**CDAC MUMBAI**

**Assignment 2**

**Concepts of Operating System**

**What will the following commands do?**

** echo "Hello, World!"**

This prints the Hello,World! On the console.

** name="Productive"**

**T**his command assigns the string "Productive" to a variable named name

** touch file.txt**

Creates a file named file.txt

** ls -a**

**ls** lists the files and directories in the current directory.

**-a** include hidden files .

** rm file.txt**

Delete the file.txt

** cp file1.txt file2.txt**

Copt the file1 content in file2

** mv file.txt /path/to/directory/**

If there exists the directory named “directory” then it moves the file1.txt to that directory.

** chmod 755 script.sh**

Sets the file permissions for owner to read , write , execute . for group and others read and execute .

** grep "pattern" file.txt**

Grep used to find the specific word in file . basically it finds the word pattern in file.txt

** kill PID**

It terminates the process which having PID

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

**mkdir mydir** used to make a new directory

**cd mydir** moves in mydir directory   
**touch file.txt** creates a file named file.txt

**echo “Hello,World!”** used to print the message on console

**> :** delets the previous content from the file and adds the new one in file.txt

**Cat file.txt**  prints the entire file on console

** ls -l | grep ".txt"**

**ls** lists the files and directories in the current directory.

**-l** used to see the file permission   
 **grep ".txt"** used to find any file or word

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

Concate the output of the file1.txt and file2.txt & sort the only unique recors from file and prints it on console

** ls -l | grep "^d"**

**ls -l**  Lists files and directories in long format, showing permissions, owner, size, and modification time.

**grep "^d"** → Filters the output to **show only directories.**

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

used to **search for a specific pattern recursively** within a directory and its subdirectories.

grep : to search for a pattern in files.

-r (or --recursive) : Recursively searches inside all subdirectories.

"pattern" : The text or regular expression to search for.

/path/to/directory/ → The directory where the search should begin.

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

Prints only duplicate content from file1 and file2

** chmod 644 file.txt**

Sets the file permissions for owner to read , write. for group and others read only.

** cp -r source\_directory destination\_directory**

used to copy directories and their contents recursively from one location to another.

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

find : Command to search for files and directories.

/path/to/search : The directory where the search begins.

-name "\*.txt" : Searches for files ending with .txt. The \* is a wildcard that matches any filename.

** chmod u+x file.txt**

Sets the file permissions for owner to execute . for group and others read and execute .

** echo $PATH**

echo : Prints text or variables to the terminal.

$PATH : Refers to the **PATH environment variable**, which lists directories separated by colons (:)

***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**

Cd is to change directories

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. – **False**

-r is for directories

**Identify the Incorrect Commands:**

1. **chmodx is used to change file permissions. – Incorrect**

chmod is correct

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

cp is used to copy files and directories

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

mkdir used to create directories , touch used to create new file

**4. catx is used to concatenate files. - Incorrect**

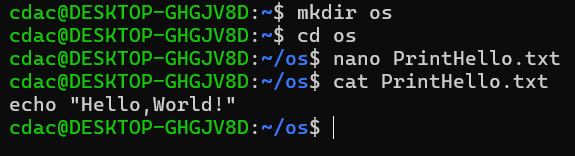
Cat is used to show the file data

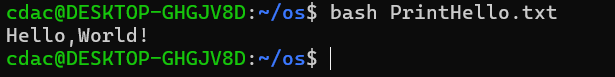
1. **rn is used to rename files. -Incorrect**

