What will the following commands do?

- echo "Hello, World!"

→It will print Hello, World to the console

- name="Productive"

→it will store the String “Productive” in name variable

- touch file.txt

→it will create a file.txt file

- ls -a

→it will display all files including hidden

- rm file.txt

→it will remove file.txt

- cp file1.txt file2.txt

→it will copy the contents of the file1.txt to file2.txt

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

→it will move the file.txt to directory directory

- chmod 755 script.sh

→gives full permission to the owner and only read, execute permission to others

- grep "pattern" file.txt

→checks pattern word in file.txt

- kill PID

→ kills a process

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

→make a directory named ‘mydir’ and jumps into it. In that creates a file named ‘file.txt’ and add contents “Hello, World!” to the file.txt and prints the contents of the file to the console.

- ls -l | grep ".txt"

→list owner information of .txt files

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

→will display the contents of both files which are unique in sorted way

- ls -l | grep "^d"

→list all the directories

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

→ recursively searches for the pattern word in a directory

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

→ will display the contents of both files which are duplicate in sorted way

- chmod 644 file.txt

→The owner can only read and write, and other can only read the file.txt

- cp -r source_directory destination_directory

→it will copy files recursively.

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

→ it will find all the .txt files into the provided path

- chmod u+x file.txt

→it will add permission to the owner to execute the file.txt

- echo $PATH

→it allows to check the current value of PATH variable

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, cp is used to copy

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.

→True

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.

→chmod is used to change file permissions

2. cpy is used to copy files and directories.

→cp is to copy files and directories

3. mkfile is used to create a new file.

→touch is used to create new files

4. catx is used to concatenate files.

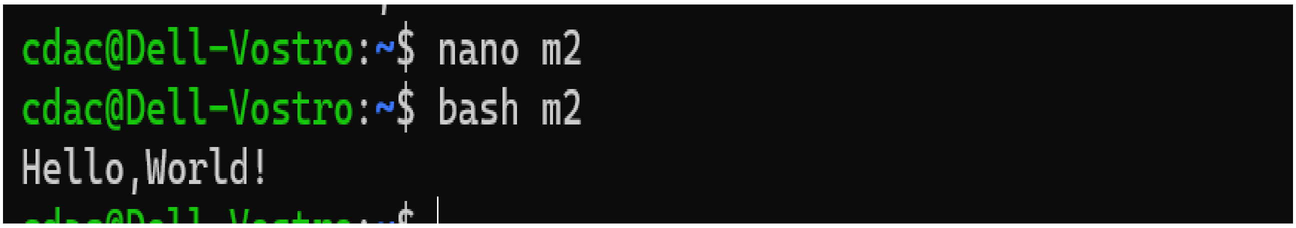
→cat is used to concatenate the files

5. rn is used to rename files.

→mv is used to rename the files

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

→ echo “Hello, World!”

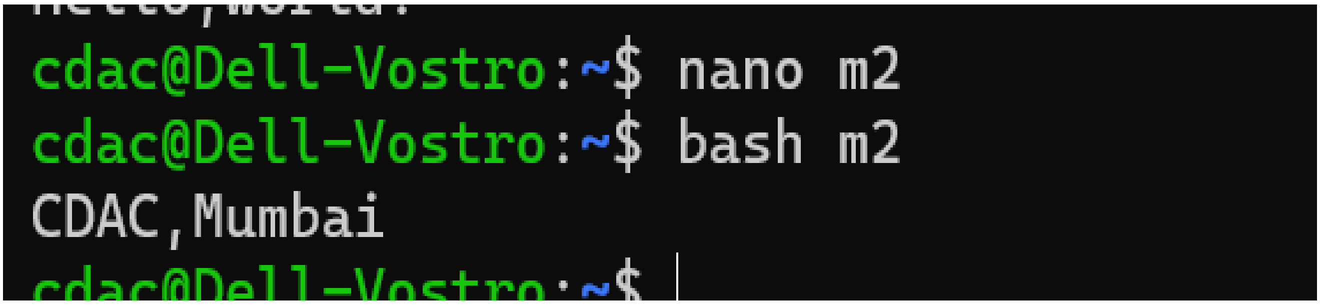


```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Hello,World!
cdac@Dell-Vostro:~$
```

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

→ name="CDAC, Mumbai"

