Assignment 2

Part A

What will the following commands do?

1. echo "Hello, World!" ➔ It will print Hello World
2. name="Productive" ➔ It will assign name variable as Productive
3. touch file.txt ➔ For creating text file named file.txt
4. ls -a ➔ **ls**: The basic command used to list the files and directories in the current directory. -a includes the hidden files in the testing
5. rm file.txt ➔ to remove the file.txt file
6. cp file1.txt file2.txt ➔ copy the content of file1.txt to new file2.txt
7. mv file.txt /path/to/directory/ ➔ This will move file.txt from current directory to the directory of which path is given
8. chmod 755 script.sh ➔ chmod command is used to change the file permissions. [755 is permission for setting represented in octal form. {7 for owner➔ owner gets read('r'), write('w') & execute permission} {5 for Group ➔ Members of file's group get read and execute permissions but not write} {5 for other ➔ all other user get read and execute permissions}
9. grep "pattern" file.txt ➔ grep command is used for searching text patterns. This command will search for pattern word in file and five output as lines containing pattern word.
10. kill PID ➔ kill used to signal the process. PID is the process ID u want to terminate. ps aux | grep process_name is command to find Process name
11. mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt➔ It will create directory mydir then will create file.txt with hello world and by cat hello world will be displayed
12. ls -l | grep ".txt" ➔ The command ls -l | grep ".txt" lists files in the current directory and filters the results to show only those files that have .txt in their names.
13. cat file1.txt file2.txt | sort | uniq ➔ this command combines the contents of file1.txt and file2.txt, sorts them, and then removes any duplicate lines, showing only the unique lines from the combined files.
14. ls -l | grep "^d" ➔ • `ls -l`: Lists files and directories with detailed information.`grep "^d"`: Filters the input to show only lines where the first

character is `d`. In the output of `ls -l`, `d` at the beginning of a line indicates a directory.

15. • grep -r "pattern" /path/to/directory/ ➔ This command will search for the specified pattern in all files within /path/to/directory/ and its subdirectories, displaying the lines that contain the pattern along with their filenames.
16. chmod 644 file.txt ➔ makes file.txt readable and writable by the file's owner, and readable by everyone else.
17. cp -r source_directory destination_directory ➔ copies the contents of source_directory to destination_directory, including all subdirectories and files.
18. find /path/to/search -name "*.txt" ➔ When you run this command, it will output the paths of all .txt files found within /path/to/search and its subdirectories.
19. chmod u+x file.txt➔ chmod u+x file.txt adds execute permission to file.txt for the file's owner. After running this command, the owner of the file will be able to execute it as a program or script
20. • echo $PATH ➔ displays the current value of the PATH environment variable.

Identify True or False:

1. ls is used to list files and directories in a directory.==> True

2. mv is used to move files and directories. ➔ True

3. cd is used to copy files and directories. ➔ False

4. pwd stands for "print working directory" and displays the current directory. ➔

5. grep is used to search for patterns in files. ➔ True

6. chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and execute permissions to group and others. ➔ True

7. mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist. ➔ True

8. rm -rf file.txt deletes a file forcefully without confirmation. ➔ True

Identify the Incorrect Commands:

1. chmodx is used to change file permissions. ➔ correct command is chmod
2. cpy is used to copy files and directories. ➔ correct command is cp
3. mkfile is used to create a new file. ==> correct command is touch
4. catx is used to concatenate files .==> correct command is cat
5. rn is used to rename files. ➔ correct command is mv

Part C

Question 1: Write a shell script that prints "Hello, World!" to the terminal.

⇨ #!/bin/bash

echo “Hello world”

Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

```
name="CDAC Mumbai"
echo "$name"
```

Question 3: Write a shell script that takes a number as input from the user and prints it.

```
X = 100
echo $X
```

Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.

```
#!/bin/bash
a=5
b=3
echo $((a+b))
```

Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd"

```
#!/bin/bash
echo "Enter a number:"
read num if [ $((num % 2)) -eq 0 ]; then
echo "$num is even"
else
echo "$num is odd"
fi
```

Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.

```
#!/bin/bash
a=0
for a in  1 2 3 4 5
do
      echo "Numbers are : $a"
done
```

Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.

```
a=0
while [ $a -lt 5 ]
do
echo $a
a=$(expr $a + 1)
done
```

Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".

```
#!/bin/bash
if [ -f "file.txt" ]; then
echo "File exists"
else
echo "File does not exist"
fi
```

Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.

```
#!/bin/bash
read a
if [ $a -gt 10 ] ; then
echo "Number is greater"
else
echo "Number is less"
fi
```

Question 10: Write a shell script that uses nested for loops to print a multiplication table for numbers from 1 to 5. The output should be formatted nicely, with each row representing a number and each column representing the multiplication result for that number.

```
for (( i=1; i<=5 ; i++ ))
do
    for (( j=1 ; j<=10 ; j++ ))
do
    echo $i*$j = $(($i*$j))
done
done
```

Question 11: Write a shell script that uses a while loop to read numbers from the user until the user enters a negative number. For each positive number entered, print its square. Use the break statement to exit the loop when a negative number is entered.

```
#!/bin/bash
while true; do
    read num
    if [ $num -lt 0 ]; then
    break
    fi

    if [ $num -ge 0 ]; then
    square=$((num * num))
    echo "The square of $num is $square"

    fi
done
```