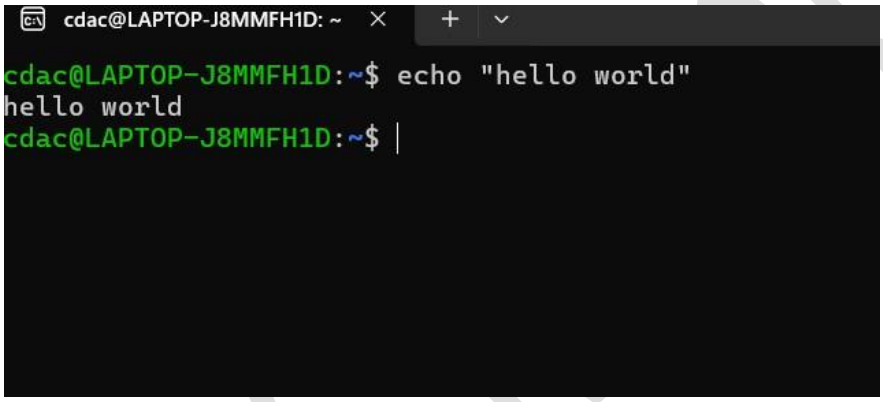
CDAC MUMBAI

Concepts of Operating System Assignment 2

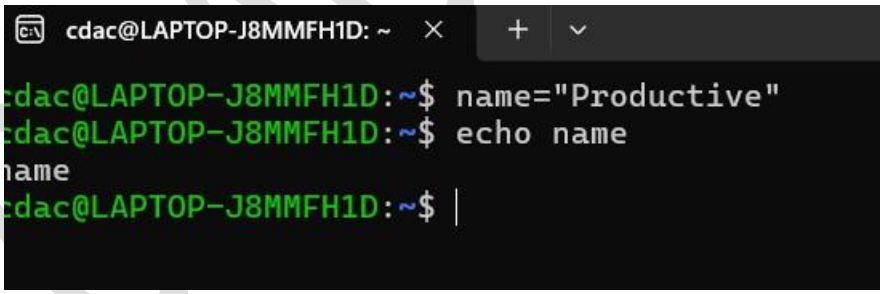
Part A

What will the following commands do?

- echo "Hello, World!"
  Ans : Prints the word “Hello, World!”



- name="Productive"
  Ans: Assigns the string “Productive” to the variable name



- touch file.txt
  Ans: Creates a file named file.txt

```
cdac@LAPTOP-J8MMFH1D:~$ touch file.txt
cdac@LAPTOP-J8MMFH1D:~$ ls -l
total 4
drwxr-xr-x 5 cdac cdac 4096 Aug 28 20:28 LinuxAssignment
-rw-r--r-- 1 cdac cdac    0 Aug 30 15:12 file.txt
cdac@LAPTOP-J8MMFH1D:~$
```

- ls -a
  Ans: ls -a shows the hidden files in the directory.

```
cdac@LAPTOP-J8MMFH1D:~$ ls -a
.   .bash_history  .bashrc  .local       .profile                  LinuxAssignment
..  .bash_logout   .cache   .motd_shown   .sudo_as_admin_successful file.txt
cdac@LAPTOP-J8MMFH1D:~$
```

- rm file.txt
  Ans: Removes the file named file.txt from the directory.

```
cdac@LAPTOP-J8MMFH1D:~$ ls -a
.   .bash_history  .bashrc  .local       .profile                  LinuxAssignment
..  .bash_logout   .cache   .motd_shown   .sudo_as_admin_successful file.txt
cdac@LAPTOP-J8MMFH1D:~$
```

- cp file1.txt file2.txt
  Ans: Copies the content of file1.txt to file2.txt.

```
cdac@LAPTOP-J8MMFH1D: ~
cdac@LAPTOP-J8MMFH1D:~$ nano file1.txt
cdac@LAPTOP-J8MMFH1D:~$ nano file2.txt
cdac@LAPTOP-J8MMFH1D:~$ cp file1.txt file2.txt
```

