FORMATIVE ASSESSMENT 6

1. Program Outcomes (POs) Addressed by the Activity

a. ability to design and conduct experiments, as well as to analyze and interpret data
b. ability to identify, formulate, and solve engineering problems
c. ability to use techniques, skills, and modern engineering tools necessary for engineering practice.
d. knowledge and understanding of engineering and management principles as a member and leader in a team, to manage projects and in multidisciplinary environments

2. Activity's Intended Learning Outcomes (AILOs)

At the end of this activity, the student shall be able to:

a. Able to perform file and directory creation, and move between directories using LINUX commands
b. Able to perform file and directory creation, and move between directories using LINUX commands
c. Able to understand LINUX file system/structure, partition, filename format, absolute and relative paths.
d. Able to create different file types and display directory content using LINUX wildcards and commands.
e. Able to create files and edit file content using vi editor, and display file content using various LINUX file readers
f. Able to utilize different file operation, data refinement commands, and redirectors in file manipulation.
g. Able to assign/modify permission and different ownerships to files and directories using symbolic and absolute file permission commands.
h. Able to create/delete user/group and modify its corresponding details.

3. Objectives of the Activity

The objectives of this activity are to:

a. Demonstrate an understanding of Disk Operating System (DOS) and LINUX history and concepts.
b. Perform file and directory creation and manipulation using DOS commands; LINUX installation in virtual machine, file and directory creation and manipulation, and system administration using LINUX commands.
c. Perform file and directory creation and manipulation using DOS commands; LINUX installation in virtual machine, file and directory creation and manipulation, and system administration using LINUX commands.

d. Perform file and directory creation and manipulation using DOS commands; LINUX installation in virtual machine, file and directory creation and manipulation, and system administration using LINUX commands..
e. Perform file and directory creation and manipulation using DOS commands; LINUX installation in virtual machine, file and directory creation and manipulation, and system administration using LINUX commands.

4. Principle of the Activity

The standard of the Linux file system consists of a set of guidelines and requirements for file and directory placement. These guidelines are intended to support interoperability of applications, system administration tools, development tools and scripts as well as greater uniformity in documentation.

Directory Commands

A. To display the contents of the current working directory

Syntax:

ls

Options that can be used with it.

-a

-A

-d

-l

-r

-R

B. To change directory

Syntax:

- cd . means the current directory
- cd .. means parent directory

- cd means will take you to your home directory
- cd – will take you to your previous directory
- cd ~ username will take you to the home directory of the user
- cd <directory_name>
- cd <directorypath>

C. To make directory/directories

Syntax:

mkdir [-option] directory1 directory2 …

Examples:

To create directories dir1, dir2, dir3, on the current directory:

Syntax:

mkdir dir1 dir2 dir3

mkdir my\ folder

To create the directory /home/tester/mydir/testdir (mydir is not yet existing):

Syntax:

mkdir –p /home/tester/mydir/testdir

D. To remove directory

Syntax:

rmdir [-option] directoryname

**This command allows removing EMPTY directories.*

Examples:

To remove the directory dir1

Syntax:

rmdir dir1

To remove the directory /home/tester/mydir/testdir and its parent directory

Syntax:

rmdir -p mydir/testdir

To remove directory that is not empty

Syntax:

rm -r directoryname

rm -rf directoryname

E. To print or display the current working directory.
Syntax: pwd

F. To clear the screen.
Syntax: clear

G. To display previously entered commands. This information is stored the ~/.bash_history file located at the home directory of each user.
Syntax: history

The following are the general terminologies and concepts used in Linux operating system:

- **PARTITION**. This is a logical division of a hard disk created so that you can have different operating system on the same hard disk, or to create the impression of having separate hard drives for file management, multiple users or other purposes.
- **FILESYSTEM.** This refers to the way in which file are named and where they are placed logically for storage and retrieval.
- **ABSOLUTE AND RELATIVE PATH**. The highest directory in the Linux directory tree is the /. To go from one path to the other, we can always start from the top (/) to the directory where we want to go (absolute), or specify the path from our current location to the directory where we want to go (relative).

- **AUTOCOMPLETE, WILDCARDS**. Autocomplete works on files and directories. Just press Tab to complete the directory or file that you are looking for. Wildcards only work on files. Wildcards such as *, ., ?, ~, .., are the ones used in Linux system
- **LINUX FILENAMES**. Linux allows filenames to be up to 256 characters long. These characters can be lower and upper case letters, numbers, and other characters, usually the dash (-), the underscore (_), and the dot (.). Linux filenames do not follow the concept of file extensions as in DOS. IN other words, period or dot (.) is just like an ordinary character. A filename preceded by a dot (.) will become a hidden file. Also, Linux filenames are case sensitive, unlike in DOS or Windows.

  vi Editor

  You can use **vi** editor to edit an existing file or to create a new file from scratch. You can also use this editor to just read a text file.

Starting the vi Editor:

There are following way you can start using vi editor:

Command	Description
vi filename	Creates a new file if it already does not exist, otherwise opens existing file.
vi -R filename	Opens an existing file in read only mode.
view filename	Opens an existing file in read only mode.
:f filename	Renames current file to filename.

Following is the example to create a new file **testfile** if it already does not exist in the current working directory:

```
$vi testfile
```

As a result you would see a screen something like as follows:

```
~
~
~
~
~
"testfile" [New File]
```

You will notice a tilde (~) on each line following the cursor. A tilde represents an unused line. If a line does not begin with a tilde and appears to be blank, there is a space, tab, newline, or some other nonviewable character present.

So now you have opened one file to start with. Before proceeding further let us understanding few minor but important concepts explained below.

Operation Modes:

While working with vi editor you would come across following two modes:

- **Command mode:** This mode enables you to perform administrative tasks such as saving files, executing commands, moving the cursor, cutting and pasting lines or words, and finding and replacing. In this mode, whatever you type is interpreted as a command.

- **Insert mode:** This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and finally it is put in the file.

The **vi** always starts in command mode. To enter text, you must be in insert mode. To come in insert mode you simply ***type i***. To get out of insert mode, press the Esc key, which will put you back into command mode.

Hint: If you are not sure which mode you are in, press the **Esc key** twice, and then you'll be in command mode. You open a file using vi editor and start type some characters and then come in command mode to understand the difference.

Getting Out of vi:
The command to quit out of vi is ***:q***. Once in command mode, type colon, and 'q', followed by return. If your file has been modified in any way, the editor will warn you of this, and not let you quit. To ignore this message, the command to quit out of vi without saving is ***:q!***. This lets you exit vi without saving any of the changes.

The command to save the contents of the editor is ***:w***. You can combine the above command with the quit command, or ***:wq*** and return.

The easiest way to save your changes and exit out of vi is the ZZ command. When you are in command mode, type ZZ and it will do the equivalent of :wq.

You can specify a different file name to save to by specifying the name after the :w.
For example, if you wanted to save the file you were working as another filename called filename2, you would type ***:w filename2*** and return. Try it once.

File Operation Commands

The **cp** command allows copying file/s and directories from one location to another.
Syntax:

cp source destination

The **mv** command allows to move and rename files
Syntax:

mv oldname newname
mv source destination

The **touch** command allows creating an empty file. It also allows updating the time stamp on existing file.

Syntax:

touch filename1 filename2

The **rm** command will delete a file forever.

Syntax:

rm [-option] filename1 filename2 …

Examples:

To delete the file aa:

Syntax: rm aa

To prompt first before removing the file bb:

Syntax: rm –I bb

To remove all files and all sub-directories and their contents:

Syntax: rm -r *

To remove forcefully all files and all sub-directories and their contents:

Syntax: rm -rf *

***Caution*:** Exercise caution when executing the last two commands. Should you exercise either of them from the root directory (/), your system will definitely crash.

The **echo** command displays the string or text specified after it. It also used to reference and display the values of variables. It is commonly used in programs, or shell scripts, were user input is needed.

Syntax:

echo [string]
echo $variablename

Examples:

To re-echo the word "hello" on the command line:

Syntax: echo hello

To display the value of the variable "x"

Syntax: x=hello
Syntax: echo $x

The **cmp** command checks two files to see if they differ. It does a byte-by-byte comparison of file1 and file2. If the files differ, cmp outputs the location at which the first difference occurs.

Syntax:

cmp [options] file1 [file2]

The **file** command determines the file type of a given file. It reads the first few bytes of a file to determine the file type.

Syntax:

file [filename]

Redirection Standard Input/Output

There are three main redirection symbols >, >>, <

Redirector Symbol >

Syntax: Linux-command > filename

To output Linux-commands result (output of command or shell script) to file. Note that if file already exist, it will be overwritten else new file is created.

Example:

To send output of ls command

Syntax: ls > myfiles

Note: Now if 'myfiles' file exist in your current directory it will be overwritten without any type of warning.

Redirector Symbol >>

Syntax: Linux-command >> filename

To output Linux-commands result (output of command or shell script) to END of file. Note that if file exist , it will be opened and new information/data will be written to END of file, without losing previous information/data, And if file is not exist, then new file is created.

Example:

To send output of date command to already exist file give command

Syntax: date >> myfiles

Note: Now if 'myfiles' file exist in your current directory it will be overwritten without any type of warning.

Redirector Symbol <

Syntax: Linux-command < filename

To take input to Linux-command from file instead of keyboard.

Example:

To take input for cat command give

Syntax: cat < myfiles

Note: Now if '**myfiles**' file exist in your current directory it will be overwritten without any type of warning.

| (Pipe Symbol)

A pipe is the same as redirecting standard output.

