**CDAC MUMBAI Name: Shreya Raj**

**Concepts of Operating System**

**Assignment 2**

**Part A**

**What will the following commands do?**

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

It will print “Hello, World!” on the screen

* **name="Productive"**

It will save the word Productive in the variable name

* **touch file.txt**

Creates a file named as file.txt

* **ls -a**

This will list all files including the hidden files

* **rm file.txt**

This will delete or remove the file named file.txt

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

It will copy the content of file1.txt into file2.txt

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

This will move the file.txt to the specific directory

* **chmod 755 script.sh**

It will change the permission for the file script.sh

7{owner: read, write, execute}

5{group: read and write}

5{others: read and write}

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

This will search the line having the specific word “pattern” in file.txt

* **kill PID**

This will kill or terminate the process with the given process PID

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

This will create a directory named mydir and the command cd mydir will help to fo inside the directory mydir, and the command touch file.txt will create a file named file.txt and the command echo will help to print the “Hello, World!” in the file.txt and then the command cat will help to display the content inside the file.txt

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

It will show files in long format and will flter out those having .txt in the name

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

The cat command will combine contents of file1.txt and file2.txt and the sort command will sort them alphabetically and then the command uniq will remove the duplicate lines.

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

It will list files in long format and then grep will help in filtering the file starting with d (directories)

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

It will search for word pattern in the file (recursively)

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

cat will join both the files and sort will sort and uniq -d will show only the duplicate lines

* **chmod 644 file.txt**

This will set permission to file.txt as:

6 {owner: read and write}

4 {group: read}

4 {others: read}

* **cp -r source\_directory destination\_directory**

This will copy the entire folder source\_directory into destination\_directory recursively

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

This will search all the .txt files

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

This will give the owner the permission to execute the file.

* **echo $PATH**

echo will print the value of PATH and $Path shows thw value stored in path variable.

In Linux, PATH is an environmental variable which will store list of directories.

**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 is used to change directory**

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**

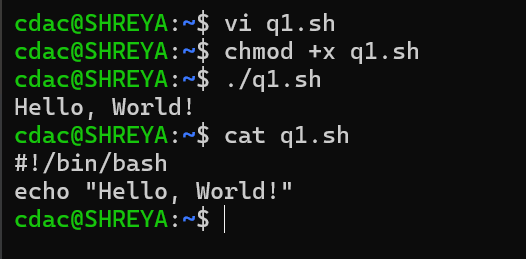
3. mkfile is used to create a new file. **touch filename**

