CDAC MUMBAI

Concepts of Operating System
Assignment 2

Sakshi Jadhav-KH

Part A

What will the following commands do?

Command	Description
echo "Hello, World!"	Prints "Hello, World!" to the terminal
name="Productive"	Assigns the string "Productive" to the variable name.
touch file.txt	Creates an empty file named file.txt or updates its timestamp if it exists.
ls -a	Lists all files and directories, including hidden ones .
rm file.txt	Deletes the file file.txt.
cp file1.txt file2.txt	Copies file1.txt to file2.txt.
Mv file.txt /path/to/directory/	Moves file.txt to the specified directory.
chmod 755 script.sh	Chmod - Change file permission,755 - Grants read, write, execute to owner, and read & execute to group and others for script.sh.
grep "pattern" file.txt	Searches for "pattern" in file.txt and displays matching lines.
kill PID	Terminates the process with the given PID.
mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt	Creates mydir directory, navigates into it, creates file.txt, writes "Hello, World!" to it, and displays its contents.
ls -l | grep ".txt"	ls -l - Lists files in the current directory in long format. grep ".txt" - Filters the output to show only lines containing .txt, which means it displays only .txt files.

cat file1.txt file2.txt | sort | uniq	cat file1.txt file2.txt - Concatenates (merge) the contents of both files. sort - Sorts the combined contents alphabetically. uniq - Removes duplicate lines
grep -r "pattern" /path/to/directory/	Recursively searches for "pattern" in all files within the directory.
chmod 644 file.txt	Chmod - change the permission ,644 - Grants read & write to owner, and read-only to group and others for file.txt.
Cp -r source_directory destination_directory	Recursively copies source_directory to destination_directory.
find /path/to/search -name "*.txt"	Searches for files ending in .txt within the given path.
chmod u+x file.txt	Chmod change the permission ,u+x Grants execute permission to the file owner (u).
echo $PATH	Displays the system's executable search paths.
ls -l | grep "^d"	ls -l Lists files and directories in long format. grep "^d" Filters lines starting with d, which indicates directories.
cat file1.txt file2.txt | sort | uniq –d	cat file1.txt file2.txt - Combines the contents of both files. sort - Sorts the combined contents. uniq -d - Displays only duplicate lines (lines that appear in both files).

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 permissions to group and others.

→ False

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 : used to change file permissions

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

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

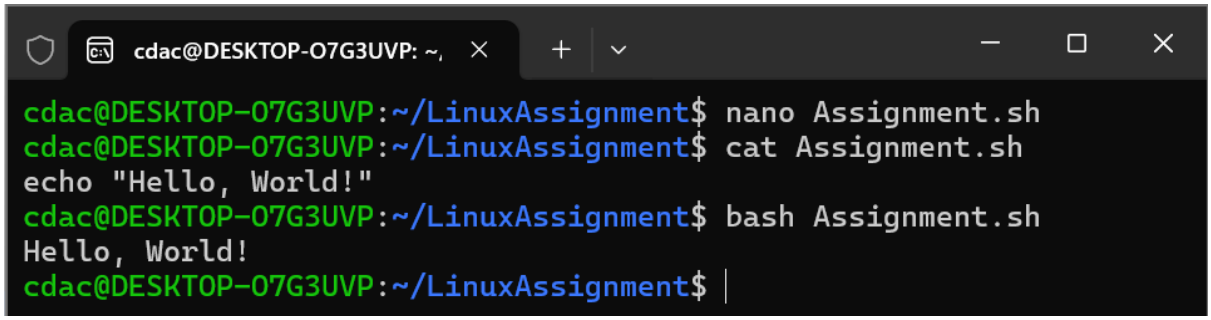
4. **catx** is used to concatenate files.
→ cat : used to concatenate and display file contents

