Activity No. 11	
Command Line Skills	
Course Code: CPE 201A	**Program: CPE**
Course Title: COMPUTER SYSTEM ADMINISTRATION AND TROUBLESHOOTING	**Date Performed:10/23/2025**
Section: CPE11S1	**Date Submitted:10/23/25**
Name: Will Stuart D. Ponce Jr.	**Instructor: Engr Jimlord Quejado**

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:

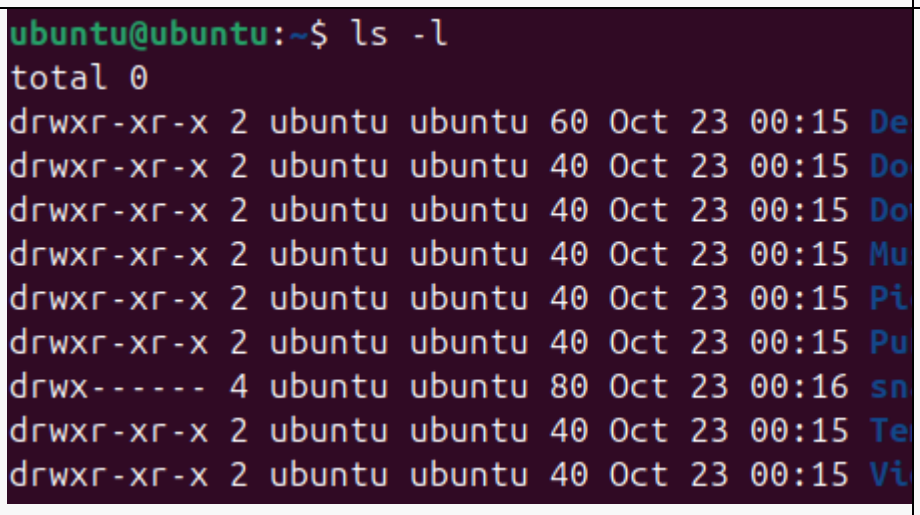
username
Name of the user that runs the shell
hostname
Name of the host on which the shell runs. There is also a command hostname , with which you can show or set the system's host name.
current_directory
The directory that the shell is currently in. A ~ means that the shell is in the current user's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

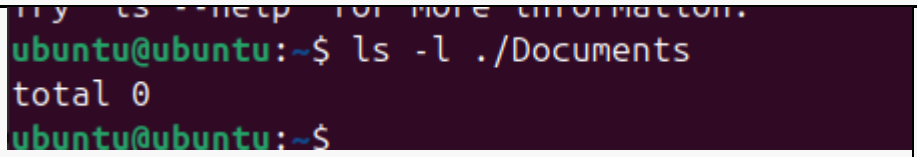
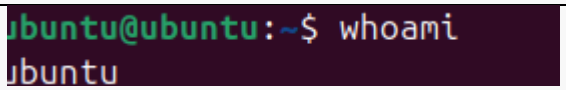
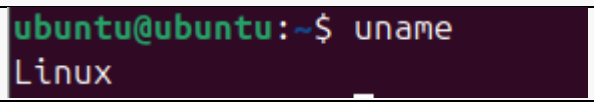
4. Resources:

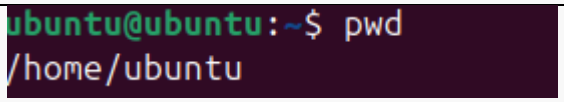
Personal Computer with installed Virtual Box
Ubuntu Server or Desktop virtual machine

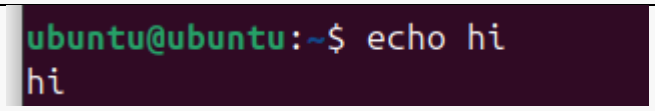
5. Procedure:

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.

