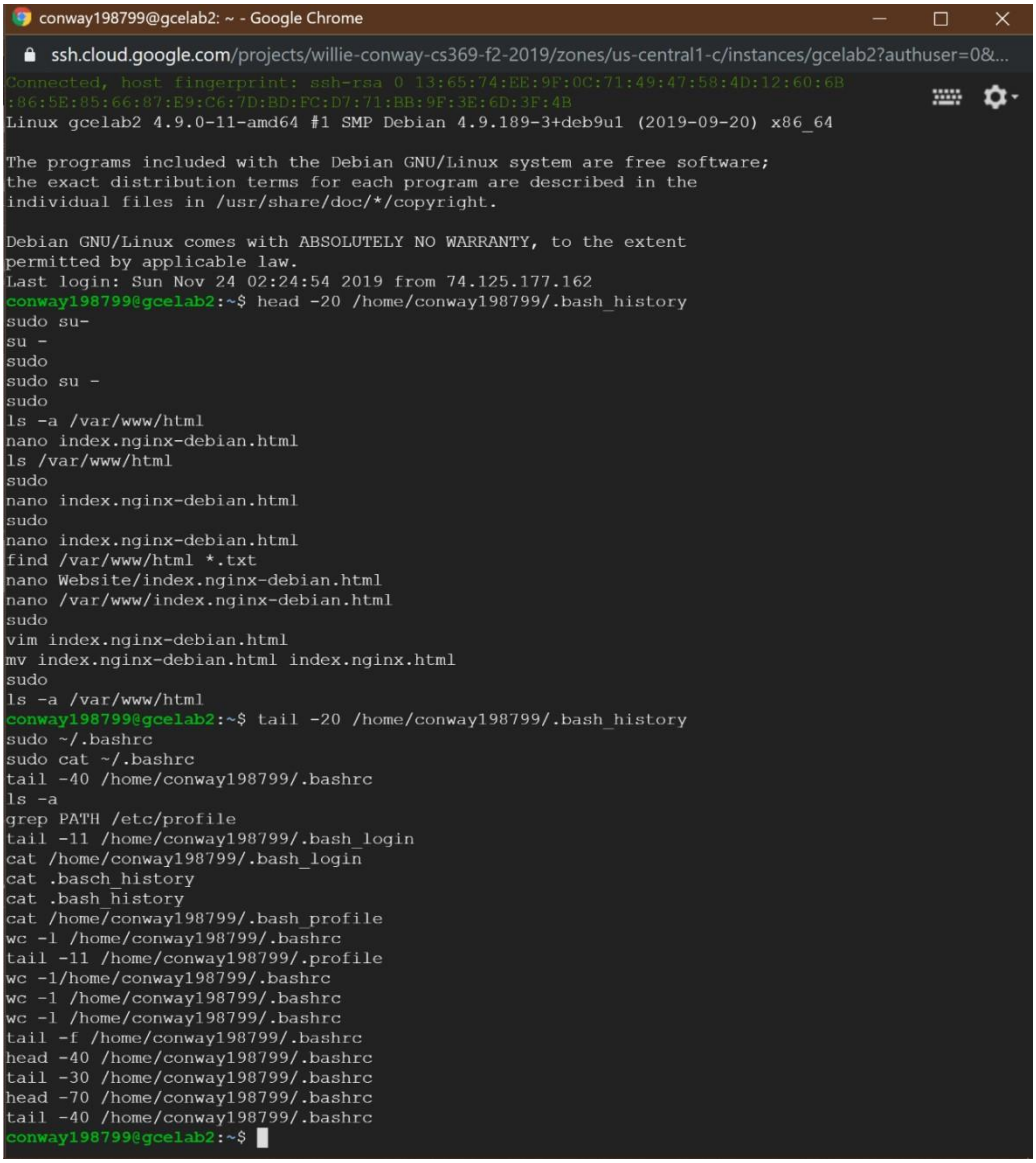
and a number of active and inactive aliases. This appears to be the same in the text, with the exception of extra lines added to the file. The wc (word count) command in UNIX is a command line utility for printing newline, word and byte counts for files. It can return the number of lines in a file, the number of characters in a file and the number of words in a file. It can also be combined with pipes for general counting operations. So, there's 113 lines instead of 110.)