It is nothing but a temporary storage place where the output of one command is stored and then passed as the input for second command.

Pipes are used to run more than two commands (multiple commands) from same command line.

Syntax: command1 | command2

Example:

Syntax: cat hello.txt | cat > h.txt

The output from the first command cat hello.txt will be piped or temporarily stored and the stored value will serve as an input to the next command. Say the output of the command **cat hello.txt** is ***hello*** then the content of **h.txt** is also ***hello***

Data Refinement Commands

The **sort** command sorts and/or merges one or more text files in sequence.

Syntax:

sort filename

The **uniq** command displays a file, removing all but one copy of successive repeated lines. If the file has been sorted, **uniq** ensures that no two lines that it displays are the same.

Syntax:

uniq filename

grep command

- Used for pattern searching.
- Users can use this command to search a set of files for one or more phrases or patterns.
- If the pattern exists, then grep will print all the lines that contain the said pattern.

Syntax: grep pattern <filename>

where:

pattern is the phrase or pattern the user wants to find.

filename is the name of the target file.

5. Materials/Equipment

1 unit Personal Computer
DOS / Command Prompt
Storage unit

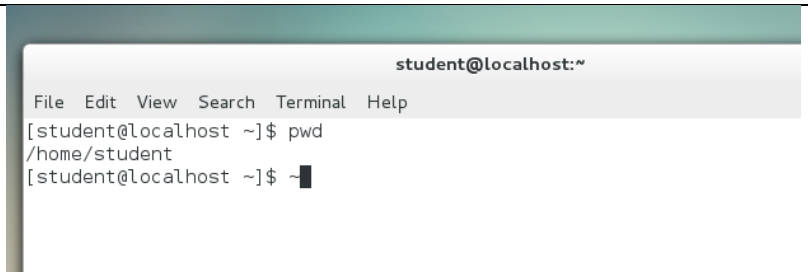
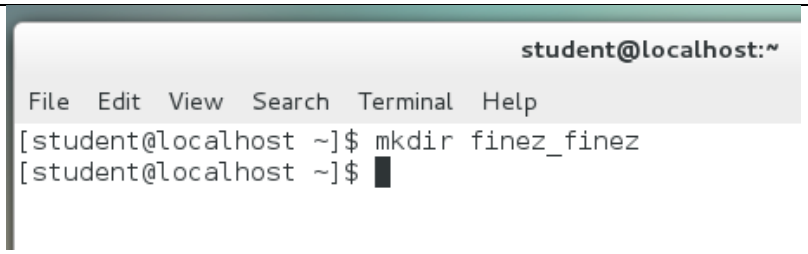
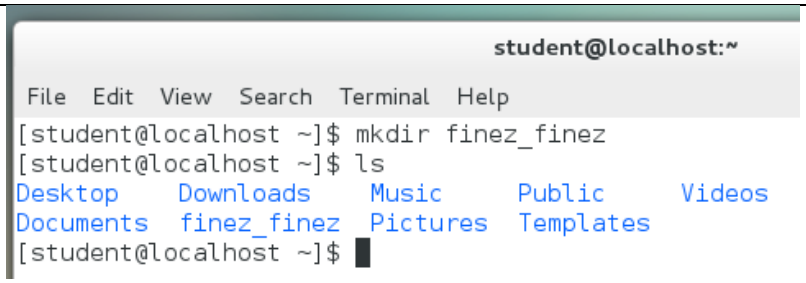
6. Circuit Diagrams / Figures / Source Codes (if Applicable)

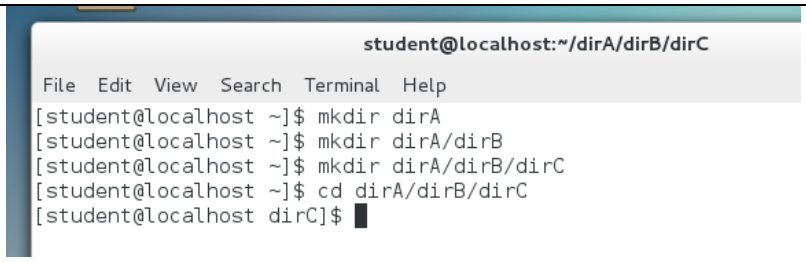
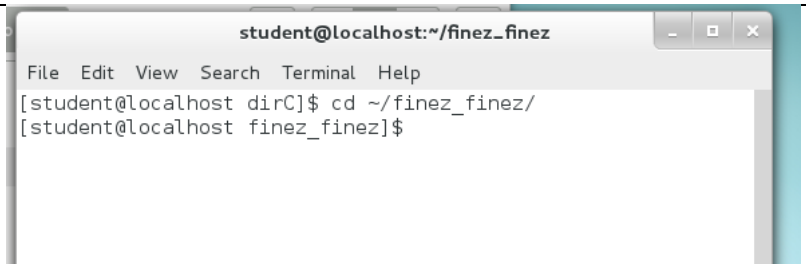
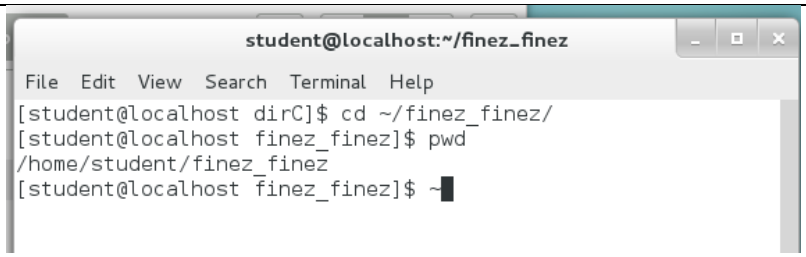
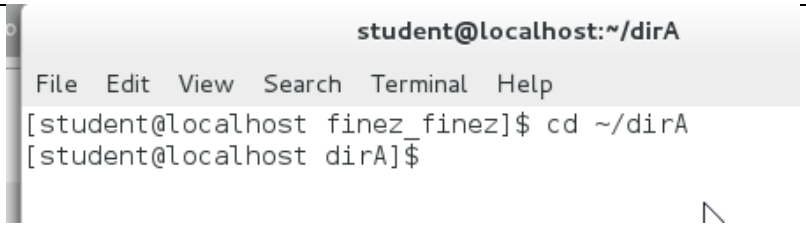
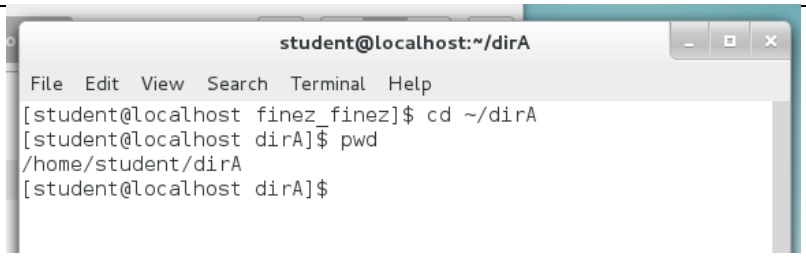
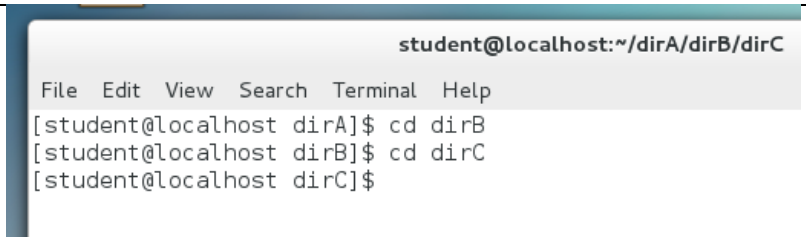
Figure 1. *Title*

