CDAC MUMBAI

Concepts of Operating System

Assignment 2

Part A

What will the following commands do?

- echo "Hello, World!"
  This command prints the text "Hello, World!" to the terminal.
- name="Productive"
  This assigns the value "Productive" to a variable named name. You can reference it later using $name.
- touch file.txt
  Creates an empty file named file.txt if it doesn't already exist. If it does exist, touch updates the file's last modified timestamp.
- ls -a
  Lists all files and directories in the current directory, including hidden ones
- rm file.txt
  Deletes the file named file.txt.
- cp file1.txt file2.txt
  Copies the contents of file1.txt to file2.txt. If file2.txt doesn't exist, it's created.
- mv file.txt /path/to/directory/
  Moves file.txt to the specified directory. This can also be used to rename files.
- chmod 755 script.sh
  Changes the permissions of the file script.sh to 755, which means:
  Owner: read, write, and execute (7).
  Group: read and execute (5).
  Others: read and execute (5).
- grep "pattern" file.txt
  Searches for the string "pattern" in file.txt and prints any matching lines.
- kill PID
  Sends a termination signal to the process with the specified Process ID (PID), effectively stopping it.
- mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt
  Creates a directory named mydir.
  Changes into the mydir directory.
  Creates an empty file named file.txt.
  Writes "Hello, World!" into file.txt.
  Displays the contents of file.txt to the terminal.
- ls -l | grep ".txt
  Lists all files and directories in long format and filters the list to show only items with .txt in their names.
- grep -r "pattern" /path/to/directory/
  Recursively searches for the string "pattern" in all files within /path/to/directory/ and its subdirectories.
- cat file1.txt file2.txt | sort | uniq –d

Concatenates the contents of file1.txt and file2.txt.
Sorts the combined contents.
Displays only the lines that are repeated (duplicates).

- chmod 644 file.txt
  Changes the permissions of file.txt to 644, which means:
  Owner: read and write (6).
  Group: read-only (4).
  Others: read-only (4).

- cp -r source_directory destination_directory
  Recursively copies the contents of source_directory to destination_directory. If destination_directory doesn't exist, it's created.

- find /path/to/search -name "*.txt"
  Searches for all files with a .txt extension within /path/to/search and its subdirectories

- chmod u+x file.txt
  Adds execute permission to the file file.txt for the owner (u stands for user).

- echo $PATH
  Displays the current PATH environment variable, which lists the directories the shell searches to find executable files.

Part B

True or False Statements:

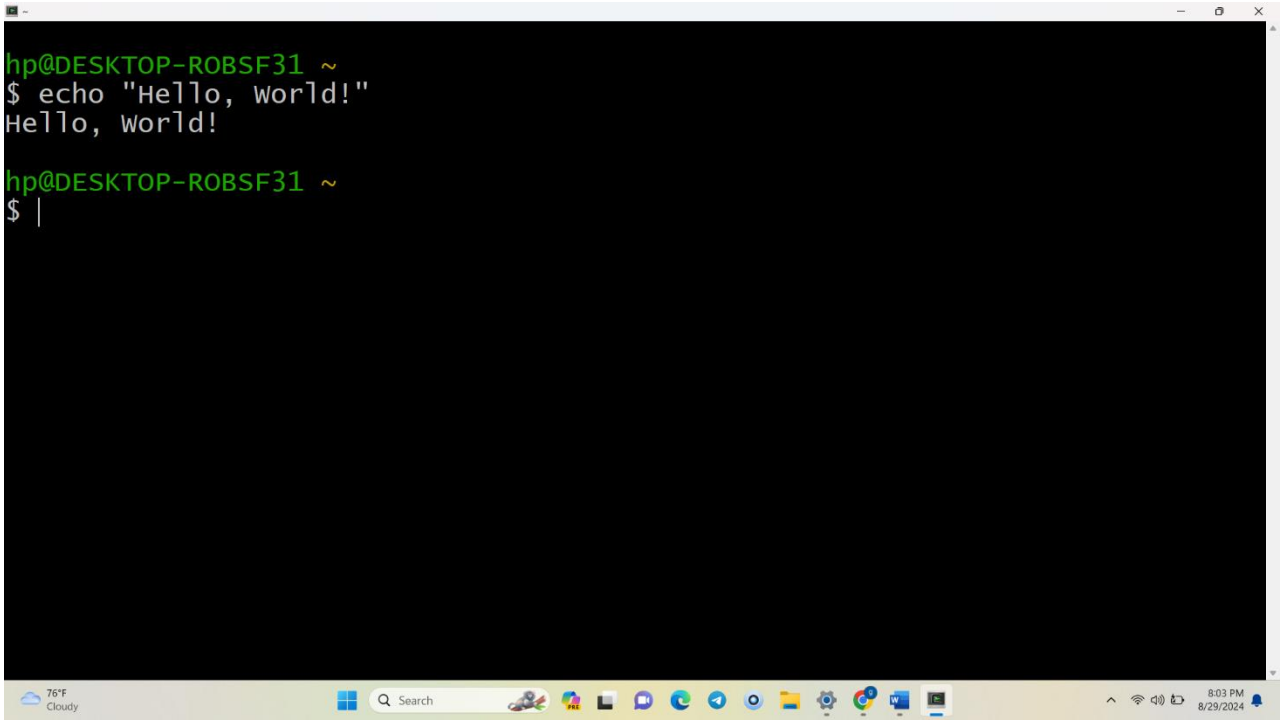
1. True: ls is used to list files and directories in a directory.
2. True: mv is used to move files and directories.
3. False: cd is used to change the current directory, not to copy files and directories.
4. True: pwd stands for "print working directory" and displays the current directory.
5. True: grep is used to search for patterns in files.
6. True: chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.
7. True: mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist.
8. True: rm -rf file.txt deletes a file forcefully without confirmation.