~/.bash_logout



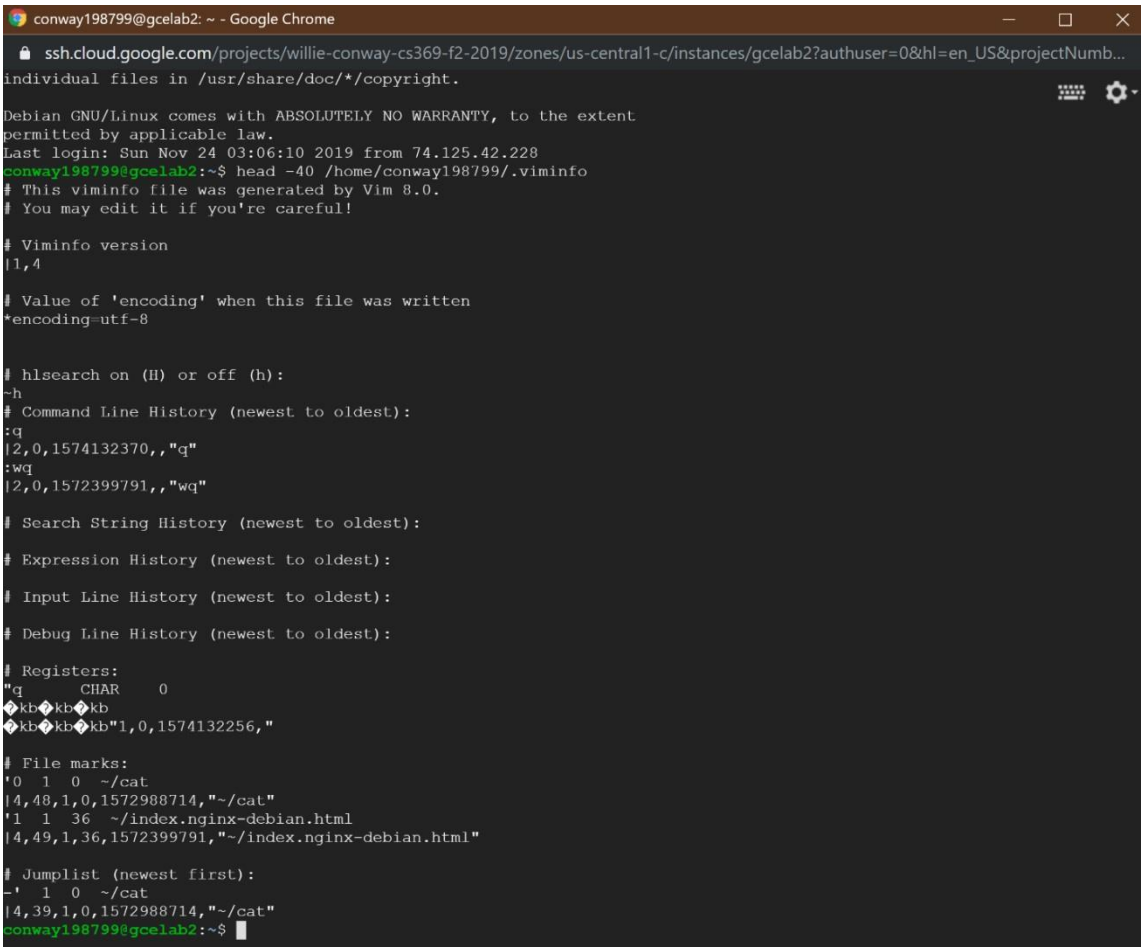
(When exiting bash, it can execute ~/.bash_logout. Debian use this opportunity to clear the console screen. This appears to be the same as the text.)

~/.bash_history



(The ~/.bash_history file is created by Bash, a Unix-based shell program commonly used on Linux operating systems; stores a history of user commands entered at the command prompt; used for viewing old commands that have been executed. I used Linux's head and tail commands to print out the first 20 lines and last 20 lines of the contents, as this content to this file is quite lengthy. This happens to be an additional hidden file that's added to my /home/conway198799 directory. This is not shown in the text.)