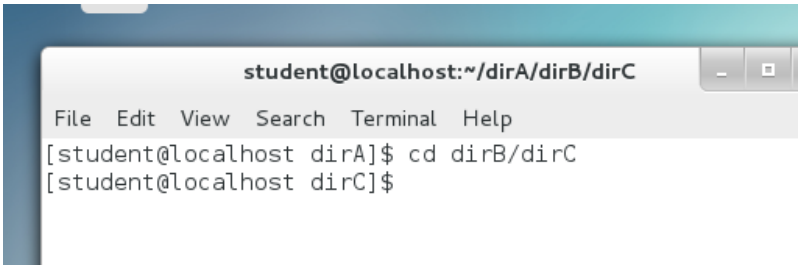
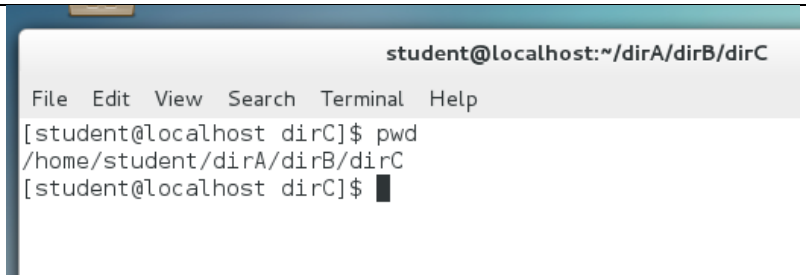
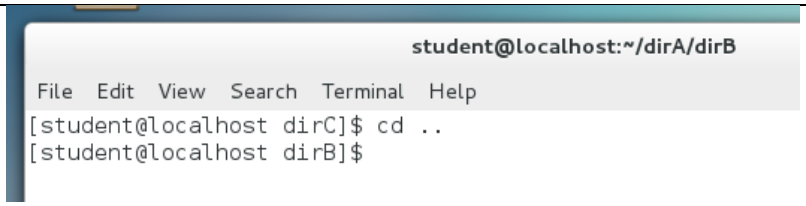
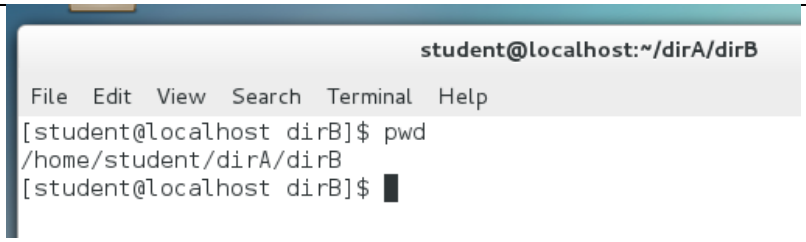
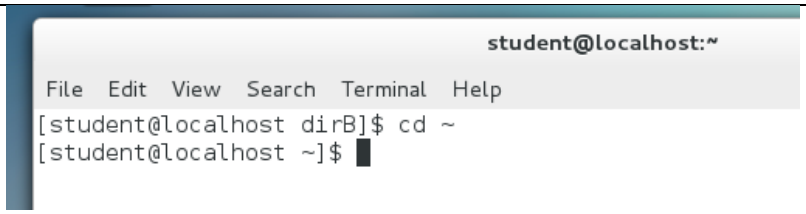
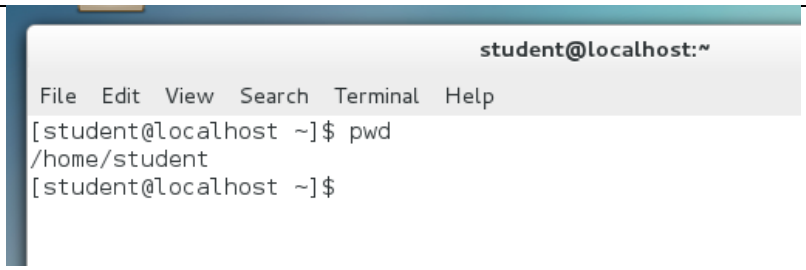
7. Activity Report

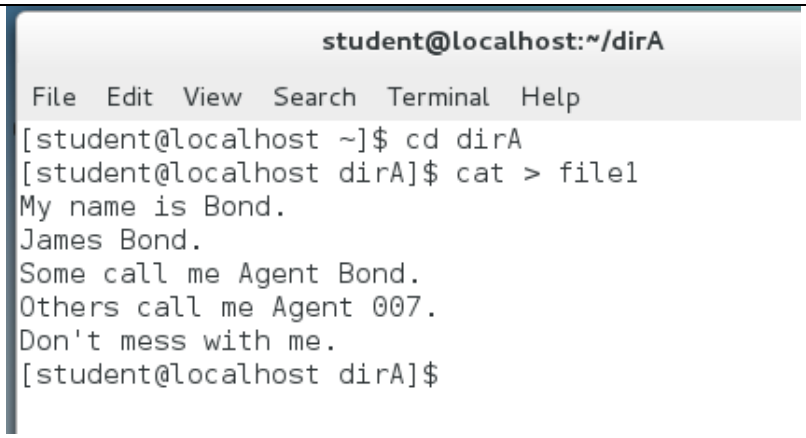
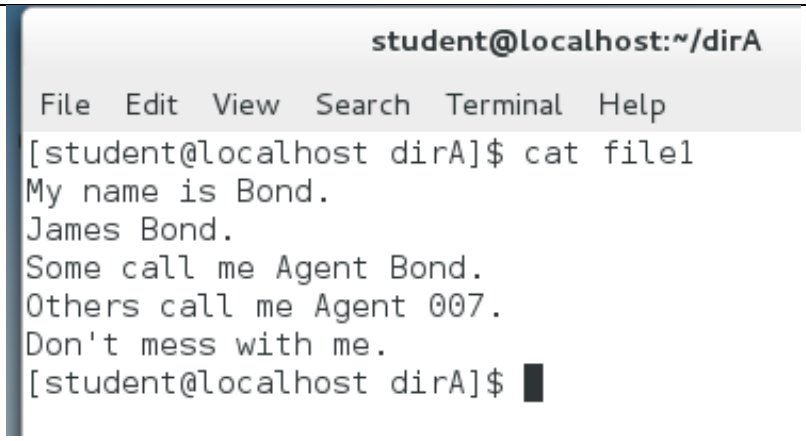
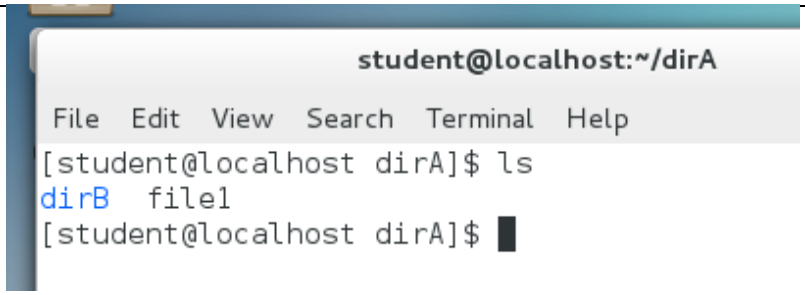
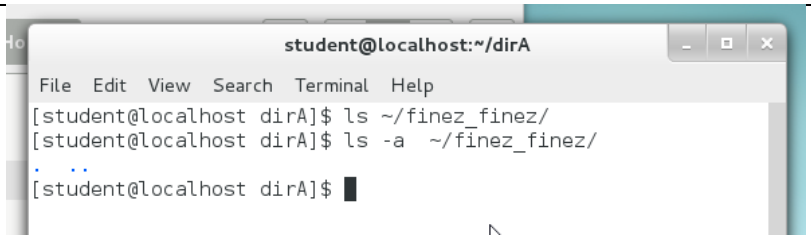
Section: TW32	**Date Performed: 26 March 2025**
Course Code: IT0047	**Date Submitted: 26 March 2025**
Course Title: Computer Systems and Platform Technologies	
Instructor: Mr. Jon Errol Troy Santiago	
Student Name: FINEZ, JORELL ANDREI	**Activity No.: FA6**

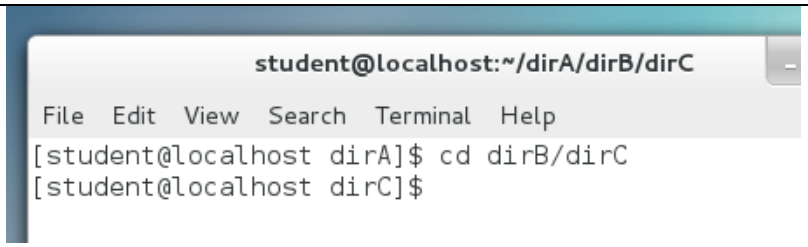
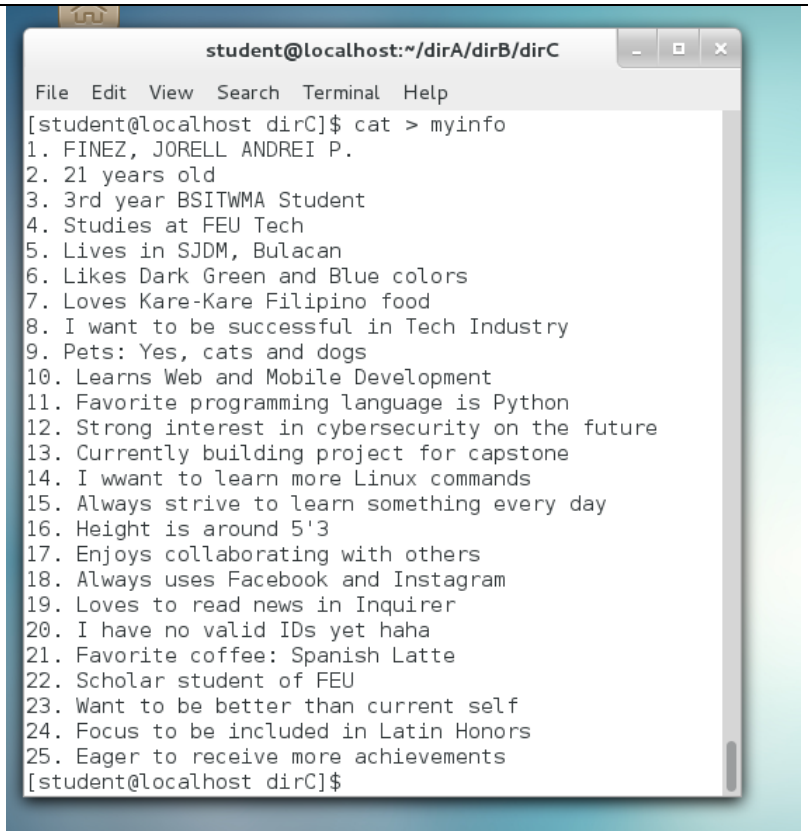
8. Data and Results

