CDAC MUMBAI

Concepts of Operating System

ASSIGNMENT - PART A

What will the following commands do?

- **echo "Hello, World!"**

→ Prints Hello, World! to the terminal.

- **name="Productive"**

→ Assigns the string "Productive" to the variable name in the shell.

- **touch file.txt**

→ Creates an empty file named file.txt (or updates its timestamp if it already exists).

- **ls -a**

→ Lists all files and directories, including hidden ones (those starting with .)

- **rm file.txt**

→ It is use to deletes file.txt permanently.

- **cp file1.txt file2.txt**

→ Copies file1.txt to file2.txt

- **mv file.txt /path/to/directory/**

→ mv command is used rename or move a file.

Moves file.txt to the specified directory.

- **chmod 755 script.sh**

→ Sets file permissions for script.sh to rwxr-xr-x (owner can read,write, execute; others can read and execute).

- **grep "pattern" file.txt**

→ grep command is used to search for specific patterns or

regular expressions in text files & display the matching lines.

Searches for the word "pattern" inside file.txt and displays matching lines.

- **kill PID**

→ Terminates the process with the given PID (Process ID).

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

→ Creates mydir, navigates into it, creates file.txt, writes "Hello, World!" inside it, and then displays the contents.

- **ls -l | grep ".txt"**

→ Lists all files in long format (-l) and filters only txt files.

- **cat file1.txt file2.txt | sort | uniq**

→ Concatenates file1.txt and file2.txt, sorts the lines, and removes duplicates.

- **ls -l | grep "^d"**

→ Lists only directories (^d indicates directory in ls -l output).

- **grep -r "pattern" /path/to/directory/**

→ Recursively searches for "pattern" inside all files

under /path/to/directory/.

- **cat file1.txt file2.txt | sort | uniq –d**

→ Shows only duplicate lines from file1.txt and file2.txt after sorting.

- **chmod 644 file.txt**

→ Sets file permissions to rw-r--r-- (owner can read and write, others can only read).

- **cp -r source_directory destination_directory**

→ Recursively copies source_directory to destination_directory.

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

→ Finds all txt files in /path/to/search and its subdirectories.

- **chmod u+x file.txt**

→ Gives the owner (u) execute (x) permission for file.txt.

- **echo $PATH**

→ Displays the system's PATH variable, which contains directories where executable files are stored.

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**

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. : **chmod**

2. cpy is used to copy files and directories : **cp**

3. mkfile is used to create a new file. **: using touch create new empty file, using mkdir create new directory**

4. catx is used to concatenate files. : **cat**

5. rn is used to rename files. **using mv command we can rename the file & move to another file**

Part C

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

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q1.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q1.sh
echo "Hello, World!"
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q1.sh
Hello, World!
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q2.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q2.sh
name="CDAC Mumbai"
echo $name
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q2.sh
CDAC Mumbai
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q3.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q3.sh
echo Enter a num1
read Num1
echo Enter a num2
read Num2
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q3.sh
Enter a num1
45
Enter a num2
78
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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

```
ohini@LAPTOP-U0E18KK2:~/myScripts$ nano q4.sh
ohini@LAPTOP-U0E18KK2:~/myScripts$ cat q4.sh
echo Enter a Number
read num1
echo Enter number2
read num2

sum=$((num1+num2))

echo sum of $num1 and $num2 is $sum

ohini@LAPTOP-U0E18KK2:~/myScripts$ bash q4.sh
Enter a Number
3
Enter number2
5
sum of 3 and 5 is 8
```

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

```
ohini@LAPTOP-U0E18KK2:~/myScripts$ nano test.sh
ohini@LAPTOP-U0E18KK2:~/myScripts$ cat test.sh
echo Enter a number
read num

if [ $((num % 2)) -eq 0 ]; then
    echo "$num is an even number"
else
    echo "$num is an odd number"
fi

ohini@LAPTOP-U0E18KK2:~/myScripts$ bash test.sh
Enter a number
20
20 is an even number
```

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

```
ohini@LAPTOP-U0E18KK2:~/myScripts$ nano q6.sh
ohini@LAPTOP-U0E18KK2:~/myScripts$ cat q6.sh
for i in {1..5}
do
     echo $i
done
ohini@LAPTOP-U0E18KK2:~/myScripts$ bash q6.sh
1
2
3
4
5
ohini@LAPTOP-U0E18KK2:~/myScripts$
```

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

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q7.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q7.sh
num=1
while [ $num -le 5 ]
do
    echo $num
    ((num++))
done
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q7.sh
1
2
3
4
5
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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".

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q8.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q8.sh
if [ -e "file.txt" ]; then
    echo "File exists"
else
    echo "File does not exist"
fi
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q8.sh
File does not exist
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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.

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q9.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q9.sh
echo "Enter a number:"
read num

if [ $num -gt 10 ]; then
    echo "The number is greater than 10."
else
    echo "The number is 10 or less."
fi
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q9.sh
Enter a number:
23
The number is greater than 10.
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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.

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q10.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q10.sh
#!/bin/bash

# Printing the header
echo "Multiplication Table (1 to 5)"
echo "-----------------------------"

# Outer loop for rows (numbers 1 to 5)
for ((i=1; i<=5; i++))
do
    # Inner loop for columns (multiplication results)
    for ((j=1; j<=5; j++))
    do
        # Printing the multiplication result with proper spacing
        printf "%4d" $((i * j))
    done
    echo  # Newline after each row
done

ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q10.sh
Multiplication Table (1 to 5)
-----------------------------
   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
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

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.

```
ohini@LAPTOP-U0E18KK2:~/myPrograms$ nano q11.sh
ohini@LAPTOP-U0E18KK2:~/myPrograms$ cat q11.sh
#!/bin/bash

# Prompting user for input
echo "Enter numbers to calculate their square (enter a negative number to exit):"

while true  # Infinite loop, will break when a negative number is entered
do
    # Read user input
    read -p "Enter a number: " num

    # Check if the number is negative
    if [ "$num" -lt 0 ]; then
        echo "Negative number entered. Exiting..."
        break  # Exit the loop
    fi

    # Calculate and display the square
    square=$((num * num))
    echo "Square of $num is: $square"
done
ohini@LAPTOP-U0E18KK2:~/myPrograms$ bash q11.sh
Enter numbers to calculate their square (enter a negative number to exit):
Enter a number: 3
Square of 3 is: 9
Enter a number: 5
Square of 5 is: 25
Enter a number: 6
Square of 6 is: 36
Enter a number: -1
Negative number entered. Exiting...
ohini@LAPTOP-U0E18KK2:~/myPrograms$
```

Part E