~/.viminfo



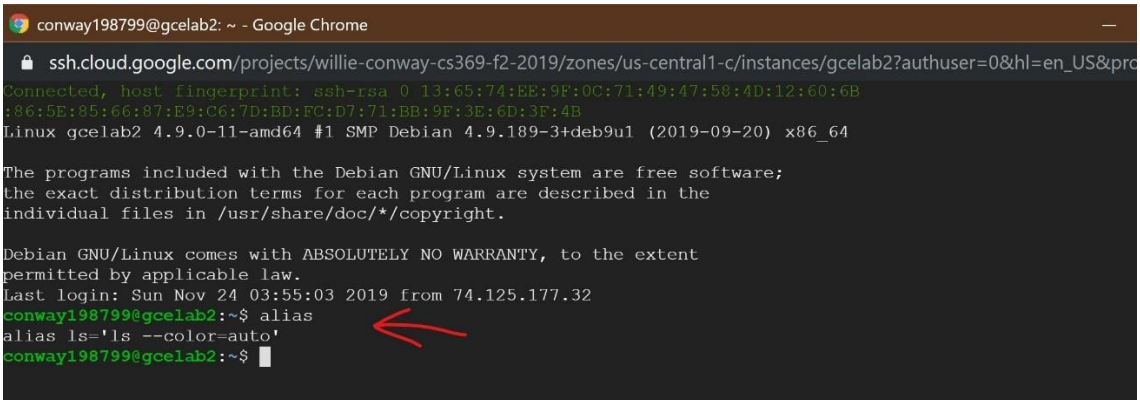
(The default location for the viminfo file is ~/.viminfo. The viminfo file is used to store the command line history, search string history, input-line history, contents of non-empty registers, marks for several files, file marks (pointing to locations in files), the buffer list, and global variables. The contents to this file is quite lengthy so I used the Linux head command to print the first 20 lines. This appears to be a hidden file and an additional file to my /home/conway198799 directory. This file is not shown in the text.)

.nano and .ssh

(If nano is started for the first with sudo then a root owned directory named .nano is created in the user's $HOME directory. SSH keys are typically configured in an authorized_keys file in .ssh subdirectory in the user's home directory. Typically, a system administrator would first create a key using ssh-keygen and then install it as an authorized key on a server. These are additional hidden directories. Neither of these are shown in the text.)

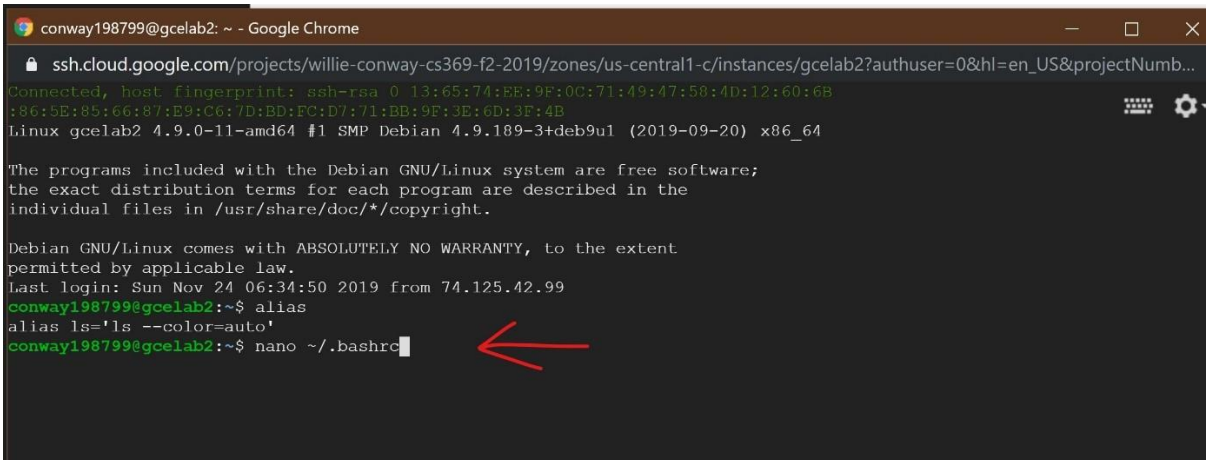
(The other files are from previous assignment exercises.)