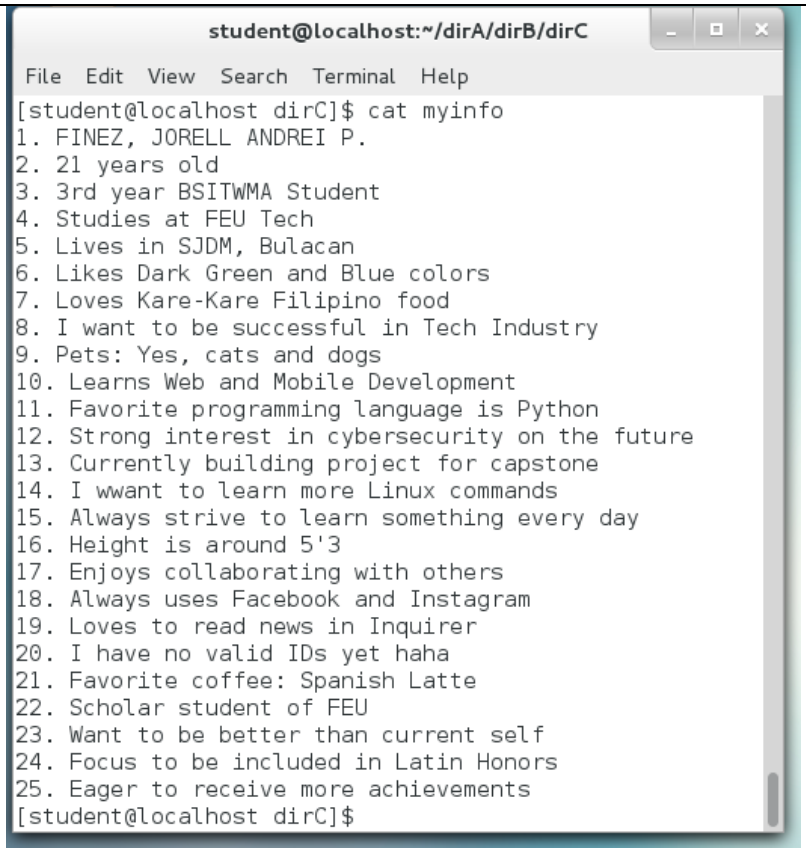
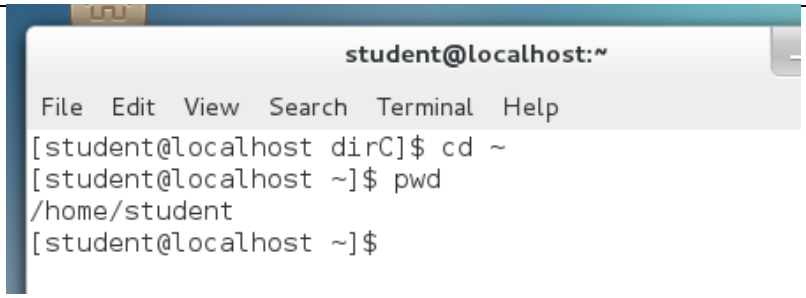
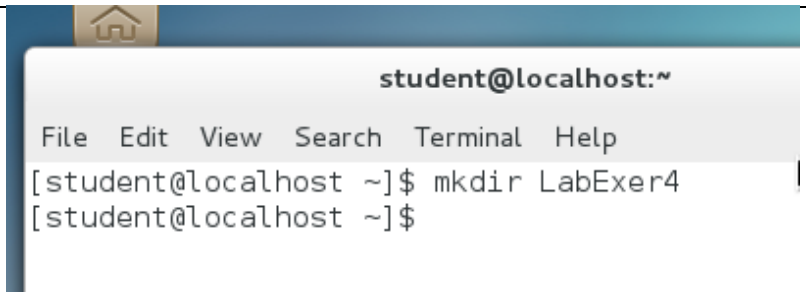
PART 1: Directory Commands	Paste your captured executed command
1. Type the command ***pwd*** at your default prompt :~$	
2. What is the meaning of ***pwd***?	"**pwd**" stands for "print working directory" and is a command used to display the full path of your current directory
3. Create a directory at your current directory**.** *(Use your **surname1_surname2** as your directory name)*	
4**.** Type ***ls*** to check the content of your current working directory.	

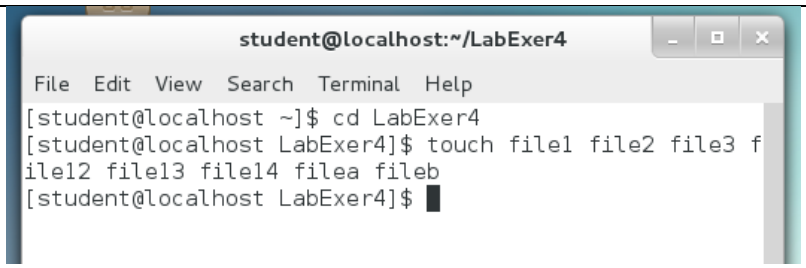
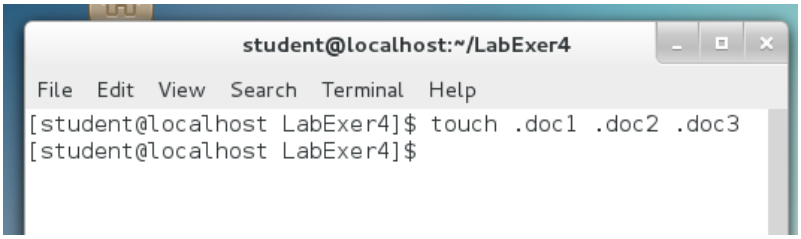
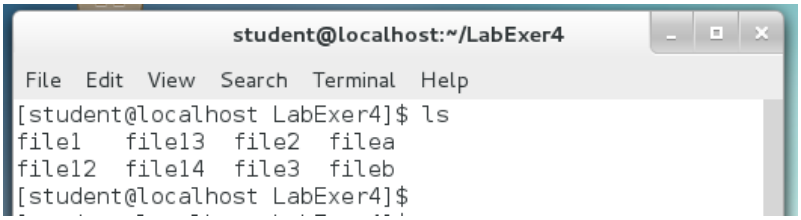
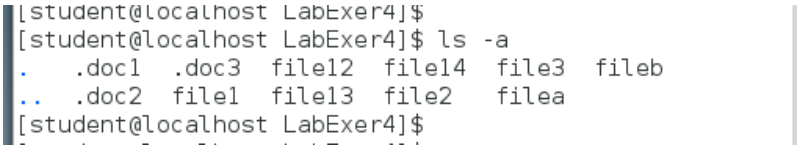
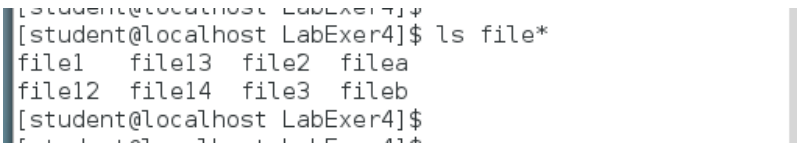
5. Create a directory tree. where **dirC** will be created under **dirB** and **dirB** is created under **dirA.**	
6. Move to your surname directory.	
7. Display your present working directory?	
8. Move to the **dirA** directory.	
9. Display your present working directory?	
10. Move to **dirB** and **dirC** sub-directories.	

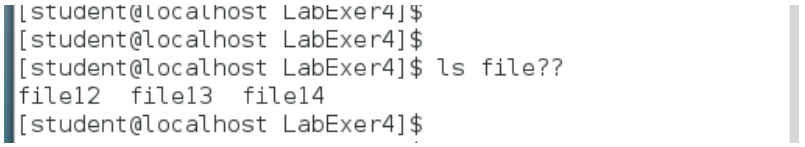
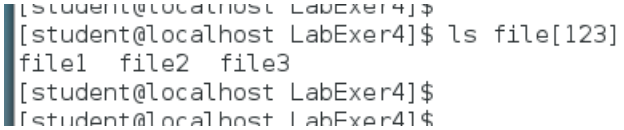
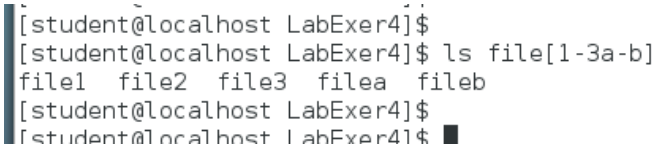
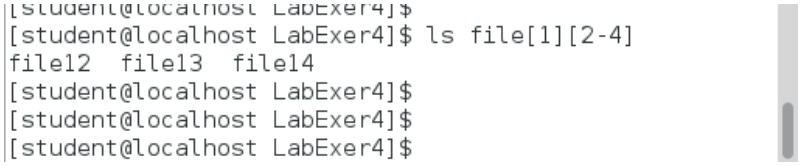
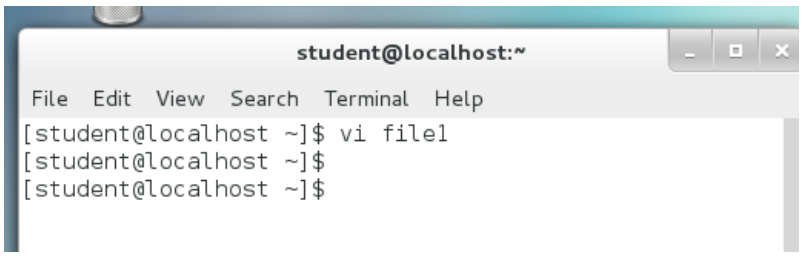
	Or we can also do this: 
11. Display your present working directory?	
12. Move one level hierarchy down from the current directory.	
13. Display your present working directory?	
14. Go back to your home directory.	
10. Display your present working directory?	