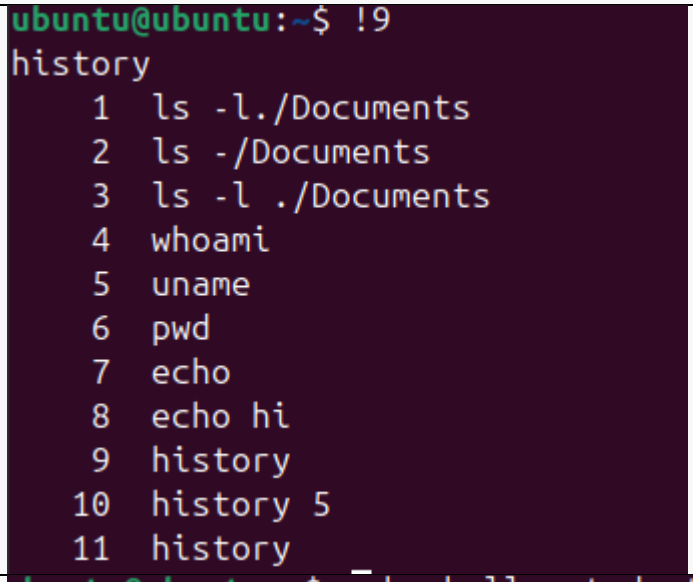
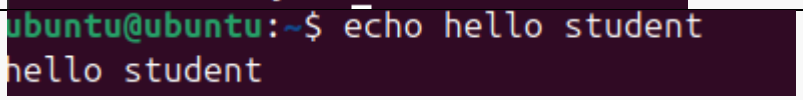
Command	Screenshot	Explanation
1. ls -l		used to list information about files and directories within the file system. The ls utility is a part of the GNU core utilities package which is installed on all Linux distributions .

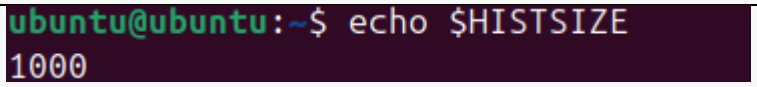
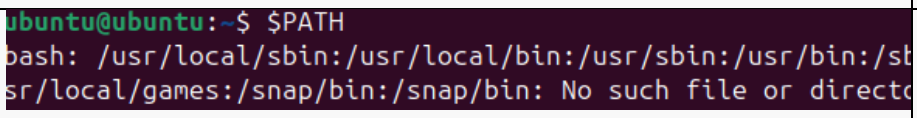
2. ls -l ./Documents		"ls" stands for "list segments" which essentially means display all non unix items inside of the current directory. This being either directories, files or hidden files if you pass it the flag -a like so "ls -a".
3. whoami		The whoami command is a utility found in most Unix-like operating systems, as well as Microsoft Windows and ReactOS. Its purpose is to display information about the user who is currently logged on or executing the command.
4. Uname		The uname command in

		Unix-like operating systems, including Linux, stands for "Unix name." It is a utility used to print system information, specifically details about the operating system kernel and underlying hardware.
5. pwd		The pwd command stands for Print Working Directory. It is a fundamental command in Unix-like operating systems, including Linux and macOS. When executed in a terminal or command-line interface, pwd displays the absolute

		path of the current directory (folder) where the user is currently located within the file system hierarchy. This path starts from the root directory (represented by /) and shows the full sequence of directories leading to the current location.
6. echo Hi		The echo command in Linux is a fundamental utility used to display lines of text or strings, including the values of variables, to the standard output (typically the terminal). It is a built-in command in most shells, such as Bash, and is

