CDAC MUMBAI

**Concepts of Operating System Assignment 2**

**Part A**

**What will the following commands do?**

1. `echo "Hello, World!"`: Prints "Hello, World!" to the terminal.

2. `name="Productive"`: Sets the variable `name` to "Productive".

3. `touch file.txt`: Creates or updates `file.txt`.

4. `ls -a`: Lists all files, including hidden ones.

5. `rm file.txt`: Deletes `file.txt`.

6. `cp file1.txt file2.txt`: Copies `file1.txt` to `file2.txt`.

7. `mv file.txt /path/to/directory/`: Moves `file.txt` to a directory.

8. `chmod 755 script.sh`: Makes `script.sh` executable.

9. `grep "pattern" file.txt`: Searches for "pattern" in `file.txt`.

10. `kill PID`: Terminates the process with `PID`.

11. `mkdir mydir && cd mydir && ...`: Creates `mydir`, navigates into it, creates `file.txt`, writes to it, and displays content.

12. `ls -l | grep ".txt"`: Lists `.txt` files.

13. `cat file1.txt file2.txt | sort | uniq`: Merges, sorts, and removes duplicates.

14. `ls -l | grep "^d"`: Lists directories only.

15. `grep -r "pattern" /path/to/directory/`: Recursively searches for "pattern".

16. `cat ... | sort | uniq -d`: Shows duplicate lines from sorted input.

17. `chmod 644 file.txt`: Sets read/write for owner, read-only for others.

18. `cp -r source\_directory destination\_directory`: Copies directories recursively.

19. `find /path/to/search -name "\*.txt"`: Finds `.txt` files.

20. `chmod u+x file.txt`: Makes `file.txt` executable for the user.

21. `echo $PATH`: Displays system `PATH` variable.

# **Part B**

**Identify True or False:**

**1. `ls` is used to list files and directories in a directory.**

**True - `ls` lists files and directories.**

**2. `mv` is used to move files and directories.**

**True - `mv` moves or renames files and directories.**

**3. `cd` is used to copy files and directories.**

**False - `cd` is used to change directories, not copy. `cp` is used to copy files and directories.**

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

**True - `pwd` prints the current working directory.**

**5. `grep` is used to search for patterns in files.**

**True - `grep` searches for patterns in files.**

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

**True - `chmod 755` sets these specific permissions.**

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

**True - `mkdir -p` creates nested directories as needed.**

**8. `rm -rf file.txt` deletes a file forcefully without confirmation.**

**True - `rm -rf` forcefully removes files or directories without asking for confirmation.**

# **Part C**

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

A black background with white text and green and blue text

Description automatically generated

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

A black background with white text and green and blue symbols

Description automatically generated

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

**Command: nano qes3.sh**

**This command for crate file and write shell script .**

**Command: echo $name**

**This command for show output on terminal**

A screenshot of a computer code

Description automatically generated

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

Input:

A screenshot of a computer

Description automatically generated

Output:

A black background with blue and white text

Description automatically generated

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

Input

A screenshot of a computer

Description automatically generated

Output:

A black background with blue and white text

Description automatically generated

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

Input:

A screenshot of a computer

Description automatically generated

Output:

A screen shot of a computer

Description automatically generated

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

**Input:**

A screenshot of a computer

Description automatically generated

Output:

A screen shot of a computer

Description automatically generated

**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".

Input:

A screen shot of a computer

Description automatically generated

Output:

A screen shot of a computer program

Description automatically generated

**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.

Input:

A screenshot of a computer

Description automatically generated

Output:

A black background with white text

Description automatically generated

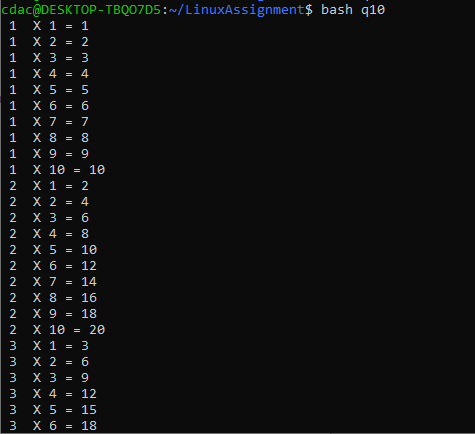
**Question 10:** Write a shell script that uses nested for loops to print a multiplication table for numbers from 1 to 5. The output should be formatted nicely, with each row representing a number and each column representing the multiplication result for that number.

Input:

A screenshot of a computer program

Description automatically generated

Output:

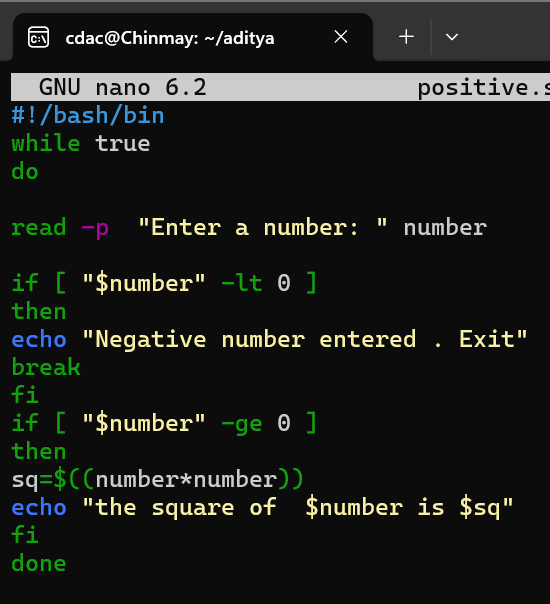


A black screen with a black background

Description automatically generated

**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.

Input:



Output:

A black background with white text and blue letters

Description automatically generated

Part E

1. Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 5 |

| P2 | 1 | 3 |

| P3 | 2 | 6 |

Calculate the average waiting time using First-Come, First-Served (FCFS) scheduling.

A screenshot of a graph

Description automatically generated

2. Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 3 |

| P2 | 1 | 5 |

| P3 | 2 | 1 |

| P4 | 3 | 4 |

Calculate the average turnaround time using Shortest Job First (SJF) scheduling.

A white sheet with black text and numbers

Description automatically generated

3. Consider the following processes with arrival times, burst times, and priorities (lower number

indicates higher priority):

| Process | Arrival Time | Burst Time | Priority |

|---------|--------------|------------|----------|

| P1 | 0 | 6 | 3 |

| P2 | 1 | 4 | 1 |

| P3 | 2 | 7 | 4 |

| P4 | 3 | 2 | 2 |

Calculate the average waiting time using Priority Scheduling.

A screenshot of a chart

Description automatically generated

4. Consider the following processes with arrival times and burst times, and the time quantum for

Round Robin scheduling is 2 units:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 4 |

| P2 | 1 | 5 |

| P3 | 2 | 2 |

| P4 | 3 | 3 |

Calculate the average turnaround time using Round Robin scheduling.

A grid of numbers and letters

Description automatically generated

5. 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?