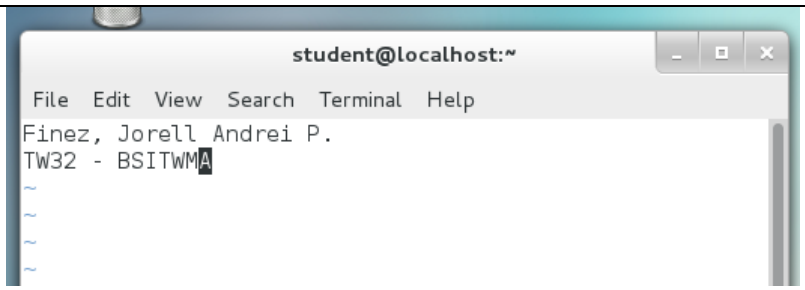
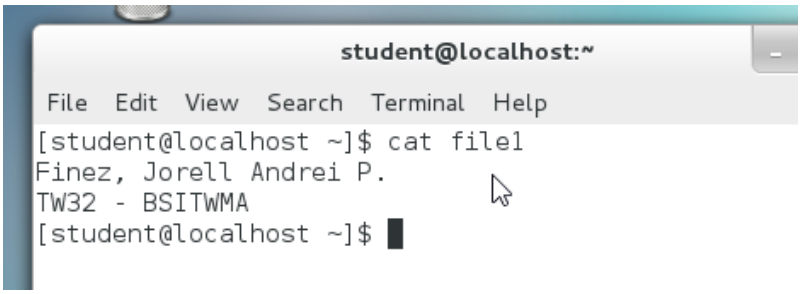
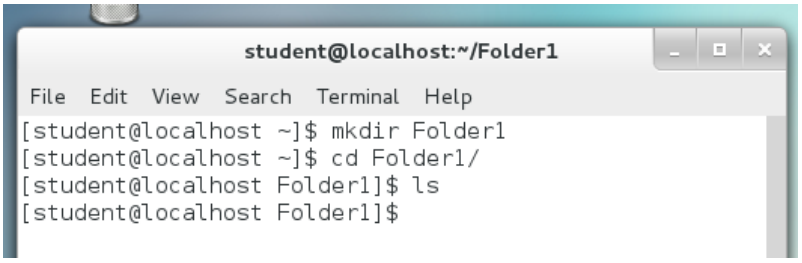
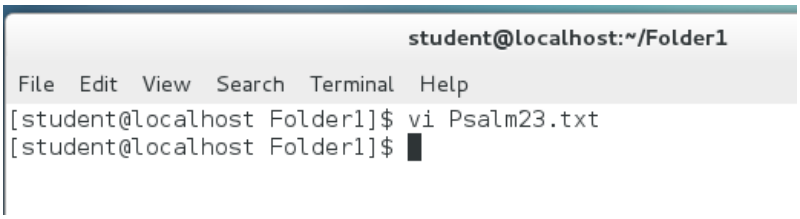
15. Create a file called file1 in **dirA**. (use cat command and type the ff. **My name is Bond.** **James Bond.** **Some call me Agent Bond.** **Others call me Agent 007.** **Don't mess with me.** **Ctrl+D**)	
16. Open the file1. (use cat command)	
17. List the file/s at **dirA.**	
18. List the files / directories in your surname directory.	

19. Go to your **dirC** directory.	
20. Create a file called **myinfo** using cat command in **dirC.** The file should contain 25 lines about yourself. Make the first line as your full name. Number each line.	

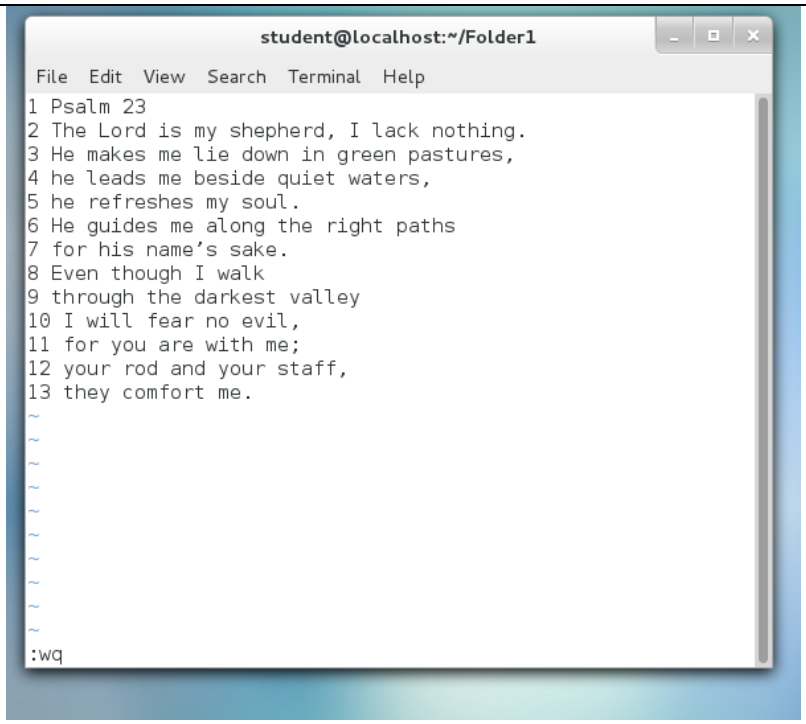
21. Open your **myinfo** file using cat command.	
PART 2: Creating Files	Paste your captured executed command
1. Go inside your home directory (/home/<*username*>).	
2. Create a directory named LabExer4.	

3. Inside the LabExer4 directory, create the following files using touch command: file1 file2 file3 file12 file13 file14 filea fileb	
4. Also, inside the LabExer4 directory, create the hidden files .doc1 .doc2 .doc3	
5. Using the ls command, display the content of the directory LabExer4.	
6. Using the ls –a command, display the content of the directory LabExer4.	
7. Using the command ls file*, display the content of the directory LabExer4.	

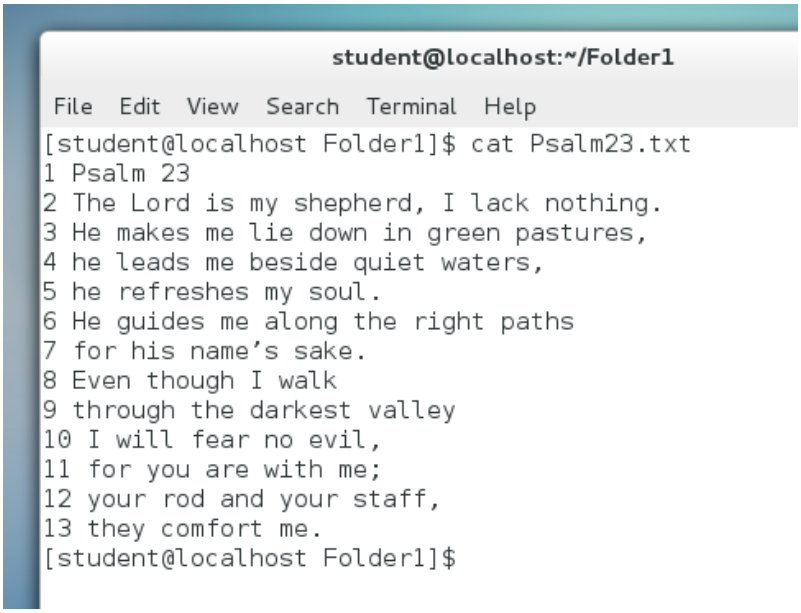
8. Using the command ls file??, display the content of the directory LabExer4.	
9. Using the command ls file[123], display the content of the directory LabExer4.	
10. Using the command ls file[1-3a-b], display the content of the directory LabExer4.	
11. Using the command ls file[1][2-4], display the content of the directory LabExer4.	
Part 3: vi Editor	Paste your captured executed command
1. To create a file using vi editor (use **file1** as filename)	

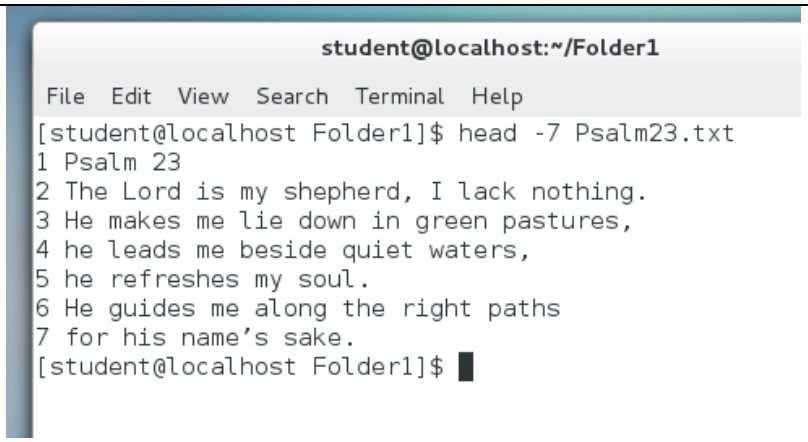
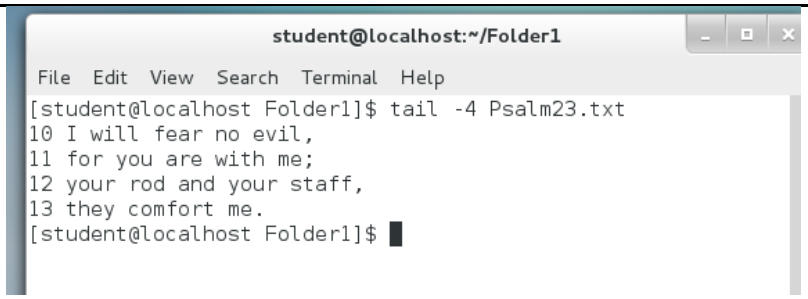
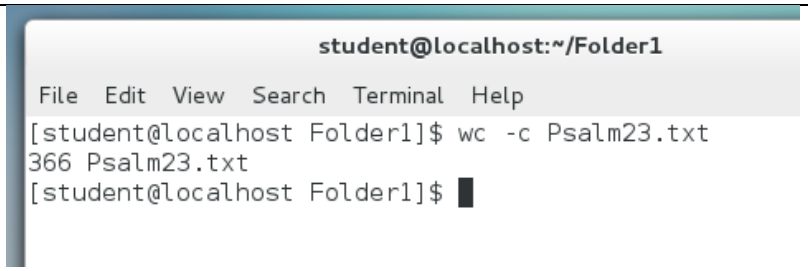
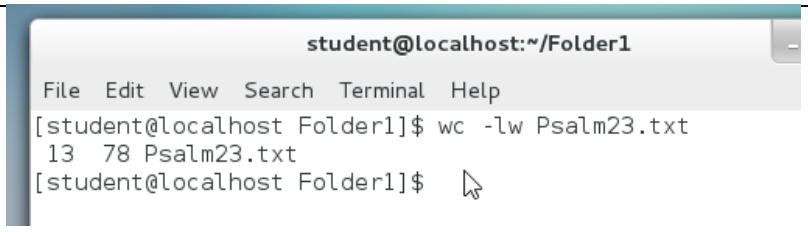
	
