Activity No. 11	
Command Line Skills	
Course Code: CPE 201A	**Program:**
Course Title: COMPUTER SYSTEM ADMINISTRATION AND TROUBLESHOOTING	**Date Performed: 11/3/2025**
Section: CPE11S5	**Date Submitted: 11/3/2025**
Name: Salgado, Andee Ramon V.	**Instructor: Engr. Lloyd Aldrin Pornobi**

1. Objective/s:

This activity aims to execute basic commands using command line interface of Linux.

2. Intended Learning Outcome/s:

The students should be able to:

2.1 Demonstrate how to use commands to explore BASH features.
2.2 Demonstrate how to use commands to display the values of Shell variables.
2.3 Demonstrate how to use quoting in Bash shells.

3. Discussion:

Command Line Interface

The Linux community promotes the CLI due to its power, speed and ability to accomplish a vast array of tasks with a single command line instruction. The CLI provides more precise control, greater speed and the ability to automate tasks more easily through scripting. By learning the CLI, a user can easily be productive almost instantly on ANY flavor or distribution of Linux.

The Shell

Once a user has entered a command , the terminal then accepts what the user has typed and passes to a shell. The shell is a program that enables text based communication between the operating system and the user. It is the command line interpreter that translates commands entered by a user into actions to be performed by the operating system. The Linux environment allows the use of many different shells. There are several different shells on Linux, these are just a few:

- Bourne-again shell (Bash)
- C shell (csh or tcsh, the enhanced csh)
- Korn shell (ksh)
- Z shell (zsh)

The most commonly used shell for Linux distributions is called the **Bash** shell. When using an interactive shell, the user inputs commands at a so-called prompt. For each Linux distribution, the default prompt may look a little different, but it usually follows this structure:

```
username@hostname current_directory shell_type
```

On Ubuntu or Debian GNU/Linux, the prompt for a regular user will likely look like this:

```
carol@mycomputer:~$
```

The superuser's prompt will look like this:

```
root@mycomputer:~#
```

On CentOS or Red Hat Linux, the prompt for a regular user will instead look like this:

```
[dave@mycomputer ~]$
```

And the superuser's prompt will look like this:

[root@mycomputer ~]#
Let's explain each component of the structure:

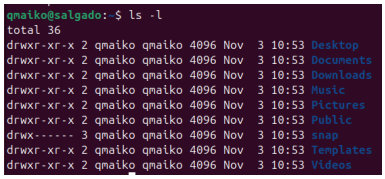
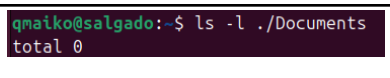
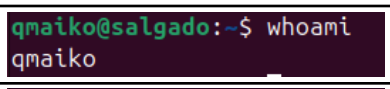
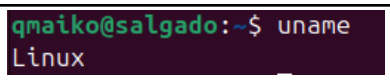
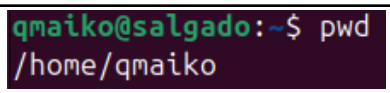
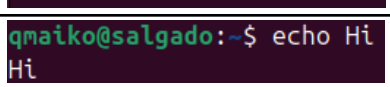
username
Name of the user that runs the shell
hostname
Name of the host on which the shell runs. There is also a command hostname , with which you can show or set the system's host name.
current_directory
The directory that the shell is currently in. A ~ means that the shell is in the current user's home directory.
shell_type
$ indicates the shell is run by a regular user.
# indicates the shell is run by the superuser root

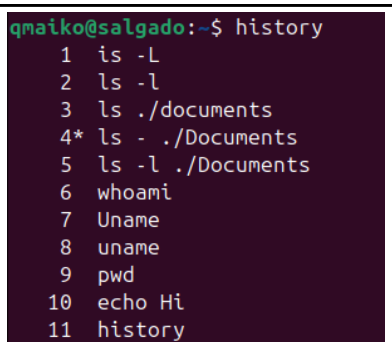
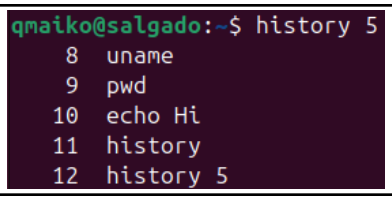
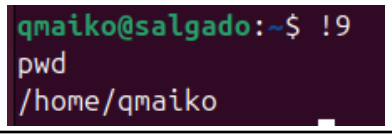
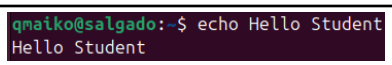
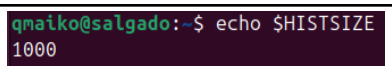
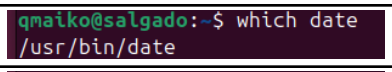
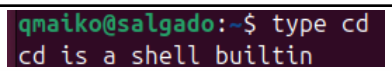
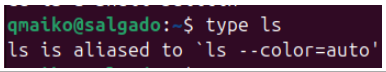
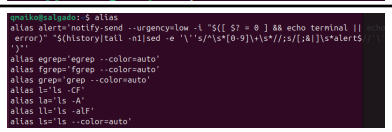
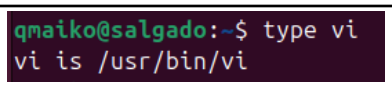
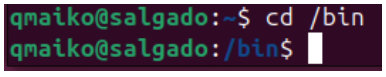
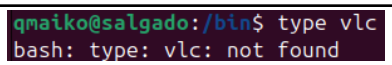
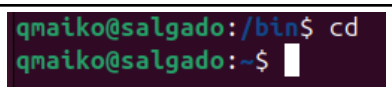
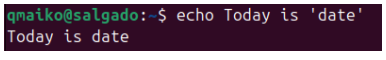
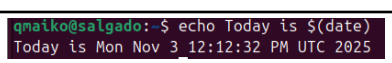
4. Resources:

