**Part A**

**What will the following commands do?**

• **echo "Hello, World!"**

It will display Hello, World!

Echo command is used to display the text ,variable on to the standard output.

**• name="Productive"**

name is the variable name and productive is the value for the variable.

The productive is the value assigned to the name.

**• touch file.txt**

Touch command is used to create a empty file .

In this command empty file named as file.txt is created.

**• ls -a**

It is use to display the list of all files and directories including hidden files.

**• rm file.txt**

rm is used to remove the file from the directory.

The above command will remove the file named file.txt.

**• cp file1.txt file2.txt**

cp : cp command is used to copy the file.

The above command copy the file1.txt into file2.txt.

**• mv file.txt /path/to/directory/**

mv : is used to move file to the another directory

**• chmod 755 script.sh**

It gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.

**• grep "pattern" file.txt**

It is used to Searching and manipulating text pattern in file.

**• kill PID**

Terminates the process with the specified Process ID (PID).

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

This command created a directory using mkdir. the file.txt is created in the mydir drectory. > sign redirected in the file.txt where the echo prints Hello, World! On the command line.

**• ls -l | grep ".txt"**

This command would select from the output of ls –l the files that has extension of .txt

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

The command combines the contents of file1.txt and file2.txt, sorts them alphabetically, and removes duplicate lines, leaving only unique lines in the output.

**• ls -l | grep "^d"**

This command would select from the output of ls –l that are directories

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

Recursively searches for pattern within all files in the specified directory and its subdirectories.

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

The command combines the contents of file1.txt and file2.txt, sorts them, and then displays only the lines that are duplicated in the combined, sorted output.

**• chmod 644 file.txt**

The command is used to change the permissions of the file file.txt.

It will give permission of Read and write to owner and read only for group and others.

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

The command is used in to copy directories and their contents from one location to another.

cp to copy and -r with all the content.

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

The command is used to search for files in a specified directory and its subdirectories that match a particular name pattern.

**• chmod u+x file.txt**

It gives permission for execution to the owner on file.txt.

**• echo $PATH**

The command is used to display the current value of the PATH environment variable.

**Part B**

**Identify True or False:**

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

True**.**

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

True

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

False. cd is used to change directory.

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

True

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

True

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

True

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

True

1. **rm -rf file.txt deletes a file forcefully without confirmation.**

True

**Identify the Incorrect Commands:**

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

chmod command is used to change file permissions.

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

cp is used to copy file and directories

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

touch or nano is used to create new file.

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

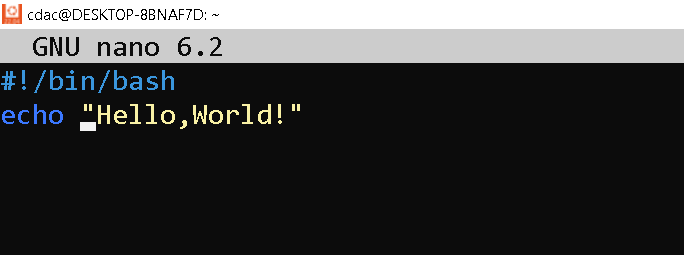
cat is used to concatenate files.

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

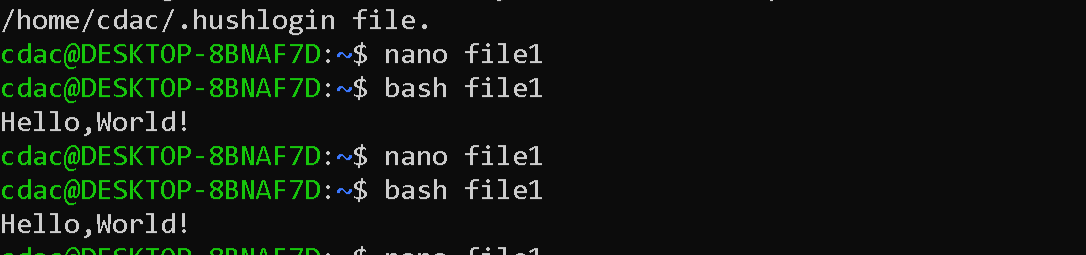
mv command is used to rename file.

**Part C**

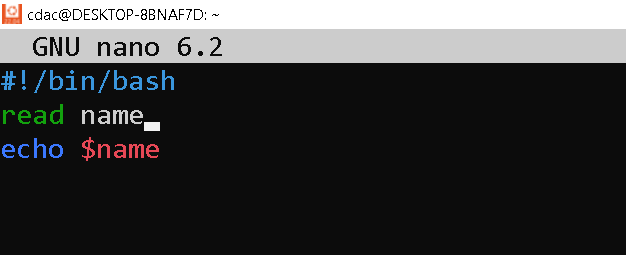
**Question 1: Write a shell script that prints "Hello, World!" to the terminal.**