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

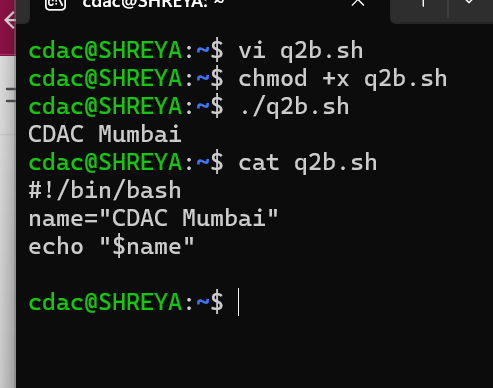
5. rn is used to rename files. **mv**

**Part C**

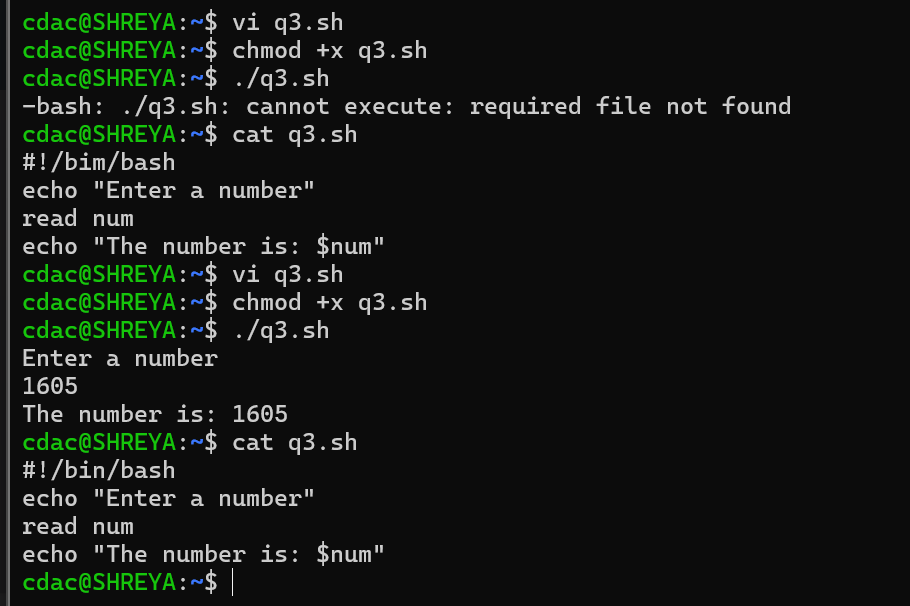
**Question 1: Write a shell script that prints "Hello, World!" to the terminal.**



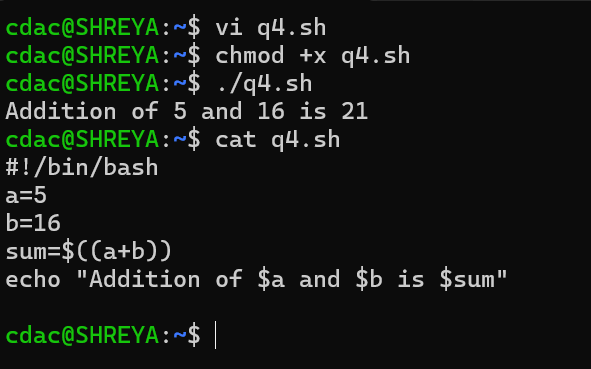
**Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.**



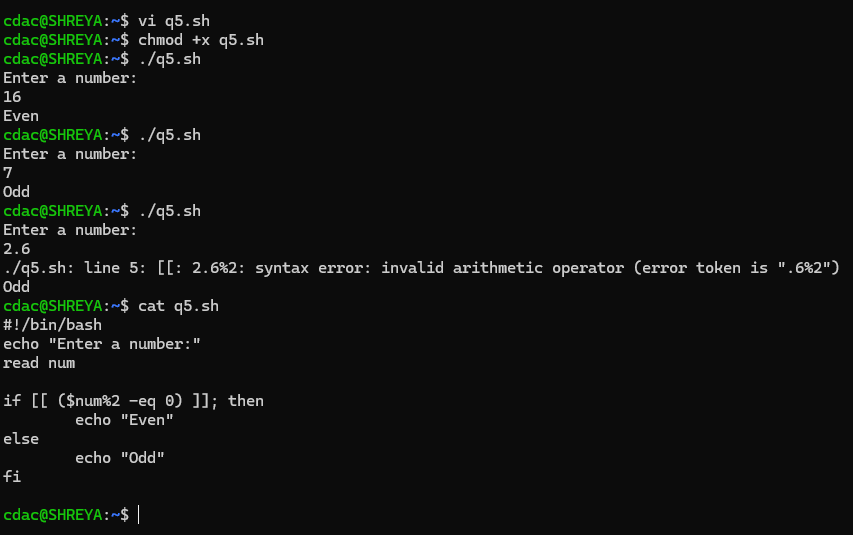
**Question 3: Write a shell script that takes a number as input from the user and prints it.**