**mv** is used to rename a file

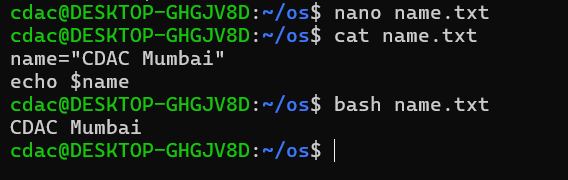
***PART C***

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

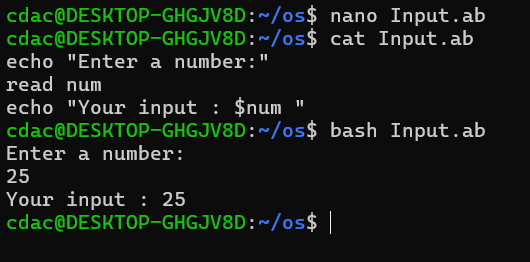
******

******

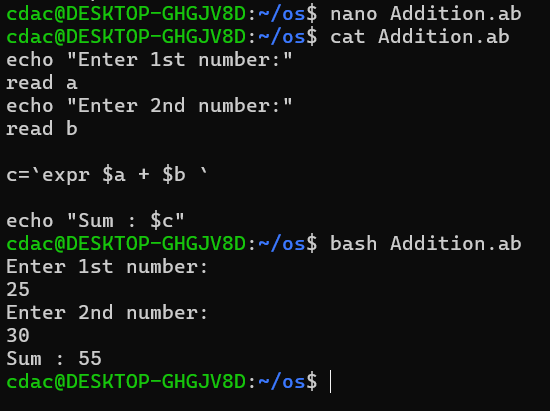
***Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.***

******

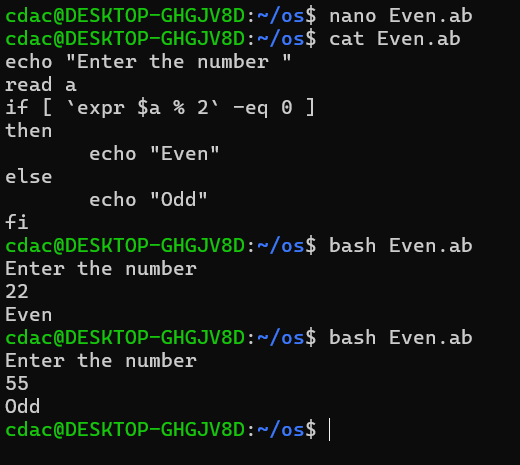
***Question 3: Write a shell script that takes a number as input from the user and prints it.***

******

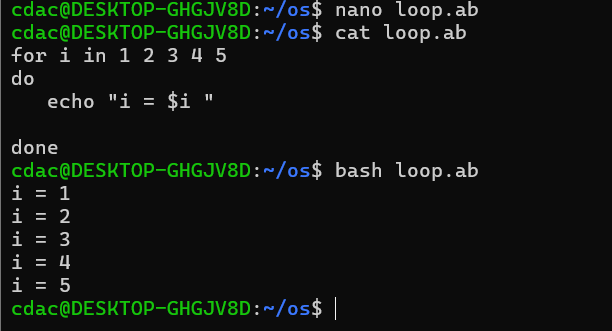
***Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.***

******

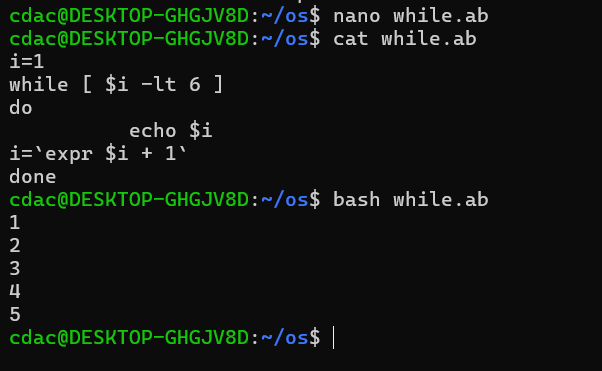
***Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".***

******

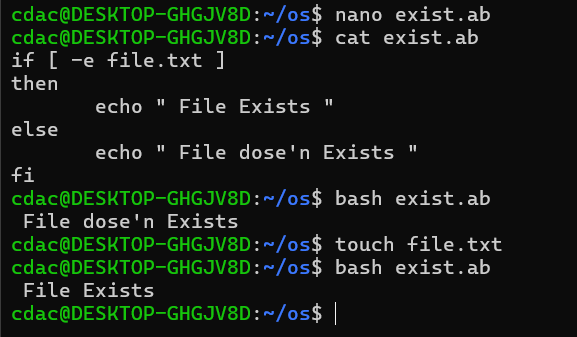
***Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.***

******

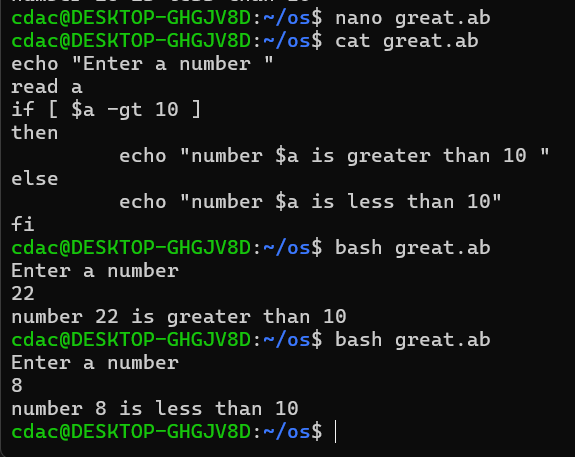
***Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.***