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

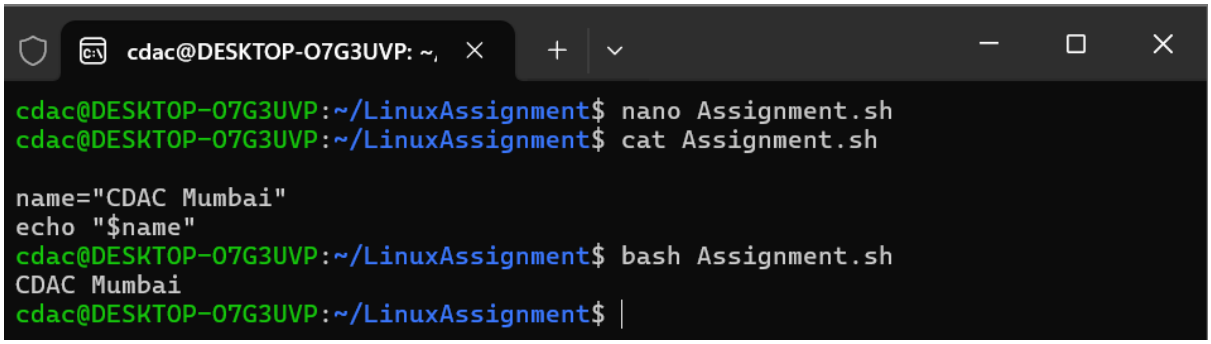
→ mv : used to rename or move files

Part C

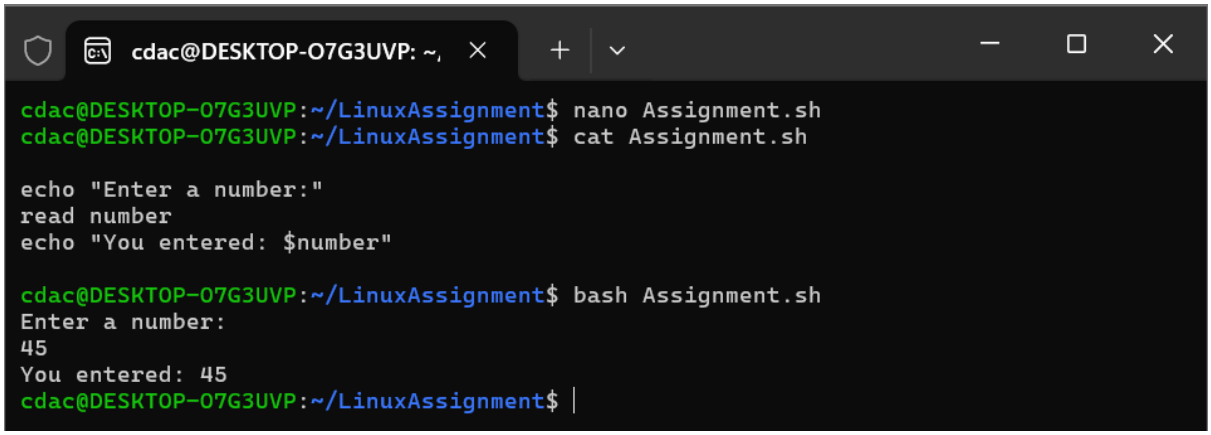
Question 1: Write a shell script that prints "Hello, World!" to the terminal.