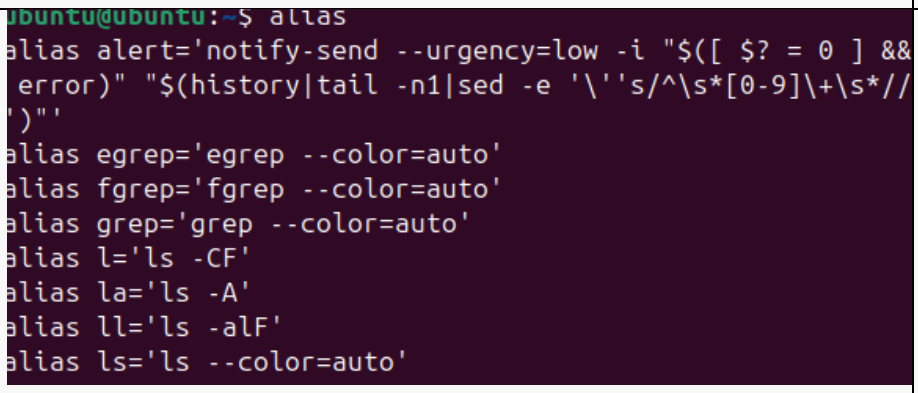
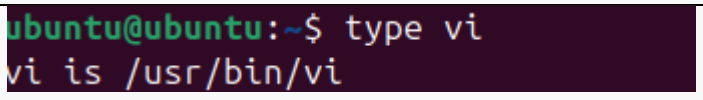
		widely used in shell scripts and command-line operations for various purposes.
7. history	ubuntu@ubuntu:~$ history 1 ls -l./Documents 2 ls -/Documents 3 ls -l ./Documents 4 whoami 5 uname 6 pwd 7 echo 8 echo hi 9 history	In Linux, "history" refers to the command that displays a list of recently executed commands in the terminal. This feature is built into shells like Bash and is useful for recalling, reusing, and re-executing commands without retyping them
8. history 5	ubuntu@ubuntu:~$ history 5 6 pwd 7 echo 8 echo hi 9 history 10 history 5	n Linux, the command history 5 displays the last five commands executed in the current shell session. The history command, when used without any

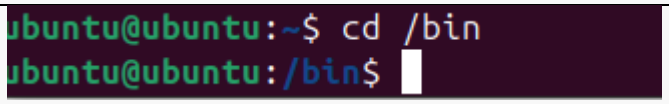
		arguments, lists all commands stored in the shell's history. When a number is appended to history, such as history 5, it restricts the output to only the most recent commands, in this case, the last five.
9. !9		In Linux, !9 or more specifically, the number 9 when used with the kill command, refers to the SIGKILL signal.
10. echo Hello Student		The command echo Hello Student prints the text "Hello Student" and is commonly used in programmin g and

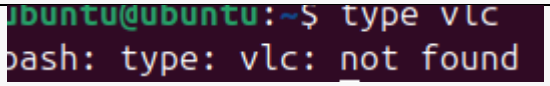
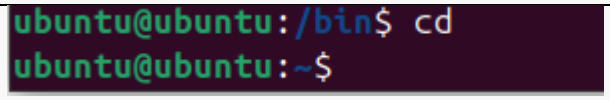
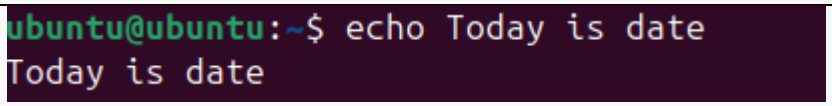
		command-line interfaces. It is also the name of a student accommodation provider in the UK, Hello Student, which offers student housing.
11. echo $HISTSIZE		To echo $HISTSIZE means to display the value of the HISTSIZE environment variable. It tells you how many command lines are remembered in your current shell's command history.
12. echo $PATH		The command echo $PATH in a Unix-like operating system (such as Linux or macOS) displays the value of the PATH

		environment variable.
13. which date	ubuntu@ubuntu:~$ which date /usr/bin/date	Which date" can refer to a specific day on a calendar (e.g., "Which date is your birthday?") or a social/romantic meeting (e.g., "I have a date on Friday"). It can also be used to refer to a period of time in general or the age of an object.
14. type cd	ubuntu@ubuntu:~$ type cd cd is a shell builtin	The acronym "cd" can have multiple meanings depending on the context, including "compact disc," a type of optical disc used for storing data. In computing, it can be the command to "change directory".