2. To display the content of file1	
3. Create a directory named Folder1 at your username directory. List the content of **Folder1** directory.	
4. Create a new file named Psalm23.txt using the vi editor and write in verbatim of the ff. including the **numbering**: Psalm 23 The Lord is my shepherd, I lack nothing. He makes me lie down in green pastures, he leads me beside quiet waters, he refreshes my soul. He guides me along the right paths for his name's sake. Even though I walk	

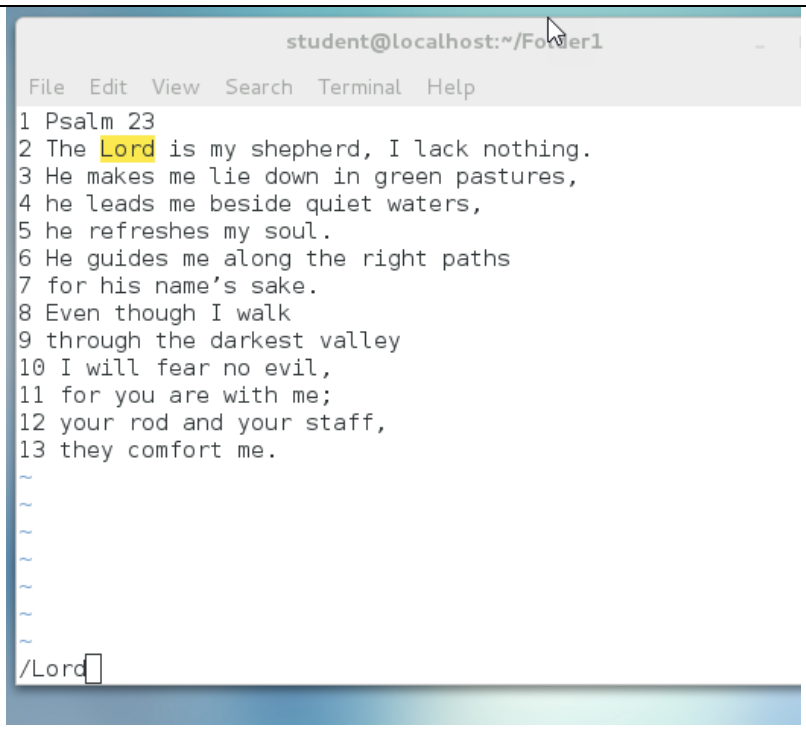
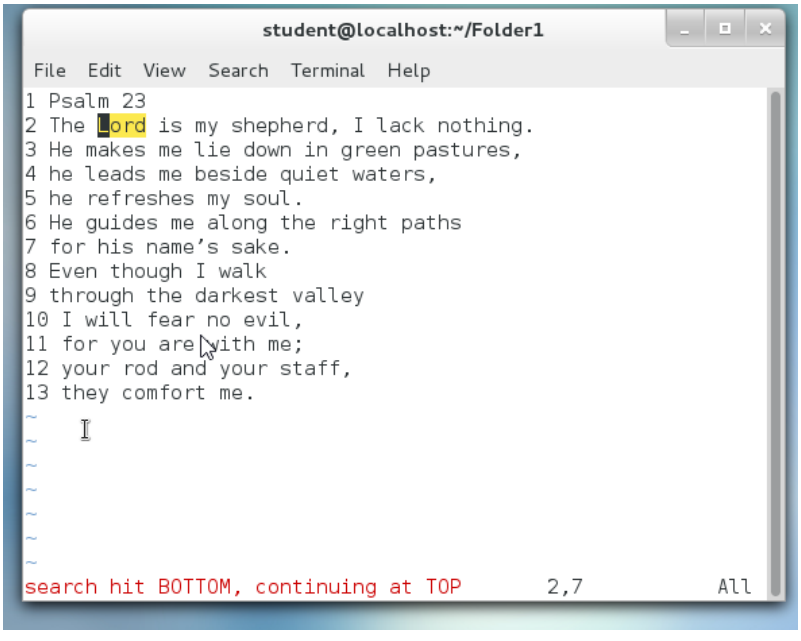
through the darkest valley
I will fear no evil,
for you are with me;
your rod and your staff,
they comfort me.

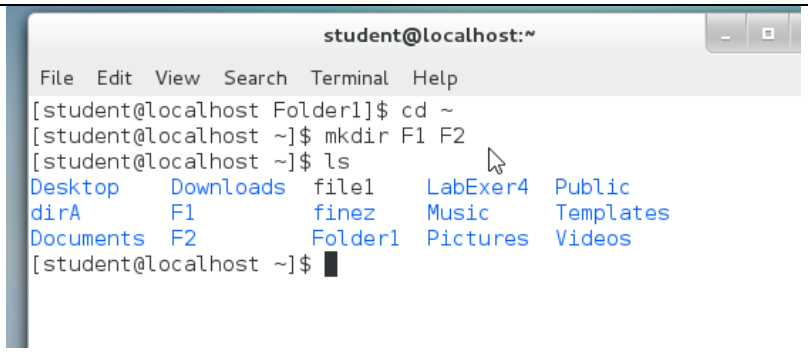
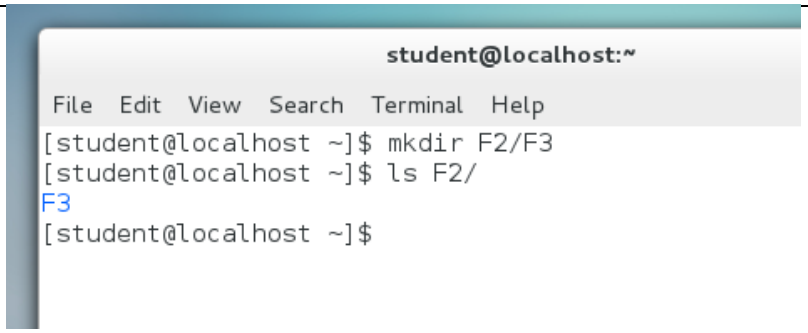


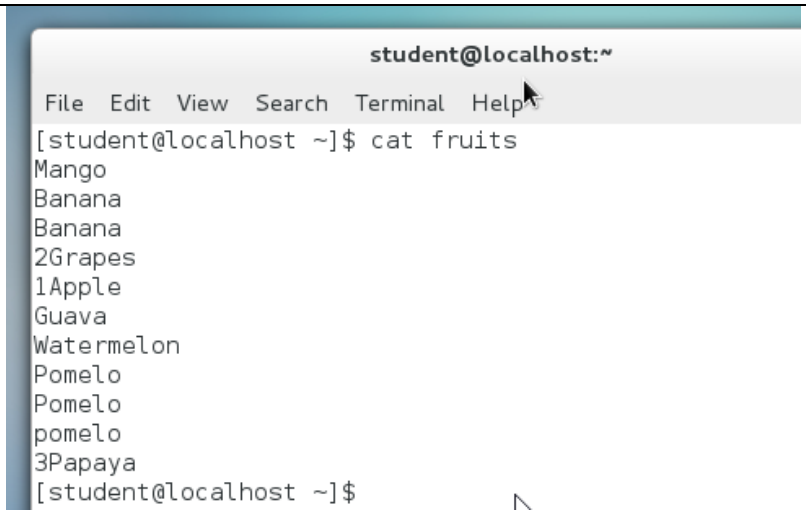
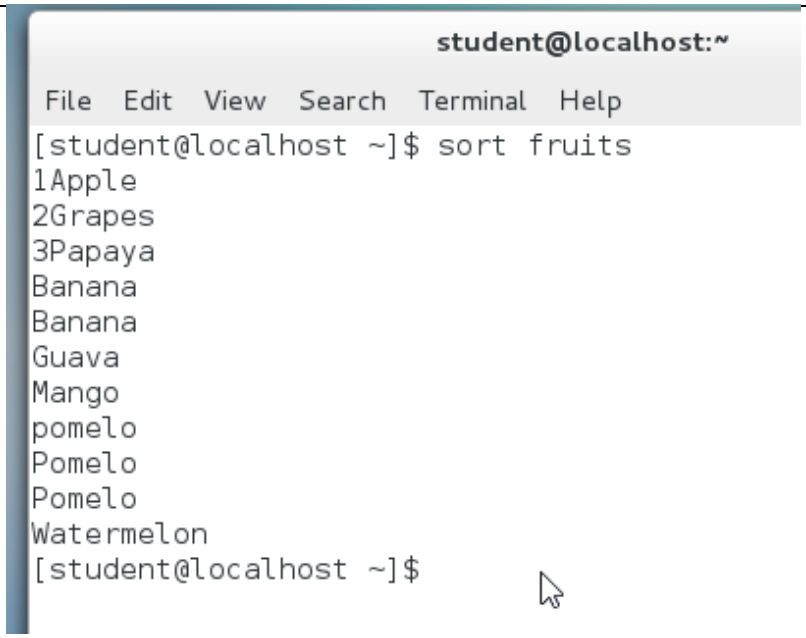
5. Display the content of the file **Psalm23.txt** using cat command