echo $name



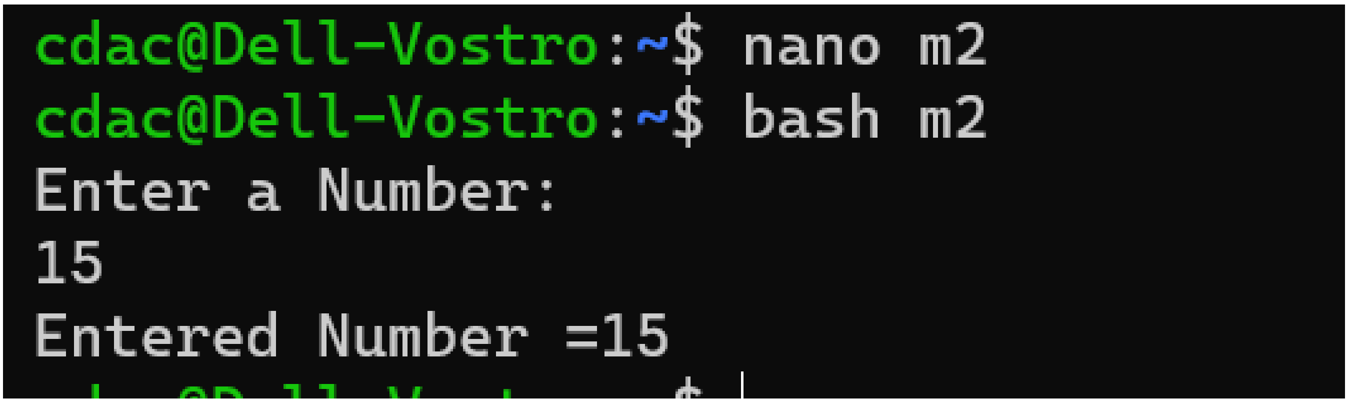
```
Hello,World!
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
CDAC,Mumbai
cdac@Dell-Vostro:~$
```

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

→ echo "Enter a Number:"

read num

echo "Entered Number=$num"

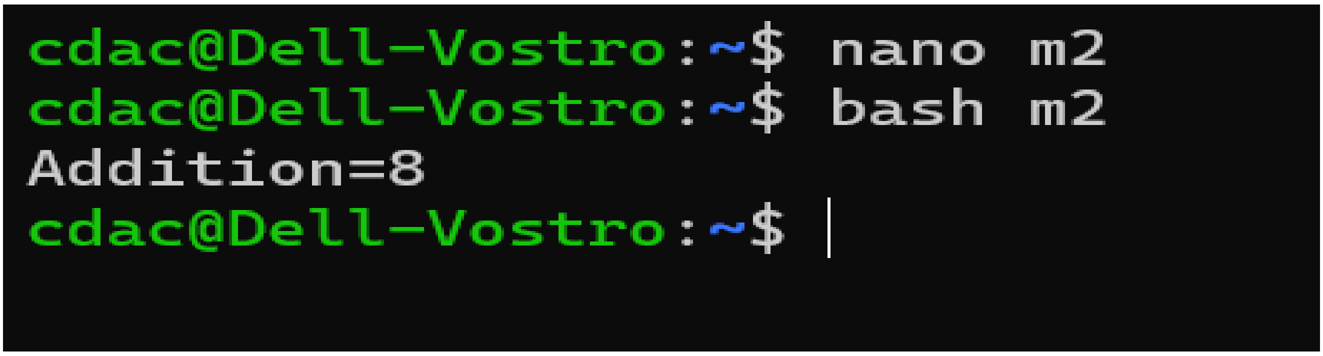


```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Enter a Number:
15
Entered Number =15
cdac@Dell-Vostro:~$
```

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

→ X=5 Y=3

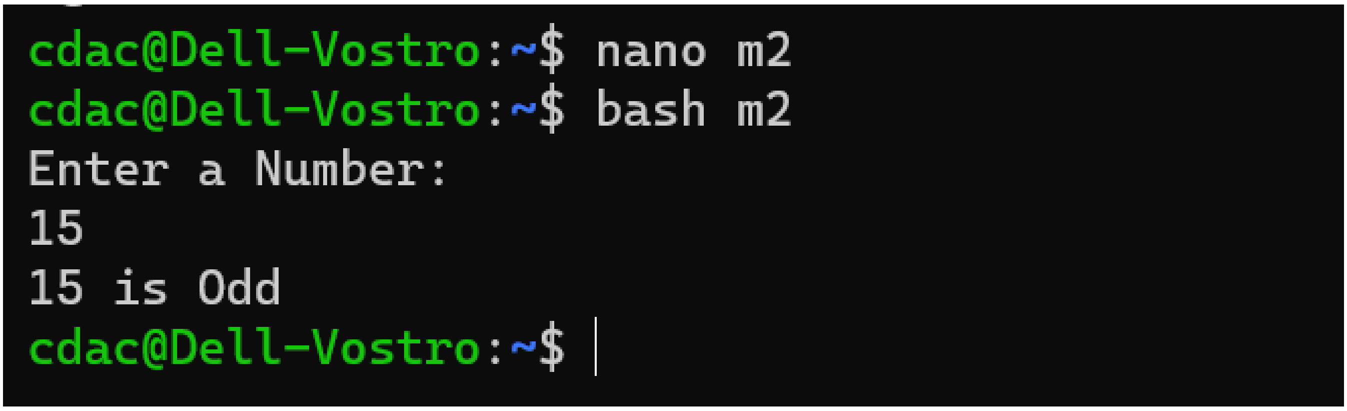
echo "Addition="$(($X+$Y))