****

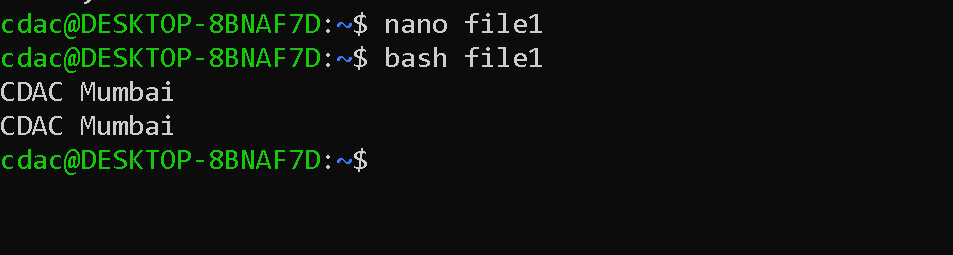
**Output:**

****

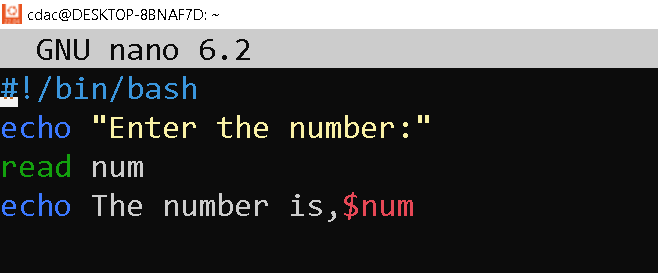
**Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.**

****

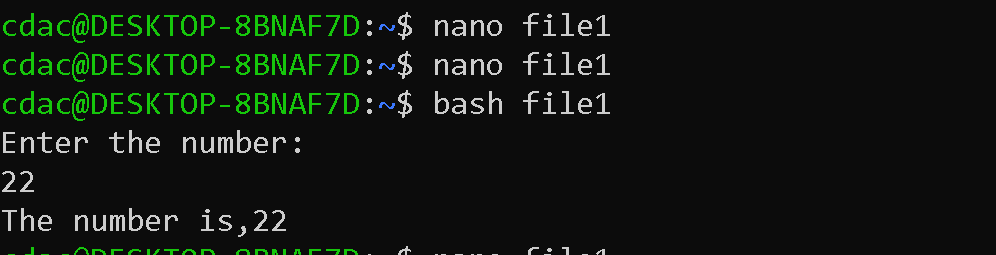
**Output:**

****

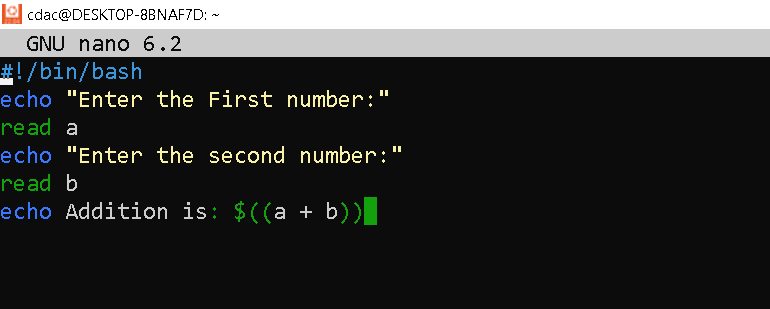
**Question 3: Write a shell script that takes a number as input from the user and prints it.**

****

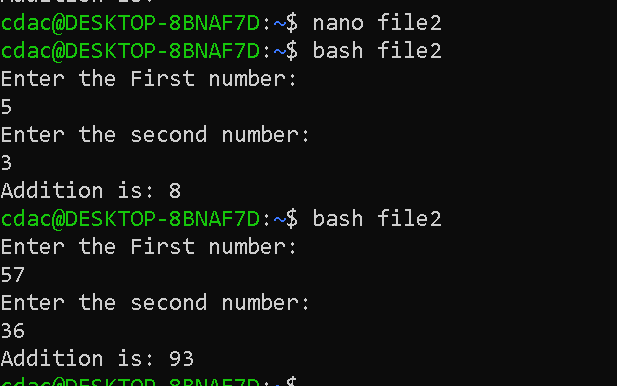
**Output:**

****

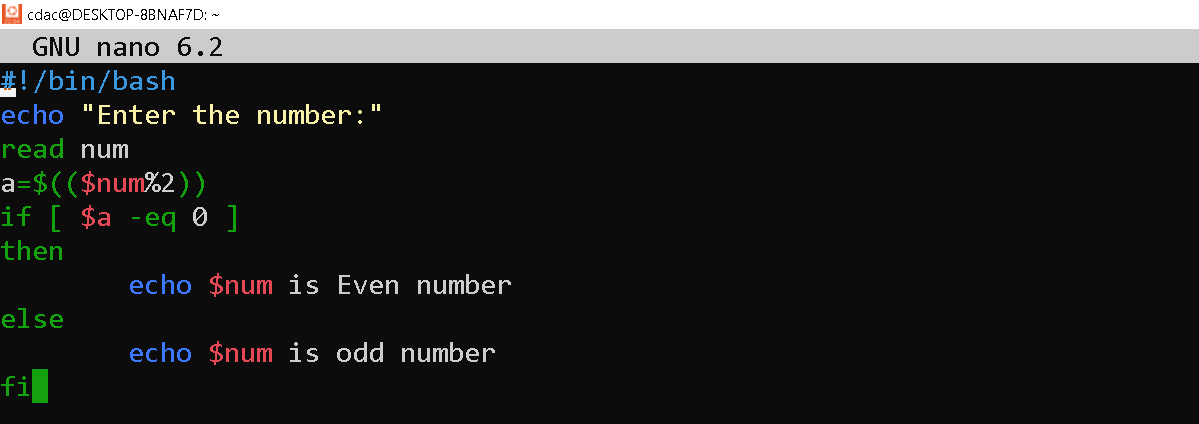
**Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.**