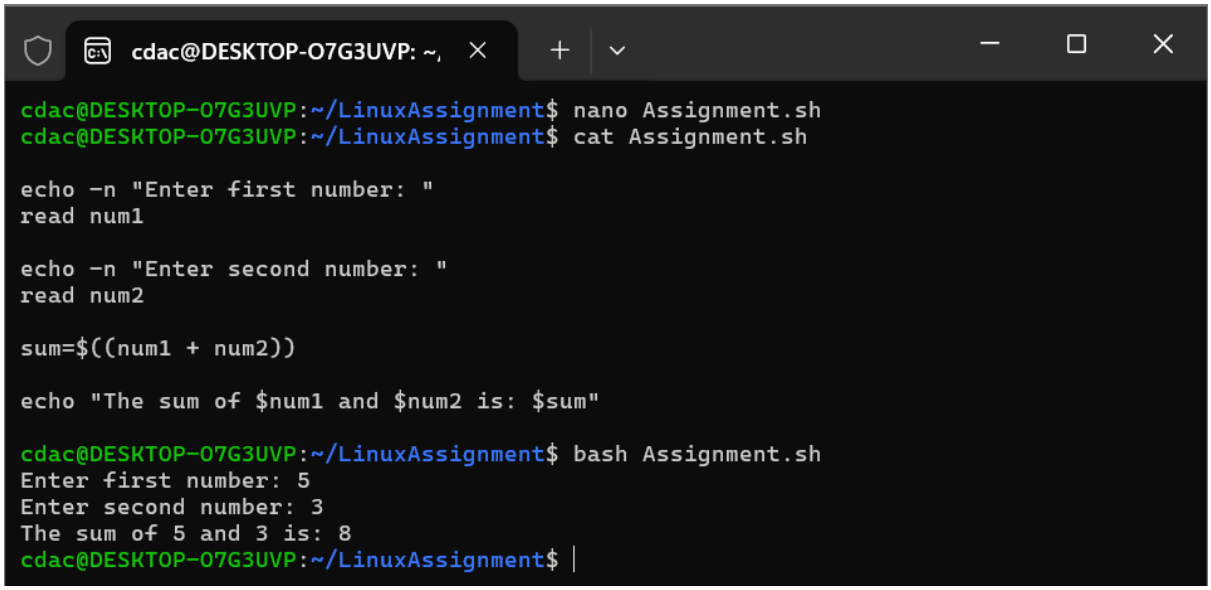
Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.



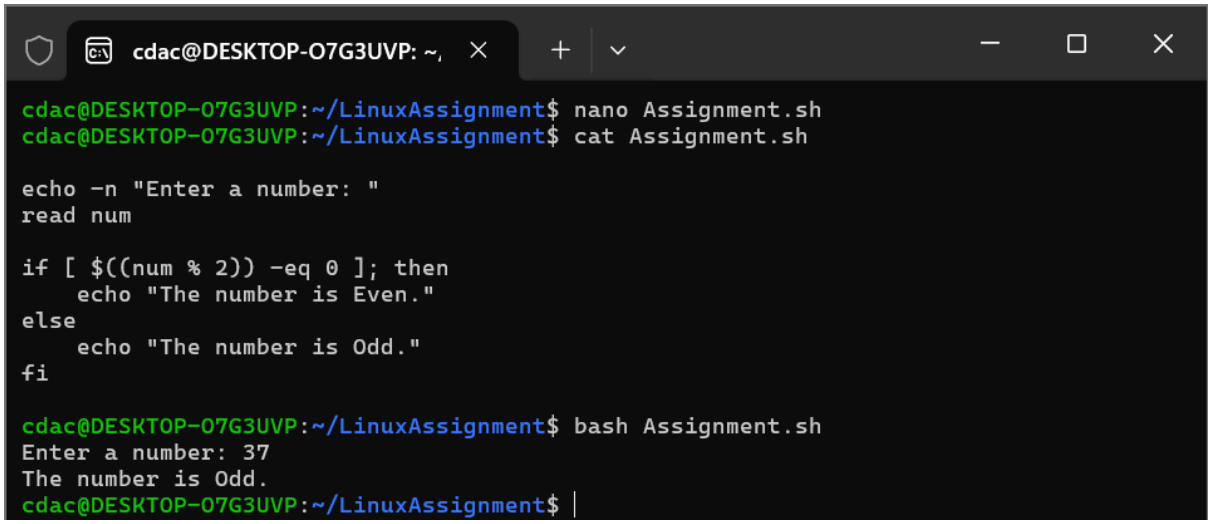
Question 3: Write a shell script that takes a number as input from the user and prints it.