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Addition=8
cdac@Dell-Vostro:~$ 
```

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

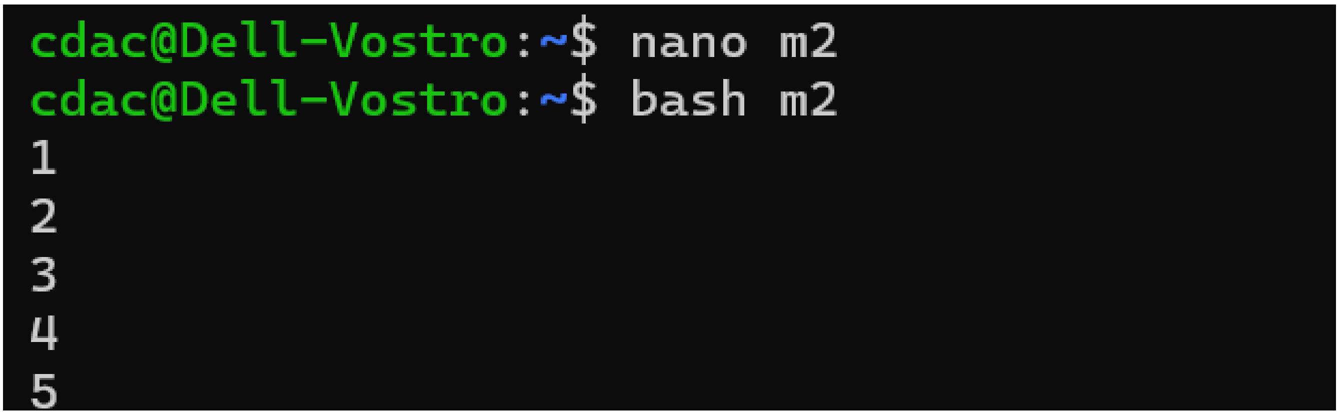
```
→ echo "Enter a Number:"
read num
if [ $(($num % 2)) == 0 ]
then
echo "$num is Even"
else
echo "$num is Odd"
fi
```



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Enter a Number:
15
15 is Odd
cdac@Dell-Vostro:~$
```

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

```
→ for (( a=1; a<=5; a++))
do
echo $a
done
```



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
1
2
3
4
5
```

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

→ a=1

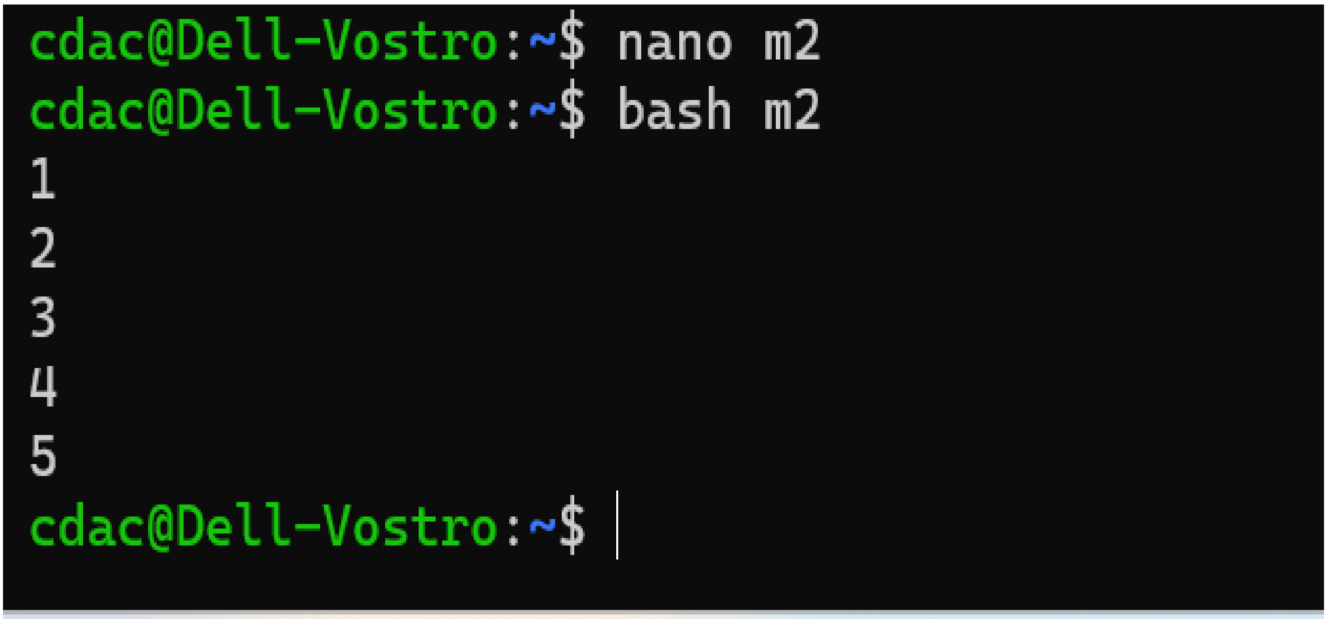
while [$a -le 5]

do

echo $a

a=$(($a+1))

done



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
1
2
3
4
5
cdac@Dell-Vostro:~$ 
```

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".