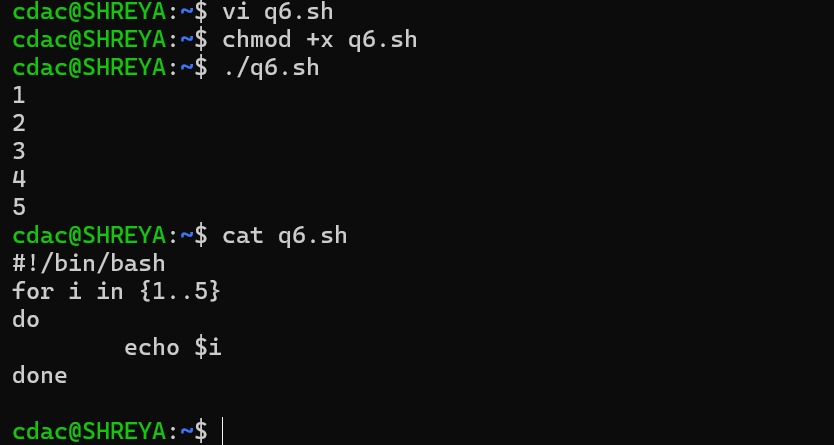
**Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.**



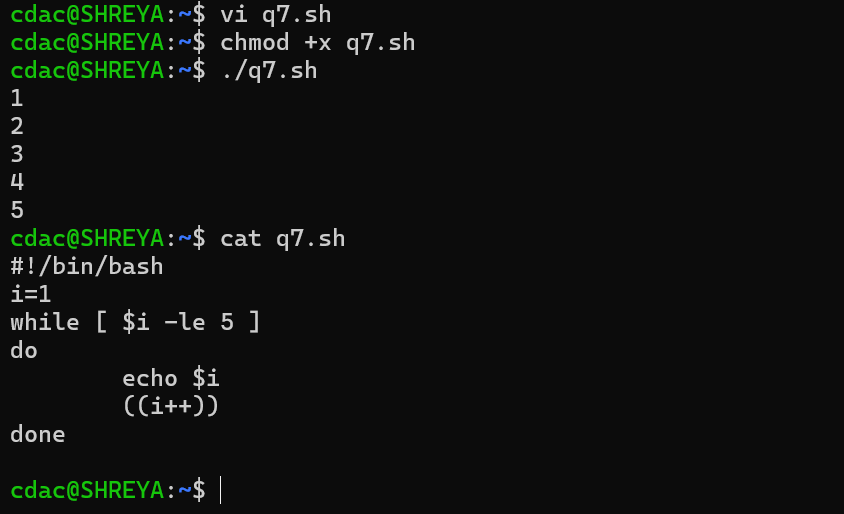
**Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".**



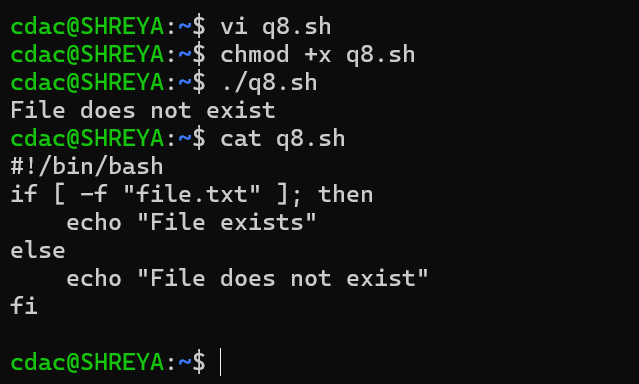
**Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.**



