**Part A**

**What will the following commands do?**

1. echo "Hello, World!"

→ Hello, World

1. name="Productive"

→ Productive string gets stored in name variable

1. touch file.txt

→ Create file name of touch.file.txt

1. ls -a

→ It will list all the files including hidden files

1. rm file.txt

→ It will remove file name of file.txt

1. cp file1.txt file2.txt

→ It will copy the content of file1.txt to file2.txt

1. mv file.txt /path/to/directory/

→ It will move file.txt to destination

1. chmod 755 script.sh

→ 7 : rwx 5 : r-x 5 : r-x so, it means chmod 755 command gives the owner full permissions

1. grep "pattern" file.txt

→ search for the literal string “pattern” in the file named file.txt

1. kill PID

→ Used to stop a process using its PID

1. mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt

→ 1. creates a new directory named mydir

2. moves into directory

3. creates file named file.txt

4. Write “Hello world” into file.txt

5. Display the contents of file.txt

1. ls -l | grep ".txt"

→ 1.lists files int the current directory long format

2. filter the list, showing only lines that contain “.txt”

1. cat file1.txt file2.txt | sort | uniq

→ 1.concatenates the contents of file1.txt and file2.txt and display file

2.Sorting the lines alphabetically

3.Removes duplicate adjacent lines

1. ls -l | grep "^d"

→ 1.lists files int the current directory long format

2.filters for line beginning with the letter d

1. grep -r "pattern" /path/to/directory/
2. cat file1.txt file2.txt | sort | uniq –d

→ 1.concatenates the contents of file1.txt and file2.txt and display file

2.Sorting the lines alphabetically

3.prints only the lines that are duplicated

1. chmod 644 file.txt

→ 6 : rw- 4 : r-- 4 : r--

1. cp -r source\_directory destination\_directory

→ coping content from source to destination file

1. find /path/to/search -name "\*.txt"

→ filter the list, showing only lines that contain “.txt”

1. chmod u+x file.txt

→ adding owner execution permission

1. echo $PATH

→ View current path

**Part B**

**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

cd : change directores

cp : copying

4. **pwd** stands for "print working directory" and displays the current directory. → true

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. →incorrect

2. cpy is used to copy files and directories. → incorrect

3. mkfile is used to create a new file. → incorrect

4. catx is used to concatenate files. → incorrect

5. rn is used to rename files. → incorrect (we use mv to rename)

**Part C**

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

→ 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.

→ echo “enter number”

read name

echo “your number is $name”

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

→ echo “enter first number”

read var1

echo “enter second number”

read var2

(( result = var1+var2 ))

echo “your result is $result”

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

→ echo “enter number”

read num

if [[ ( $num -gt 0) && ( $num%2 -eq 0) ]]; then

echo "even number"

else

echo "odd number"

fi

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

→ for var1 in {1..5}

do

echo "$var1"

done

done

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

→ i=1

while [ $i -le 5 ]

do

echo "$i"

((i++))

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".

→ 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.

→ echo “enter number”

read num

if [[ $num -gt 10 ]]; then

echo "number is $num"

else

echo "number less than 10"

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 in {1..5}

do

for j in {1..5}

do

echo -n "$((i \* j)) "

done

echo

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.

→ while true

do

read num

[ "$num" -lt 0 ] && break

echo $((num \* num))

done

**Part E**

1. Average WT = 3.33
2. Average TAT = 5.5
3. Average WT = 5.5
4. Average TAT =8.75

Consider a program that uses the fork() system call to create a child process. Initially, the parent process has a variable x with a value of 5. After forking, both the parent and child processes increment the value of x by 1. What will be the final values of x in the parent and child processes after the fork() call?

→