****

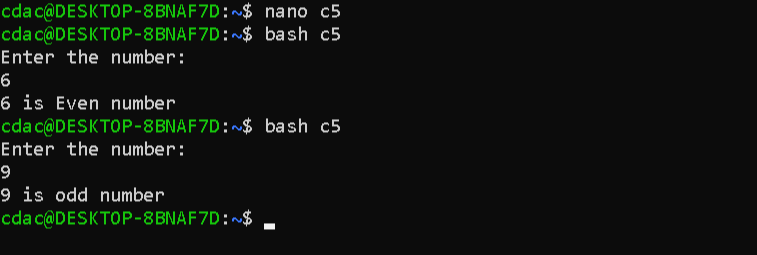
**Output:**

****

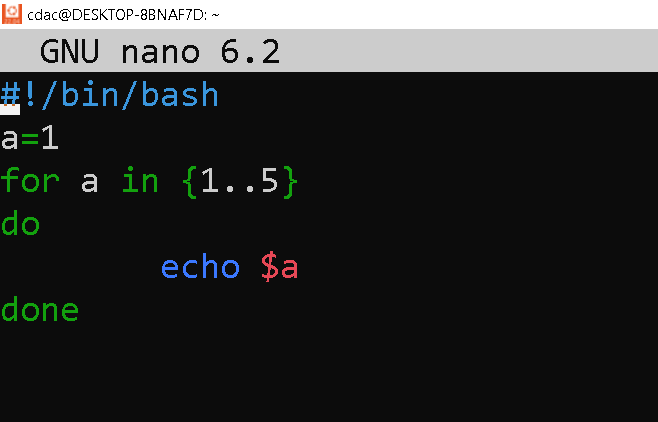
**Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".**

****

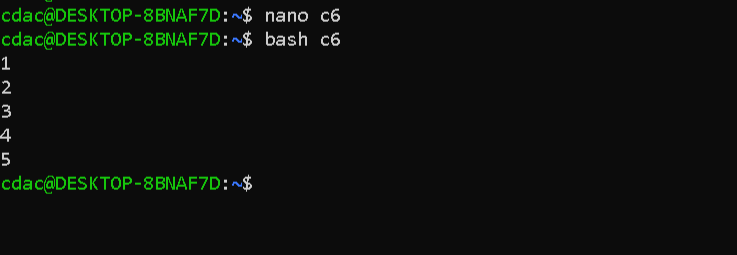
**Output:**

****

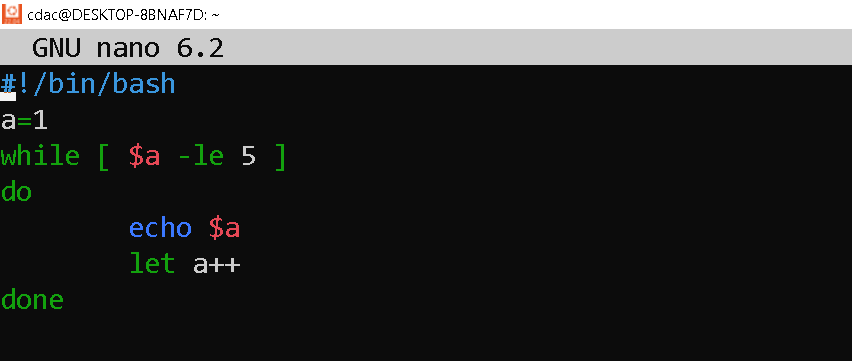
**Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.**

****

**Output:**

****

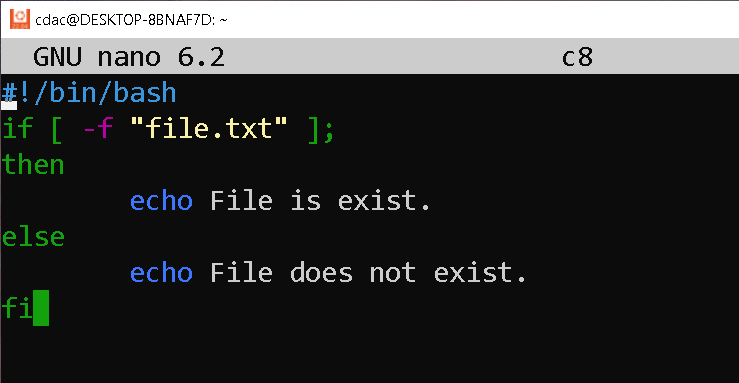
**Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.**

