Shubhada Aher_JH

CDAC MUMBAI

Concepts of Operating System

Assignment 2

Part A

What will the following commands do?

echo "Hello, World!" - Output is Hello, World on the terminal

name="Productive" - Output is Productive

touch file.txt - Creating a empty file with name file.txt

ls -a - Lists all files and directories in the current directory, including hidden files

rm file.txt - Deletes file names file.txt

cp file1.txt file2.txt - copies the content from file1.txt to file2 .txt

🕐 mv file.txt /path/to/directory/ - Moves the file file.txt to the specified directory /path/to/directory/

🕐 chmod 755 script.sh - User has read ,write and execute permissions . Group and others has read and excute only

🕐 grep "pattern" file.txt - Searches for lines in file.txt that contain the string "pattern" and prints those lines to the standard output.

🕐 kill PID -Sends a termination signal to the process with the process ID (PID)

🕐 mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt - creates mydir directory , changes current directory to mkdir ,creating empty file.txt file then add Hello World in file,txt by redirections then display the content

🕐 ls -l | grep ".txt" - Lists files and directories in long format (ls -l) and pipes the output to grep, which filters and displays only the lines containing ".txt"

🕐 cat file1.txt file2.txt | sort | uniq - mergeing 2 files , sorts them and remove dublicates

🕐 ls -l | grep "^d" -Lists files and directories in long format (ls -l) and pipes the output to grep. grep filters and displays only the lines that start with "d" (^d), which indicates directories.

🕐 grep -r "pattern" /path/to/directory/ -Recursively searches for the string "pattern" in all files within the directory /path/to/directory/ and its subdirectories, and prints the matching lines along with the file names.

🕐 cat file1.txt file2.txt | sort | uniq –d -Merge file1.txt and file2.txt ,sort the lines then display the lines that are dublicated

🕐 chmod 644 file.txt - file permission to owner -read n write , to group & other - read only

🕐 cp -r source_directory destination_directory - Recursively copies the source_directory and all its contents to destination_directory.

🕐 find /path/to/search -name "*.txt" -finding a file which extension is .txt

🕐 chmod u+x file.txt - execution is only done in user in the file.txt

🕐 echo $PATH -Prints the value of the PATH environment variable, which is a colon-separated list of directories where the shell searches for executable 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 -- its used to change the directory

4. pwd stands for "print working directory" and displays the current directory. **FALSE -- present working directory**

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.
FALSE -- -r is used for deleting directories ,not files

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.

Ans: **chmod command**

2. cpy is used to copy files and directories.

Ans: **cp command**

3. mkfile is used to create a new file.

Ans: **touch command** is used for creating new file

4. catx is used to concatenate files.

Ans: **cat command** for concatenate file

5. rn is used to rename files.

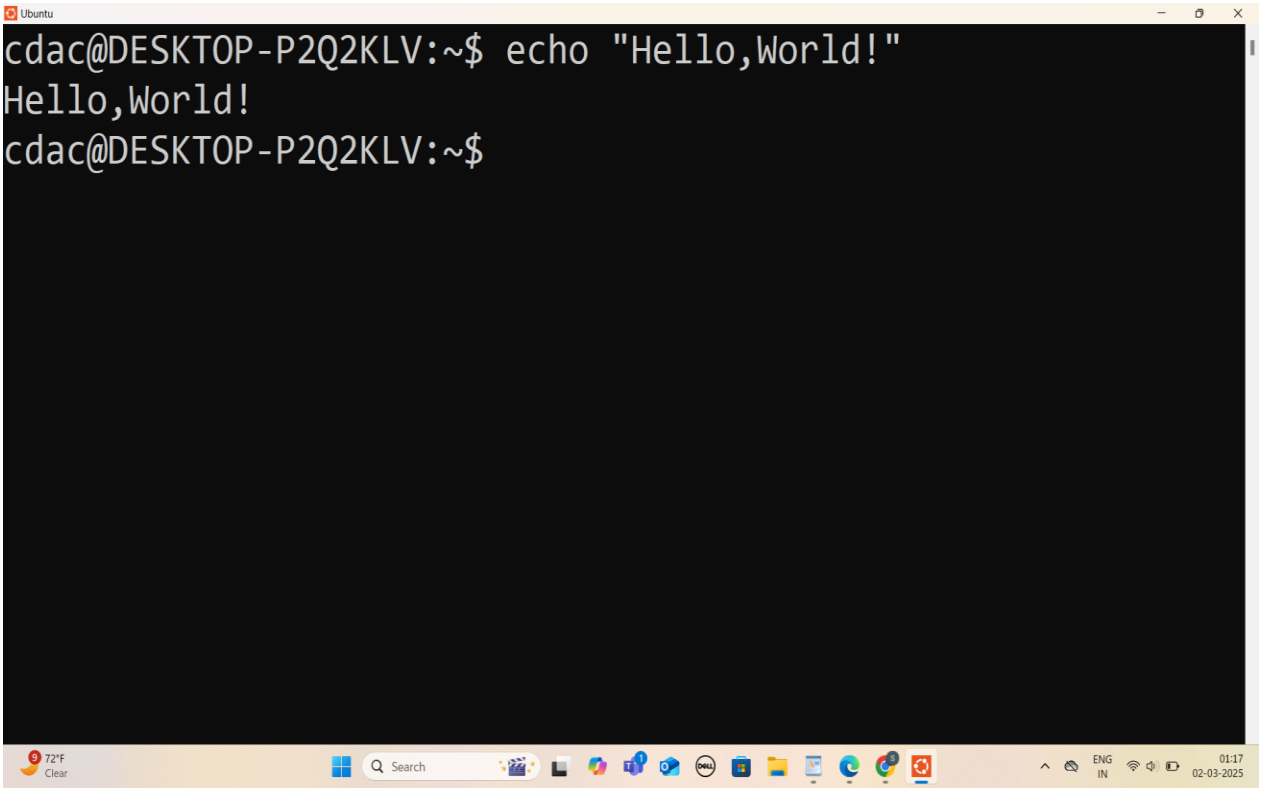
Ans: **mv command is used to rename files** when 2 files names are passed as arguments.