- mv file.txt /path/to/directory/
  Ans: Moves file.txt to the specific path and directory.

```
cdac@LAPTOP-J8MMFH1D:~$ touch report.txt
cdac@LAPTOP-J8MMFH1D:~$ mkdir documents
cdac@LAPTOP-J8MMFH1D:~$ mv report.txt documents/
cdac@LAPTOP-J8MMFH1D:~$ ls -l
total 20
drwxr-xr-x 5 cdac cdac 4096 Aug 28 20:28 LinuxAssignment
drwxr-xr-x 2 cdac cdac 4096 Aug 30 15:33 documents
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:22 file1.txt
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:23 file2.txt
-rw-r--r-- 1 cdac cdac   24 Aug 30 15:23 file2.txty
cdac@LAPTOP-J8MMFH1D:~$
```

- chmod 755 script.sh
  Ans: chmod 755 script.sh changes the permissions of owner, group and other of the file script.sh.

```
cdac@LAPTOP-J8MMFH1D:~$ nano script.sh
cdac@LAPTOP-J8MMFH1D:~$ chmod 755 script.sh
cdac@LAPTOP-J8MMFH1D:~$ ls -l
total 24
drwxr-xr-x 5 cdac cdac 4096 Aug 28 20:28 LinuxAssignment
drwxr-xr-x 2 cdac cdac 4096 Aug 30 15:33 documents
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:22 file1.txt
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:23 file2.txt
-rw-r--r-- 1 cdac cdac   24 Aug 30 15:23 file2.txty
-rwxr-xr-x 1 cdac cdac   84 Aug 30 15:38 script.sh
cdac@LAPTOP-J8MMFH1D:~$
```

- grep "pattern" file.txt
  Ans: Highlights the word “pattern” and prints the content of the file.txt.

```
cdac@LAPTOP-J8MMFH1D:~$ nano file.txt
cdac@LAPTOP-J8MMFH1D:~$ grep "pattern" file.txt
I am printing a pattern program.
cdac@LAPTOP-J8MMFH1D:~$
```

- kill PID
  Ans: The kill command will attempt to terminate the process gracefully. The process may handle the signal in a specific way, such as saving state or cleaning up resources.

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

Ans:
mkdir mydir: Creates the mydir directory.
cd mydir: Navigates into mydir.
touch file.txt: Creates an empty file.txt.
echo "Hello, World!" > file.txt: Writes "Hello, World!" into file.txt, replacing any existing content.
cat file.txt: Displays the contents of file.txt, which is "Hello, World!".

```
cdac@LAPTOP-J8MMFH1D:~$ mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt
Hello, World!
cdac@LAPTOP-J8MMFH1D:~/mydir$
```

- ls -l | grep ".txt"

Ans: Lists the type of files with the extension .txt.

```
cdac@LAPTOP-J8MMFH1D:~/mydir$ ls -l | grep ".txt"
-rw-r--r-- 1 cdac cdac 14 Aug 30 16:02 file.txt
cdac@LAPTOP-J8MMFH1D:~/mydir$
```

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

Ans: This command creates the two files file1.txt and file2.txt and sorts the unique values between them.

```
cdac@LAPTOP-J8MMFH1D:~/mydir$ touch file1.txt file2.txt
cdac@LAPTOP-J8MMFH1D:~/mydir$ cat file1.txt file2.txt | sort | uniq
cdac@LAPTOP-J8MMFH1D:~/mydir$
```

- ls -l | grep "^d"

Ans:
ls -l: Lists all files and directories in the current directory in a detailed format.
grep "^d": Filters the output to show only lines where the first character is d, indicating directories.

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

Ans: grep -r "pattern" /path/to/directory/: Recursively searches for the specified pattern in all files within the given directory and its subdirectories.

```
cdac@LAPTOP-J8MMFH1D:~$ grep -r "pattern" mydir/
mydir/Demo.text:And I am printing pattern programming
cdac@LAPTOP-J8MMFH1D:~$
```

- cat file1.txt file2.txt | sort | uniq –d
  Ans: concatenates the contents of file1.txt and file2.txt, sorts the combined lines, and then outputs only the duplicate lines.

```
cdac@LAPTOP-J8MMFH1D:~$ cat file1.txt file2.txt | sort | uniq
My name is sumit Deshmukh
Sumitttt DDeshhmukkh
cdac@LAPTOP-J8MMFH1D:~$
```