****

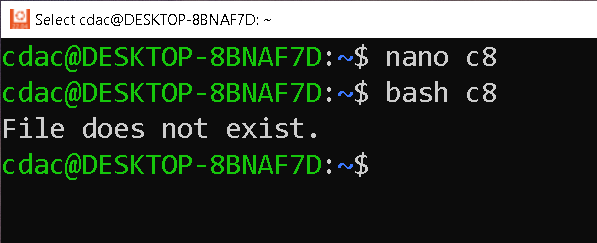
**Output:**

****

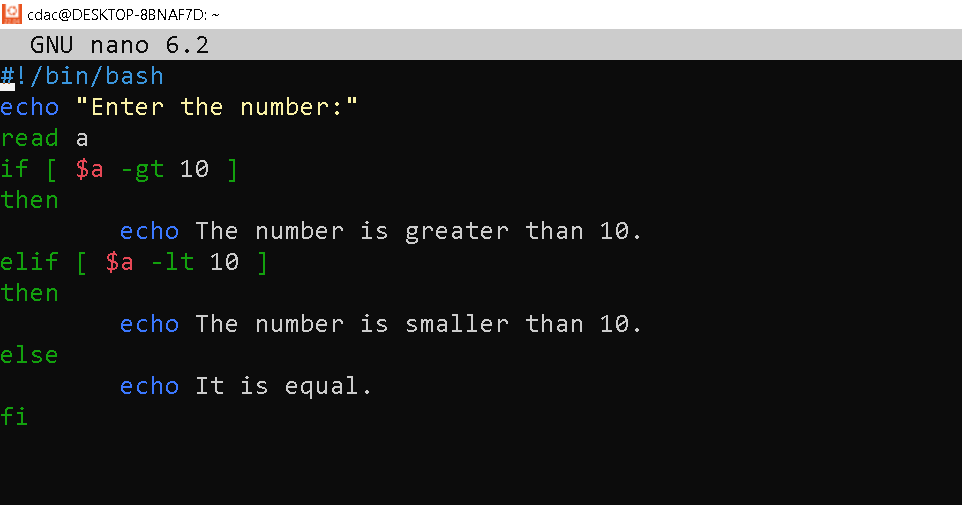
**Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".**

****

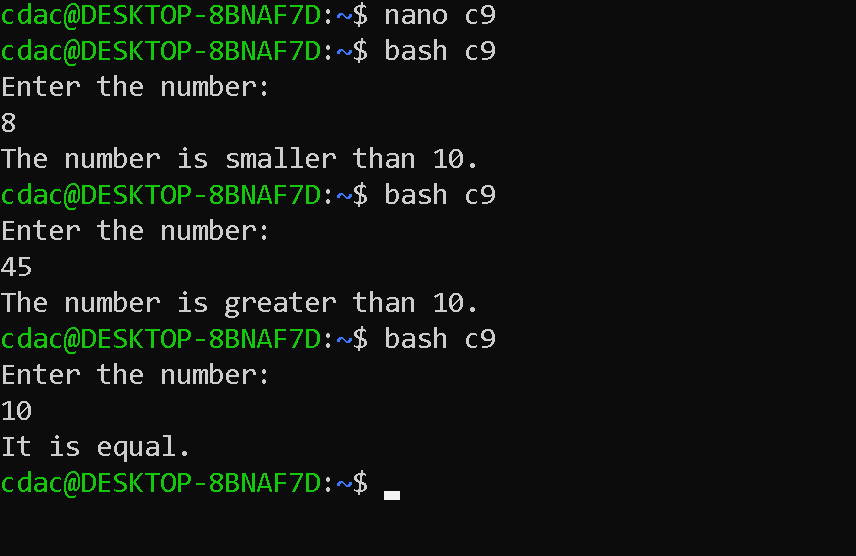
**Output:**

****

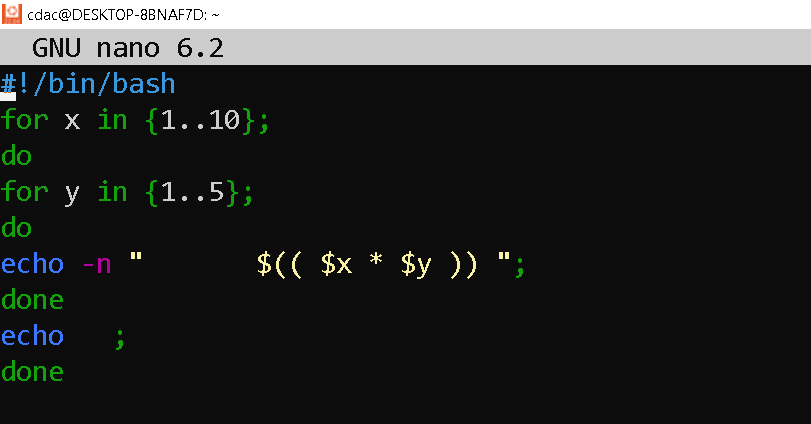
**Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.**

