**Part A**

**What will the following commands do?**

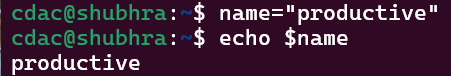
• echo "Hello, World!"

Prints the text



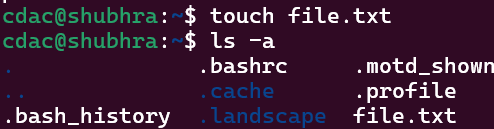
• name="Productive"

Assigns the value



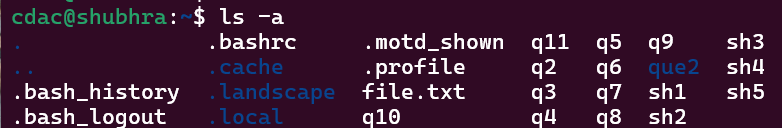
• touch file.txt

Creates a new, empty file



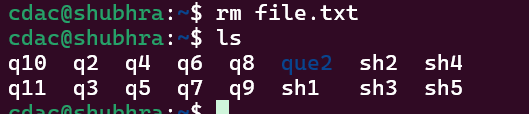
• ls -a

**all files and directories**, **including hidden ones**



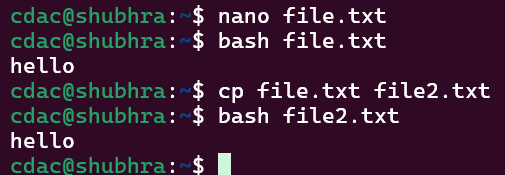
• rm file.txt

**Deletes** the file



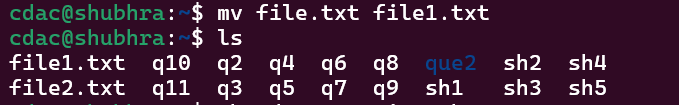
• cp file1.txt file2.txt

**Copies** the contents of file1.txt into a new file called file2.txt



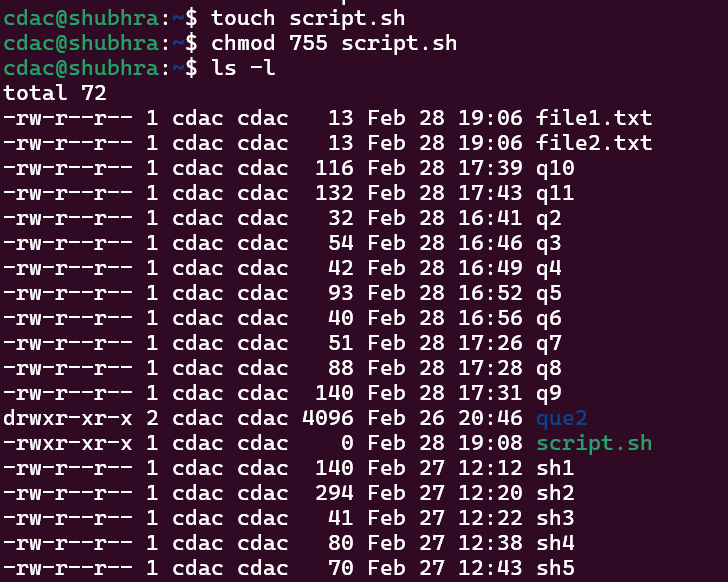
• mv file.txt /path/to/directory/

**Moves** file.txt



• chmod 755 script.sh

Sets **permissions** for script.sh



• grep "pattern" file.txt

**Searches** for the word **"pattern"** in file.txt.



• kill PID

**Terminates a process** with the given **Process ID (PID)**.

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

**Chained commands** using &&

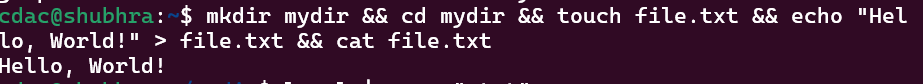
Create a directory mydir.

Move into it (cd).

Create an empty file.txt.

Write "Hello, World!" into the file (> overwrites).

Display the file’s contents (cat)



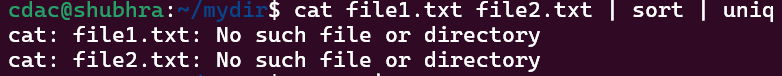
• ls -l | grep ".txt"

Lists files **in long format** and **filters** those containing ".txt" in their names.



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

Combines files, **sorts lines alphabetically**, and removes **duplicates** (uniq).