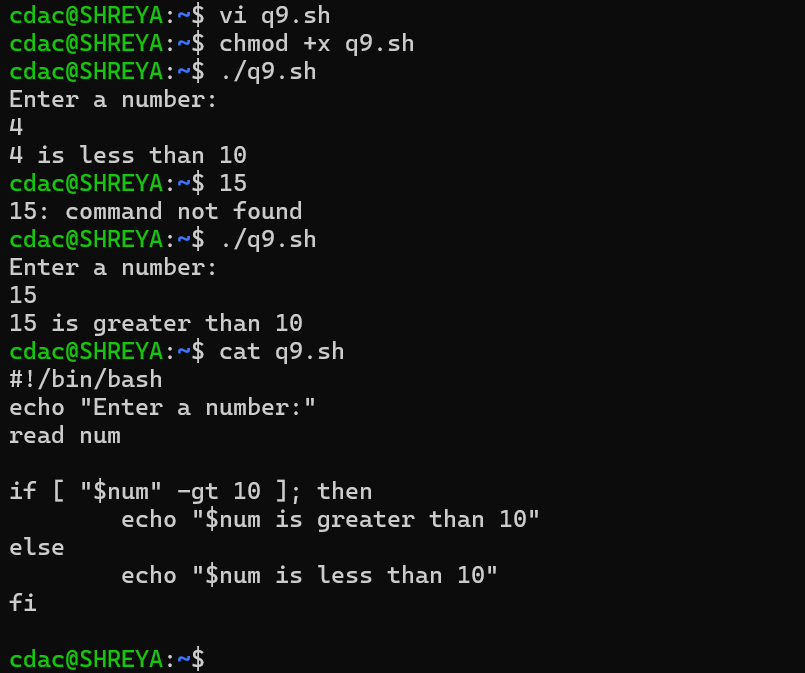
**Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.**



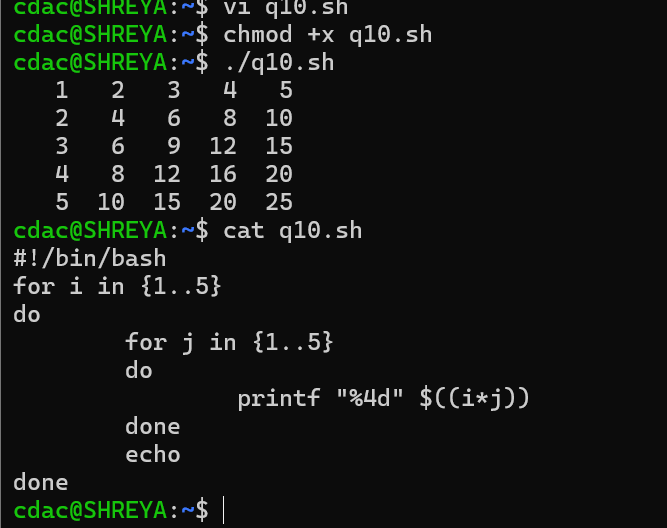
**Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".**



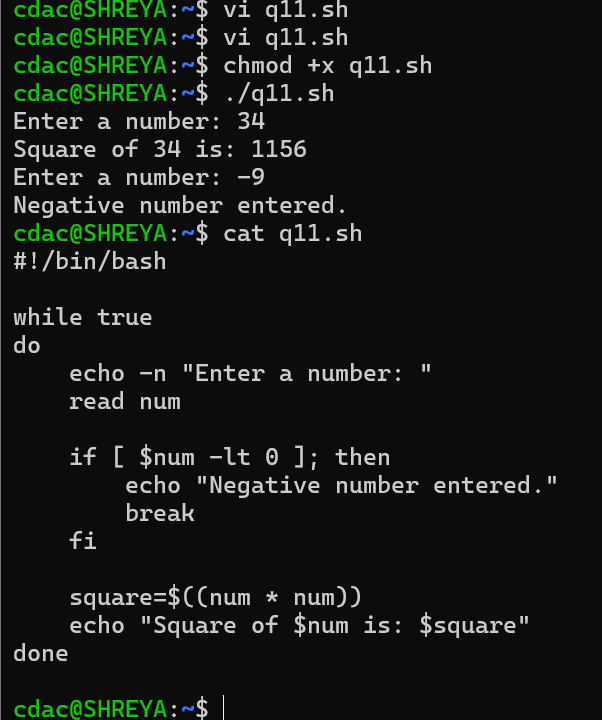
**Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.**



**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:

| 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.