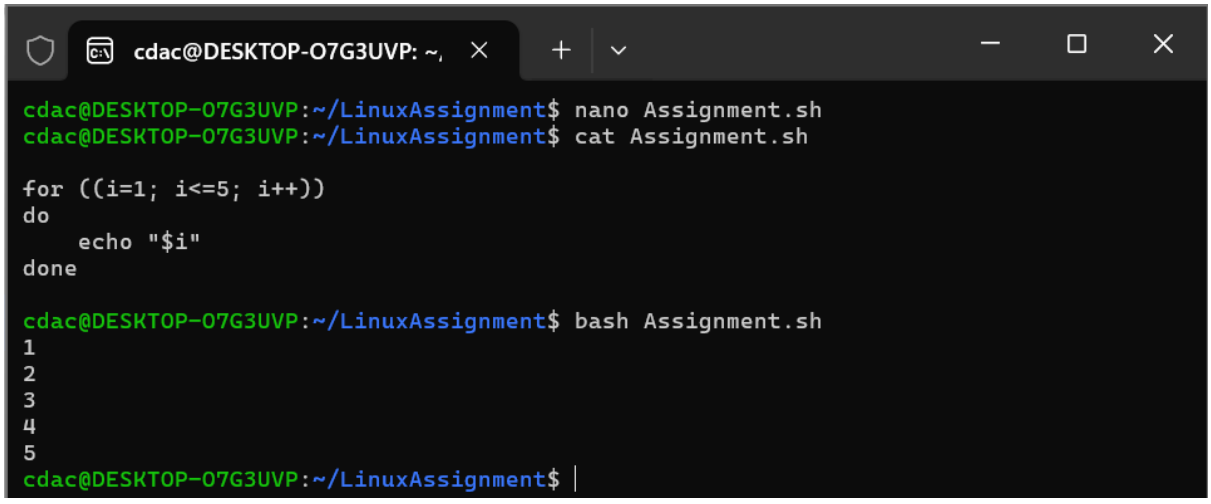
Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.



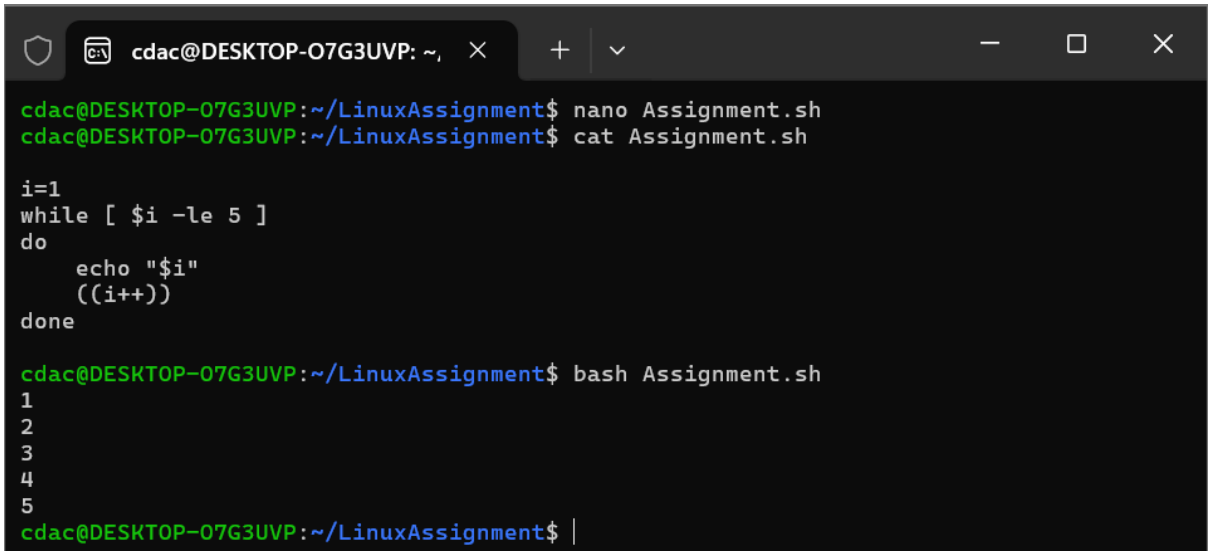
Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".