→ if [-f "file.txt"]

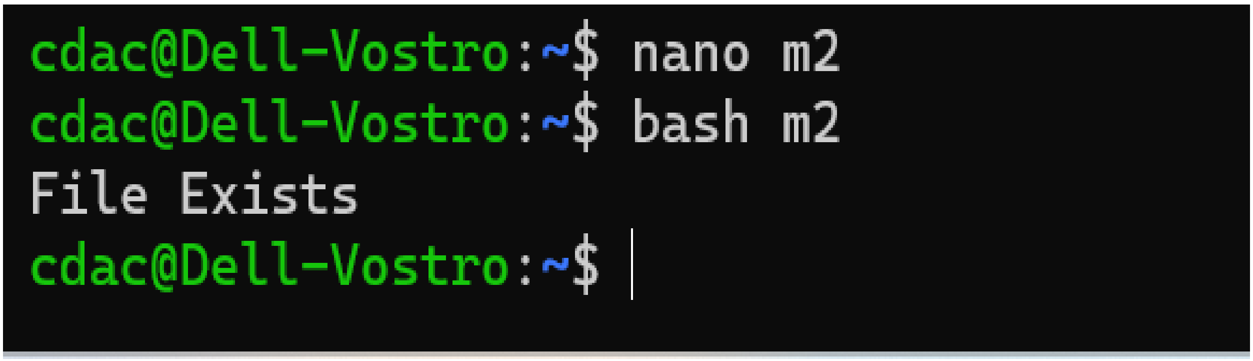
then

echo "File Exists"

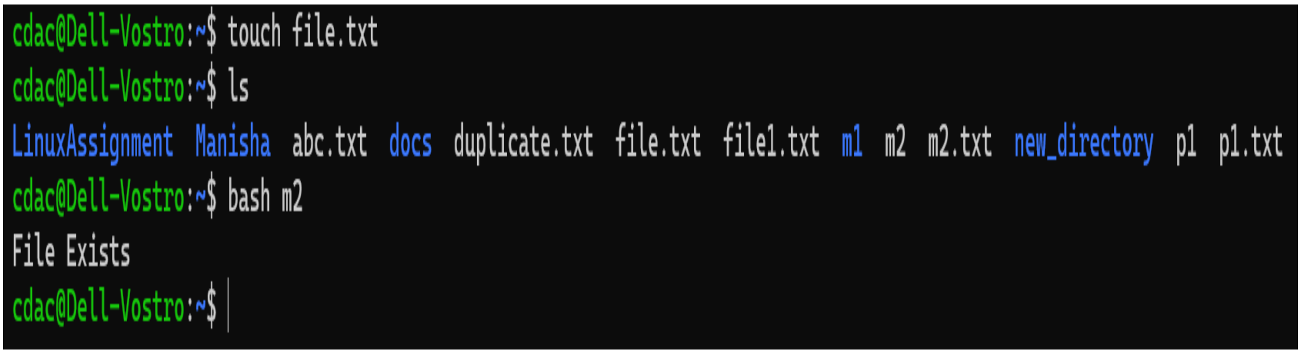
else

echo "File does not Exist"

fi



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
File Exists
cdac@Dell-Vostro:~$
```



```
cdac@Dell-Vostro:~$ touch file.txt
cdac@Dell-Vostro:~$ ls
LinuxAssignment  Manisha  abc.txt  docs  duplicate.txt  file.txt  file1.txt  m1  m2  m2.txt  new_directory  p1  p1.txt
cdac@Dell-Vostro:~$ bash m2
File Exists
cdac@Dell-Vostro:~$
```

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.

→ echo "Enter a number:"

read num

if [$num -gt 10]

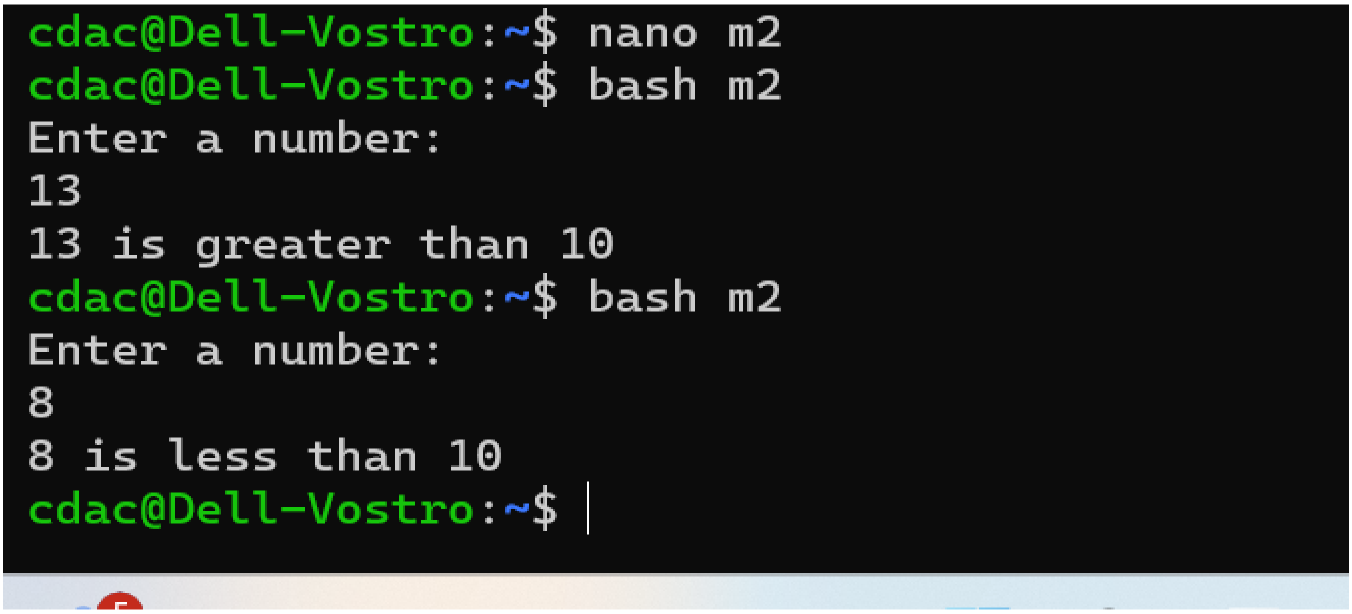
then

echo "$num is greater than 10"

else

echo "$num is less than 10"

fi



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Enter a number:
13
13 is greater than 10
cdac@Dell-Vostro:~$ bash m2
Enter a number:
8
8 is less than 10
cdac@Dell-Vostro:~$
```