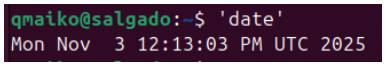
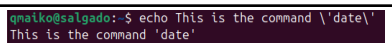
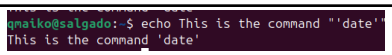
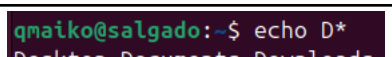
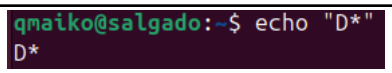
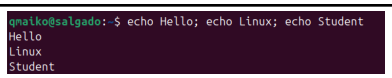
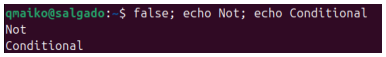
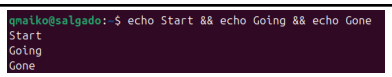
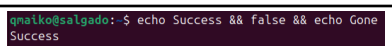
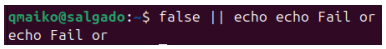
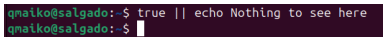
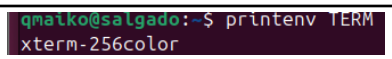
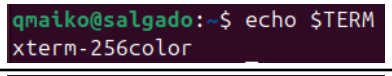
Personal Computer with installed Virtual Box
Ubuntu Server or Desktop virtual machine

5. Procedure:

1. Login using your username and password.
2. Use terminal emulator application (if you are using desktop version)
3. Execute the following commands. Copy a screenshot as output after you execute the given command. Create a brief explanation of the command.

Command	Screenshot	Explanation
1. ls -l		lists files "-l" just means the long version of the list
2. ls -l ./Documents		lists files, but in the documents directory
3. whoami		Displays the username of the machine
4. Uname		shows system information, in this case, "Linux"
5. pwd		shows the current directory
6. echo Hi		basically the print command

7. history		shows the previous commands
8. history 5		shows the previous x commands, in this case, x = 5
9. !9		"!" repeats the previous command, in this case, "history" then 9
10. echo Hello Student		basically the print command
11. echo $HISTSIZE		shows the $HISTSIZE
12. echo $PATH		print the path of something
13. which date		shows the location of a variable
14. type cd		shows what type a command is
15. type ls		shows what type a command is
16. alias		shows the current aliases of the system
17. type vi		shows what type a command is
18. cd /bin		moves the current path of the command line to /bin
19. type vlc		this is supposed to show what kind of command vlc is, but it seems vlc is not a command in my system
20. cd		brings us back 1 directory
21. echo Today is `date`		the quotes are to make sure date is unchanged
22. echo Today is $(date)		$(date) is the actual date, so that's what it prints

23. echo This is the command \`date\`		this just gets the date
24. echo This is the command \`date\`		this made it so that the quotes are kept in the output of the command
25. echo This is the command "\`date\`"		another way of keeping the quotes using double quotes
26. echo D*		Paths/folders that have D in the start
27. echo "D*"		directs to quote the command directly
28. echo Hello; echo Linux; echo Student		the ";" separates each command
29. false; echo Not; echo Conditional		from what I can tell, it sets a false, then echos commands
30. echo Start && echo Going && echo Gone		&& means "and", can be a bitwise operator
31. echo Success && false && echo Bye		since there is a false, it fails the and operation
32. false || echo Fail Or		|| is an or operator, so both commands are ran
33. true || echo Nothing to see here		the || operator fails so the next command is not ran
34. printenv		it prints the environment variables
35. printenv TERM		print the value of the corresponding environment variable
36. echo $TERM		same as last
37. env		also shows the environment variables

6. Supplementary Activity:

Copy screen shot(s) of the following tasks:

1. An alias can be used to map longer commands to shorter key sequences. Use an alias to represent a very long command.

```
qmaiko@salgado:~$ alias guesswhat='echo Chicken butt'
qmaiko@salgado:~$ guesswhat
Chicken butt
```

2. Create a new directory in the Documents directory. Rename the directory as CPE_201A _(lastname). Create a new file inside the CPE_201A_(lastname) directory. Rename the file as sample1_lastname.txt. Display the content of the CPE_201A _(lastname) directory by executing one line of command only.

```
qmaiko@salgado:~/Documents$ mkdir CPE11S5_Salgado && touch CPE11S5_Salgado/Sampl
e1_Salgado.txt && ls ~/Documents/CPE11S5_Salgado
Sample1_Salgado.txt
```

3. Execute a command to display the working shell.

```
qmaiko@salgado:~/Documents$ echo $SHELL
/bin/bash
```

4. Shell variables, called environment variables, have the string data type and typically are named with capital letters and the _ (underline) character. Names are case sensitive. The env command will list all the environment variables. The printenv command will list all or will list only the names on its command line. List all environment variables. Which start with P?

```
qmaiko@salgado:~$ printenv | grep ^P
PWD=/home/qmaiko
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/us
r/local/games:/snap/bin:/snap/bin
```

7. Conclusion:

As useful as it may be, most of the given commands here are not quite usable with the current state that most Linux operating systems operate. Most already have file managers, and an average user will most likely won't touch the terminal.

8. Assessment (Rubric for Laboratory Performance):