• ls -l | grep "^d"

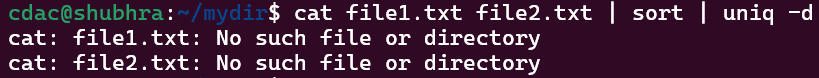
Lists only **directories**

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

**Recursively searches** for "pattern" in all files under the directory.

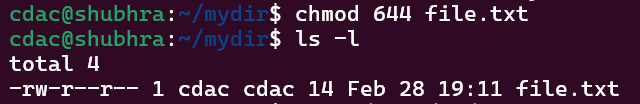
• cat file1.txt file2.txt | sort | uniq –d

Displays **only duplicate lines** from the sorted combination of two files.



• chmod 644 file.txt

Sets **permissions** for file.txt



• cp -r source\_directory destination\_directory

**Recursively copies** a directory and all its contents to a new location.

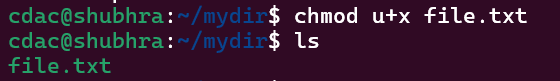
• find /path/to/search -name "\*.txt"

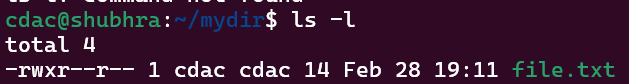
**Finds all files** with a .txt extension **within a directory**



• chmod u+x file.txt

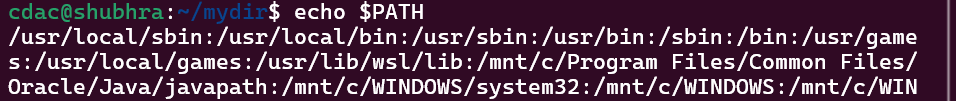
Adds **execute (x) permissions to the owner (u)** of the file.





• echo $PATH

Displays your **system's PATH variable** — a list of directories the shell searches for **executables**.



**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.

**INCORRECT.** Correct command is **chmod.**

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

**INCORRECT.** Correct command is **cp**.

3. **mkfile** is used to create a new file.

**INCORRECT.** Correct command is **touch**.

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

**INCORRECT.** Correct command is **cat**.

5. **rn** is used to rename files.

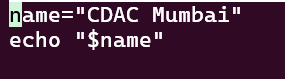
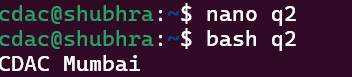
**INCORRECT.** Correct command is **mv**.

**Part C**

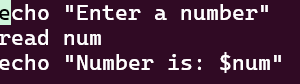
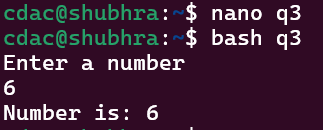
Question 1: Write a shell script that prints "Hello, World!" to the terminal.



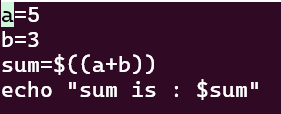
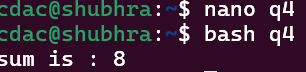
Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

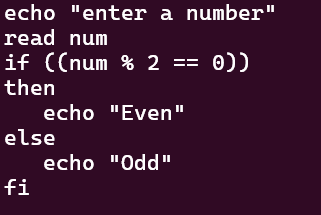
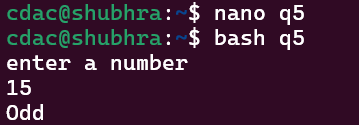
Question 3: Write a shell script that takes a number as input from the user and prints it.

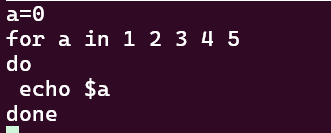
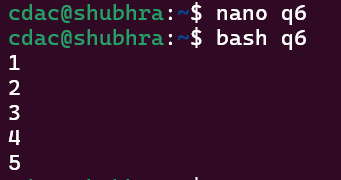
Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.

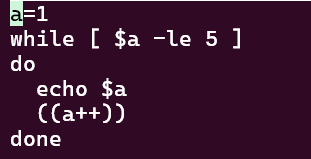
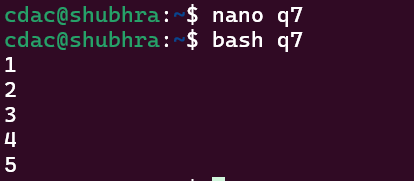
Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".

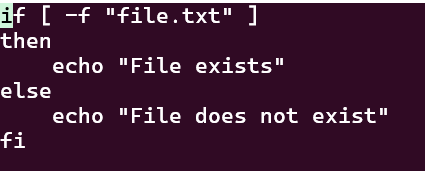
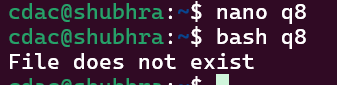
Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.

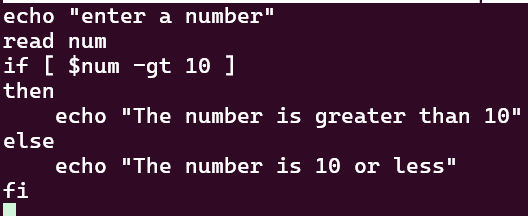
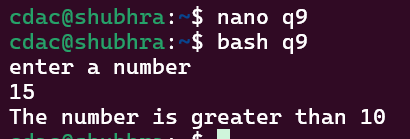
Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.