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.

→for ((i=1; i<=5; i++))

do

echo "----Table of $i ----- "

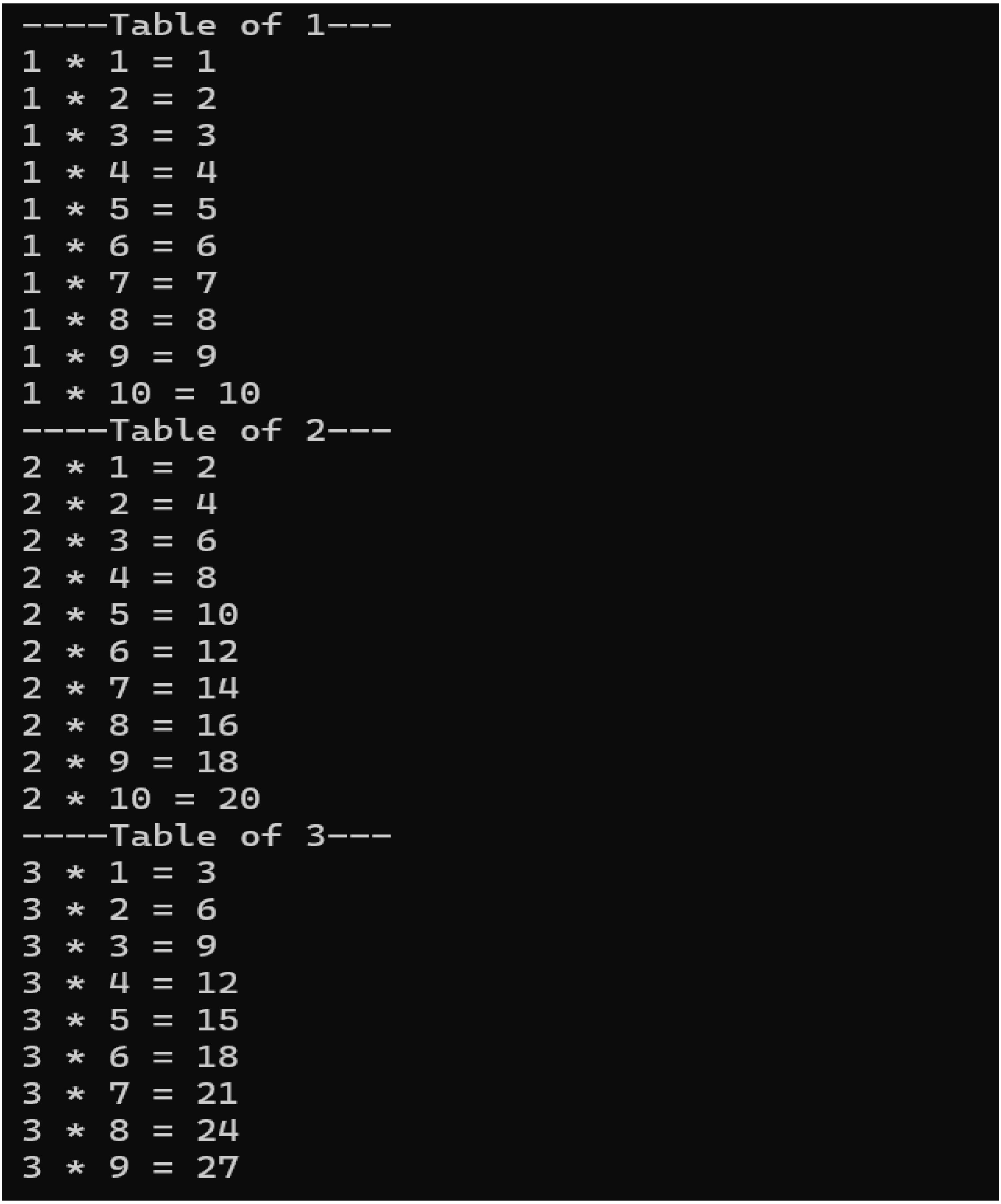
for ((j=1; j<=10; j++))