****

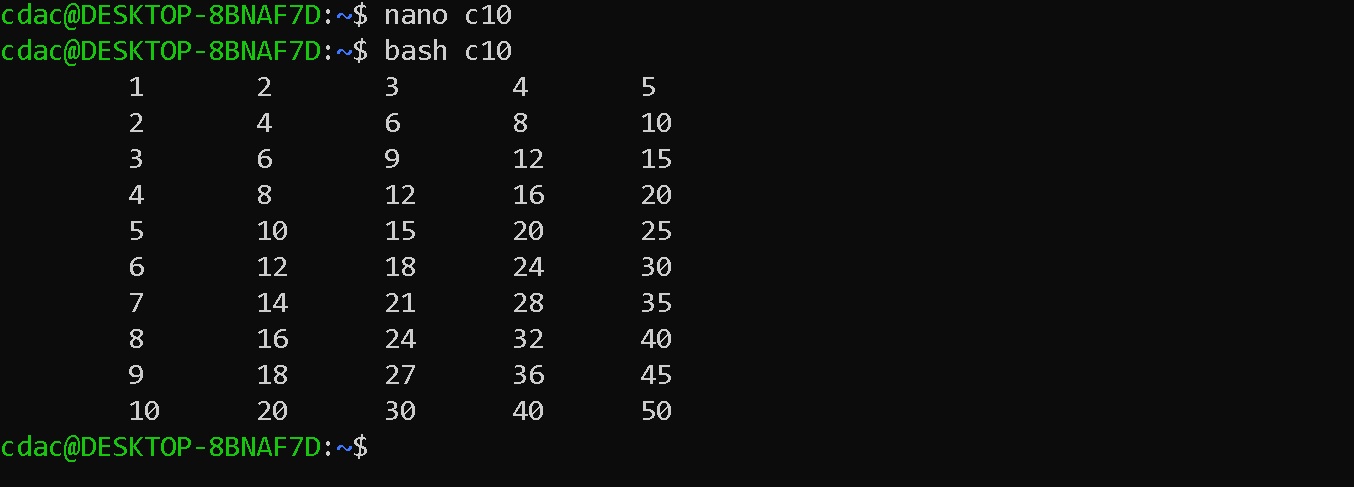
**Output:**

****

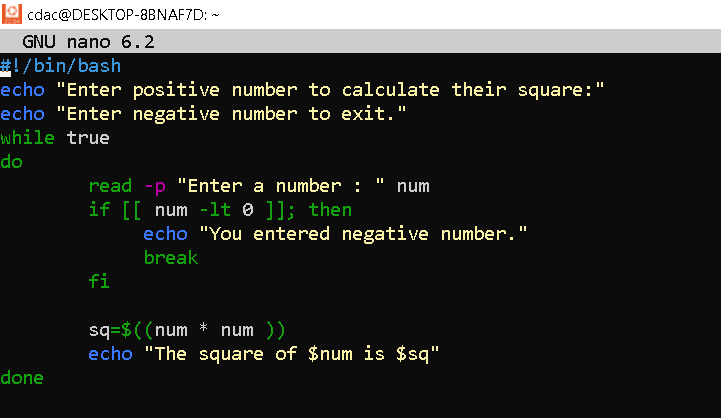
**Question 10: Write a shell script that uses nested for loops to print a multiplication table for numbers from 1 to 5. The output should be formatted nicely, with each row representing a number and each column representing the multiplication result for that number.**

****

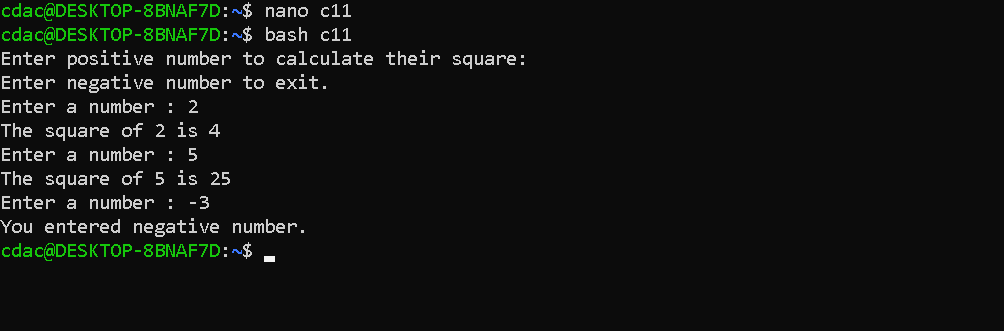
**Output:**

****

**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.**

****

**Output:**

****

**Part E**

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

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

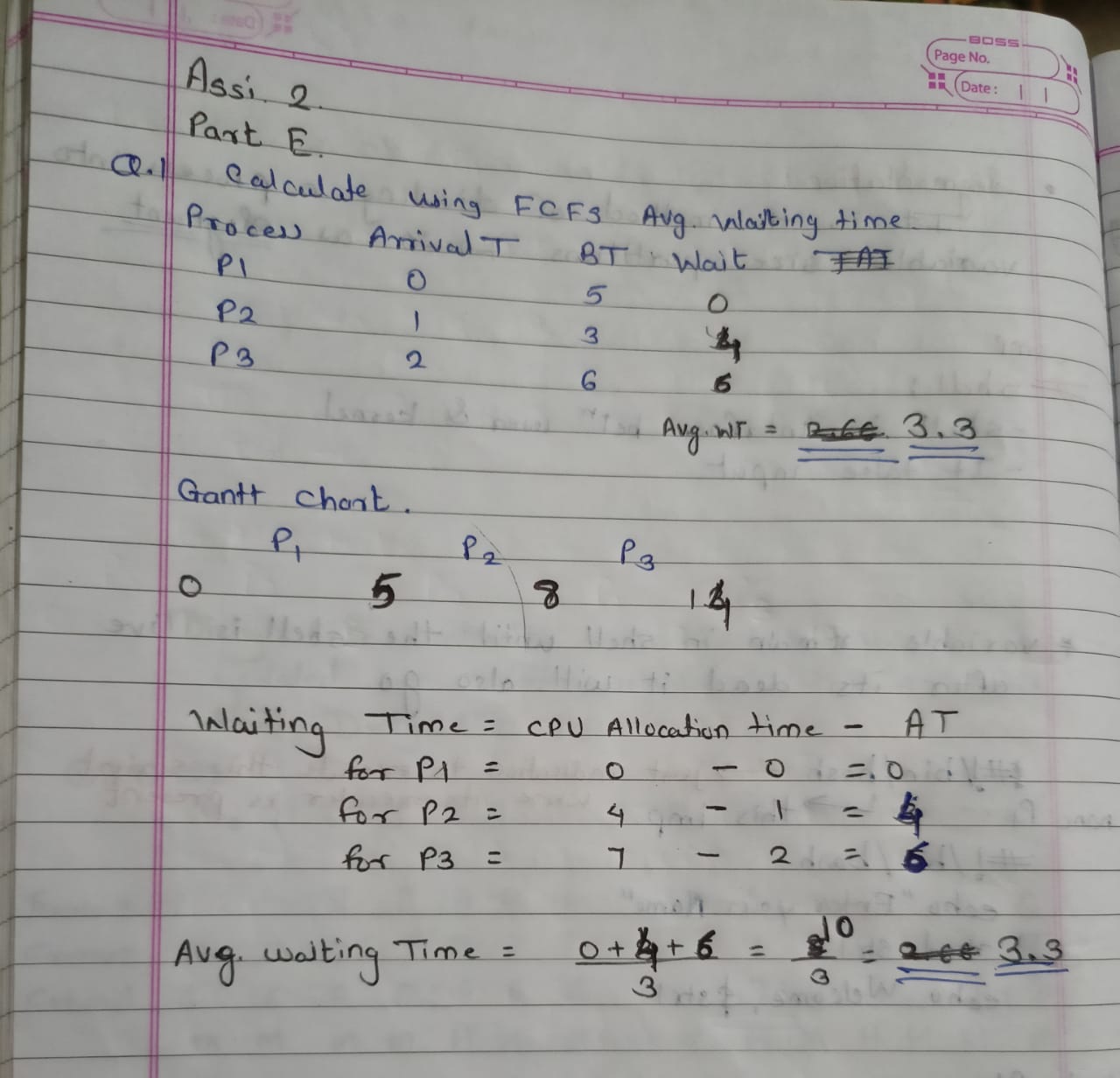
**|---------|--------------|------------|**