Identify the Incorrect Commands:

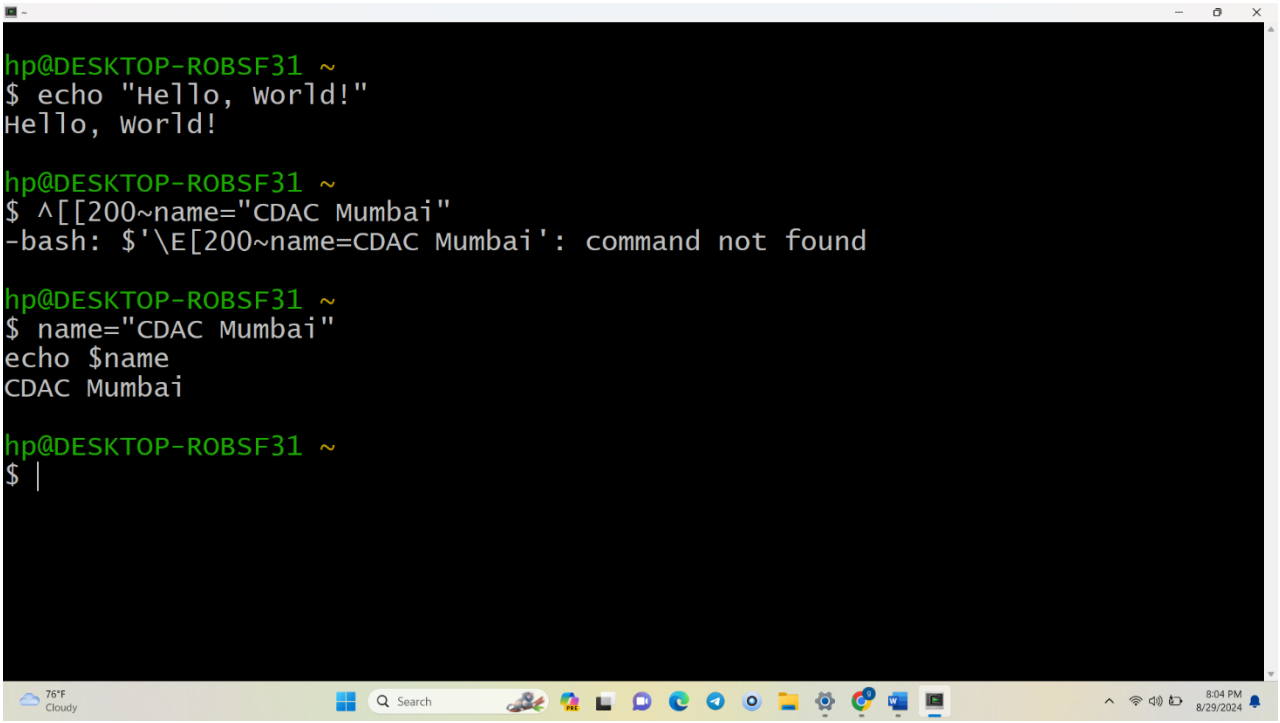
1. **chmodx** is incorrect. The correct command to change file permissions is chmod.
2. **cpy** is incorrect. The correct command to copy files and directories is cp.
3. **mkfile** is incorrect. The correct way to create a new file is using touch or echo "content" > filename.
4. **catx** is incorrect. The correct command to concatenate files is cat.
5. **rn** is incorrect. The correct command to rename files is mv (or rename in some systems).