do

echo "$i * $j = $(($i*$j))"

done

done



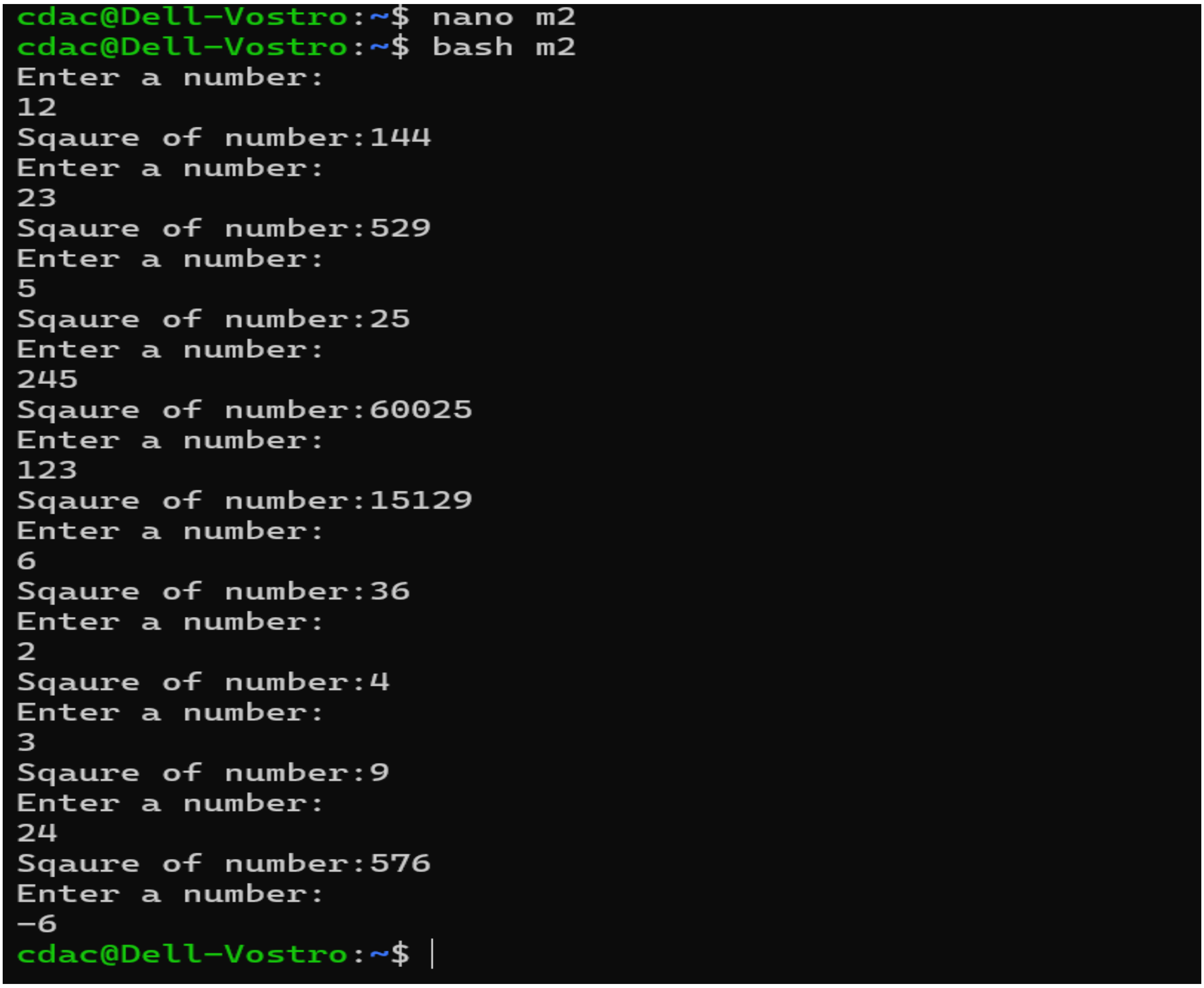
```
----Table of 1---
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
1 * 4 = 4
1 * 5 = 5
1 * 6 = 6
1 * 7 = 7
1 * 8 = 8
1 * 9 = 9
1 * 10 = 10
----Table of 2---
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20
----Table of 3---
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
```

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.