Part C

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

ANS: cdac@DESKTOP-P2Q2KLV:~$ echo "Hello,World!"

Hello,World!



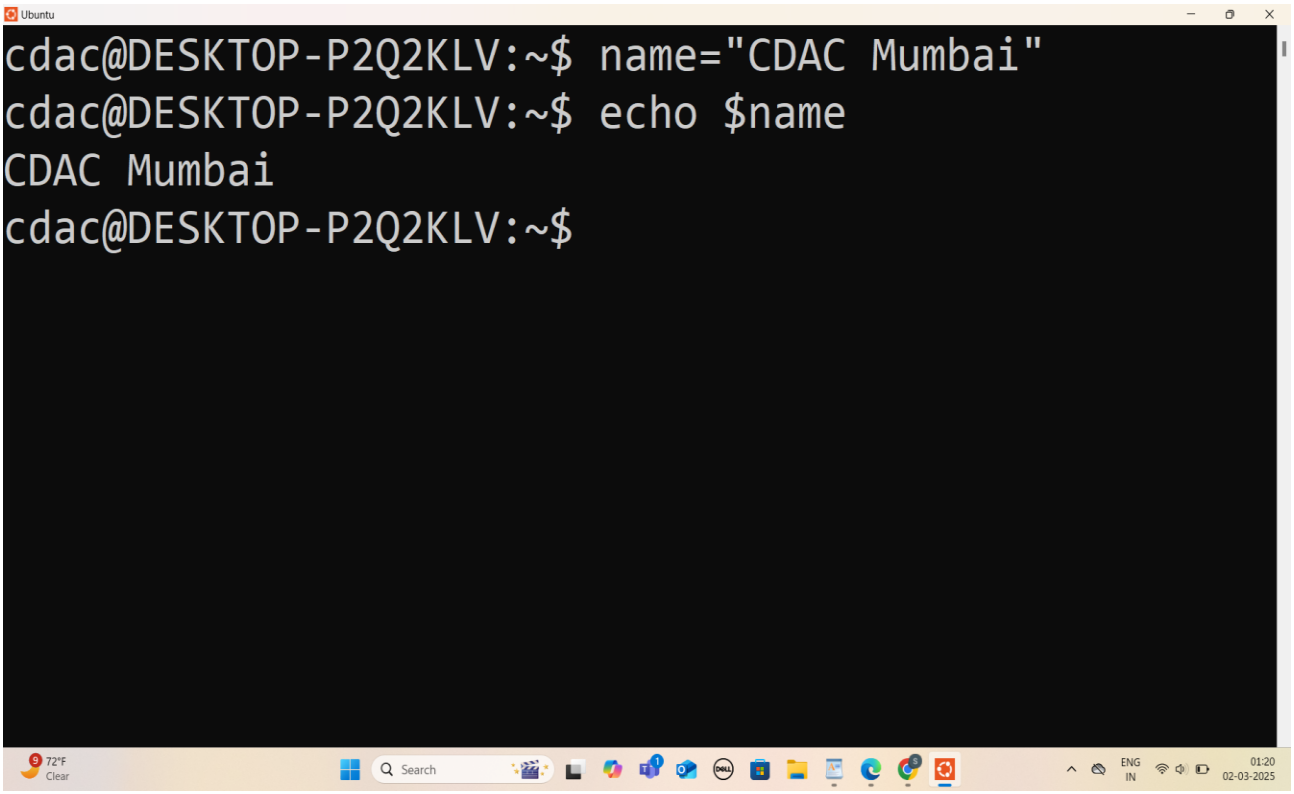
Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