Part C

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



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



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

```
$ echo "Hello, World!"
Hello, World!

hp@DESKTOP-ROBSF31 ~
$ ^[[200~name="CDAC Mumbai"
-bash: $'\E[200~name=CDAC Mumbai': command not found

hp@DESKTOP-ROBSF31 ~
$ name="CDAC Mumbai"
echo $name
CDAC Mumbai

hp@DESKTOP-ROBSF31 ~
$ echo "Enter a number:"
read number
echo "You entered: $number"
Enter a number:
3
You entered: 3

hp@DESKTOP-ROBSF31 ~
$ |
```

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

```
hp@DESKTOP-ROBSF31 ~
$ num1=5
num2=3
sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is: $sum"
The sum of 5 and 3 is: 8

hp@DESKTOP-ROBSF31 ~
$
```

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

```
hp@DESKTOP-ROBSF31 ~
$ num1=5
num2=3
sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is: $sum"
The sum of 5 and 3 is: 8

hp@DESKTOP-ROBSF31 ~
$ echo "Enter a number:"
read number
if [ $((number % 2)) -eq 0 ]; then
    echo "Even"
else
    echo "Odd"
fi
Enter a number:
2
Even

hp@DESKTOP-ROBSF31 ~
$
```

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

```
hp@DESKTOP-ROBSF31 ~
$ for i in {1..5}; do
    echo $i
done
1
2
3
4
5

hp@DESKTOP-ROBSF31 ~
$ |
```

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

```
    echo $i
done
1
2
3
4
5

hp@DESKTOP-ROBSF31 ~
$ i=1
while [ $i -le 5 ]; do
    echo $i
    i=$((i + 1))
done
1
2
3
4
5

hp@DESKTOP-ROBSF31 ~
$ |
```

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".

```
hp@DESKTOP-ROBSF31 ~
$ #!/bin/bash
if [ -f "file.txt" ]; then
    echo "File exists"
else
    echo "File does not exist"
fi
File does not exist

hp@DESKTOP-ROBSF31 ~
$
```

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.

```
hp@DESKTOP-ROBSF31 ~
$ echo "Enter a number:"
read number

# Check if the number is greater than 10
if [ "$number" -gt 10 ]; then
    echo "The number is greater than 10."
else
    echo "The number is 10 or less."
fi
Enter a number:
2
The number is 10 or less.

hp@DESKTOP-ROBSF31 ~
$ |
```

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.

```
    echo "The number is 10 or less"
fi
Enter a number:
5
The number is 10 or less

hp@DESKTOP-ROBSF31 ~
$ #!/bin/bash
for i in {1..5}; do
    for j in {1..5}; do
        printf "%4d" $((i * j))
    done
    echo ""
done
   1   2   3   4   5
   2   4   6   8  10
   3   6   9  12  15
   4   8  12  16  20
   5  10  15  20  25

hp@DESKTOP-ROBSF31 ~
$
```

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.

```
hp@DESKTOP-ROBSF31 ~
$ #!/bin/bash
while true; do
    echo "Enter a number:"
    read number
    if [ $number -lt 0 ]; then
        break
    fi
    echo "The square of $number is $((number * number))"
done
Enter a number:
7
The square of 7 is 49
Enter a number:
4
The square of 4 is 16
Enter a number:
77
The square of 77 is 5929
Enter a number:
0
```