→while [1]

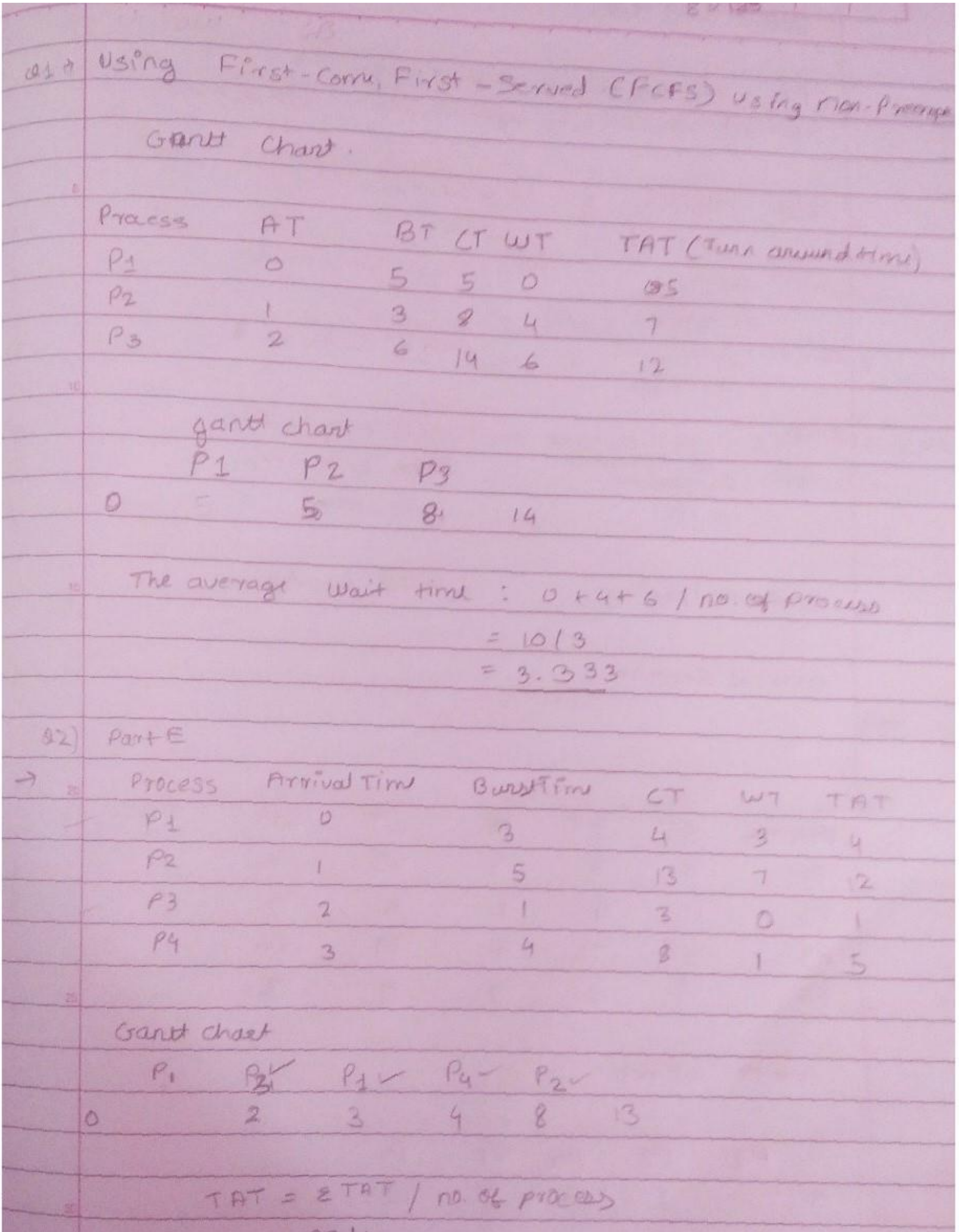
do

```
echo "Enter

a number:"

read num

if (( $num < 0 ))

then

break

else

echo "Sqaure of number:"$(($num*$num))

fi

done
```



```
cdac@Dell-Vostro:~$ nano m2
cdac@Dell-Vostro:~$ bash m2
Enter a number:
12
Sqaure of number:144
Enter a number:
23
Sqaure of number:529
Enter a number:
5
Sqaure of number:25
Enter a number:
245
Sqaure of number:60025
Enter a number:
123
Sqaure of number:15129
Enter a number:
6
Sqaure of number:36
Enter a number:
2
Sqaure of number:4
Enter a number:
3
Sqaure of number:9
Enter a number:
24
Sqaure of number:576
Enter a number:
-6
cdac@Dell-Vostro:~$
```

Part E:



Q1 → Using First-Come, First-Served (FCFS) using non-Preempt

Gantt Chart.

