

Concepts of Operating System

Assignment 2

Part A

What will the following commands do?

- `echo "Hello, World!"`

Ans: Print text to the terminal.

In this case it will print : Hello, World!

- `name="Productive"`

Ans: This command Assigns String productive to the name variable.

- `touch file.txt`

Ans: Creates an empty file named file.txt

- `ls -a`

Ans: list all files and directories with hidden ones

- `rm file.txt`

Ans: Deletes the file named file.txt

- cp file1.txt file2.txt

Ans: Copies file1.txt content to file2.txt

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

Ans: Mv is a move command

File.txt is the file being moved here

/path/to/direction is the destination directory

- chmod 755 script.sh

Ans: This command changes the permission of the script.sh to allow the owner to read, write, and execute the file , while others can only read and execute it.

- grep "pattern" file.txt

Ans: Grep command searches for the string “pattern” in file.txt and then displays the matching specific word or lines in that file

- kill PID

Ans: This command Terminates the process with the specific Process ID (PID)

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

Ans: mkdir Creates a directory named mydir

by using `cd` we enter in that directory

By `touch file.txt` command we create `file.txt`

`echo` write `Hello,World!` into file

and later `cat` command display all the contents of file

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

Ans: `ls -l` command list all files and directories in detailed format and `grep` command filters those containing `.txt`

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

Ans: concatenates `file1.txt` and `file2.txt`, sorts the combined content and removes the duplicate ones

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

Ans: list all the files and directories in detailed format and `grep "^d"` command searches for only directories

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

Ans: This command will search for pattern word in all directory and subdirectory given as destination path.

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

Ans: concatenates file1.txt and file2.txt , sorts them and displays only duplicate lines as it is written -d

- `chmod 644 file.txt`

Ans: set file.txt permission so that the owner can read and write while others can only read it

- `cp -r source_directory destination_directory`

Ans: Recursively copies the source_directory contents to destination_directory

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

Ans: searches for files with .txt extension within the path given

- `chmod u+x file.txt`

Ans: This command is used for to give permission to owner to execute file.txt

- `echo $PATH`

Ans: This command Displays the system PATH environment variable, which list directories searched for executable files. It give list of directories separated by (:)

Part B

Identify True or False:

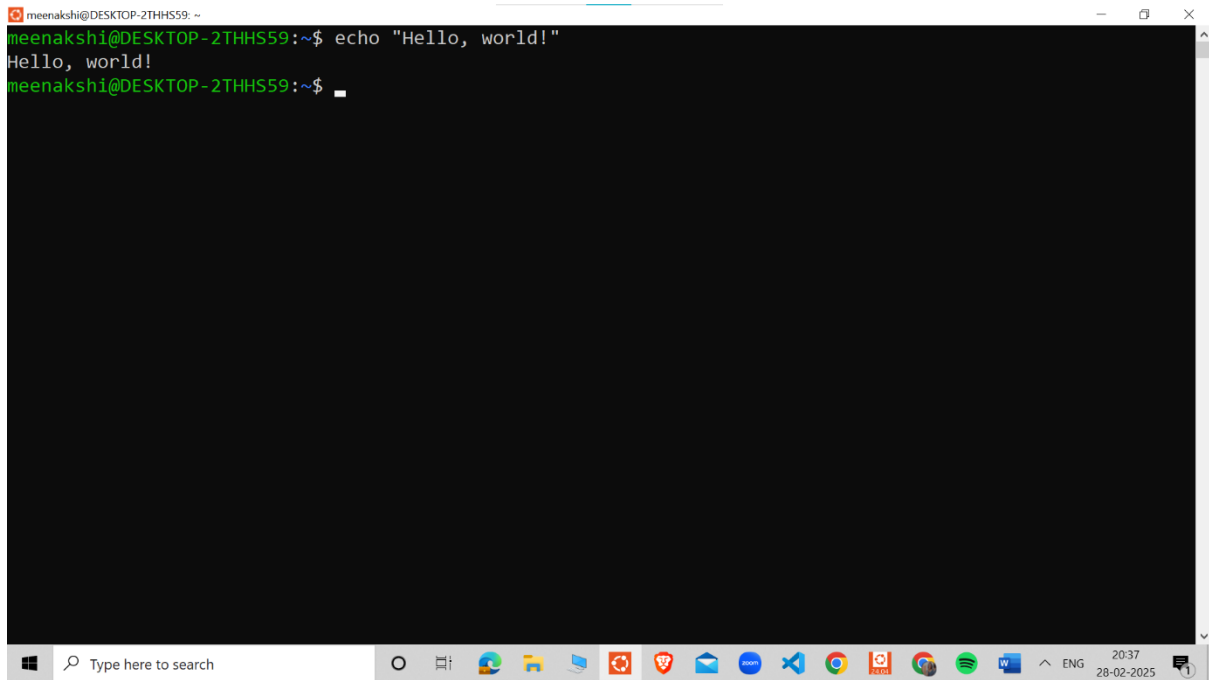
1. ls is used to list files and directories in a directory. - True
2. mv is used to move files and directories. - True
3. cd is used to copy files and directories. - False
4. pwd stands for "print working directory" and displays the current directory. - true
5. grep is used to search for patterns in 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...
2. `cpy` is used to copy files and directories. - Incorrect
3. `mkfile` is used to create a new file. - Incorrect
4. `catx` is used to concatenate files. - Incorrect
5. `rn` is used to rename files. - Incorrect