**| 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 |**

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

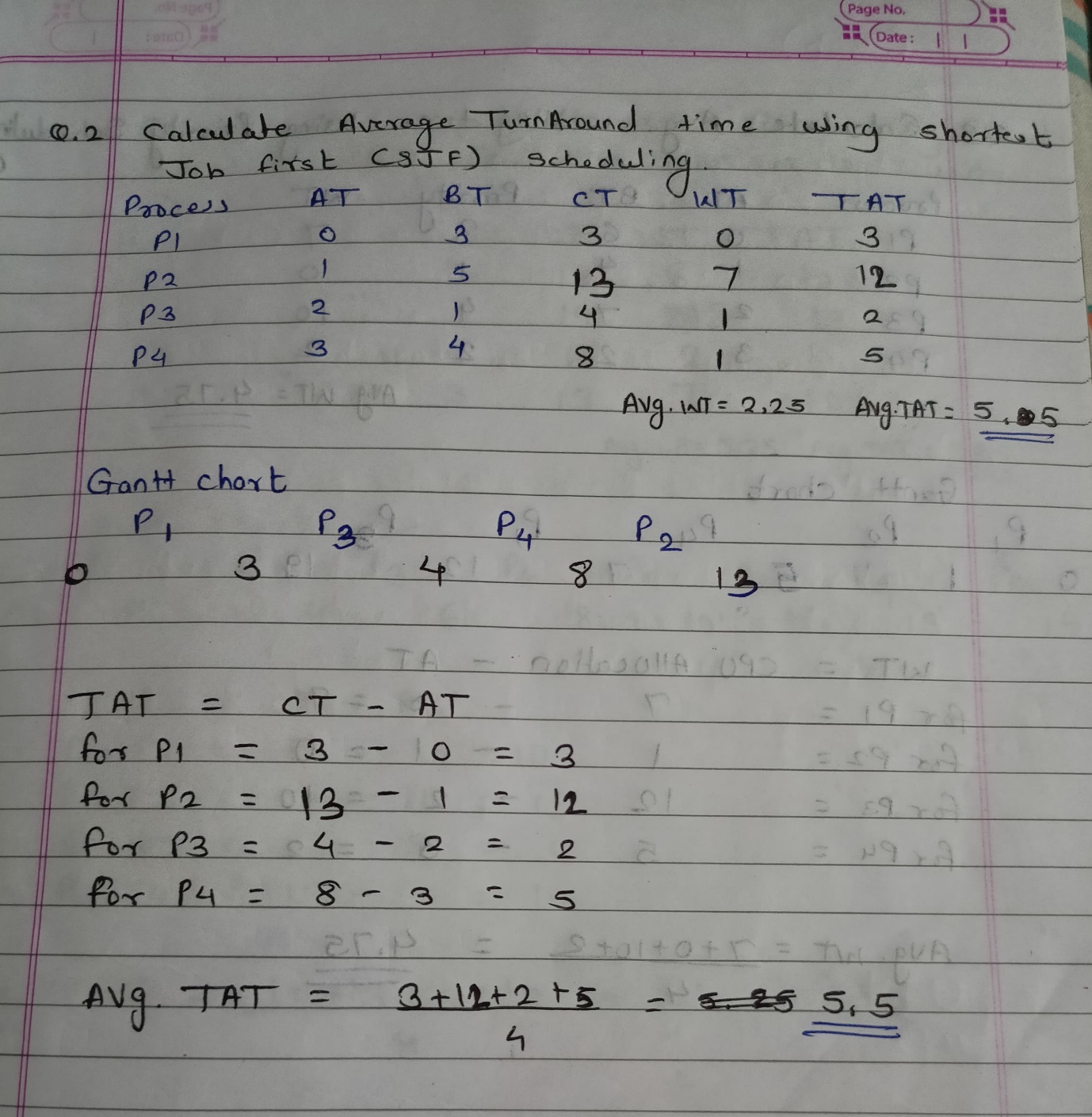
**| P1 | 0 | 3 |**

**| 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 |**

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