7. Issue the alias command and show the results.

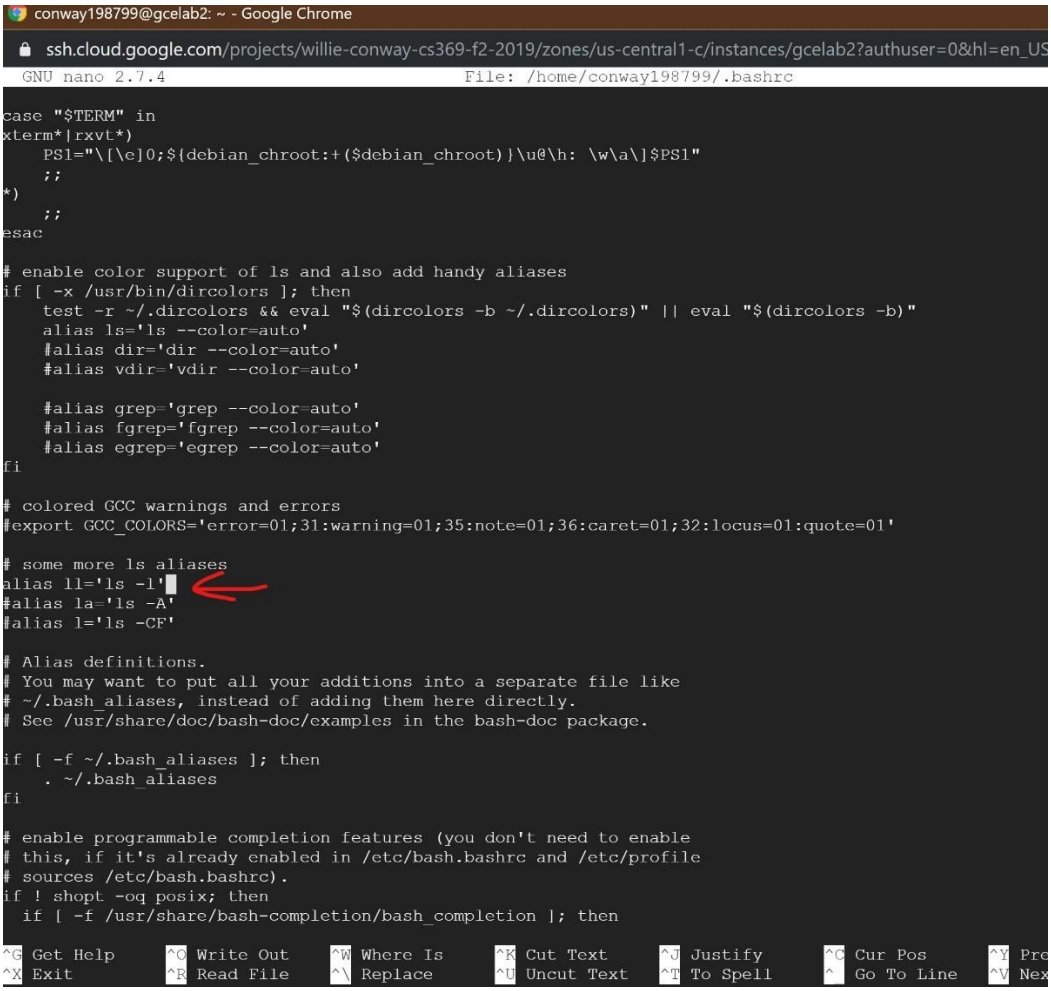


(Using Linux alias command. Thier appears to be only one alias created.)

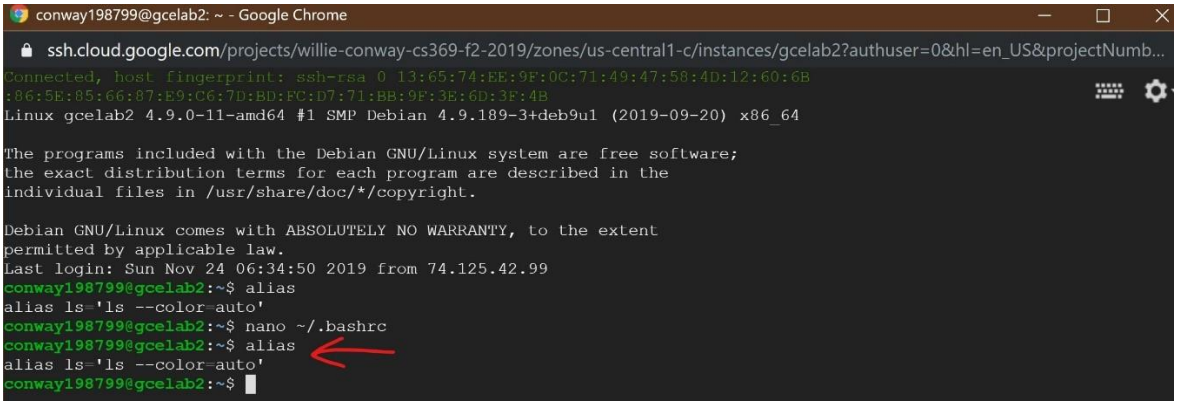
Add your own alias that will get established every time you open a new shell window. Edit the appropriate file to uncomment one of the predefined aliases. Save the file. Reissue the alias command. You shouldn't see your new alias. This is because these configuration files are normally only sourced once when you log in or create a new shell. After you make a change, you can certainly log out and in again, though this is overkill. You just need to issue the source command instead.