Part C

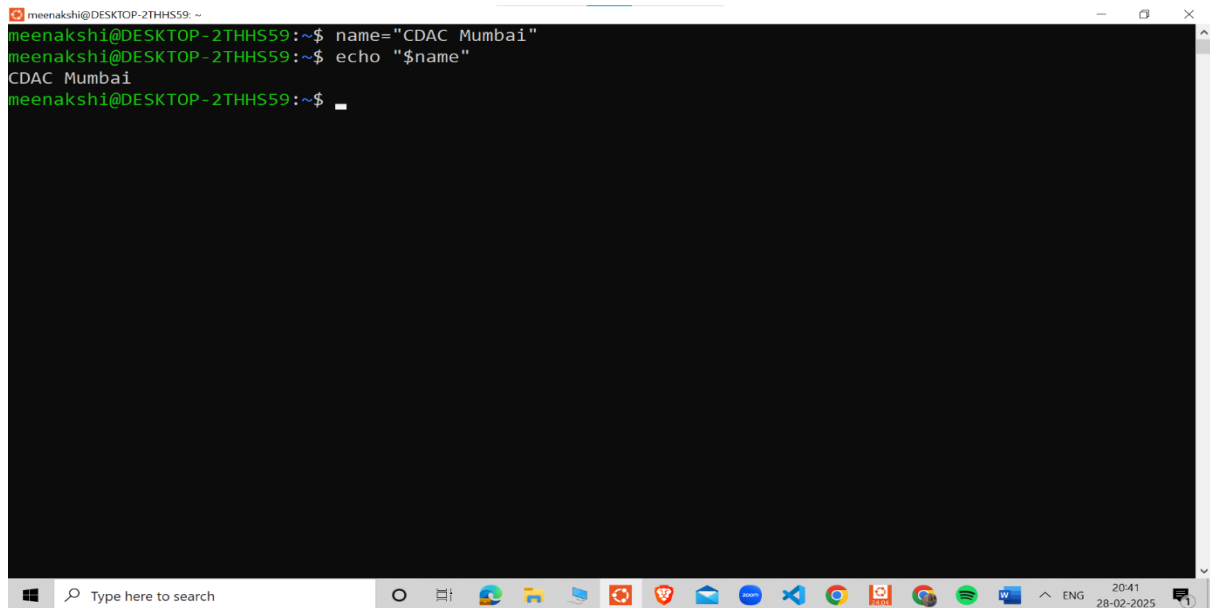
Question 1: Write a shell script that prints "Hello, World!" to the terminal.