******

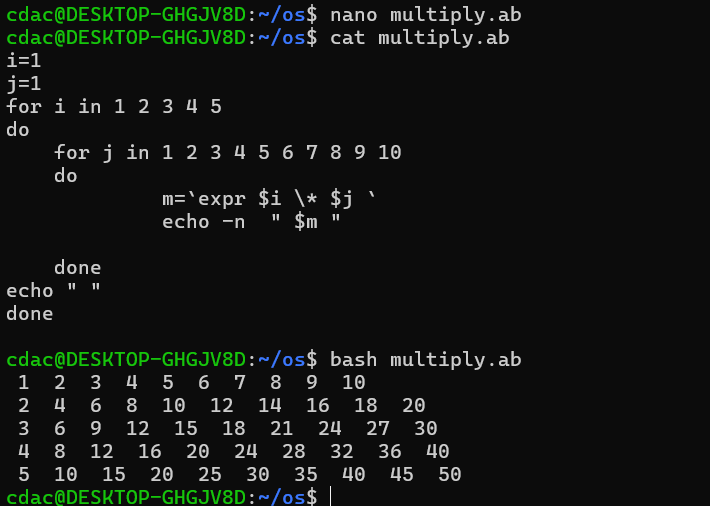
***Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".***

******

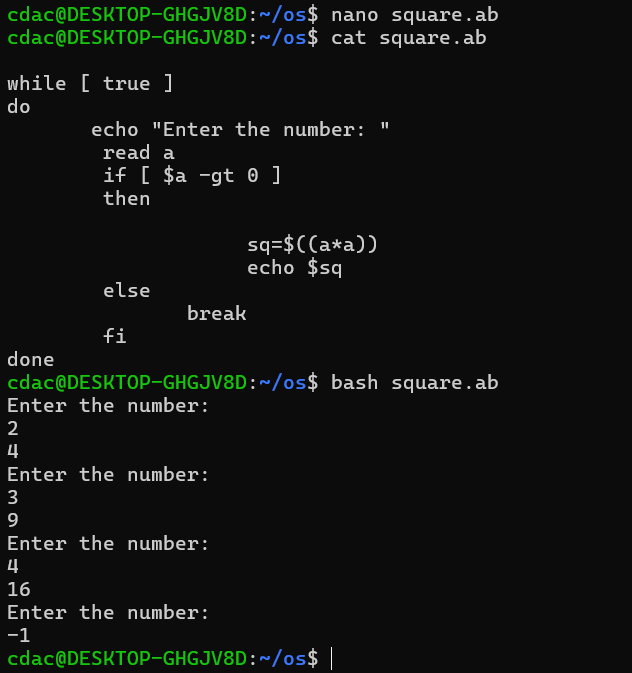
***Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.***

******

***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 E**

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

**| Process | Arrival Time | Burst Time |**

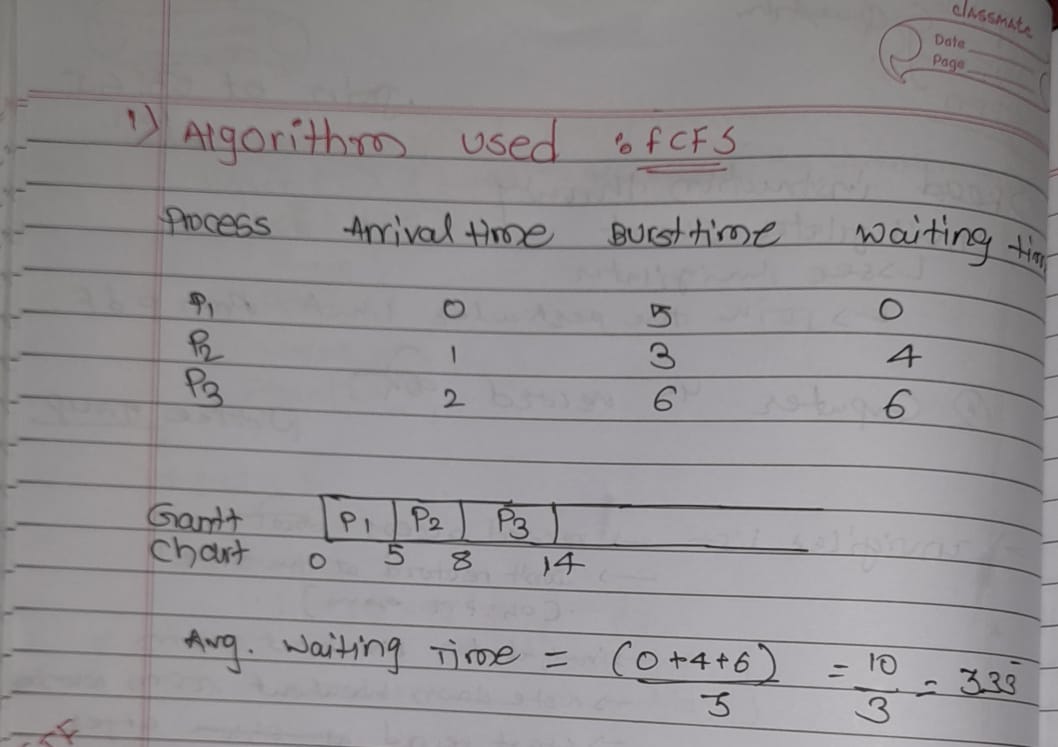
**|---------|--------------|------------|**