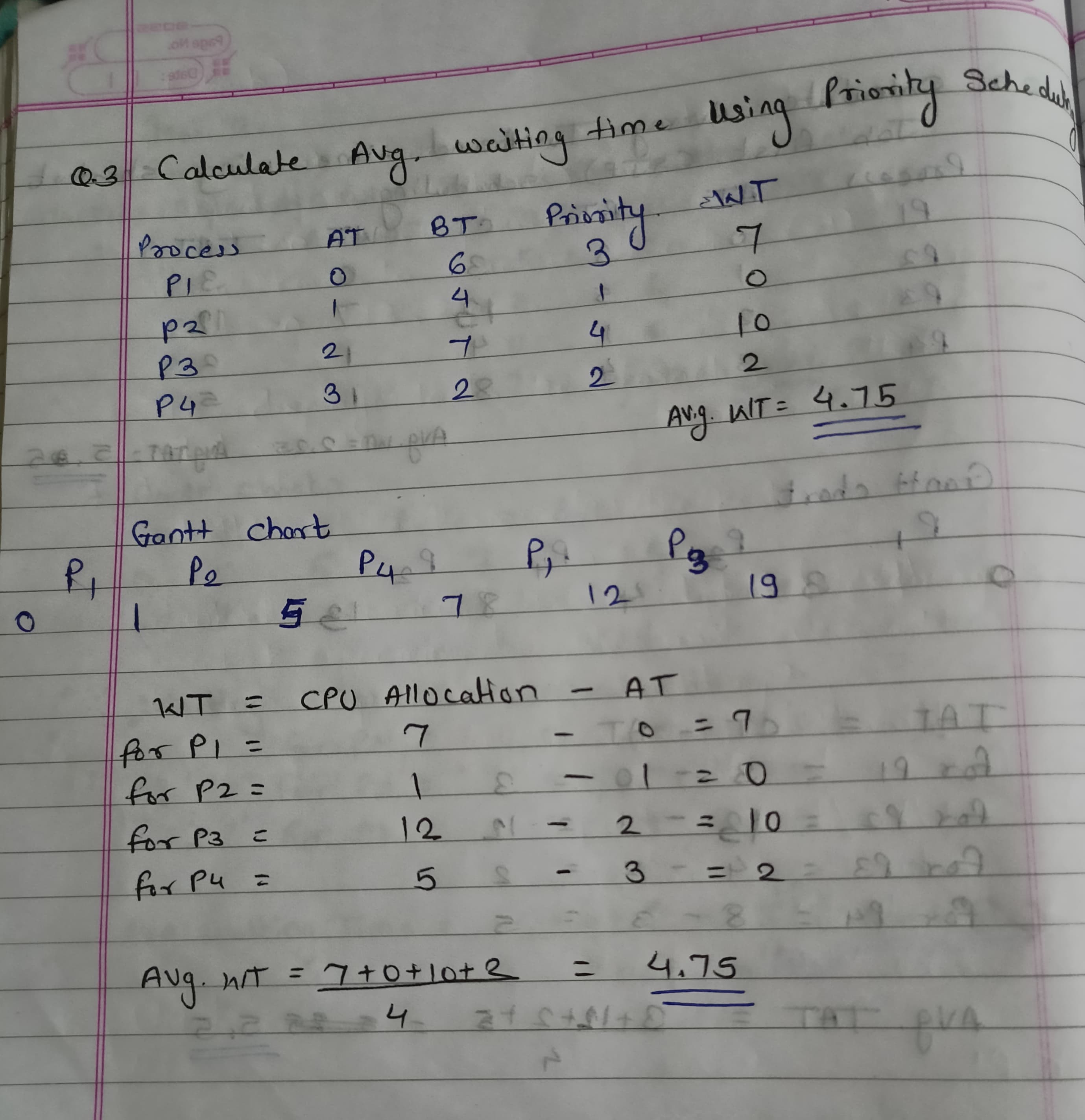
**| P1 | 0 | 6 | 3 |**

**| 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 |**

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

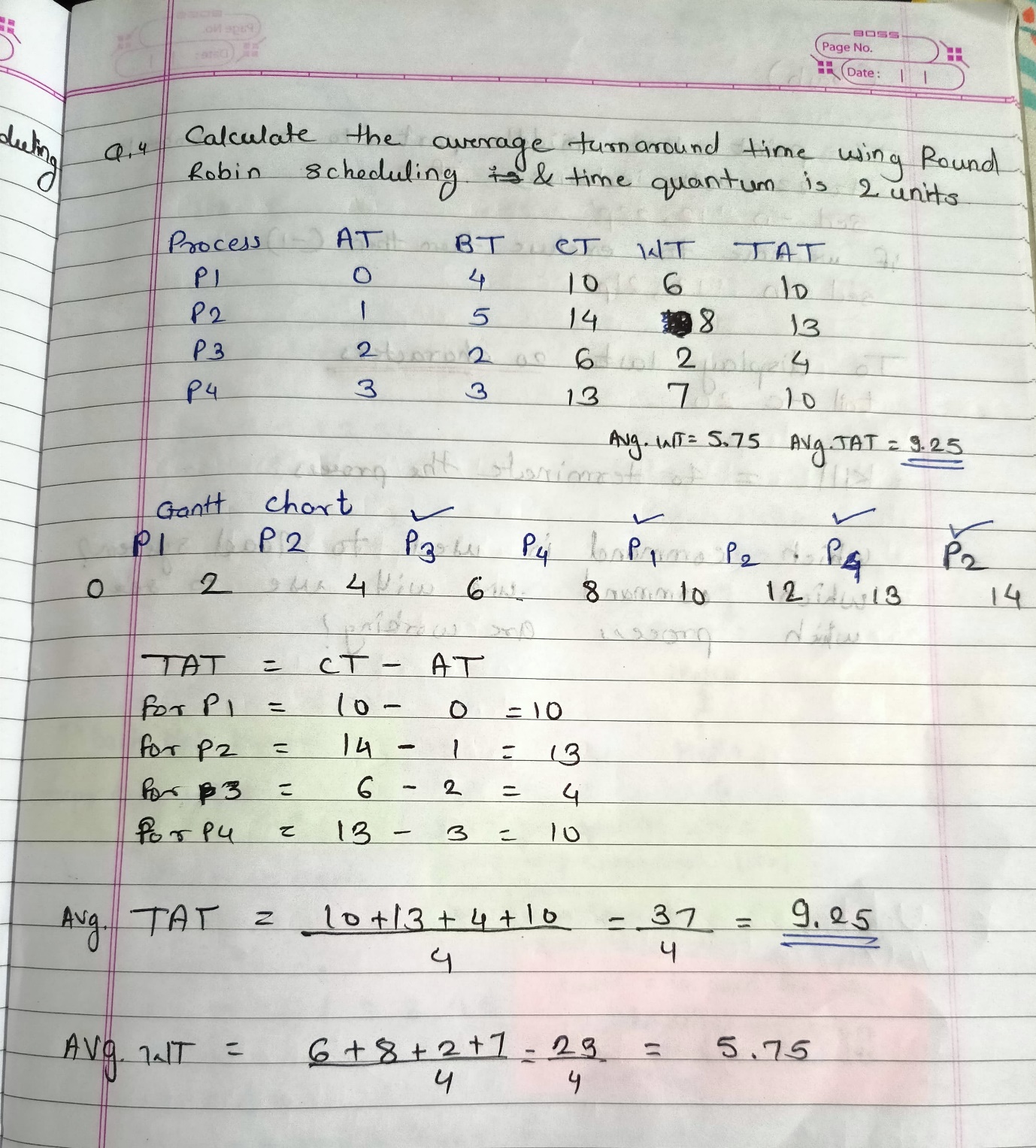
**| P1 | 0 | 4 |**

**| P2 | 1 | 5 |**

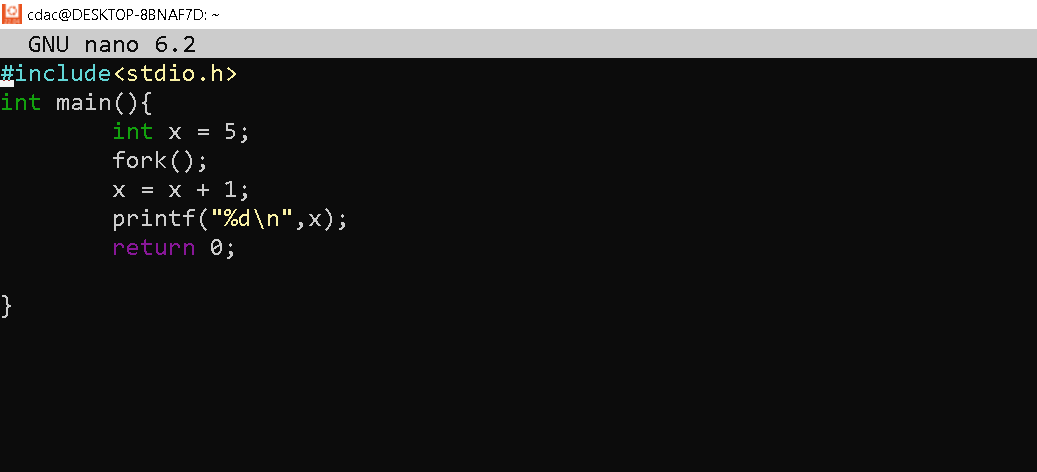