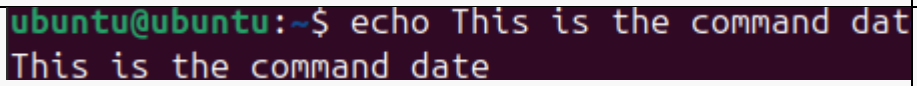
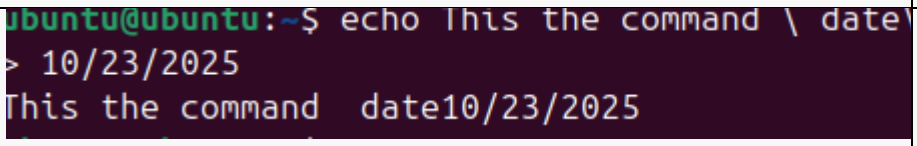
		In finance, "CD" stands for "certificate of deposit," a type of savings account. Additionally, in a personality assessment, a "CD" style may refer to a Conscientious-Dominant profile.
15. type ls	`ubuntu@ubuntu:~$ type ls` `ls is aliased to `ls --color=auto'`	In a computer command line, ls is a command that lists files and directories within a directory, similar to how a file explorer shows a file list. It's short for "list" and is one of the most fundamental commands for navigating and managing files in operating

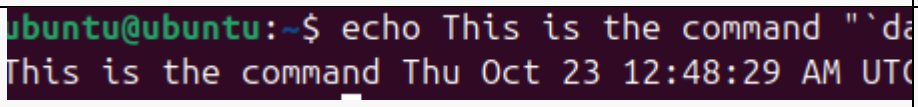
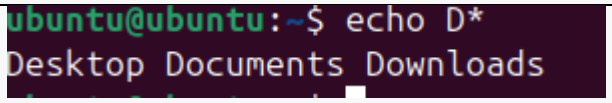
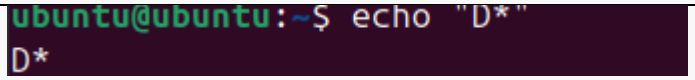
		systems like Linux and Unix. Other meanings include "Long Story" in social media, or "letter signed" and "library science" in abbreviations.
16. alias		In Linux, an alias is a user-defined shortcut or a custom name assigned to a longer command or sequence of commands. It allows users to replace frequently used, lengthy, or complex commands with a shorter, more memorable alternative.
17. type vi		Type VI" can refer to several different things

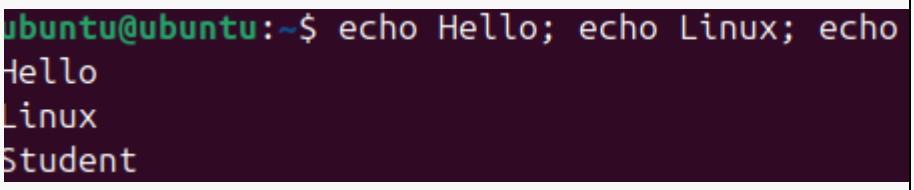
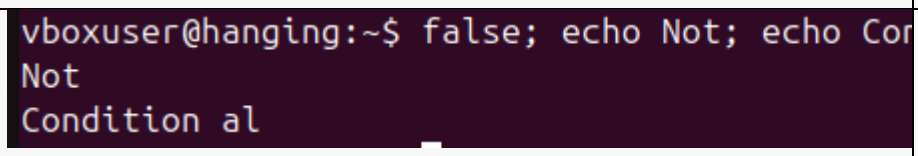
		depending on the context, most commonly the Fitzpatrick skin type VI (very dark skin that never burns) or the Type VI secretion system (T6SS) (a complex protein delivery machine found in many bacteria)
18. cd /bin		"cd" can mean two different things: "change directory," a command to navigate folders in a command line, or "bin," short for binary, which refers to a file that stores data from a CD or DVD, notes Indeed and IONOS. When used together,