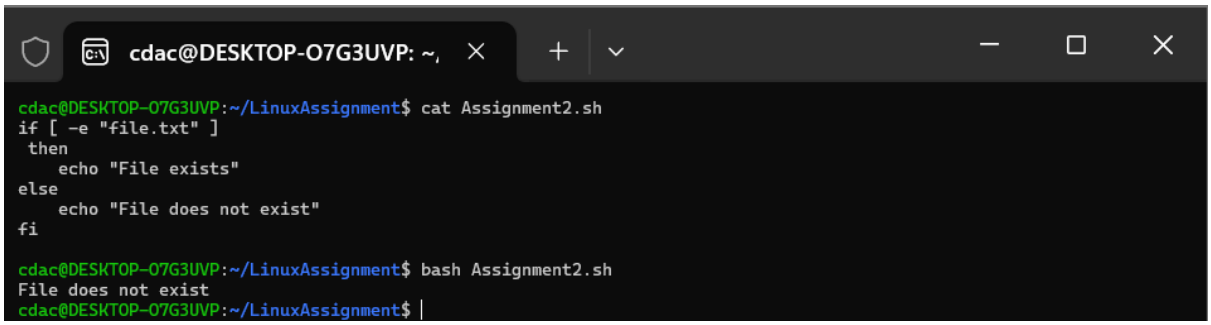
Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.



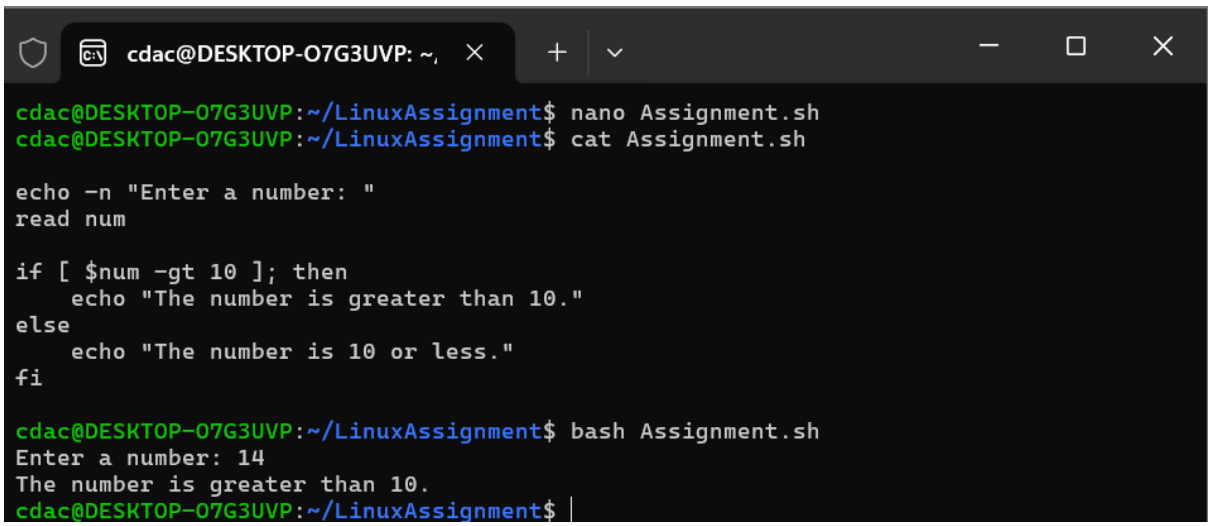
Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.