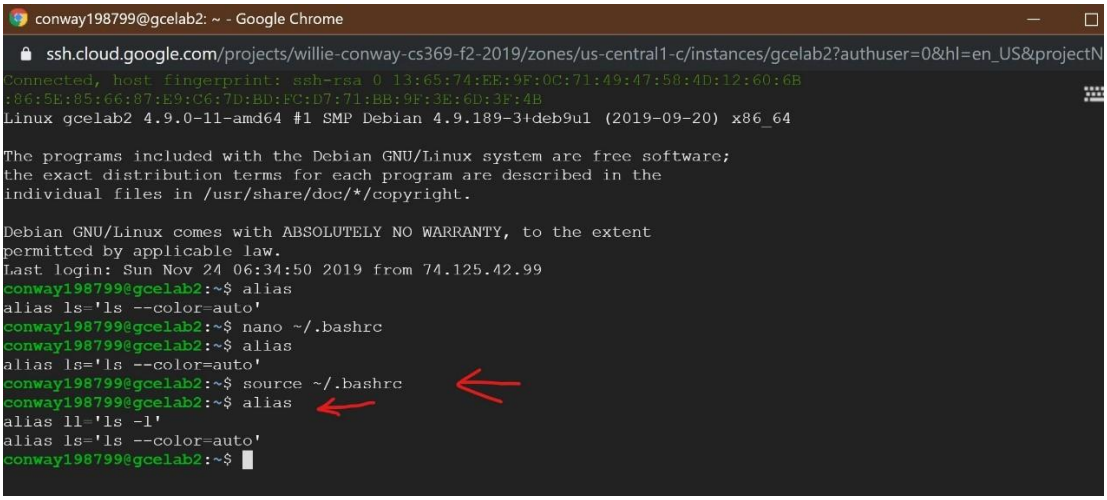
(In order to create a alias you have to edit the ~/.bashrc file, since it contains the aliases.
Your ~/.bashrc file is located in your user directory. Open it in your favorite text editor. I used nano editor to edit the file.)



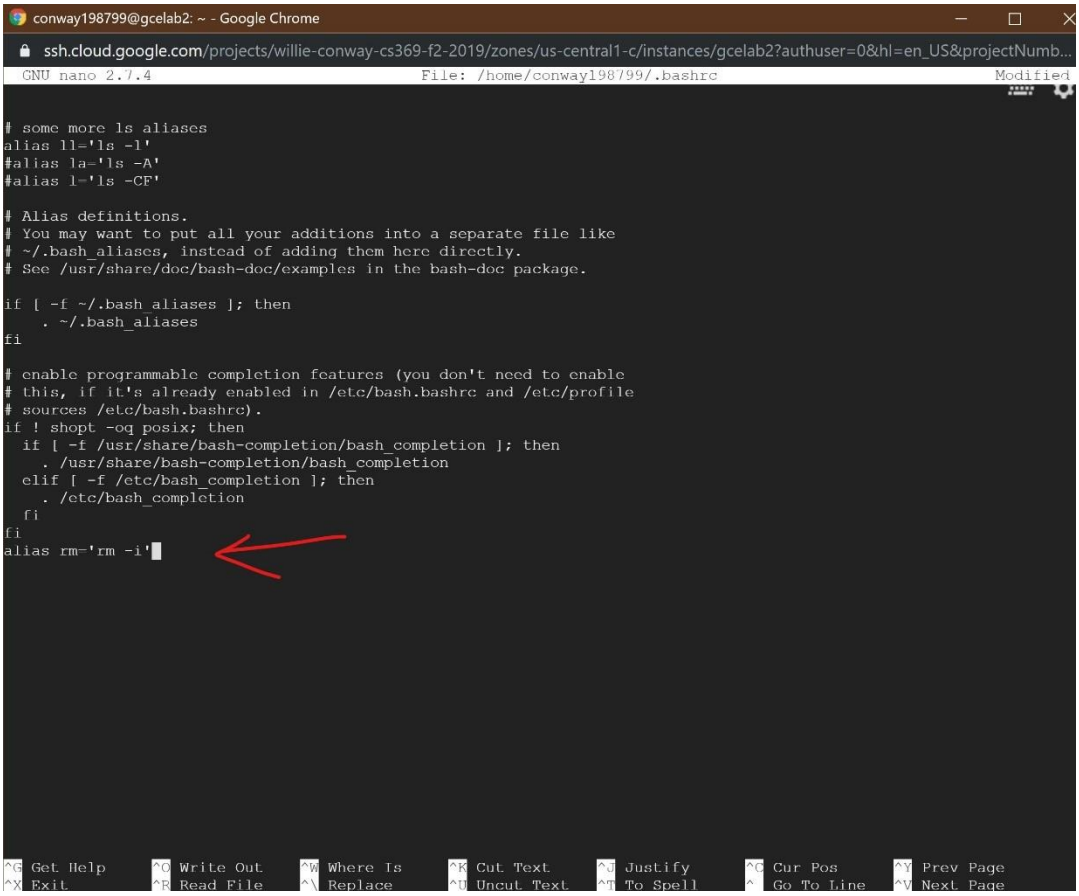
(Uncommenting one of alias from the #some more ls aliases area. I decided to choose alias ll='ls -l'. After uncommenting the file, I save the changes with Ctrl+O, then exit the file with Ctrl+X.)