A screenshot of a Windows terminal window. The title bar at the top reads "meenakshi@DESKTOP-2THHS59: ~". The terminal content shows a green prompt "meenakshi@DESKTOP-2THHS59:~\$" followed by the command "echo \"Hello, world!\"". The output "Hello, world!" is displayed on the next line. The prompt "meenakshi@DESKTOP-2THHS59:~\$" appears again on the third line, followed by a cursor. The Windows taskbar is visible at the bottom, showing the search bar, task view button, and several application icons including Edge, File Explorer, and various social media apps. The system tray on the right shows the time as 20:37 and the date as 28-02-2025.

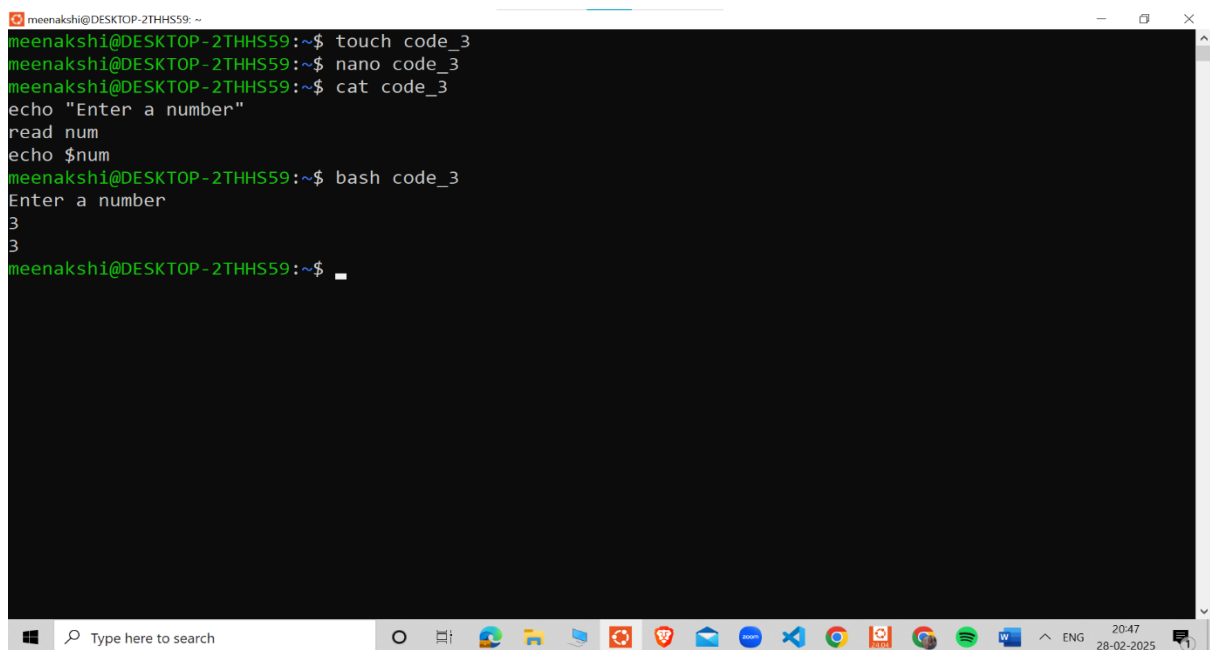
```
meenakshi@DESKTOP-2THHS59: ~  
meenakshi@DESKTOP-2THHS59:~$ echo "Hello, world!"  
Hello, world!  
meenakshi@DESKTOP-2THHS59:~$
```

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

A terminal window titled 'meenakshi@DESKTOP-2THHS59: ~' showing the execution of a shell script. The user enters 'name="CDAC Mumbai"', followed by 'echo "\$name"', which outputs 'CDAC Mumbai'. The prompt returns to '~\$'.

