CDAC MUMBAI

**Concepts of Operating System Assignment 2**

**Part A**

## What will the following commands do?

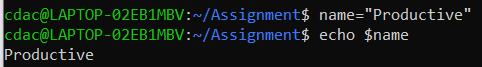
* echo "Hello, World!"

Ans : Prints the text "Hello, World!"



* name="Productive"

Ans: Assigns to the string "Productive" to the variable name



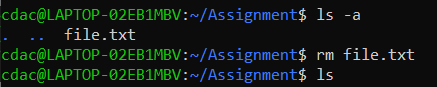
* touch file.txt

Ans : touch is use to create a file.



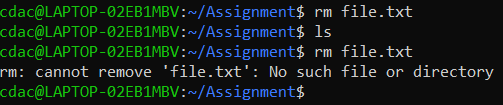
* ls -a

Ans: Lists all files and directories in the current directory, including hidden files (those starting with dot.)



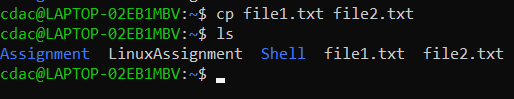
* rm file.txt

Ans: Deletes the file named file.txt



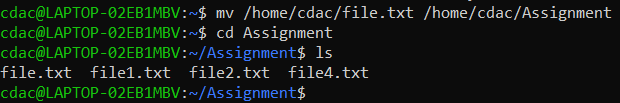
* cp file1.txt file2.txt

Ans: Command used to Copy file1.txt to file2.txt. If file2.txt exists, it will be overwritten the existing file.



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

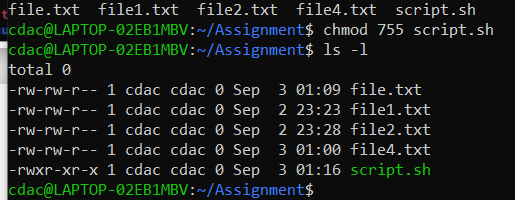
Ans: Moves file.txt to the specified directory



* chmod 755 script.sh

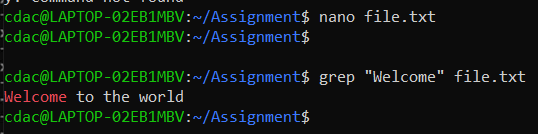
Ans: The given command Changes the permissions of file.txt to 755, giving the owner full

read, write, and execute permissions, and giving others read and execute permissions.



* grep "pattern" file.txt

Ans: Grep command searches for the string of "characters" in file.txt and displays all matching lines.



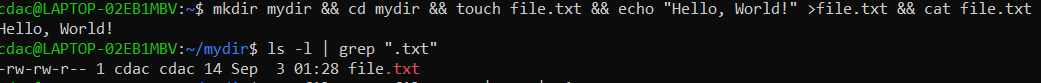
* kill PID

Ans : Terminates the process with the specified Process ID (PID)



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

Ans: The series of instructions concatenated with &&, mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt, guarantees that each command is executed only in the event that the preceding one is run.



* ls -l | grep ".txt"

Ans: Lists files in the current directory in long format (-l) and filters the output to show only those with .txt in their names.



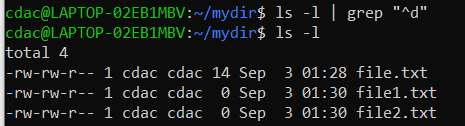
* cat file1.txt file2.txt | sort | uniq

Ans: Concatenates file1.txt and file2.txt, sorts the combined output, and removes duplicate lines



* ls -l | grep "^d"

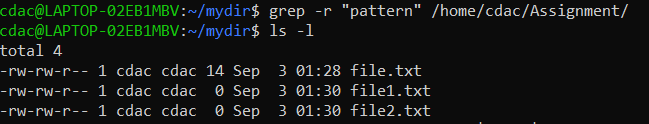
Ans: Uses the lengthy format to list files in the current directory, then filters the output to display only directories (denoted by a "d" in the permissions column).