(Issued the Linux alias command once more, unfortunately the filed appeared to not update. I could login/logout, however there is simpler Linux command solution called source to activate the changes.)

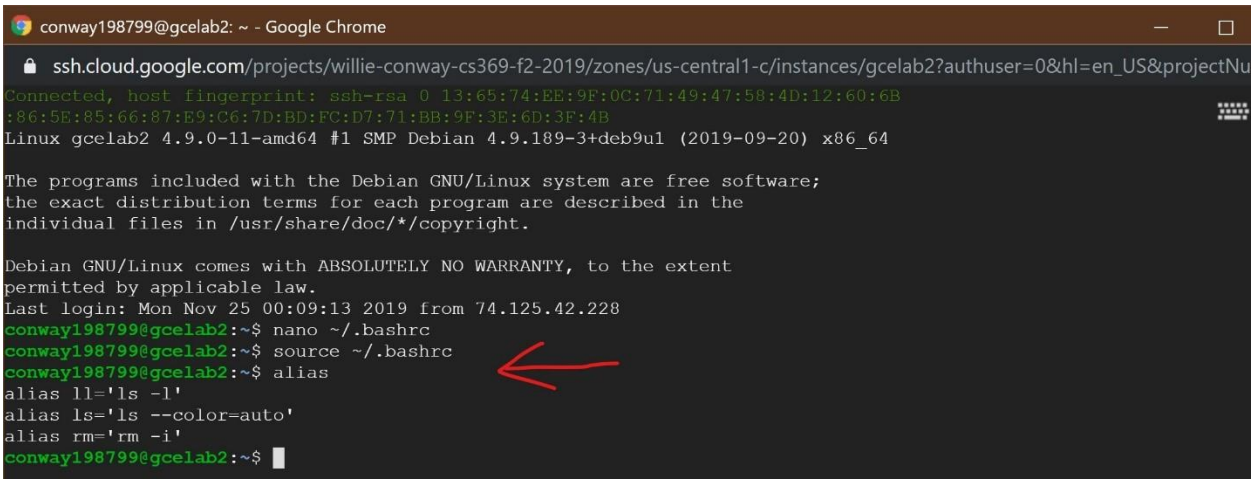


(The source is a shell built-in command which is used to read and execute the content of a file(generally set of commands), passed as an argument in the current shell script. I used the Linux command source to update the ~/.bashrc file to execute the changes right away. After, I performed the alias command to see my uncommented alias ll='ls -l'.