- chmod 644 file.txt
  Ans: The command `chmod 644 file.txt` sets the file permissions of `file.txt` to be readable and writable by the owner, and readable by the group and others.

```
cdac@LAPTOP-J8MMFH1D:~$ chmod 644 file.txt
cdac@LAPTOP-J8MMFH1D:~$ ls -l
total 32
drwxr-xr-x 5 cdac cdac 4096 Aug 28 20:28 LinuxAssignment
drwxr-xr-x 2 cdac cdac 4096 Aug 30 15:33 documents
-rw-r--r-- 1 cdac cdac   59 Aug 30 20:50 file.txt
-rw-r--r-- 1 cdac cdac   47 Aug 30 20:43 file1.txt
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:23 file2.txt
-rw-r--r-- 1 cdac cdac   24 Aug 30 15:23 file2.txty
drwxr-xr-x 2 cdac cdac 4096 Aug 30 16:21 mydir
-rwxr-xr-x 1 cdac cdac   84 Aug 30 15:38 script.sh
cdac@LAPTOP-J8MMFH1D:~$
```

- cp -r source_directory destination_directory
  Ans: This command is used to copy a directory and all of its contents, including subdirectories, from one location to another

- find /path/to/search -name "*.txt"
  Ans: This is used to find files with a .txt extension within a specified directory and its subdirectories

- chmod u+x file.txt
  Ans: This is used to change the permissions of a file. Here we give permission to owner to execute the file.

```
cdac@LAPTOP-J8MMFH1D:~$ chmod u+x file.txt
cdac@LAPTOP-J8MMFH1D:~$ ls -l
total 32
drwxr-xr-x 5 cdac cdac 4096 Aug 28 20:28 LinuxAssignment
drwxr-xr-x 2 cdac cdac 4096 Aug 30 15:33 documents
-rwxr--r-- 1 cdac cdac   59 Aug 30 20:50 file.txt
-rw-r--r-- 1 cdac cdac   47 Aug 30 20:43 file1.txt
-rw-r--r-- 1 cdac cdac   26 Aug 30 15:23 file2.txt
-rw-r--r-- 1 cdac cdac   24 Aug 30 15:23 file2.txty
drwxr-xr-x 2 cdac cdac 4096 Aug 30 16:21 mydir
-rwxr-xr-x 1 cdac cdac   84 Aug 30 15:38 script.sh
cdac@LAPTOP-J8MMFH1D:~$ |
```

- echo $PATH
  Ans: This command is used to displays the current value of the PATH environment variable. The PATH variable contains a list of directories that the shell searches through when you type a command.

```
cdac@LAPTOP-J8MMFH1D:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/usr/lib/wsl/lib:/mnt/c/Program Files (x86)/VMware/VMware Workstati
on/bin/:/mnt/c/Program Files/Common Files/Oracle/Java/javapath:/mnt/c/Program Files (x86)/Common Files/Oracle/Java/javapath:/mnt/c/Windows/system32:/mnt/c/W
indows:/mnt/c/Windows/System32/Wbem:/mnt/c/Windows/System32/WindowsPowerShell/v1.0/:/mnt/c/Windows/System32/OpenSSH/:/mnt/c/Program Files (x86)/NVIDIA Corpo
ration/PhysX/Common:/mnt/c/Program Files/NVIDIA Corporation/NVIDIA NvDLISR:/mnt/c/Program Files/Git/cmd:/mnt/c/WINDOWS/system32:/mnt/c/WINDOWS:/mnt/c/WINDOW
S/System32/Wbem:/mnt/c/WINDOWS/System32/WindowsPowerShell/v1.0/:/mnt/c/WINDOWS/System32/OpenSSH/:/mnt/c/Program Files/nodejs/:/mnt/c/Program Files/MySQL/MyS
QL Shell 8.0/bin/:/mnt/c/Users/Sumit/AppData/Local/Programs/Python/Python310/Scripts/:/mnt/c/Users/Sumit/AppData/Local/Programs/Python/Python310/:/mnt/c/Use
rs/Sumit/AppData/Local/Microsoft/WindowsApps:/mnt/c/Users/Sumit/AppData/Roaming/npm:/mnt/c/Users/Sumit/AppData/Local/Programs/Microsoft VS Code/bin:/snap/bi
n
cdac@LAPTOP-J8MMFH1D:~$ |
```