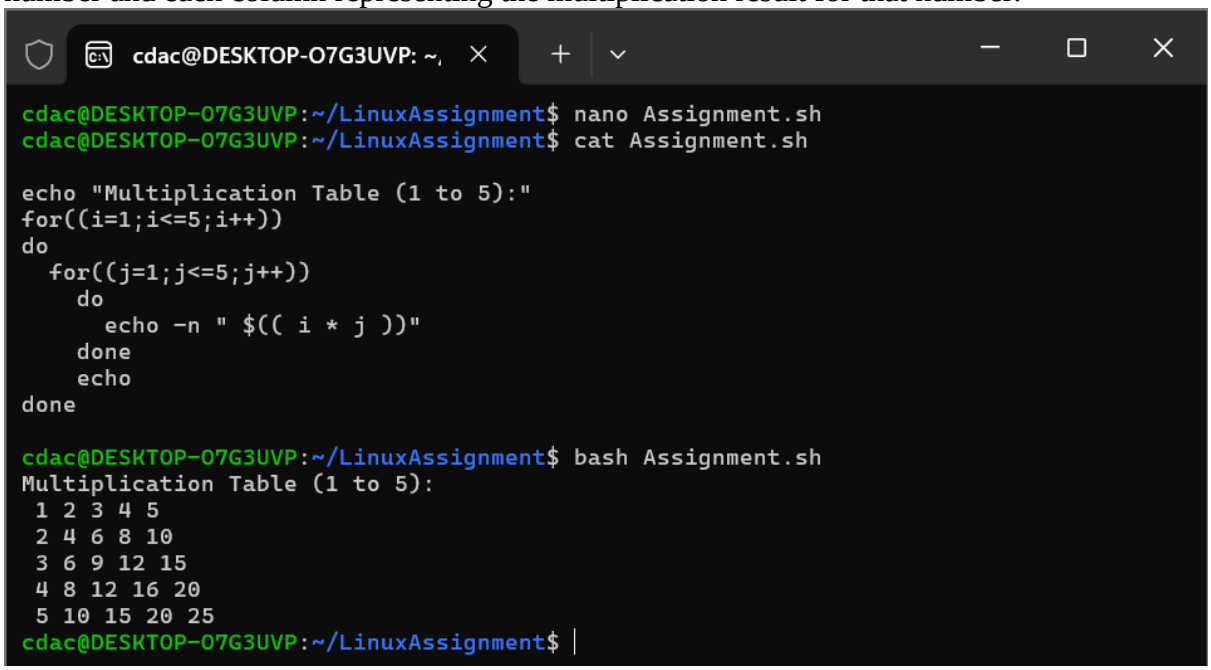
Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".