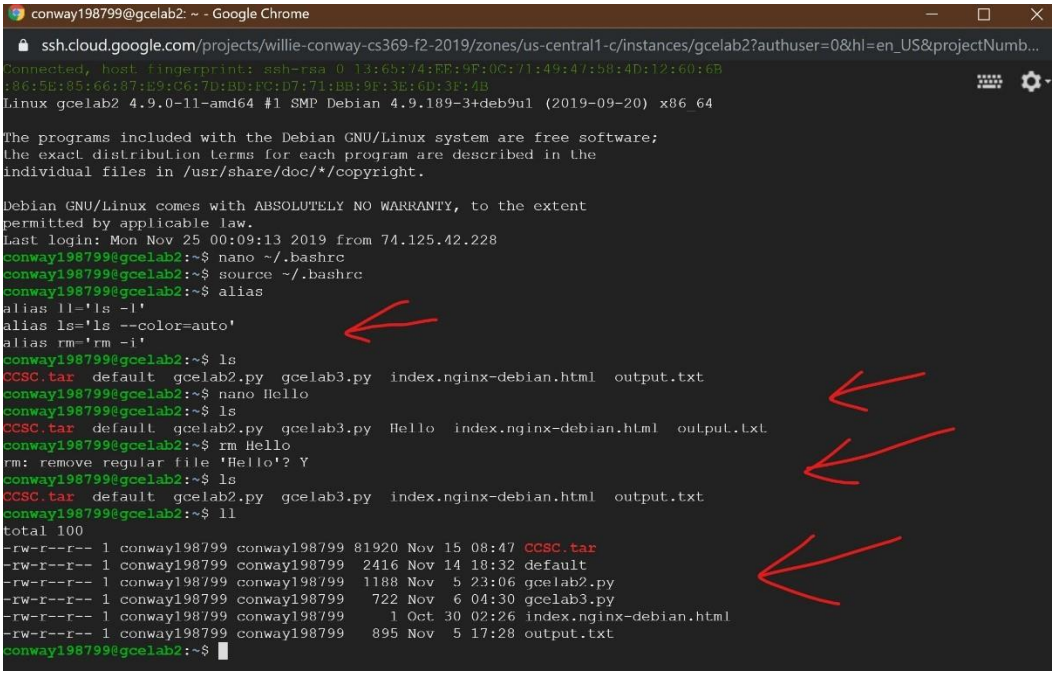
(To create my own alias, I decided to edit the Linux rm -i command. This command is responsible for removing files and directories, the -i command is added to prompt the user before removal of the file or

directory. To do this, I added alias rm = 'rm -i' to the end of the ~/.bashrc file. After, used Ctrl +O to save the file and Ctrl + X to exit the file.)



(Used source to activate the changes in the ~/.bashrc file, the Linux alias command to check if my new rm alias was added.)