Part B

Identify True or False:

1. ls is used to list files and directories in a directory.
   Ans: True
2. mv is used to move files and directories.
   Ans: True
3. cd is used to copy files and directories.
   Ans: False
4. pwd stands for "print working directory" and displays the current directory.
   Ans: False
5. grep is used to search for patterns in files.
   Ans: True

6. chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.
   Ans: True
7. mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist.
   Ans: True
8. rm -rf file.txt deletes a file forcefully without confirmation.
   Ans: True

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.
   Ans: Incorrect: The correct command for changing file permissions is chmod, not chmodx. The command chmodx does not exist.

2. cpy is used to copy files and directories.
   Ans: Incorrect: The correct command for copying files and directories is cp, not cpy. The command cpy does not exist.

3. mkfile is used to create a new file.
   Ans: Incorrect: The correct command for creating a new file is touch or echo with redirection, not mkfile. The command mkfile does not exist in standard Unix/Linux environments.

4. catx is used to concatenate files.
   Ans: Incorrect: The correct command for concatenating files is cat, not catx. The command catx does not exist.

5. rn is used to rename files.
   Ans: Incorrect: The correct command for renaming files is mv, not rn. The command rn does not exist.

Part C

Question 1: Write a shell script that prints "Hello, World!" to the terminal.
Ans: c
dac@LAPTOP-J8MMFH1D:~$ echo "Hello, World"
Hello, World

```
cdac@LAPTOP-J8MMFH1D:~$ echo "Hello, World"
Hello, World
cdac@LAPTOP-J8MMFH1D:~$
```

Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.
Ans:
cdac@LAPTOP-J8MMFH1D:~$ name="CDAC Mumbai"
cdac@LAPTOP-J8MMFH1D:~$ echo $name
CDAC Mumbai

```
cdac@LAPTOP-J8MMFH1D:~$ echo "Hello, World"
Hello, World
cdac@LAPTOP-J8MMFH1D:~$ name="CDAC Mumbai"
cdac@LAPTOP-J8MMFH1D:~$ echo $name
CDAC Mumbai
cdac@LAPTOP-J8MMFH1D:~$
```

Question 3: Write a shell script that takes a number as input from the user and prints it.
Ans:
#!/bin/bash
echo "Please enter a number:"
read number
echo "You entered: $number"

```
cdac@LAPTOP-J8MMFH1D:~$ nano input
cdac@LAPTOP-J8MMFH1D:~$ bash input
Please enter a number:
7
You entered: 7
```

Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.
Ans:
#!/bin/bash
echo "Please enter the first number:"
read num1
echo "Please enter the second number:"
read num2

sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is $sum."

```
cdac@LAPTOP-J8MMFH1D:~$ nano input
cdac@LAPTOP-J8MMFH1D:~$ bash input
Please enter the first number:
3
Please enter the second number:
5
The sum of 3 and 5 is 8.
cdac@LAPTOP-J8MMFH1D:~$
```

Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".
Ans:
#!/bin/bash
read -p "Enter a number: " number
if [$((number % 2)) -eq 0]; then
** echo "Even"**
else
** echo "Odd"**
fi

```
cdac@LAPTOP-J8MMFH1D:~$ bash input
Enter a number: 4
Even
cdac@LAPTOP-J8MMFH1D:~$ bash input
Enter a number: 7
Odd
cdac@LAPTOP-J8MMFH1D:~$
```

Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.
Ans:
#!/bin/bash
a=0
for a in 1 2 3 4 5
do
echo $a

done

```
cdac@LAPTOP-J8MMFH1D:~$ bash input
1
2
3
4
5
```

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

```
#!/bin/bash

a=0
while [ $a -lt 5 ]
do
echo $a
a=$((a + 1))
done
```

```
cdac@LAPTOP-J8MMFH1D:~$ nano input
cdac@LAPTOP-J8MMFH1D:~$ bash input
0
1
2
3
4
cdac@LAPTOP-J8MMFH1D:~$
```

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".
Ans:

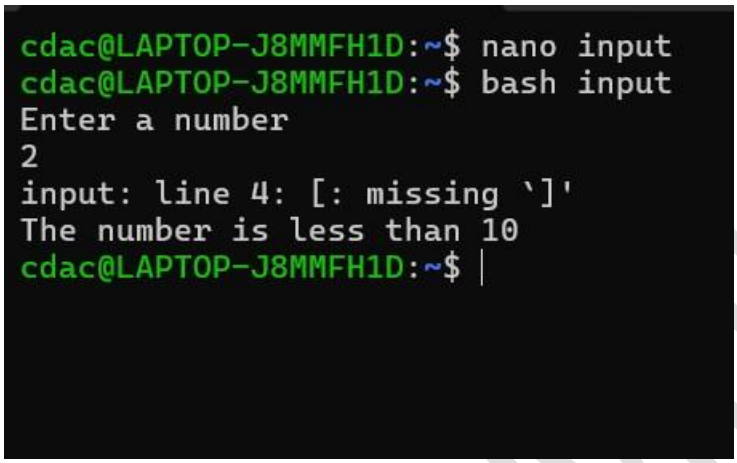
```
#!/bin/bash
if [ -f "file.txt" ];
 then
echo "File exists"
else
echo "File does not exist"
fi
```

```
cdac@LAPTOP-J8MMFH1D:~$ nano input
cdac@LAPTOP-J8MMFH1D:~$ bash input
File exists
cdac@LAPTOP-J8MMFH1D:~$
```

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.

Ans:

```
#!/bin/bash
echo "Enter a number"
read number
if [ "$number" -gt 10];
then
echo "The number is greater than 10"
else
echo "The number is less than 10"
fi
```



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.

Ans:

```
#!/bin/bash
max=5
for (( i=1; i<=max; i++ )); do
 for (( j=1; j<=max; j++ )); do
echo -n "$((i * j)) "
done
echo
```

done

```
cdac@LAPTOP-J8MMFH1D:~$ nano input
cdac@LAPTOP-J8MMFH1D:~$ bash input
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
cdac@LAPTOP-J8MMFH1D:~$
```

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.

Ans:

```
#!/bin/bash
while true;
do
read -p "Enter a number (negative to quit): " number
if [ "$number" -lt 0 ];
then
echo "Negative number entered. Exiting..."
break
fi
if [ "$number" -ge 0 ]; then
square=$((number * number))
echo "The square of $number is $square"
else
echo "Invalid input. Please enter a valid number."
fi
done
```

```
cdac@LAPTOP-J8MMFH1D:~$ bash input
Enter a number (negative to quit): -9
Negative number entered. Exiting...
cdac@LAPTOP-J8MMFH1D:~$ bash input
Enter a number (negative to quit): 2
The square of 2 is 4
Enter a number (negative to quit):
```

Part D

Common Interview Questions (Must know)

1. What is an operating system, and what are its primary functions?
2. Explain the difference between process and thread.
3. What is virtual memory, and how does it work?
4. Describe the difference between multiprogramming, multitasking, and multiprocessing.
5. What is a file system, and what are its components?
6. What is a deadlock, and how can it be prevented?
7. Explain the difference between a kernel and a shell.
8. What is CPU scheduling, and why is it important?
9. How does a system call work?
10. What is the purpose of device drivers in an operating system?
11. Explain the role of the page table in virtual memory management.
12. What is thrashing, and how can it be avoided?
13. Describe the concept of a semaphore and its use in synchronization.
14. How does an operating system handle process synchronization?
15. What is the purpose of an interrupt in operating systems?
16. Explain the concept of a file descriptor.
17. How does a system recover from a system crash?
18. Describe the difference between a monolithic kernel and a microkernel.
19. What is the difference between internal and external fragmentation?
20. How does an operating system manage I/O operations?
21. Explain the difference between preemptive and non-preemptive scheduling.
22. What is round-robin scheduling, and how does it work?
23. Describe the priority scheduling algorithm. How is priority assigned to processes?
24. What is the shortest job next (SJN) scheduling algorithm, and when is it used?
25. Explain the concept of multilevel queue scheduling.
26. What is a process control block (PCB), and what information does it contain?
27. Describe the process state diagram and the transitions between different process states.
28. How does a process communicate with another process in an operating system?
29. What is process synchronization, and why is it important?
30. Explain the concept of a zombie process and how it is created.
31. Describe the difference between internal fragmentation and external fragmentation.
32. What is demand paging, and how does it improve memory management efficiency?
33. Explain the role of the page table in virtual memory management.
34. How does a memory management unit (MMU) work?
35. What is thrashing, and how can it be avoided in virtual memory systems?
36. What is a system call, and how does it facilitate communication between user programs and the operating system?
37. Describe the difference between a monolithic kernel and a microkernel.
38. How does an operating system handle I/O operations?
39. Explain the concept of a race condition and how it can be prevented.