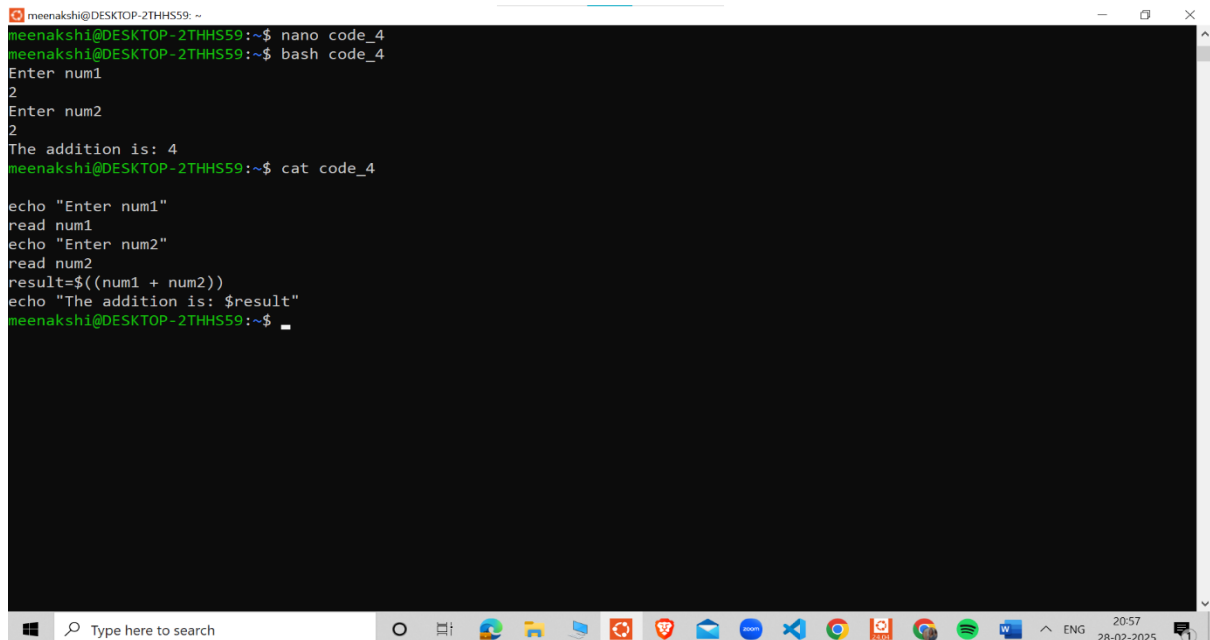
```
meenakshi@DESKTOP-2THHS59: ~$ name="CDAC Mumbai"
meenakshi@DESKTOP-2THHS59: ~$ echo "$name"
CDAC Mumbai
meenakshi@DESKTOP-2THHS59: ~$
```

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

A terminal window titled 'meenakshi@DESKTOP-2THHS59: ~' showing the creation and execution of a shell script. The user creates 'code_3' with 'touch', opens it with 'nano', and adds code: 'echo "Enter a number"', 'read num', and 'echo \$num'. After saving and exiting, the user runs 'bash code_3', which prompts 'Enter a number', receives input '3', and prints '3'.

```
meenakshi@DESKTOP-2THHS59: ~$ touch code_3
meenakshi@DESKTOP-2THHS59: ~$ nano code_3
meenakshi@DESKTOP-2THHS59: ~$ cat code_3
echo "Enter a number"
read num
echo $num
meenakshi@DESKTOP-2THHS59: ~$ bash code_3
Enter a number
3
3
meenakshi@DESKTOP-2THHS59: ~$
```