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

| Process | Arrival Time | Burst Time |

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

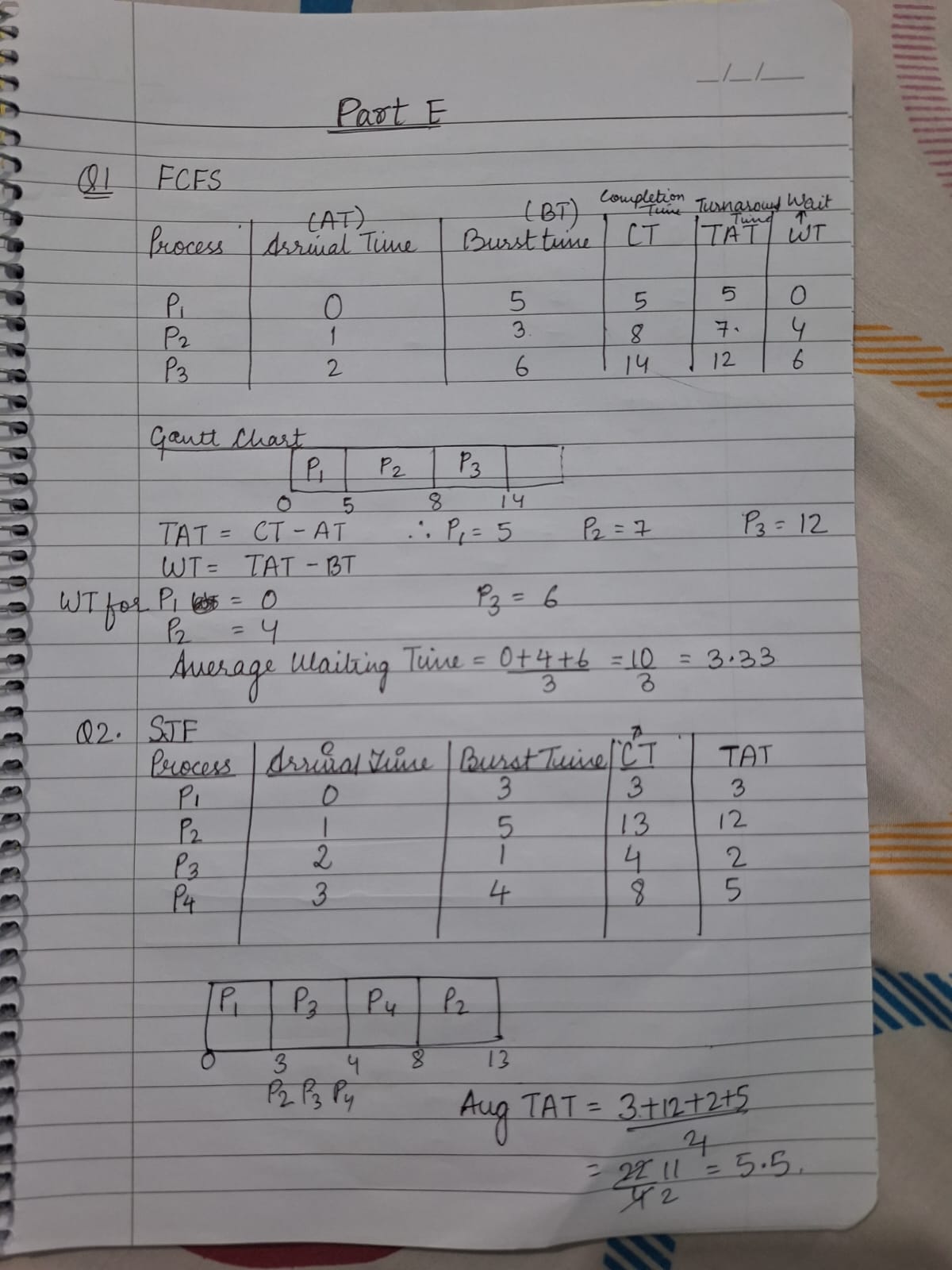
| P1 | 0 | 3 |

| P2 | 1 | 5 |

| P3 | 2 | 1 |

| P4 | 3 | 4 |

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



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.

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.