40. Describe the role of device drivers in an operating system.
41. What is a zombie process, and how does it occur? How can a zombie process be prevented?
42. Explain the concept of an orphan process. How does an operating system handle orphan processes?
43. What is the relationship between a parent process and a child process in the context of process management?
44. How does the fork() system call work in creating a new process in Unix-like operating systems?
45. Describe how a parent process can wait for a child process to finish execution.
46. What is the significance of the exit status of a child process in the wait() system call?
47. How can a parent process terminate a child process in Unix-like operating systems?
48. Explain the difference between a process group and a session in Unix-like operating systems.
49. Describe how the exec() family of functions is used to replace the current process image with a new one.
50. What is the purpose of the waitpid() system call in process management? How does it differ from wait()?
51. How does process termination occur in Unix-like operating systems?
52. What is the role of the long-term scheduler in the process scheduling hierarchy? How does it influence the degree of multiprogramming in an operating system?
53. How does the short-term scheduler differ from the long-term and medium-term schedulers in terms of frequency of execution and the scope of its decisions?
54. Describe a scenario where the medium-term scheduler would be invoked and explain how it helps manage system resources more efficiently.

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.

Ans:

PART E

1.

Process	AT	BT	CT	TAT	WT
P1	0	5	5	5	0
P2	1	3	8	7	4
P3	2	6	16	14	8

Solution : (First come first Served)

Gantt Chart - | P1 | P2 | P3 |
0 5 8 16

Avg W.T = $\frac{8+4+0}{3} = 4$

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:

2.

Process	AT	BT	CT	TAT	WT
P1	0	3	3	3	0
P2	1	5	13	12	7
P3	2	1	4	2	1
P4	3	4	8	5	1

Solution :- (Shortest Job First)

Gantt chart -

P1	P3	P4	P2

0 3 4 8 13

Avg. TAT = $\frac{3+12+2+5}{4} = 5.5$

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:

3.

Process	Priority	AT	BT	CT	TAT	WT
P1	3	0	6	6	6	0
P2	1	1	4	10	9	5
P3	4	2	7	16	14	7
P4	2	3	2	12	9	7

Solution :- (Priority scheduling)

Gantt Chart: | P1 | P2 | P4 | P3 |

0 6 10 12 16

$$\text{Avg W.T} = \frac{0+5+7+7}{4} = 4.75$$

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:

4.

Process	AT	BT	CT	TAT	WT
P1	0	~~4~~ 0	10	10	6
P2	1	~~5~~ 3	14	13	8
P3	2	~~2~~ 0	6	4	2
P4	3	~~3~~ 0	13	10	7

Solution :- (Round Robin scheduling)

Gantt chart:

P1	P2	P3	P4	P1	P2	P4	P2

0 2 4 6 8 10 12 13 14

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

.

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:

```
#!/bin/bash

x=5  # Initialize x with value 5

(
   # Child process block (subshell)
   x=$((x + 1))  # Increment x by 1 in the child process
   echo "Child process: x = $x"
) &

# Parent process block
x=$((x + 1))  # Increment x by 1 in the parent process
echo "Parent process: x = $x"

# Wait for the child process to finish
```

Wait

```
cdac@LAPTOP-J8MMFH1D:~$ nano Prime
cdac@LAPTOP-J8MMFH1D:~$ bash Prime
Parent process: x = 6
Child process: x = 6
cdac@LAPTOP-J8MMFH1D:~$
```

Submission Guidelines:

- Document each step of your solution and any challenges faced.
- Upload it on your GitHub repository

Additional Tips:

- Experiment with different options and parameters of each command to explore their functionalities.
- This assignment is tailored to align with interview expectations, CCEE standards, and industry demands.
- If you complete this then your preparation will be skyrocketed.