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

Ans: The command grep -r "pattern" /path/to/directory/ is used to search for a specific text pattern within all files in a directory and its subdirectories.

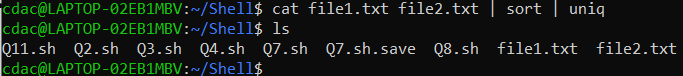
Command: grep -r "pattern" /home/cdac/Assignment/



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

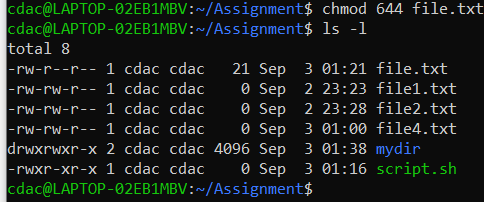
Ans: Concatenates file1.txt and file2.txt, sorts the combined output, and displays only duplicate lines

with the help of sort and uniq commands.



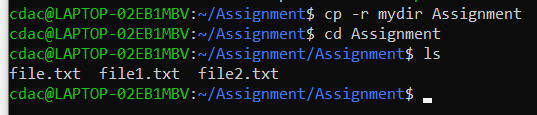
* chmod 644 file.txt

Ans: the Chmod 644 Changes the permissions of file.txt to 644, giving the owner read and write permissions, and giving others read-only permissions



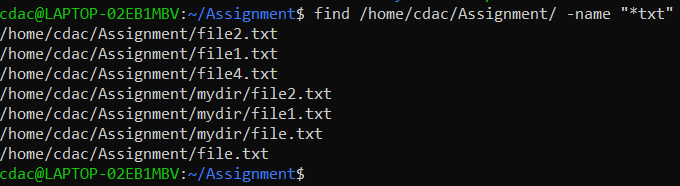
* cp -r source\_directory destination\_directory

Ans: The command "cp -r source\_directory destination\_directory" copies the whole source directory along with all of its contents, then stores the copied copy in destination\_directory.



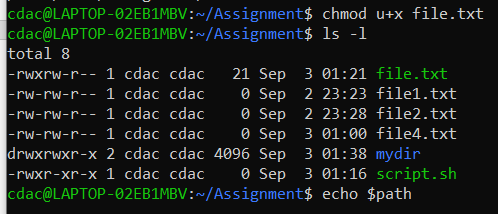
* find /path/to/search -name "\*.txt"

Ans: Searches for all files with .txt extension within the specified directory and its subdirectories.



* chmod u+x file.txt

Ans: Adds execute permission for the (user) of file.txt



* echo $PATH

Ans: When we used this command, echo $PATH displayed the list of folders (or path) that your computer searches for software to launch.

# 

# Part B

## Identify True or False:

1. **ls** is used to list files and directories in a directory.

**Ans**: True

1. **mv** is used to move files and directories.

**Ans**: True

1. **cd** is used to copy files and directories.

**Ans:** False

1. **pwd** stands for "print working directory" and displays the current directory.

**Ans**: True

1. **grep** is used to search for patterns in files.

**Ans:** True

1. **chmod 755 file.txt** gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.

**Ans:** True

1. **mkdir -p directory1/directory2** creates nested directories, creating directory2 inside directory1 if directory1 does not exist.

**Ans:** True

1. **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:** chmod

1. **cpy** is used to copy files and directories.

**Ans:** cp

1. **mkfile** is used to create a new file.

**Ans:** touch

1. **catx** is used to concatenate files.