**| P1 | 0 | 5 |**

**| P2 | 1 | 3 |**

**| P3 | 2 | 6 |**

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

****

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

**| Process | Arrival Time | Burst Time |**

**|---------|--------------|------------|**

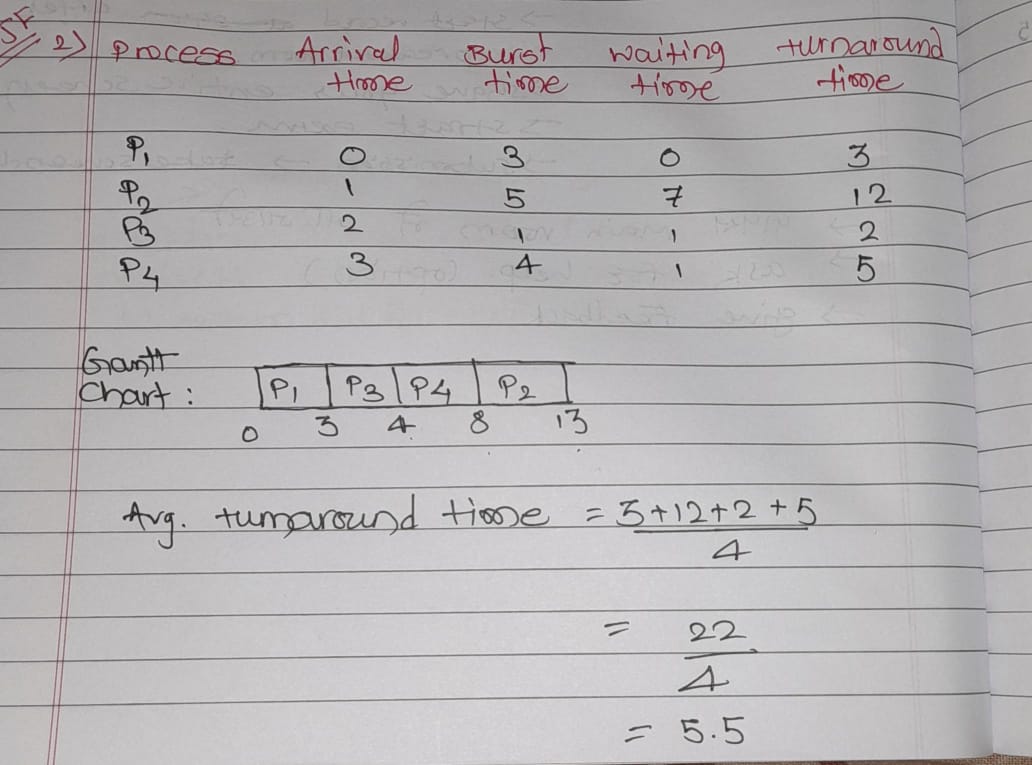
**| P1 | 0 | 3 |**

**| P2 | 1 | 5 |**

**| P3 | 2 | 1 |**

**| P4 | 3 | 4 |**

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

****

**3. Consider the following processes with arrival times, burst times, and priorities (lower number**

**indicates higher priority):**