Process	AT	BT	CT	WT	TAT (Turn around time)
P1	0	5	5	0	5
P2	1	3	8	4	7
P3	2	6	14	6	12

gantt chart

P1 P2 P3

0 5 8 14

The average wait time : 0 + 4 + 6 / no. of process

= 10 / 3

= 3.333

Q2) Part E

→

Process	Arrival Time	Burst Time	CT	WT	TAT
P1	0	3	4	3	4
P2	1	5	13	7	12
P3	2	1	3	0	1
P4	3	4	8	1	5

Gantt chart

P1 P3 ✓ P1 ✓ P4 ✓ P2 ✓

0 2 3 4 8 13

TAT = ΣTAT / no. of process

Process	AT	BT	PT	CT	WT
P1	0	6	3	12	6
P2	1	4	1	5	0
P3	2	7	4	19	10
P4	3	2	2	7	2

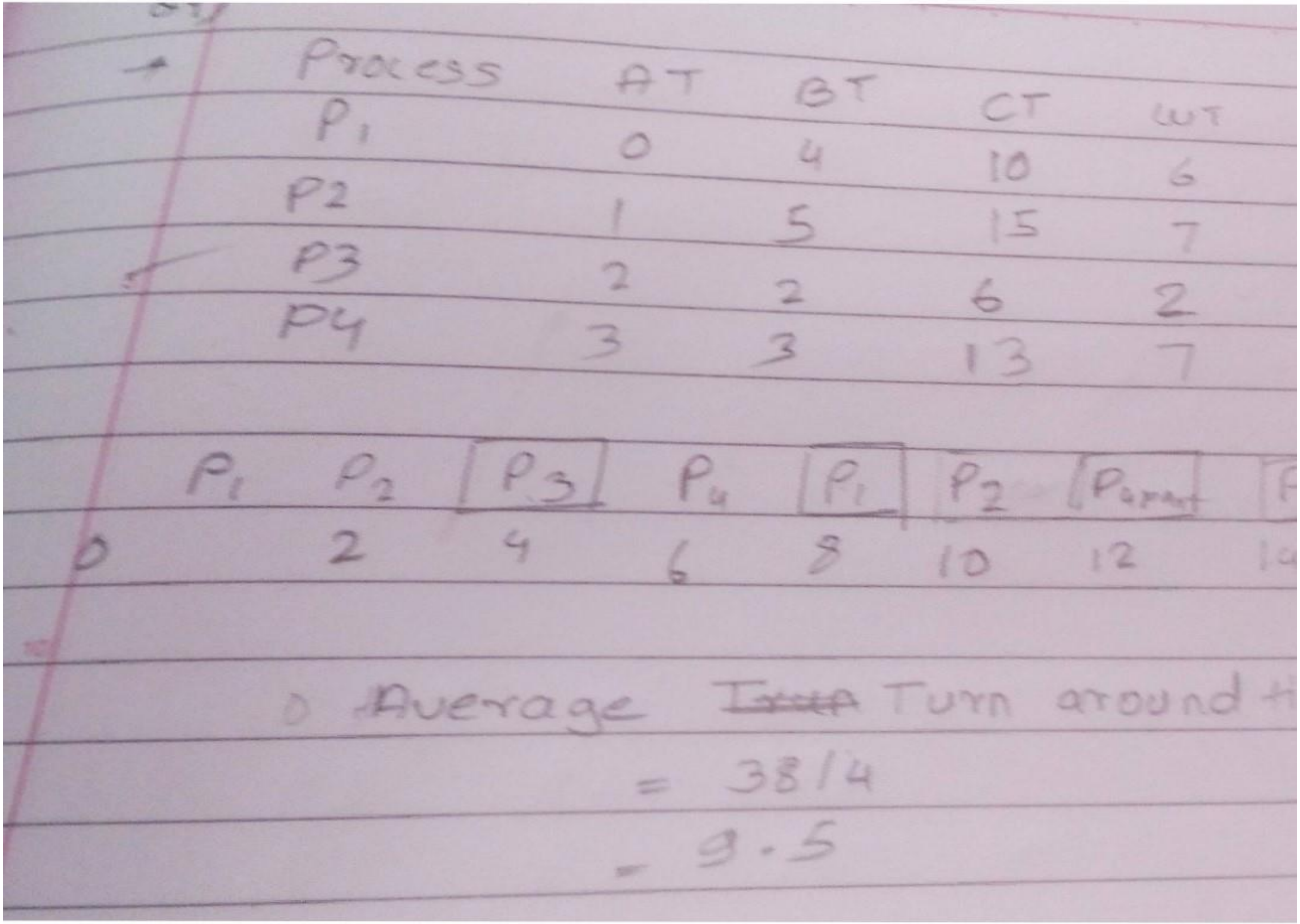
Gantt chart.

P1 P2 P4 P1 P3

0 1 5 7 12 19

Average wait time = ΣWT / No. of

= 18/4 = 4.



Process	AT	BT	CT	WT
P1	0	4	10	6
P2	1	5	15	7
P3	2	2	6	2
P4	3	3	13	7

P1 P2 $\boxed{P_3}$ P4 $\boxed{P_1}$ $\boxed{P_2}$ $\boxed{P_4}$ $\boxed{P}$

0 2 4 6 8 10 12 1

o Average ~~[illegible]~~ Turn around t

= 38/4

= 9.5