**Ans:** cat

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

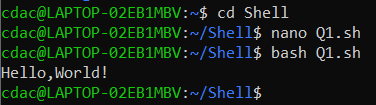
**Ans:** mv

# Part C

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

**Command:** nano Q1.sh

bash Q1.sh

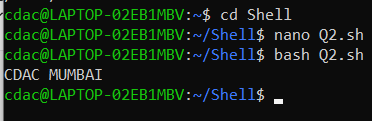


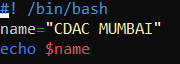


**Question 2:** Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

**Command**: nano Q2.sh

bash Q2.sh





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

Command: nano Q3.sh

bash Q3.sh

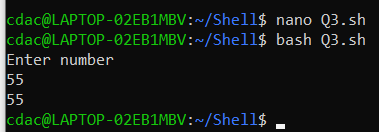
This command used to create file and for shell scripting

#!/bin/bash

echo "Enter number"

read number

echo $number



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

Command: nano Q4.sh

bash Q4.sh

Script that perform addition of two numbers

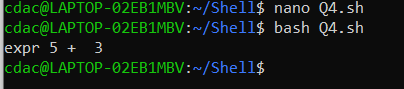
#!/bin/nash

a=4

b=7

c=$[a+b]

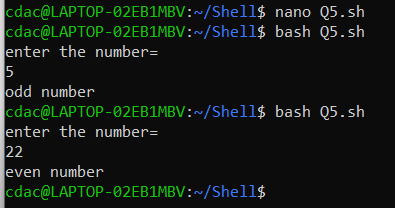
echo $c



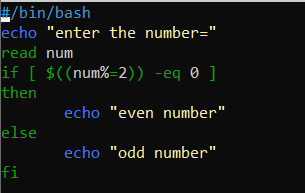
**Question 5:** Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".

Command: nano Q5.sh

bash Q5.sh



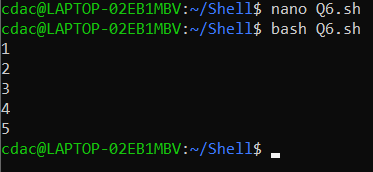
Shell script foe even and odd



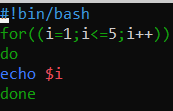
**Question 6:** Write a shell script that uses a for loop to print numbers from 1 to 5.

Command: nano Q6.sh

bash Q6.sh



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.

Command: nano Q7.sh

bash Q7.sh

In shell script while loop is used to print from number from 1 to 5

#!/bin/bash

i=1

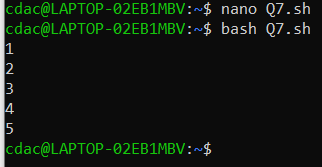
while [ $i -le 5 ]

do

echo $i

i=$(($i+1))

done



**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".

Command: nano Q8.sh

bash Q8.sh

Script:

#!/bin/bash

f="h.sh"

if [ -f "$f" ]

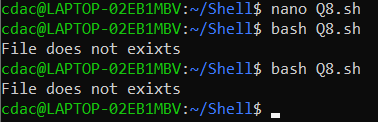
then

echo "File exixts"

else

echo "File does not exixts"

fi

****

**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.

Command: nano Q9.sh

bash Q9.sh

Shell script:

#!/bin/bash

echo "Enter a number:"

read number

if [ $number -gt 10 ]

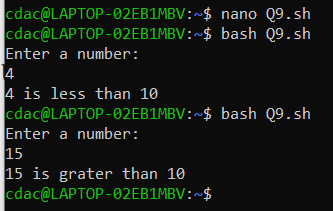
then

echo "$number is grater than 10"

else

echo "$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.

Command: nano Q10.sh

bash Q10.sh

Shell Script:

#!/bin/bash

for ((i=1; i<=5; i++))

do

for ((j=1; j<=5; j++))

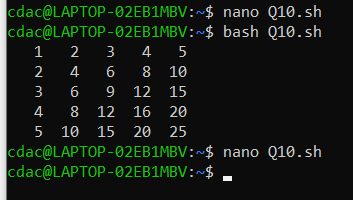
do

printf "%4d" "$((i\*j))"

done

echo

done



**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.

Command: nano Q11.sh

bash Q11.sh

Shell script:

#!/bin/bash

echo "Enter a number"

read num

while [ "$num" -gt 0 ]

do

sq=$((num \* num))

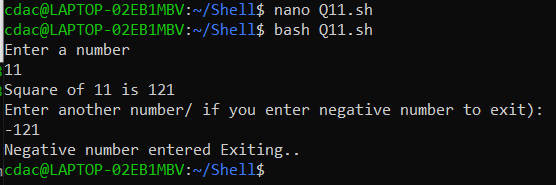
echo "Square of $num is $sq"

echo "Enter another number/ if you enter negative number to exit):"

read num

done

echo "Negative number entered Exiting.."