8. Document in which file you added your alias, and show that it now works.

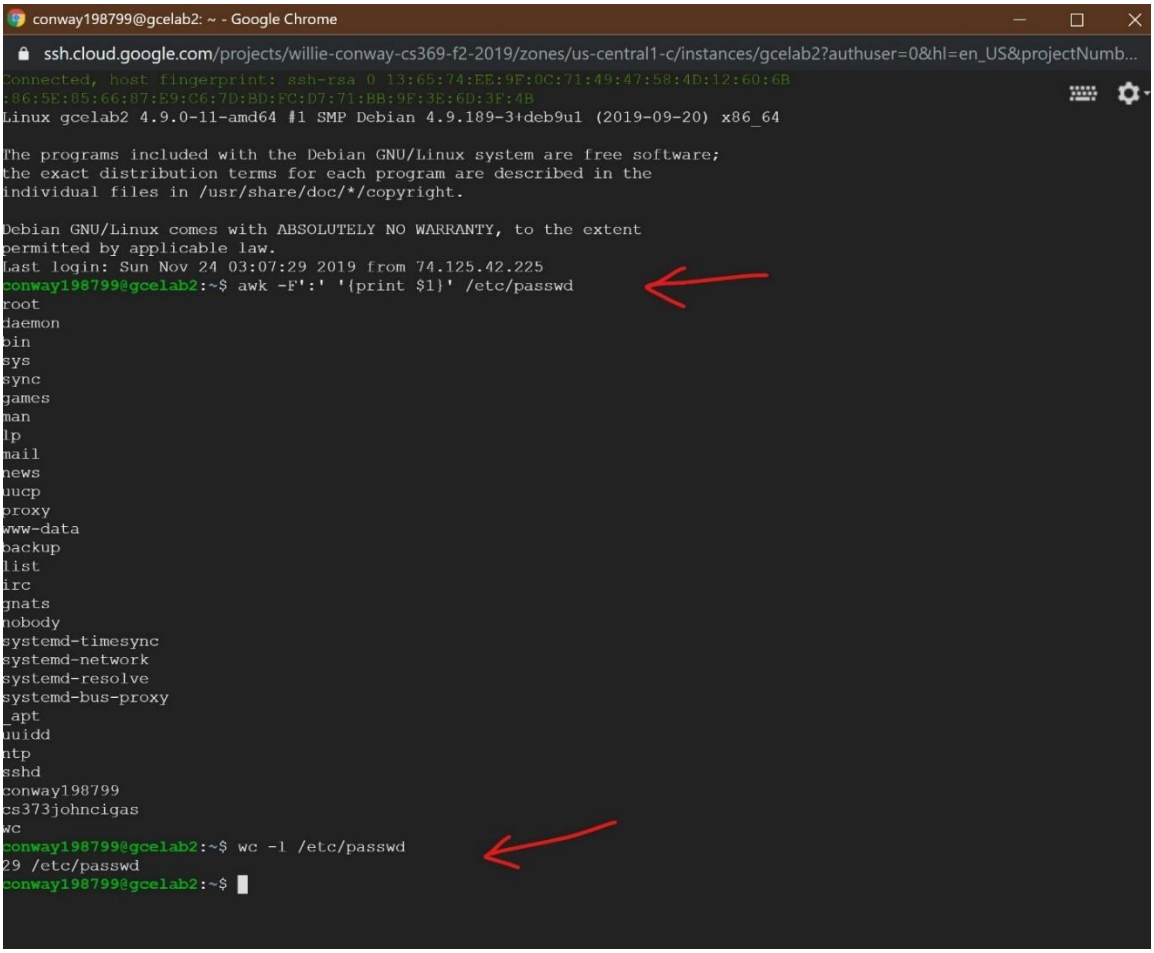


(The file which I added my aliases to was the ~/.bashrc file. The screenshot shows my created alias rm='rm -i' by entering rm to activate the Linux rm -i command ,and uncommented alias ll='ls -l' by entering just ll, to activate the Linux ls -l command. This Linux command uses a long listing format to list the file contents in the directory. To show the process of the alias rm, I had to create the file Hello with nano editor, then delete the file using the rm alias. The Linux ls command shows where the file was stored, as it was listed amongst the present files in the home

directory, to the file being deleted/remove from the directory. You can see that the system prompts the user to answer yes or no before moving forward with deletion/removal.)

Utility program (7 points)

9. Write a Linux command or series of commands, or a python program that prints a count of the number of system user accounts listed in /etc/passwd. Show your code and its output.



(In order to complete this exercise, I had to count the number of users that were disclosed in the /etc/passwd file. I just want a list of the names, so that it would make the counting a lot easier. To do so, I needed to separate the rest of the content. To do this, I needed to used Linux awk command. Awk is a scripting language used for manipulating data and generating reports.The awk command programming language requires no compiling, and allows the user to use variables, numeric functions, string functions, and logical operators. Awk is a utility that enables a programmer to write tiny but effective programs in the form of statements that define text patterns that are to be searched for in each line of a document. The -F command acts as a field separator for the input field. Since I only wanted the first field of the users, which is the usernames, I use {print $1} to let the system know that I wanted the first field from the /etc/passwd file.)

(Next, I use the Linux wc (word count) command to count and list the number of user accounts within the file /etc/passwd file.)

Reflection

At the end of your Words document, include the answers to the following questions:

1. In a sentence or two, what did you learn? *This was difficult exercise to accomplish as there was a lot to learn. This was my first time creating a permanent alias, editing a bash file. Usually when it comes to editing important system files, I usually try to avoid since I'm not aware of what I was doing. Using Linux is becoming more fascinating as I complete these exercises because I'm becoming more familiar with commands and finding out much more about what each file purpose is.*
2. In a sentence or two, what did you like about this project? *Although utilizing the awk command wasn't part of the subject I really thought of it as a unique way of coding, that I was able to achieve a simple task with one line. In this project I was learning as I was going, so maybe I wrote more then what I was expected. I like to document everything, because I like to see the process and it makes following the task simple.*
3. In a sentence or two, what did you find confusing or would like to see done differently regarding this project? *I'm still a little confused about question 6, not sure if I did it correctly or not. It seemed like it was asking me to just compare the Debian files in Chapter 30 of Linux fundamentals – user profile to the similar and additional or missing files that were in my /home/conway198799 directory. Not being sure, I decided to review every file from text that was explained in Liniux fundamentals chapter 30 and compare if I had those files or not in my home directory. I also notified the additional files that appeared to be hidden. I was aware of my previous assignment files, so I didn't list them. I think* the wording just confused me.