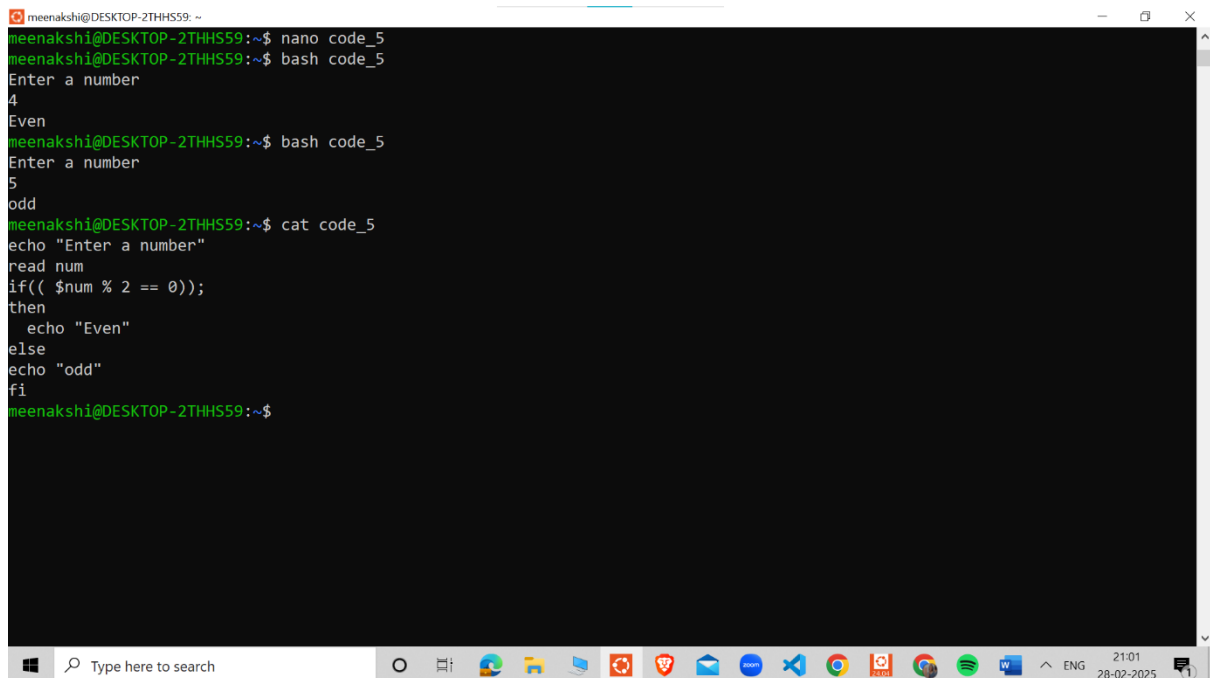

Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.



A terminal window titled 'meenakshi@DESKTOP-2THHS59: ~' showing the execution of a shell script. The user enters 'nano code_4' to create the script, then 'bash code_4' to run it. The script prompts for 'num1' and 'num2', both of which are entered as '2'. The script then outputs 'The addition is: 4'. Finally, the user enters 'cat code_4' to view the script's content, which is displayed as follows:

```
meenakshi@DESKTOP-2THHS59: ~$ nano code_4
meenakshi@DESKTOP-2THHS59: ~$ bash code_4
Enter num1
2
Enter num2
2
The addition is: 4
meenakshi@DESKTOP-2THHS59: ~$ cat code_4
echo "Enter num1"
read num1
echo "Enter num2"
read num2
result=$((num1 + num2))
echo "The addition is: $result"
meenakshi@DESKTOP-2THHS59: ~$
```

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



A terminal window titled 'meenakshi@DESKTOP-2THHS59: ~' showing the execution of a shell script. The user enters 'nano code_5' to create the script, then 'bash code_5' to run it. The script prompts for 'Enter a number'. In the first run, '4' is entered and the output is 'Even'. In the second run, '5' is entered and the output is 'odd'. Finally, the user enters 'cat code_5' to view the script's content, which is displayed as follows:

```
meenakshi@DESKTOP-2THHS59: ~$ nano code_5
meenakshi@DESKTOP-2THHS59: ~$ bash code_5
Enter a number
4
Even
meenakshi@DESKTOP-2THHS59: ~$ bash code_5
Enter a number
5
odd
meenakshi@DESKTOP-2THHS59: ~$ cat code_5
echo "Enter a number"
read num
if(( $num % 2 == 0 ));
then
    echo "Even"
else
    echo "odd"
fi
meenakshi@DESKTOP-2THHS59: ~$
```

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

```
meenakshi@DESKTOP-2THHS59: ~  
meenakshi@DESKTOP-2THHS59:~$ nano code_6  
meenakshi@DESKTOP-2THHS59:~$ bash code_6  
1  
2  
3  
4  
5  
meenakshi@DESKTOP-2THHS59:~$ cat code_6  
for a in 1 2 3 4 5  
do  
echo $a  
done  
meenakshi@DESKTOP-2THHS59:~$
```