ANS:

cdac@DESKTOP-P2Q2KLV:~$ name="CDAC Mumbai"

cdac@DESKTOP-P2Q2KLV:~$ echo $name

CDAC Mumbai



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

ANS:

cdac@DESKTOP-P2Q2KLV:~$ nano abc.txt

cdac@DESKTOP-P2Q2KLV:~$ cat abc.txt

echo "Enter a Number?"

read Number

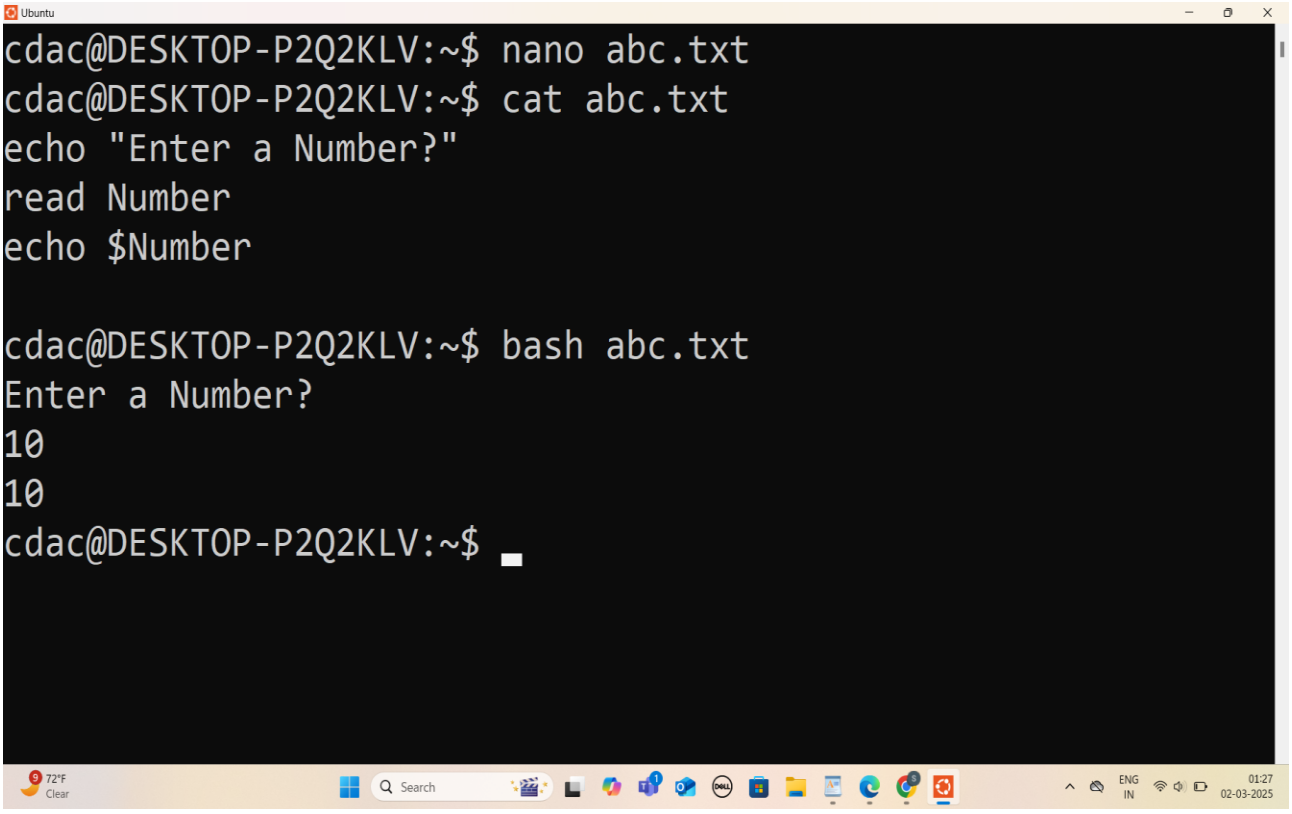
echo $Number

cdac@DESKTOP-P2Q2KLV:~$ bash abc.txt

Enter a Number?