# Part E

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

| Process | Arrival Time | Burst Time |

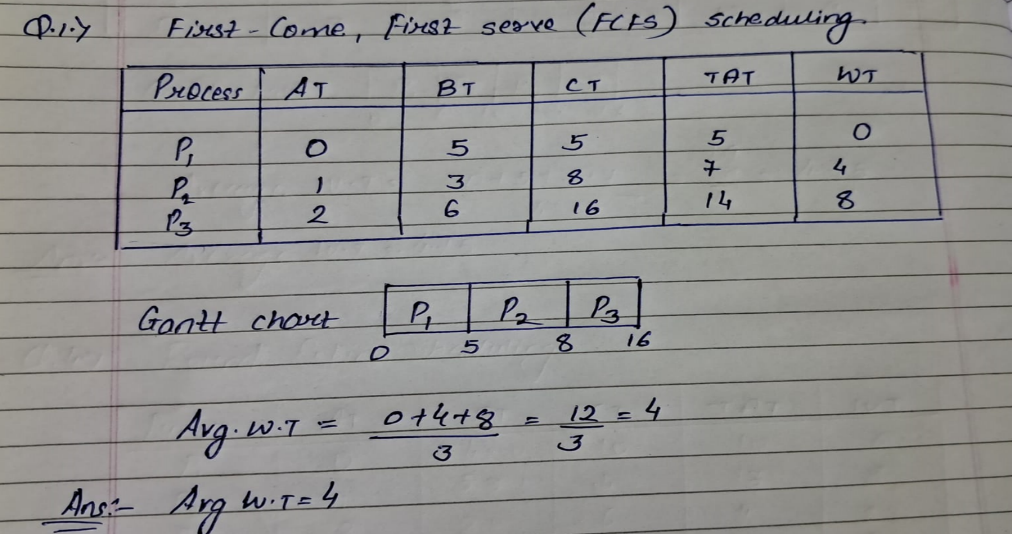
| | | |

| P1 | 0 | 5 |

| P2 | 1 | 3 |

| P3 | 2 | 6 |

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



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

| Process | Arrival Time | Burst Time |