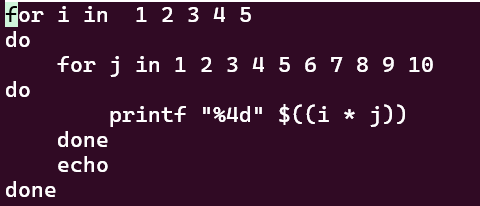
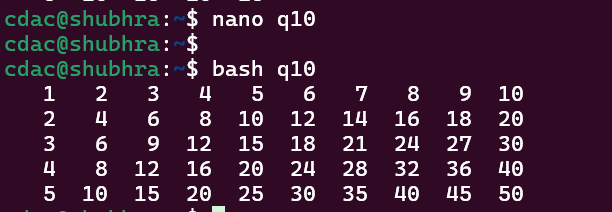
Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".

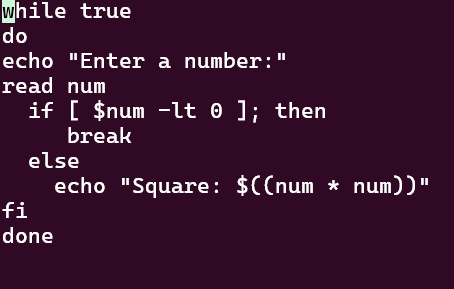
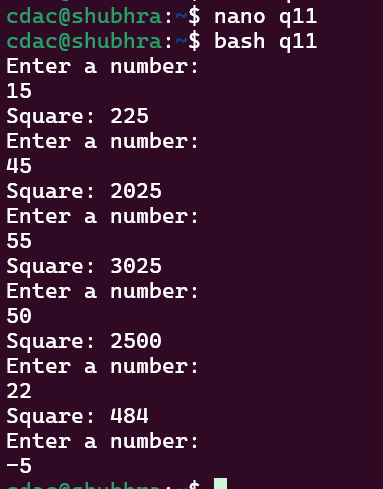
Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.

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.

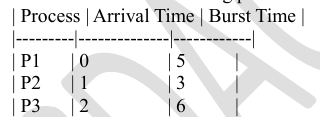
 

Question 11: Write a shell script that uses a while loop to read numbers from the user until the user enters a negative number. For each positive number entered, print its square. Use the break statement to exit the loop when a negative number is entered.

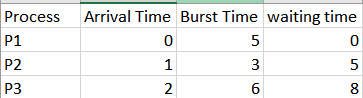
**Part E**

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



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

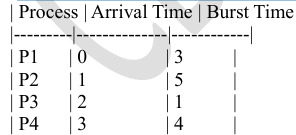
ANS:

****

Gantt chart: ****

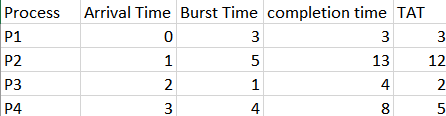
Avg waiting time= 0+5+8 =13/3= 4.3

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



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

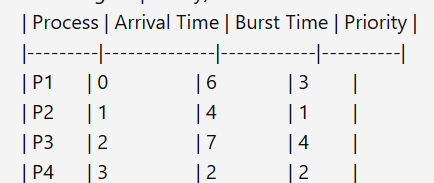
ANS:



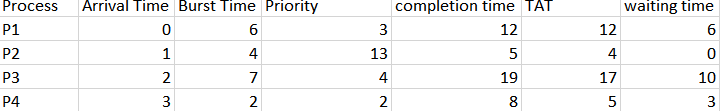
Gantt chart: 

Average TAT=43+12+2+5​=422​=5.5

1. Consider the following processes with arrival times, burst times, and priorities (lower number indicates higher priority):



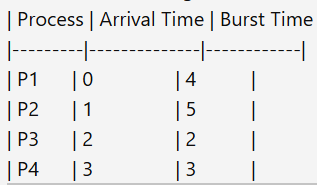
Calculate the average waiting time using Priority Scheduling.



Gantt chart:

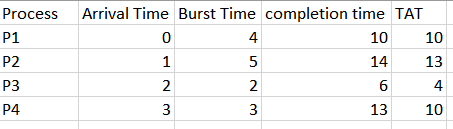
Average WT=46+0+10+3​=419​=4.75

1. Consider the following processes with arrival times and burst times, and the time quantum for Round Robin scheduling is 2 units:



Calculate the average turnaround time using Round Robin scheduling.

ANS:



Gantt chart:



Average TAT=410+13+4+10​=437​=9.25

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?