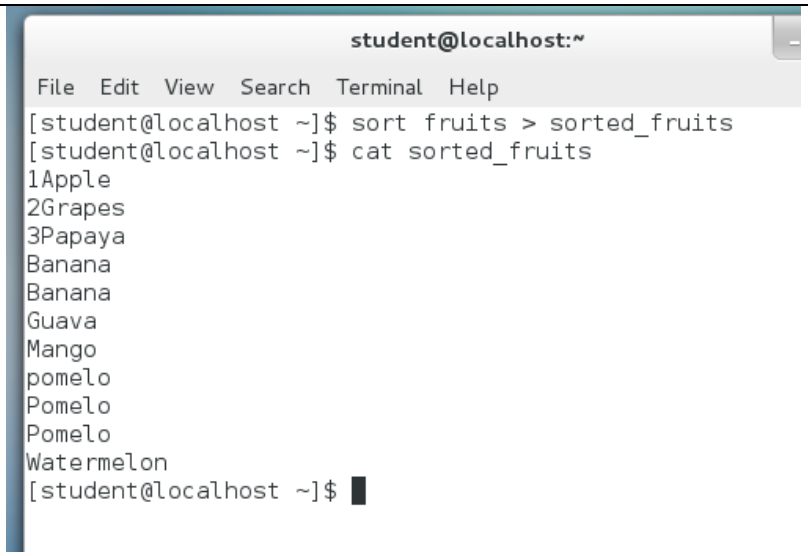
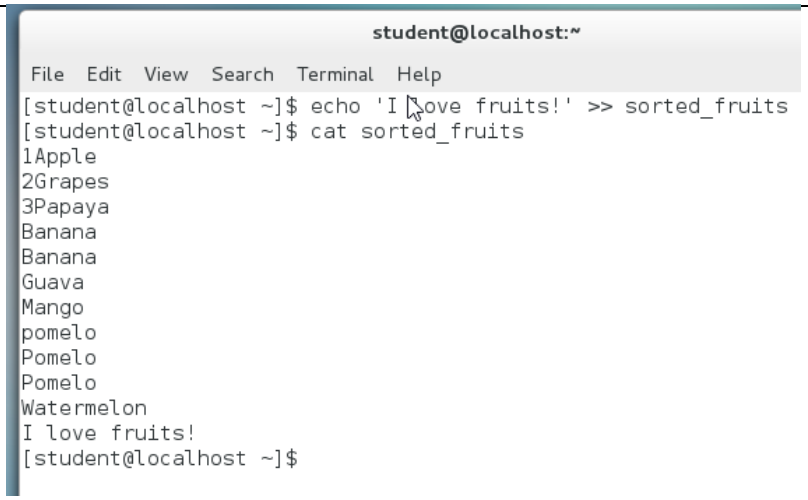
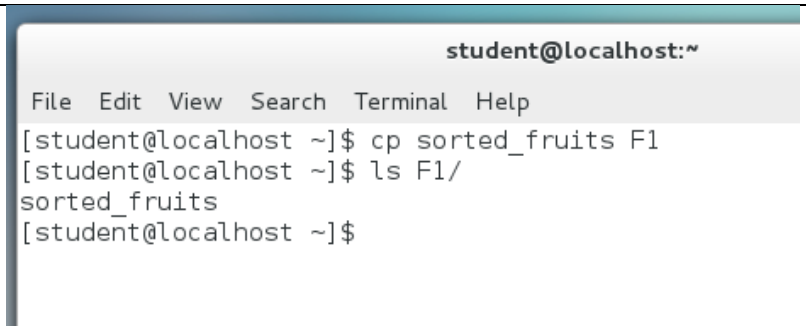


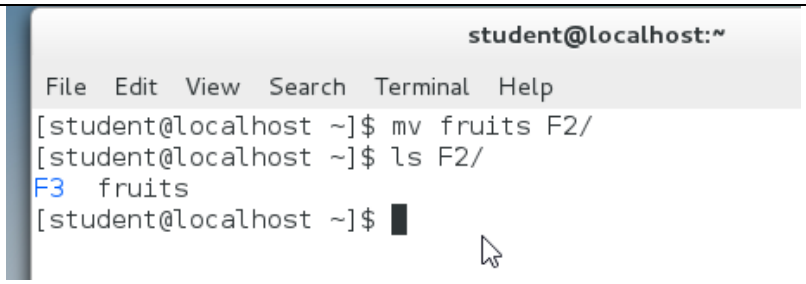
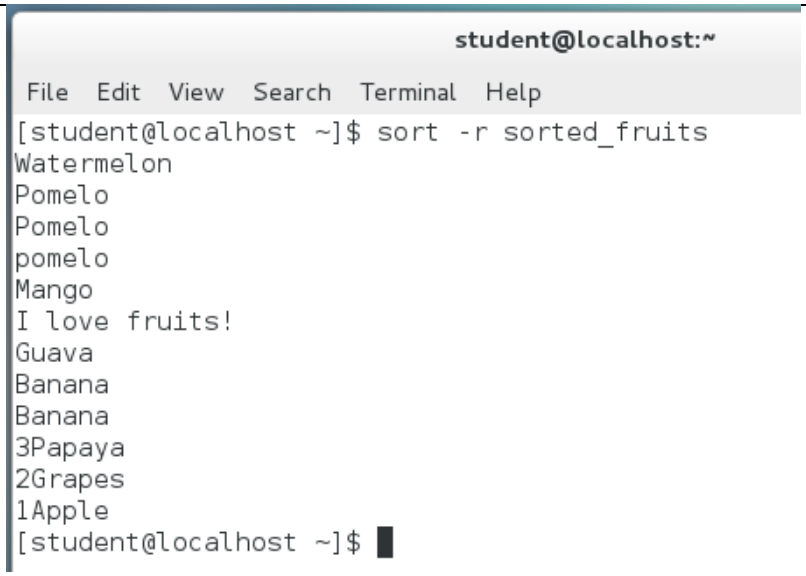
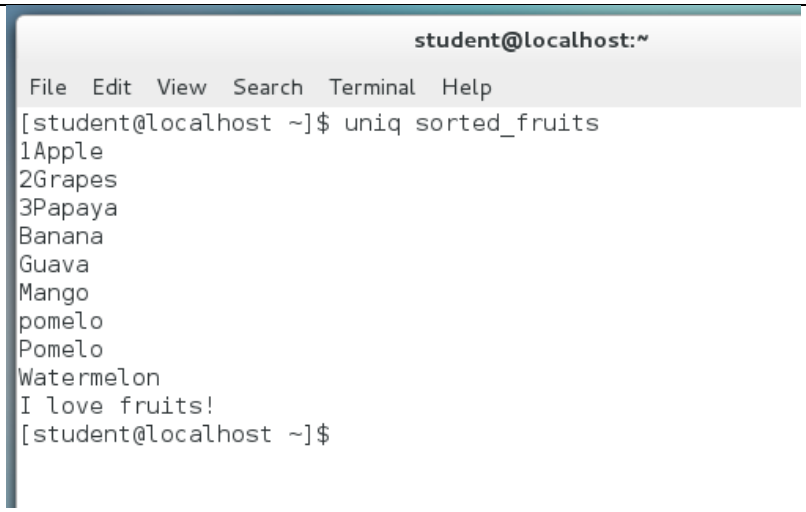
6. Display the first 7 lines of the **Psalm23.txt**	
7. Display the last 4 lines of the **Psalm23.txt**	
8. Count the number of characters in file **Psalm23.txt**	
9. In just one line of command, count the number of lines and words are there in **Psalm23.txt**	