| | | |

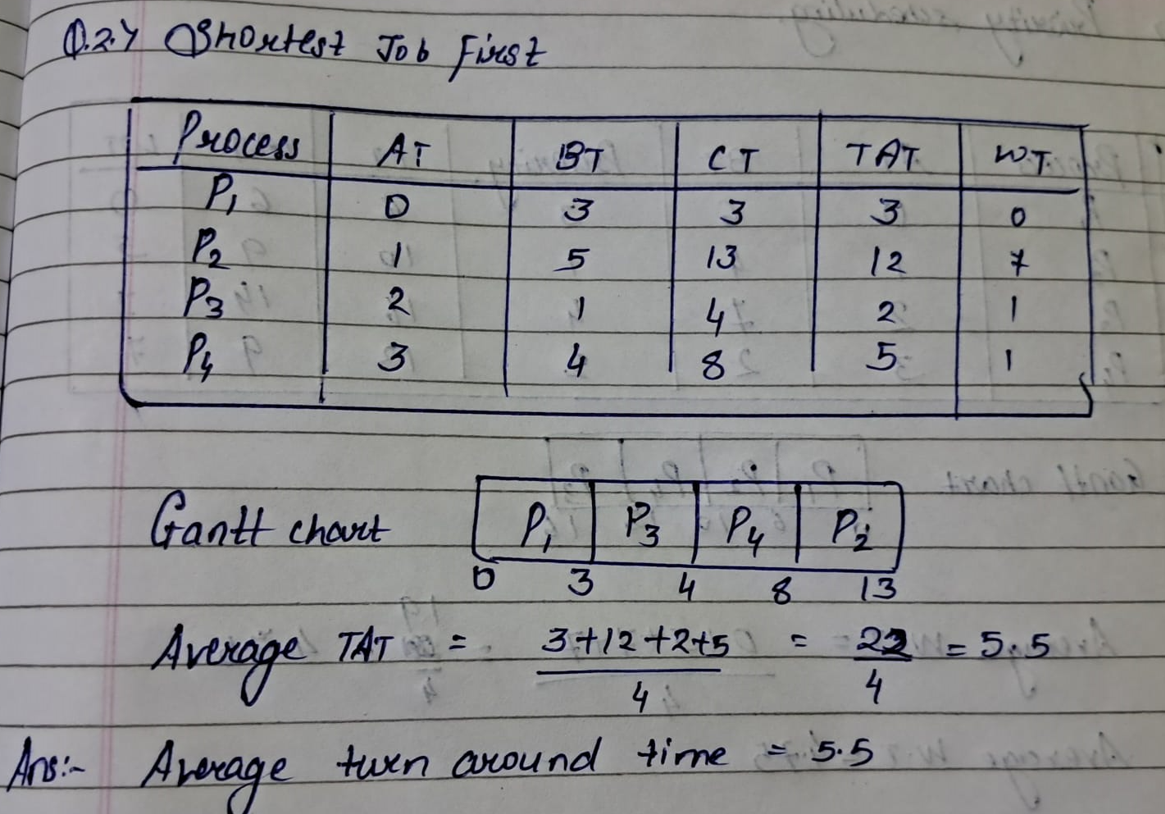
| P1 | 0 | 3 |

| P2 | 1 | 5 |

| P3 | 2 | 1 |

| P4 | 3 | 4 |

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



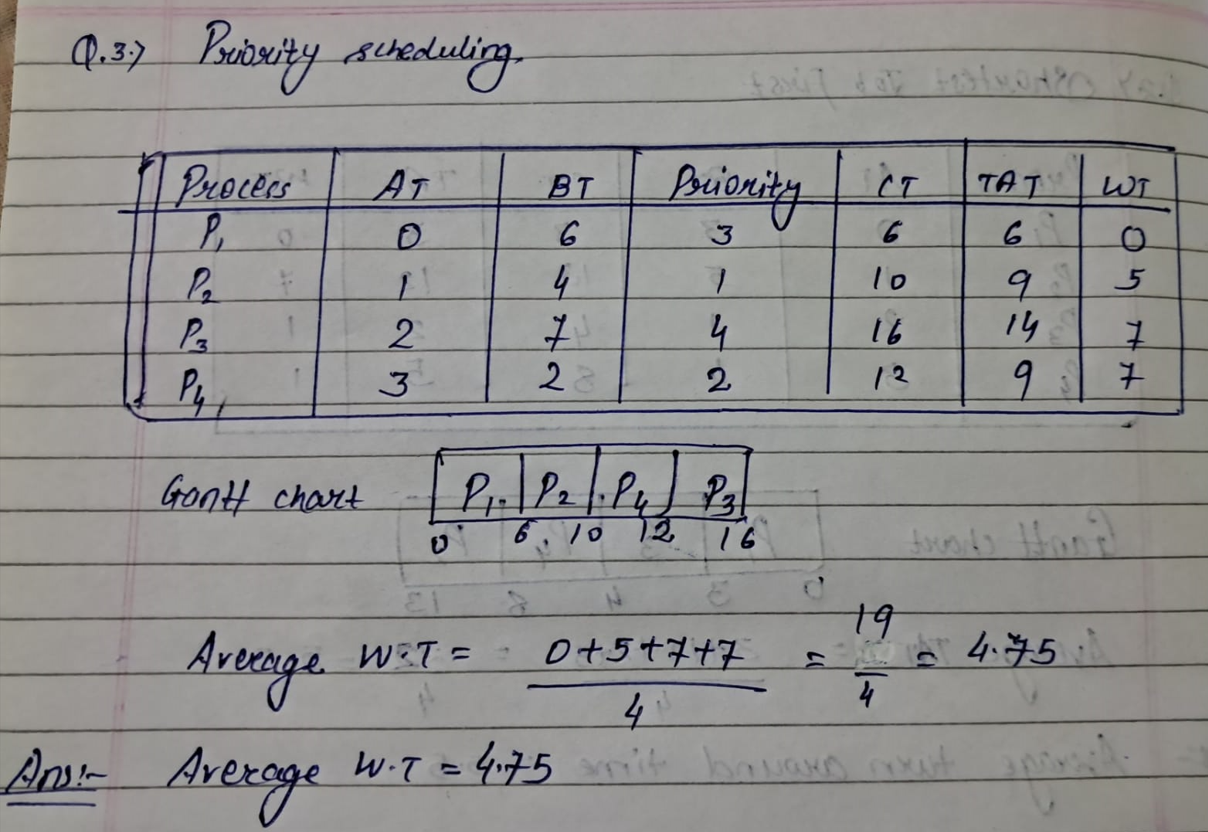
1. Consider the following processes with arrival times, burst times, and priorities (lower number indicates higher priority):