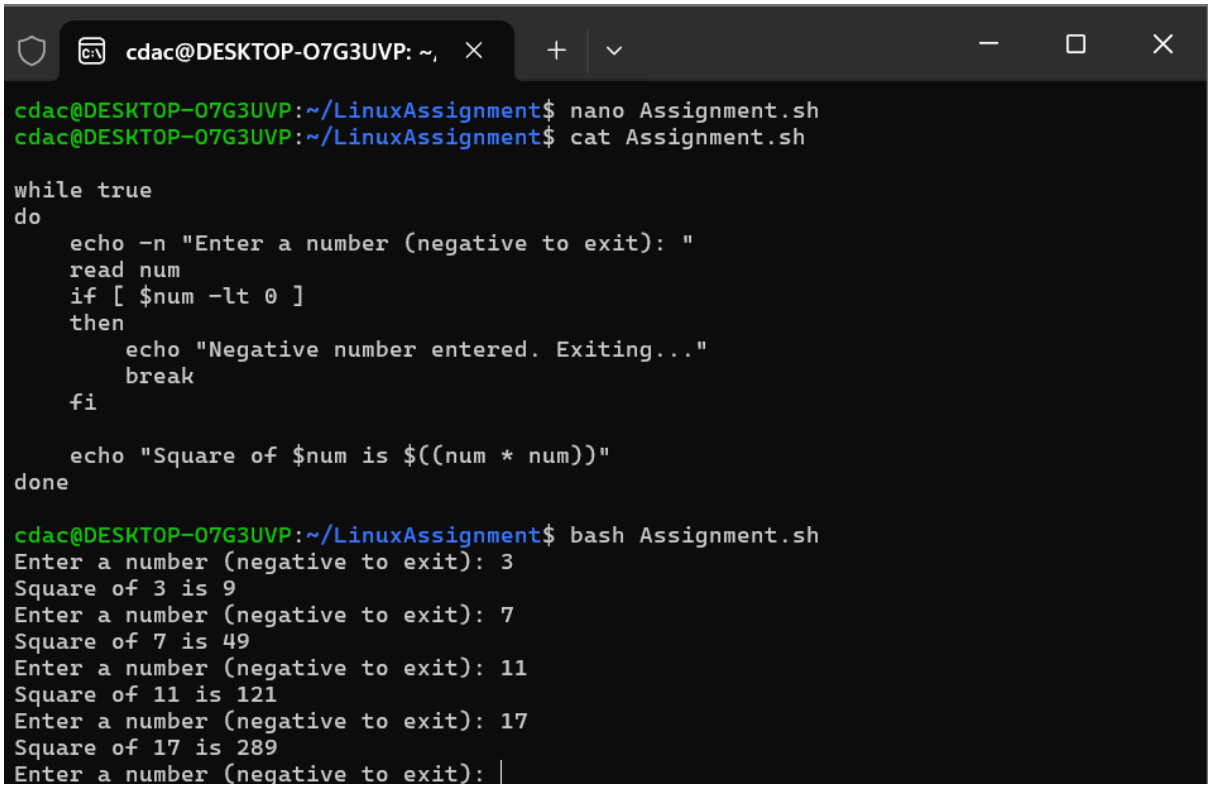
Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.



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 D

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.

Ans:

Process	Arrival Time	Burst time	Waiting Time
P1	0	5	0
P2	1	3	4
P3	2	6	6

Gantt chart:

0		5		8		14
	P1		P2		P3	

Average waiting time = (0+4+6)/3
= 3.33

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.
Ans ;

Process	Arrival Time	Burst time	Waiting time	TAT
P1	0	3	0	3
P2	1	5	7	12
P3	2	1	1	2
P4	3	4	1	5

	0		3		4		8	13
Gantt chart		P1		P3		P4		P2

Average TAT = (3+12+2+3) /4
= 5.5

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.

Ans ; Non-Preemptive

Process	Arrival Time	Burst time	Priority	Response time	Waiting time	TAT
P1	0	6	3	0	0	6
P2	1	4	1	5	5	9
P3	2	7	4	10	10	17
P4	3	2	2	7	7	9

	0	6	10	12	19
Gantt chart	P1	P3	P4	P2	

- Average waiting Time = (0+5+10+7) /4 = 5.5

Preemptive:

Process	Arrival Time	Burst time	Priority	Response time	Waiting time	TAT
P1	0	6	3	6	6	10
P2	1	4	1	0	0	4
P3	2	7	4	10	10	17
P4	3	2	2	2	2	4

	0	2	5	7	12	19
Gantt chart	P1	P2	P4	P1	P3	

- Average waiting Time = (6+0+10+2)/4 = 4.5

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.

Ans :

Process	Arrival Time	Burst time	Response time	Waiting time	TAT
P1	0	4	0	6	10
P2	1	5	1	8	13
P3	2	2	2	3	4
P4	3	3	6	7	10

	0	2	4	6	8	10	12	13	14
Gantt c	P1	P2	P3	P4	P1	P2	P4	P2	

- Average Turnaround Time (TAT) = (10+13+4+10)/4 = 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?

Ans : When the fork() system call is used, it creates a child process that has its own copy of the parent's memory

-Before fork():The parent process has x = 5.

- fork() is called A child process is created, and it inherits x = 5 from the parent.

-Both processes execute independently:

- **Parent process:** Increments x → x = 6.
- **Child process:** Increments x → x = 6.

Since both processes have their own separate copies of x in memory, their modifications do not affect each other. Thus, both the parent and child will have x = 6 in their respective address spaces.