10

10



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

result.

```
cdac@DESKTOP-P2Q2KLV:~$ nano file1.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file1.txt
echo "Enter a Number1"
read number1

echo "Enter a Number2"
read number2

sum=`expr $number1 + $number2`
echo sum of $number1 and $number2 is $sum

cdac@DESKTOP-P2Q2KLV:~$ bash file1.txt
Enter a Number1
5
Enter a Number2
3
sum of 5 and 3 is 8
cdac@DESKTOP-P2Q2KLV:~$
```

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

prints "Odd".

```
cdac@DESKTOP-P2Q2KLV:~$ nano file2.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file2.txt
echo "Enter a number1"
read number1

if [ `expr $number1 % 2` -eq 0 ]
then
    echo "$number1 is even"
else
    echo "$number1 is odd"
fi
cdac@DESKTOP-P2Q2KLV:~$ bash file2.txt
Enter a number1
5
5 is odd
cdac@DESKTOP-P2Q2KLV:~$
```

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

```
cdac@DESKTOP-P2Q2KLV:~$ nano file3.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file3.txt

for i in 1,2,3,4,5
do
   echo $i
done
cdac@DESKTOP-P2Q2KLV:~$ bash file3.txt
1,2,3,4,5
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
```

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

```
cdac@DESKTOP-P2Q2KLV:~$ nano file4.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file4.txt
a=1
while [ $a -lt 6 ]
do
   echo $a
a=`expr $a + 1`
done

cdac@DESKTOP-P2Q2KLV:~$ bash file4.txt
1
2
3
4
5
cdac@DESKTOP-P2Q2KLV:~$
```

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".

```
cdac@DESKTOP-P2Q2KLV:~$ nano file.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file.txt
if [ -e file.txt ]
then
    echo "File exists"
else
    echo "File doesn't exists"
fi
cdac@DESKTOP-P2Q2KLV:~$ bash file.txt
File exists
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
```

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.

```
cdac@DESKTOP-P2Q2KLV:~$ nano file11.txt
cdac@DESKTOP-P2Q2KLV:~$ cat file11.txt
echo "Enter a number"
read number

if [ $number -gt 10 ]
then
    echo "$number is greater than 10"
else
    if [ $number -eq 10 ]
    then
        echo "$number is equal"
    else
        echo "$number is less than 10"
    fi
fi

cdac@DESKTOP-P2Q2KLV:~$ bash file11.txt
Enter a number
4
4 is less than 10
cdac@DESKTOP-P2Q2KLV:~$
```

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.

```
cdac@DESKTOP-P2Q2KLV:~$ nano filee.txt
cdac@DESKTOP-P2Q2KLV:~$ cat filee.txt
for i in {1..5}
do
   for j in {1..5}
   do
      result=`expr $i \* $j`
      echo -n "$result"
   done
   echo
done
cdac@DESKTOP-P2Q2KLV:~$ bash filee.txt
12345
246810
3691215
48121620
510152025
cdac@DESKTOP-P2Q2KLV:~$
cdac@DESKTOP-P2Q2KLV:~$
```

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.

```
cdac@DESKTOP-P2Q2KLV:~$ nano ab.txt
cdac@DESKTOP-P2Q2KLV:~$ cat ab.txt
while [ true ]
do
   echo "enter a number"
   read number
   if [ $number -lt 0 ]
   then
       break
   fi
done
echo "program terminated"

cdac@DESKTOP-P2Q2KLV:~$ bash ab.txt
enter a number
4
enter a number
4
enter a number
5
enter a number
66
enter a number
-1
program terminated
cdac@DESKTOP-P2Q2KLV:~$
```

Part E

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

| Process | Arrival Time | Burst Time |

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

| P1

|

|

| 6

|

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

scheduling.

| 0

⇒ waiting time = Response time − Arrival time

Process	Arrival time	Burst time	Response time	waiting Time
P1	0	5	0	0-0 = 0
P2	1	3	5	5-1 = 4
P3	2	6	8	8-2 = 6
				Avg = 3.3

$$Avg = \frac{10}{3} = 3.3$$

Avg Waiting Time = 3.3

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

| Process | Arrival Time | Burst Time |

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

| P1

| 3

| P2

| P3

| P4

| 1

| 2

| 3

| 5

| 1

| 4

|

|

|

|

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

process	Arrival time	Burst time	Response Time	waiting time	TAT
P1	0	3	2	2-0 = 0	3+0 = 3
P2	1	5	6	6-1 = 5	5+5 = 10
P3	2	1	0	0-2 = 2	1+2 = 3
P4	3	4	5	5-3 = 2	4+2 = 6
					= 22