**| P3 | 2 | 2 |**

**| P4 | 3 | 3 |**

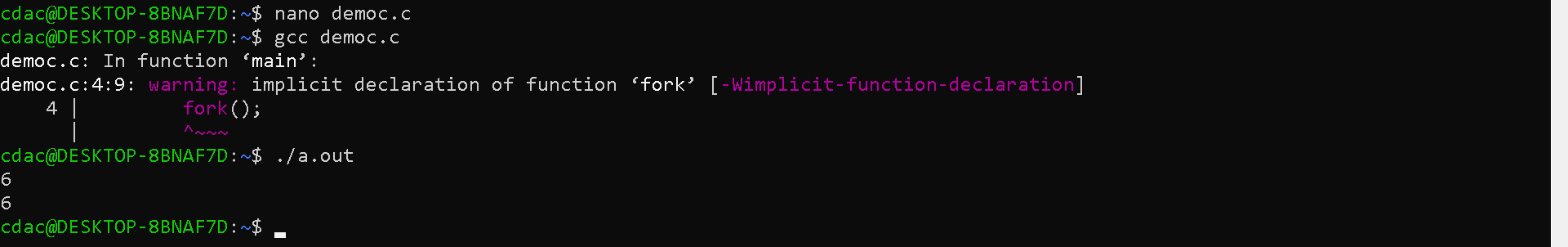
**Calculate the average turnaround time using Round Robin scheduling.**

****

**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?**

****

**Output:**

****