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

```
Select meenakshi@DESKTOP-2THHS59: ~  
meenakshi@DESKTOP-2THHS59:~$ nano code_7  
meenakshi@DESKTOP-2THHS59:~$ cat code_7  
a=1  
while [ $a -lt 6 ]  
do  
echo $a  
a=`expr $a + 1`  
done  
meenakshi@DESKTOP-2THHS59:~$ bash code_7  
1  
2  
3  
4  
5  
meenakshi@DESKTOP-2THHS59:~$
```

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".

```
meenakshi@DESKTOP-2THHS59: ~  
meenakshi@DESKTOP-2THHS59:~$ nano code_8  
meenakshi@DESKTOP-2THHS59:~$ cat code_8  
if [ -e "file.txt" ]; then  
    echo "file exists"  
else  
    echo "file does not exist"  
fi  
meenakshi@DESKTOP-2THHS59:~$ bash code_8  
file does not exist  
meenakshi@DESKTOP-2THHS59:~$ nano code_8  
meenakshi@DESKTOP-2THHS59:~$ bash code_8  
\file does not exist  
meenakshi@DESKTOP-2THHS59:~$
```

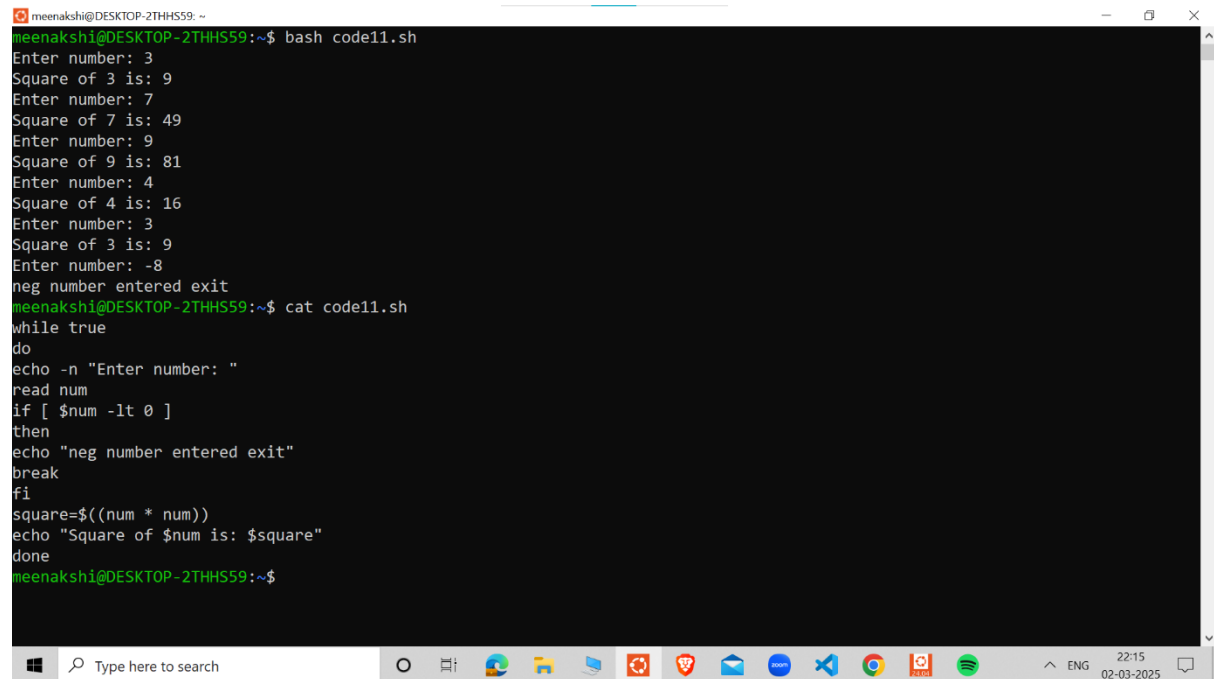
Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.

```
meenakshi@DESKTOP-2THHS59: ~  
meenakshi@DESKTOP-2THHS59:~$ bash code_9.sh  
Enter a num  
11  
number is greater than 10  
meenakshi@DESKTOP-2THHS59:~$ bash code_9.sh  
Enter a num  
9  
number is less than than or equal to 10  
meenakshi@DESKTOP-2THHS59:~$ cat code_9.sh  
echo "Enter a num"  
read num  
if [ $num -gt 10 ]  
then  
    echo "number is greater than 10"  
else  
    echo "number is less than than or equal to 10"  
fi  
meenakshi@DESKTOP-2THHS59:~$
```