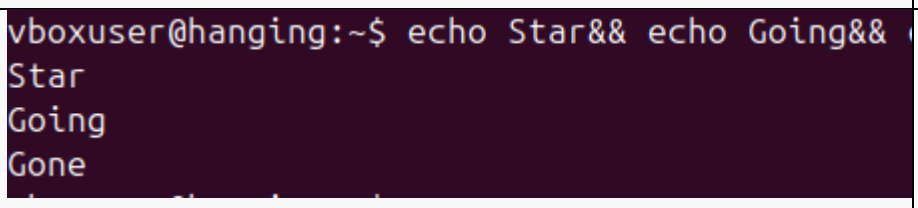
		such as in the command cd bin, it means changing the current directory to one named "bin".
19. type vlc		To launch the VLC media player in Linux, you can either use the graphical user interface (GUI) or the command line
20. cd		In Linux, cd is a fundamental shell command that stands for "change directory." Its primary function is to navigate the file system by changing the current working directory of the shell.
21. echo Today is `date`		In a command-line environment like a Unix

		or Linux shell, echo Today is \date`means that the shell will first execute thedatecommand and then use the output of that command as an argument for theecho` command
22. echo Today is $(date)	ubuntu@ubuntu:~$ echo Today is$(date) Today isThu Oct 23 12:41:34 AM UTC 2025	The echo command in Linux is a fundamental utility used to display lines of text or strings, including the values of variables, to the standard output (typically the terminal). It is a built-in command in most shells, such as Bash, and is widely used in shell scripts and command-line operations

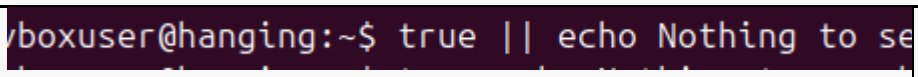
		for various purposes.
23. echo This is the command "`date`"		`date` will just expand to the output of the date command. However, it removes extra space characters at places where there are more than one consecutive space character in the output. (This is because the command substitution is subject to word splitting, and because of how the echo command handles multiple arguments.)
24. echo This is the command \`date\`		The command echo \date`` prints the current date and time to the terminal. It combines two separate commands

		in a specific way:
25. echo This is the command "`date`"		The command echo "This is the command \`date\`" combines two fundamental shell functionalities: echo and command substitution using backticks (` `).
26. echo D*		The command echo D* in a Unix-like shell (such as Bash) will print a list of all files and directories in the current working directory that begin with the letter "D".
27. echo "D*"		The command echo D* in a Unix-like shell (such as Bash) does not literally print "D*".

		Instead, the shell performs globbing (also known as wildcard expansion) before the echo command is executed.
28. echo Hello; echo Linux; echo Student		The command echo D* in a Unix-like shell (such as Bash) does not literally print "D*". Instead, the shell performs globbing (also known as wildcard expansion) before the echo command is executed.
29. false; echo Not; echo Conditional		The statement false; echo Not; echo Conditional meaning does not have a conditional meaning; instead, it executes three separate, sequential

		shell commands. The semicolon (;) is a command separator, telling the shell to execute the command on its left and then the command on its right, regardless of the success or failure of the first command
30. echo Start && echo Going && echo Gone		The command echo Start && echo Going && echo Gone is a sequence of three separate commands run in a shell, like Bash in Linux or the Command Prompt (CMD) in Windows. The && operator ensures that each subsequent

		command is only executed if the one before it succeeds
31. echo Success && false && echo Bye	vboxuser@hanging:~$ echo Success&& false&& ec Success	The command echo Success && false && echo Bye is a sequence of three commands chained together using the logical AND (&&) operator. In this sequence
32. false || echo Fail Or	vboxuser@hanging:~$ false || echo Fail Or Fail Or	In a shell command like false || echo Fail, the || is the logical OR operator. This construct is a common shell programming technique that executes the second command (echo Fail) only if the first command (false) fails