**| Process | Arrival Time | Burst Time | Priority |**

**|---------|--------------|------------|----------|**

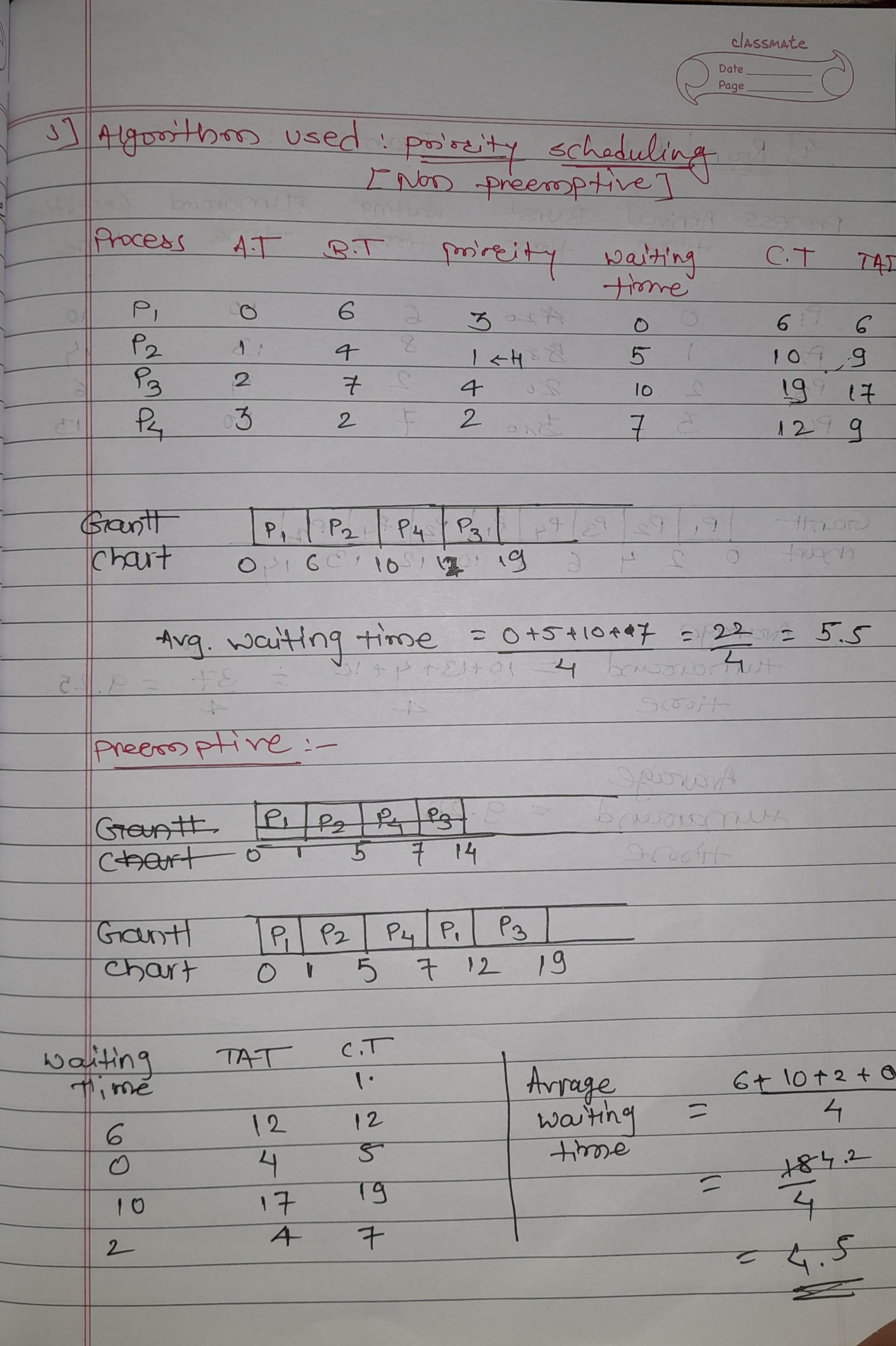
**| P1 | 0 | 6 | 3 |**

**| P2 | 1 | 4 | 1 |**

**| P3 | 2 | 7 | 4 |**

**| P4 | 3 | 2 | 2 |**

**Calculate the average waiting time using Priority Scheduling.**

****

**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 |**

**|---------|--------------|------------|**

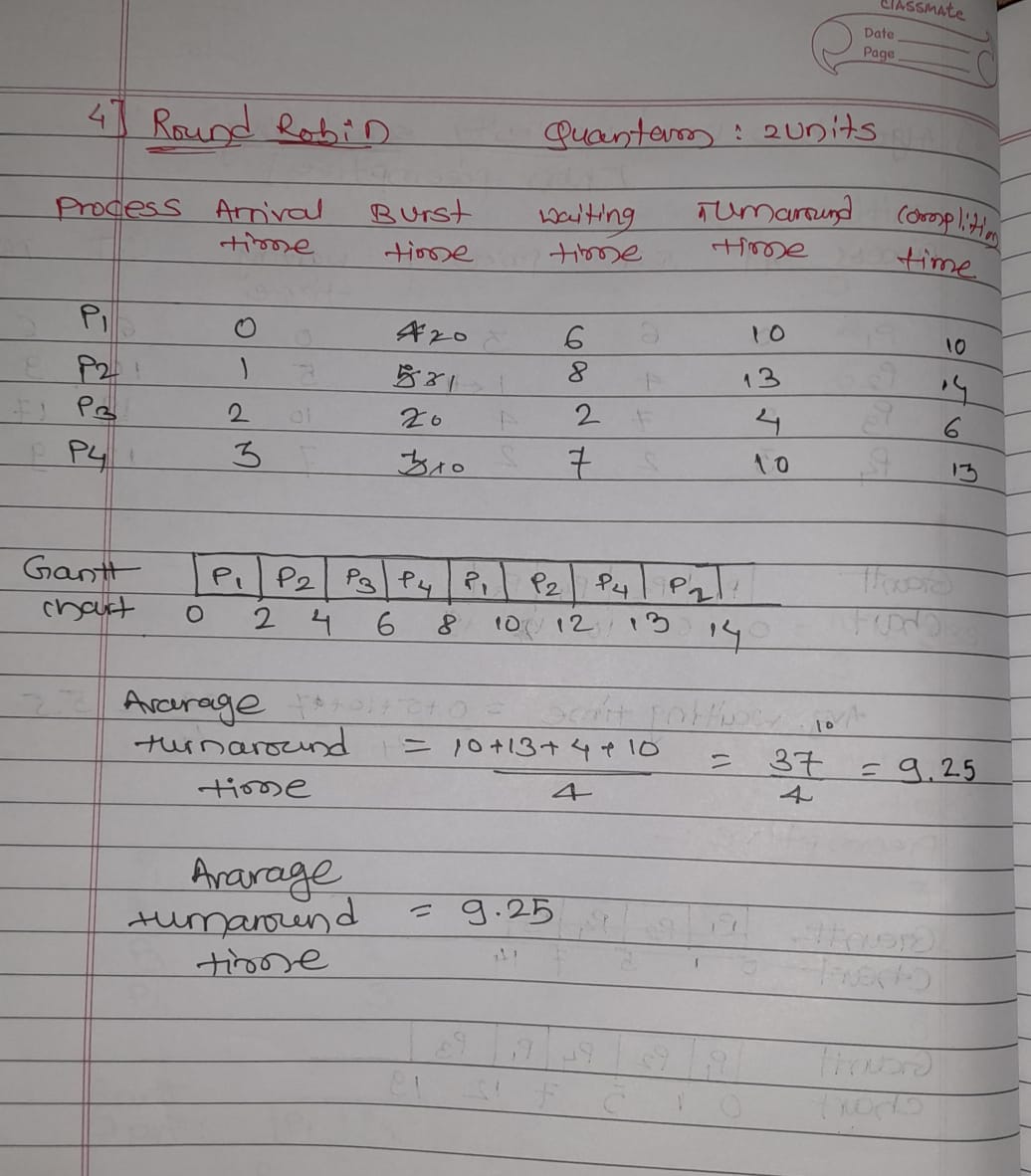
**| P1 | 0 | 4 |**

**| P2 | 1 | 5 |**

**| P3 | 2 | 2 |**

**| P4 | 3 | 3 |**

**Calculate the average turnaround time using Round Robin scheduling.**

****

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?**

**Answer :-**

1. The parent process starts with x = 5.
2. When fork() is called, a new child process is created, and it also has x = 5.
3. Both the parent and child execute their code independently.
4. Each process increments x by 1:
   * In the child process, x = 5 + 1 = 6.
   * In the parent process, x = 5 + 1 = 6.

Since both processes have their own copies of x, their changes do not reflect in each other. So the value of x in parent is as same as in child

i.e. x= 6