| 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



1. 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 |

| | | |

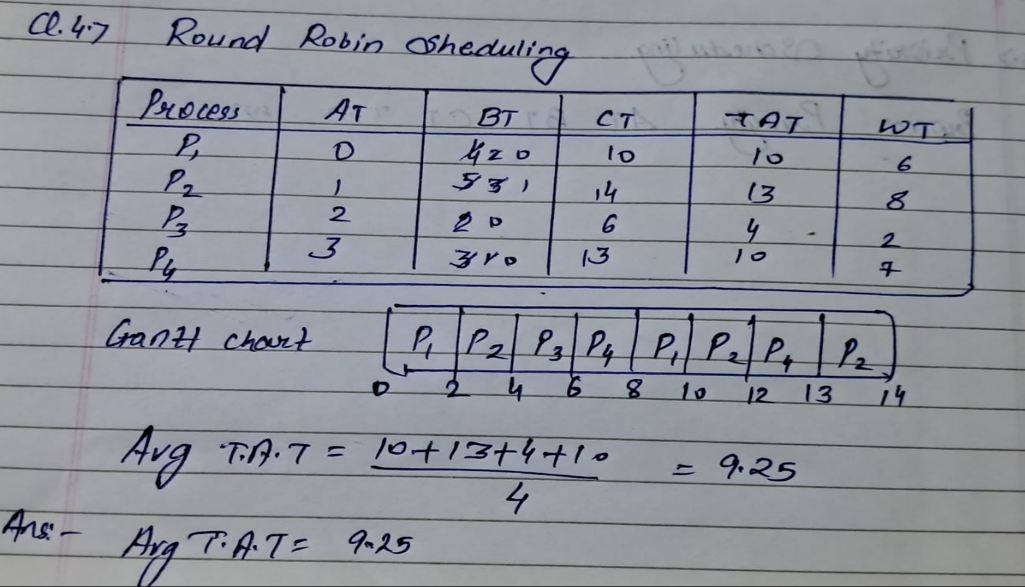
| P1 | 0 | 4 |

| P2 | 1 | 5 |

| P3 | 2 | 2 |

| P4 | 3 | 3 |

Calculate the average turnaround time using Round Robin scheduling.



1. 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**: After fork ( ) call

#include <stdio.h>

void main()

{

int x = 5;

fork();

x = x+1;

printf("x = %d\n",x);

}

## 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.