Gantt diagram	P3	P1	P4	P2
	0	2	5	6

TAT = 22

Avg TAT = $\frac{22}{4}$

$\boxed{\text{Avg TAT} = 5.5}$

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.

process	Arrival time	Burst time	Response Time	waiting time	TAT
P1	0	3	2	2-0 = 0	3+0 = 3
P2	1	5	6	6-1 = 5	5+5 = 10
P3	2	1	0	0-2 = 2	1+2 = 3
P4	3	4	5	5-3 = 2	4+2 = 6
					= 22

Gantt diagram	P3	P1	P4	P2
	0	2	5	6

TAT = 22

Avg TAT = $\frac{22}{4}$

$\boxed{\text{Avg TAT} = 5.5}$

4. Consider the following processes with arrival times and burst times, and the time quantum for

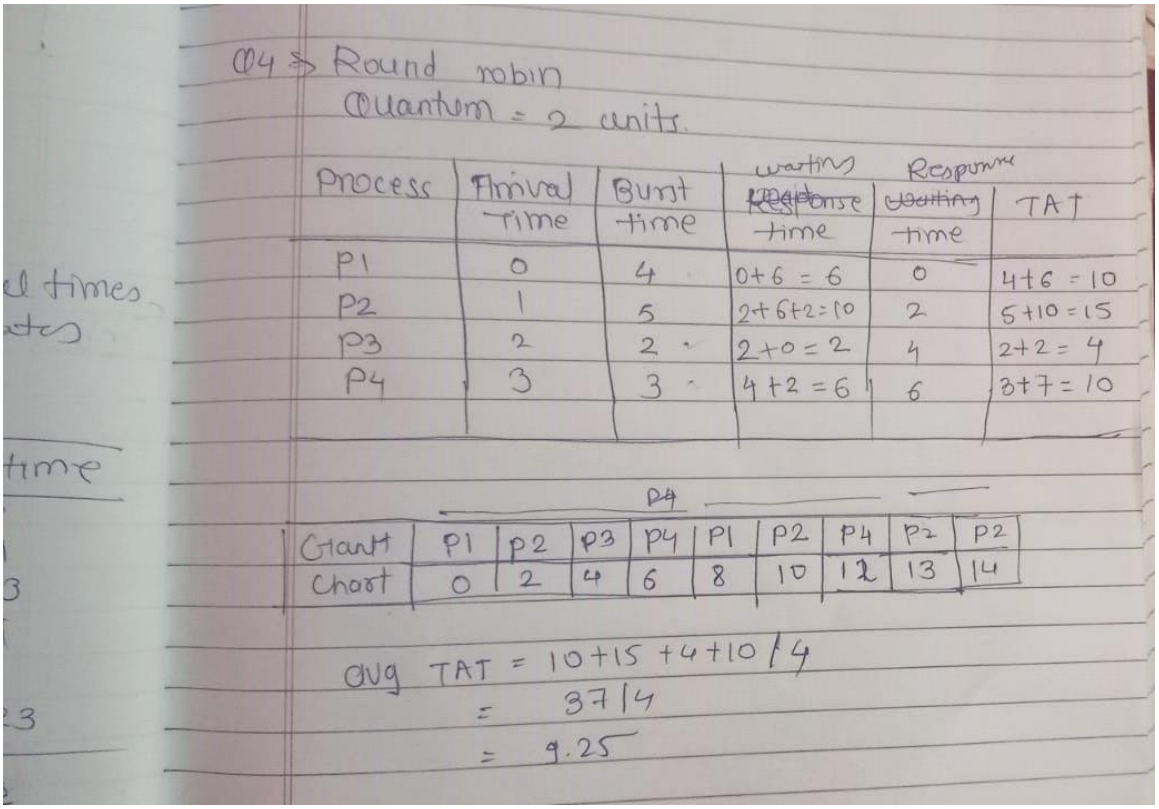
Round Robin scheduling is 2 units:

| 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.



Q4 ⇒ Round robin
Quantum = 2 units.

Process	Arrival Time	Burst time	~~Response~~ waiting time	~~Waiting~~ Response time	TAT
P1	0	4	0+6 = 6	0	4+6 = 10
P2	1	5	2+6+2=10	2	5+10 = 15
P3	2	2	2+0 = 2	4	2+2 = 4
P4	3	3	4+2 = 6	6	3+7 = 10

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

Avg TAT = 10+15+4+10/4
= 37/4
= 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.

o Before forking, the parent has a variable x = 5. After the

fork, both the parent and child have separate copies of

x, still equal to 5.

o Each process then increments x by 1, so both the parent

and child have x = 6, but in their own separate memory.

o In parent process, x=6. In child process, x=6