33. true || echo Nothing to see here		The command . true || echo Nothing to see here will do nothing and produce no output. This is due to how shell commands are processed, specifically the use of the . (source) command, the true command, and the || (logical OR) operator
34. printenv		The printenv command is a standard utility on Unix-like operating systems (including Linux and macOS) used to print all or part of the system's environment variables. Environment variables are dynamic key-value

		pairs that influence the behavior of the system and processes running within it.
35. printenv TERM	vboxuser@hanging:~$ printenv TERM xterm-256color	The command printenv TERM means "print the value of the TERM environment variable". In a Unix-like operating system like Linux, TERM is a critical environment variable that tells applications what type of terminal is being used.
36. echo $TERM	vboxuser@hanging:~$ echo $TERM xterm-256color	The echo $TERM command in Linux displays the value of the TERM environment variable. This variable specifies the type of terminal emulation

		being used for the current session.
37. env	LOGNAME=vboxuser XDG_SESSION_DESKTOP=ubuntu XDG_SESSION_TYPE=wayland SYSTEMD_EXEC_PID=2337 XAUTHORITY=/run/user/1000/.mutter-Xwaylandauth.723WE3 HOME=/home/vboxuser USERNAME=vboxuser IM_CONFIG_PHASE=1 LANG=C.UTF-8 XDG_CURRENT_DESKTOP=ubuntu:GNOME MEMORY_PRESSURE_WATCH=/sys/fs/cgroup/user.slice/user-1000.slice/user@1000.service/se service/memory.pressure VTE_VERSION=7600 WAYLAND_DISPLAY=wayland-0 GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/c3017587_fec1_415c_8054_85e16d07a2a GNOME_SETUP_DISPLAY=:1 XDG_SESSION_CLASS=user TERM=xterm-256color USER=vboxuser GNOME_TERMINAL_SERVICE=:1.103 DISPLAY=:0 SHLVL=1 GSM_SKIP_SSH_AGENT_WORKAROUND=true QT_IM_MODULE=ibus XDG_RUNTIME_DIR=/run/user/1000 DEBUGINFOD_URLS=https://debuginfod.ubuntu.com XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/ PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/lo GDMSESSION=ubuntu DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus _=/usr/bin/env	The env command in Linux is a utility used to manage environment variables. Environmen t variables are dynamic named values that influence the behavior of programs and processes within the operating system

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.
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.
3. Execute a command to display the working shell.
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?

```
vboxuser@hanging:~/Documents$ ls
'CPE_201A_(Ponce'
```

```
vboxuser@hanging:~$ echo $SHELL
/bin/bash
vboxuser@hanging:~$ $0
To run a command as administrator (user "root
See "man sudo_root" for details.

vboxuser@hanging:~$ echo $0
bash
vboxuser@hanging:~$ ps -p $$
    PID TTY          TIME CMD
   3947 pts/0    00:00:00 bash
vboxuser@hanging:~$
```

```
vboxuser@hanging:~$ printenv | grep ^P
PWD=/home/vboxuser
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
```

7. Conclusion:

Based on the activity, this conclusion reflects on the completed tasks:

This activity successfully met its objective of executing basic commands using the Linux command line interface. I learned firsthand why the Linux community values the CLI for its power, speed, and precise control. The discussion clarified the role of the BASH shell as the command line interpreter that translates user input into actions. Through the procedures, I practiced essential commands like ls -l to view file details, whoami to identify the current user, pwd to display the working directory, and history to review previous commands. The supplementary activity also provided practical experience in creating directories and files and using an alias to map a shorter command to a longer one.

I was also able to demonstrate the intended learning outcomes regarding BASH features, shell variables, and quoting. I learned to display the values of shell variables by using echo with $HISTSIZE and $PATH, and how to list all environment variables using printenv and env. The exercises using different quoting methods were particularly insightful; I saw how command substitution works using backticks (``) and the $(...) notation. I also observed how double quotes ("") prevent wildcard expansion, as seen in the echo "D*" command. Finally, I learned how to control command execution by chaining them with; (semicolon), && (logical AND), and || (logical OR).

8. Assessment (Rubric for Laboratory Performance):