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.

```
meenakshi@DESKTOP-2THHS59: ~$ nano code10.sh
meenakshi@DESKTOP-2THHS59: ~$ bash code10.sh
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
meenakshi@DESKTOP-2THHS59: ~$ cat code10.sh
for a in 1 2 3 4 5
do
for b in 1 2 3 4 5
do
echo -n "$((a * b)) "
done
echo
done
meenakshi@DESKTOP-2THHS59: ~$
```

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.

A screenshot of a Windows terminal window. The title bar shows 'meenakshi@DESKTOP-2THHS59: ~'. The terminal content shows a user running 'bash code11.sh'. The script prompts for numbers: 3, 7, 9, 4, 3, and -8. For positive numbers, it prints the square (9, 49, 81, 16, 9). For -8, it prints 'neg number entered exit'. Then, the user runs 'cat code11.sh', displaying the script's source code which uses a 'while true' loop with a 'break' statement for negative numbers. The Windows taskbar is visible at the bottom with the search bar and various application icons.

```
meenakshi@DESKTOP-2THHS59: ~$ bash code11.sh
Enter number: 3
Square of 3 is: 9
Enter number: 7
Square of 7 is: 49
Enter number: 9
Square of 9 is: 81
Enter number: 4
Square of 4 is: 16
Enter number: 3
Square of 3 is: 9
Enter number: -8
neg number entered exit
meenakshi@DESKTOP-2THHS59: ~$ cat code11.sh
while true
do
echo -n "Enter number: "
read num
if [ $num -lt 0 ]
then
echo "neg number entered exit"
break
fi
square=$((num * num))
echo "Square of $num is: $square"
done
meenakshi@DESKTOP-2THHS59: ~$
```

Part E

1 Fcfs

Q1] By Fcfs scheduling algorithm.

Process	Arrival Time	Burst time	Completion Time	TAT	Waiting Time
P1	0	5	5	5	0
P2	1	3	8	7	4
P3	2	6	14	12	6

→

Process	Arrival Time	Burst time	Completion Time
P1	0	5	5
P2	1	3	8
P3	2	6	14

Avg waiting time = $\frac{0+4+6}{3} = \frac{10}{3} = 3.33$

2.SJF

Q2 By SJF

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

→

Process	Arrival Time	Burst Time	Completion Time
P1	0	3	3
P3	2	1	4
P4	3	4	8
P2	1	5	13

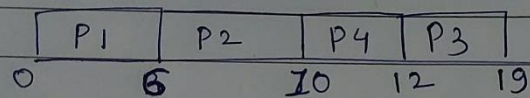
avg TAT = $\frac{3+12+2+5}{4} = \frac{22}{4} = 5.5$

3. priority Sceduling

③

Priority scheduling ..

Process	Arrival Time	Burst Time	Priority	CT	WT	TAT
P1	0	6 ✓	3 ✓	6	0	6
P2	1	4 ✓	1 ✓	10	5	9
P3	2	7 ✓	4	19	10	17
P4	3	2 ✓	2	12	7	9



$$\text{Avg waiting time} = \frac{0 + 5 + 0 + 7}{4} = \frac{22}{4} = 5.5$$

4 Round Robin Algorithm

Q4 Round Robin \rightarrow quantum = 2 units -

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

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

$$\text{Avg TAT} = \frac{10 + 13 + 4 + 10}{4} = \frac{37}{4} = 9.25$$

Q5. 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: Initially the parent process has value of $x=5$.

Then later fork() system call create a new child process and will contain value 5 and have separate memory space and it execute instruction $x = x+1$ and increments by 1

i.e 6 is final value of parent and child process and it will not affect each other as they have separate memory copies

So, the final value is 6.