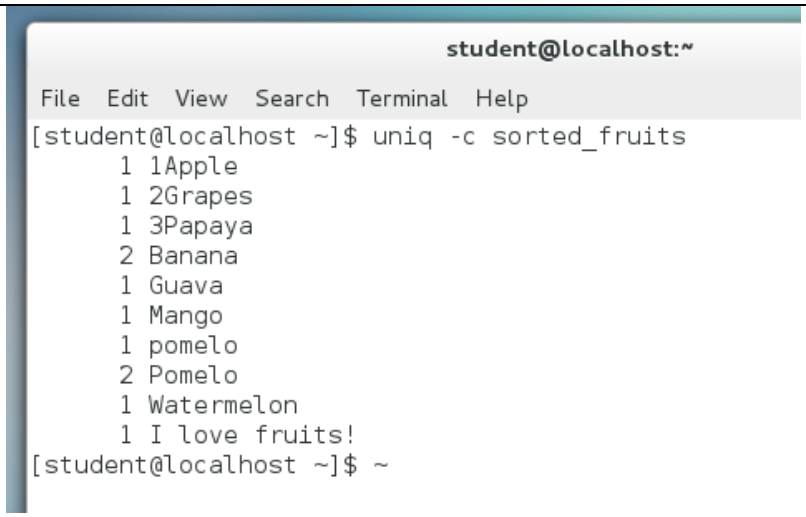
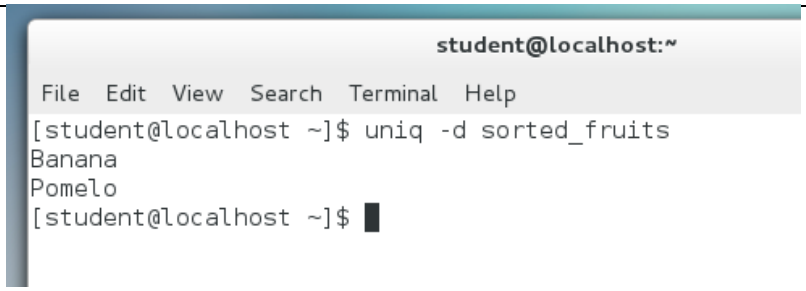
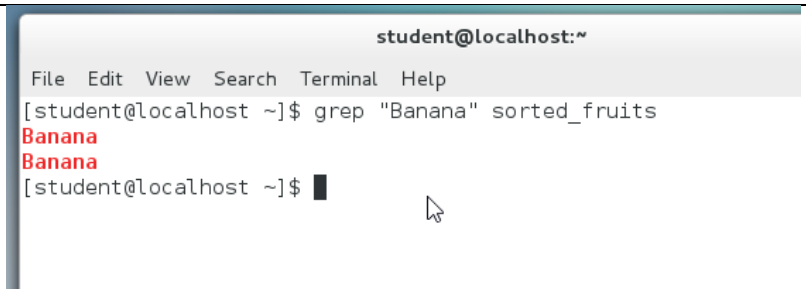
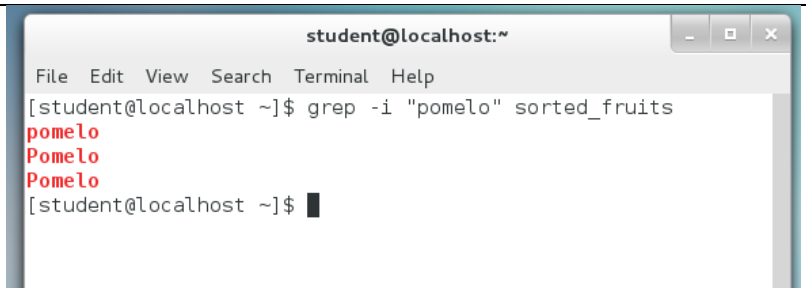
10. Using vi editor, open your file **Psalm23.txt** and search for the word **Lord** in the file content.	 
PART 4: File Operation Commands / Redirection Standard Input/Output / Data Refinement Commands	

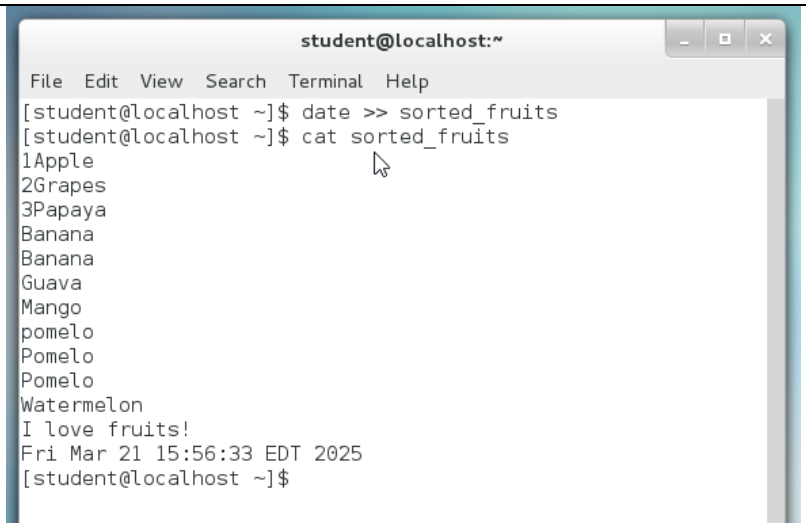
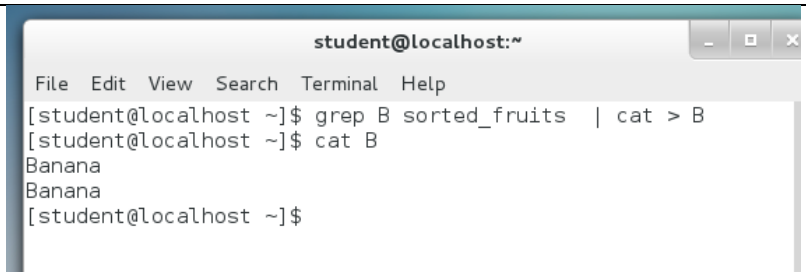
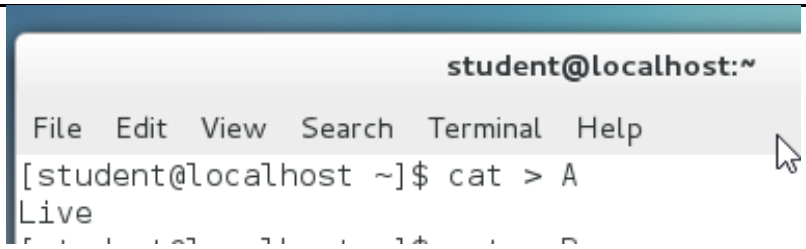
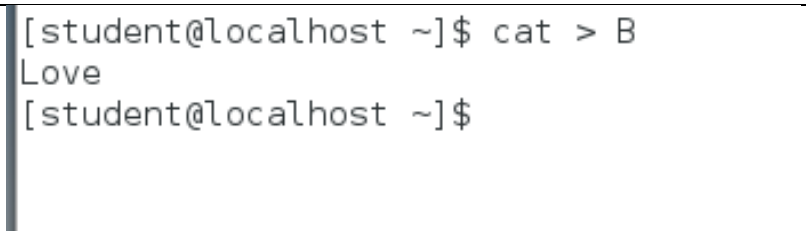
11. Create two directories in your home directory using the following folder names: • F1 • F2	
12. Under F2 folder, create another subdirectory and named it as F3	
13. Create a new file named fruits using cat command and write the following texts below: Mango Banana Banana 2Grapes 1Apple Guava Watermelon Pomelo Pomelo pomelo 3Papaya	

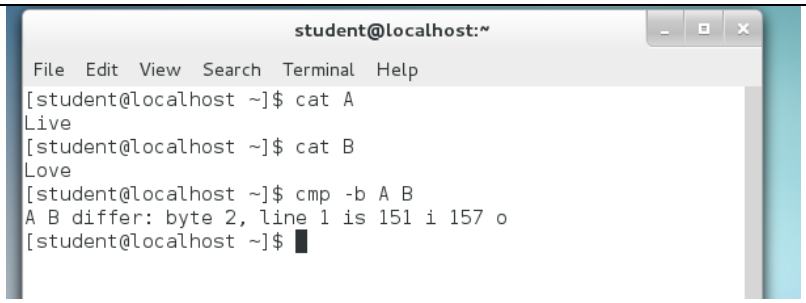
14. Display the content of the file ***fruits*** using cat command.	
15. Type ***sort fruits***, what is the output?	

16. Type ***sort fruits > sorted_fruits*** then type ***cat sorted_fruits***. What is the output?	
17. Type ***echo "I love fruits!" >> sorted_fruits*** then type ***cat sorted_fruits***. What is your observation the output?	
18. Copy the file ***sorted_fruits*** to **F1**. Display the content of **F1**.	

19. Move the file ***fruits*** to **F2**. Display the content of **F2**	
20. Sort the file ***sorted_fruits*** in reverse order.	
21. Filter the repeated lines in file ***sorted_fruits***	

22. Count the number of occurrences of each word in file ***sorted_fruits***	
23. Display all the duplicate lines in file ***sorted_fruits***	
24. Using grep command, search for the word Banana in the file ***sorted_fruits***	
25. Using grep command, search for the word "pomelo" or "Pomelo" in the file ***sorted_fruits*** regardless of the casing of letters	

26. Using redirector, append the system date to the file ***sorted_fruits***	
27. Type ***grep B sorted_fruits | cat > B***. What is your observation in the output?	 - **File named "B" is created with lines that start with "B" which is Banana**
28. Using cat command, create a filename ***A*** with a string content of **LIVE**.	
29. Using cat command, create a filename ***B*** with a string content of **LOVE**.	

30. Determine the difference between files ***A*** and ***B*** in terms of bytes.	

9. Calculations (if applicable)

10. Observations and Conclusion/s (if applicable)

Through this activity, I gained hands-on experience with essential Linux commands, improving my file management, navigation, and system administration skills. I practiced using cd, cp, mv, rm, and touch, explored redirection and pipes for efficient data handling, and mastered vi for text editing. Managing file permissions that enhanced my understanding of system security. This experience boosted my confidence in working with Linux and reinforced my ability to navigate and manipulate the file system effectively.

11. Rating (include Rubric)

Criteria	Grade
Activity Conduct (1-5)	
Correctness of Command(s)/Program(s) (1-5) x 2	
Completeness of Tasks (1-5)	
Data Analysis and Results Interpretation (1-5)	
Total Score	

Mean Score = (Total Score / 5)	
Percentage Score = (Total Score/25) * 100	
Other Comments:	