What will the following commands do

Sr.no	Command	Description
1	echo "Hello,World!"	**echo** command is used to print string in the terminal
2	name="Productive"	This create shell variable named ‘name’ and assigns it the value Productive
3	touch file.txt	Touch command used to create file. Here it create empty file named file.txt
4	ls -a	**ls** command used to display all files and directories and **-a** flag use to display hidden file as well
5	rm file.txt	Remove the file **file.txt**
6	cp file1.txt file2.txt	Cp command used for copy content from one file to another. Here Copies the contents of **file1.txt** into a new file called **file2.txt**.
7	mv file.txt /path/to/directory/	Moves **file.txt** into the specified directory
8	chmod 755 script.sh	Changes permissions of **script.sh** to Owner: **read, write, execute** Group: **read, execute** Others: **read, execute**
9	grep "pattern" file.txt	It search the word **Pattern in file.txt**
10	kill PID	It terminates the process with the given **Process ID (PID)**.
11	mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt	● First it creates directory i.e. mydir then it enter into mydir ● Second it create empty i.e. **file.txt** ● Third it write **Hello, World!** Into **file.txt** ● At end it display content of **file.txt** file
12	ls -l | grep ".txt"	● Lists files in long format ● Filters output to show only **files containing .txt**
13	cat file1.txt file2.txt | sort | uniq	Combine content **of file1.txt** and **file2.txt** Sorts lines alphabetically and then remove duplicate line
14	ls -l | grep "^d"	● Lists files in long format

		● Filters only lines starting with **d**
15	grep -r "pattern" /path/to/directory/	It search for **pattern** in all files inside the given directory
16	cat file1.txt file2.txt | sort | uniq –d	It combine file1 and file2 and sort both file and only print duplicate lines
17	chmod 644 file.txt	Changes permissions of **file.txt** to: Owner: **read, write** Group: **read** Others: **read**
18	cp -r source_directory destination_directory	It Copies entire **source_directory** into **destination_directory**
19	find /path/to/search -name "*.txt"	Searches for all files ending in **`.txt`** inside the specified directory and subdirectories.
20	chmod u+x file.txt	It allow **execute permission** for the **file owner (user)** on **file.txt**.
21	echo $PATH	It print current Path environment variable

Identify True or False

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

Answer: - True

2) mv is used to move files and directories.

Answer: - True

3) cd is used to copy files and directories

Answer: - False

4) pwd stands for "print working directory" and displays the current directory.

Answer: - False

5) grep is used to search for patterns in files.

Answer: - True

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

Answer: - True

7) mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist

Answer: - True

8) rm -rf file.txt deletes a file forcefully without confirmation.

Answer: - True

Identify the Incorrect Commands

1) chmodx is used to change file permissions

   Answer: Incorrect
   Correct command: chmod

2) cpy is used to copy files and directories.

   Answer: Incorrect
   Correct Command: cp

3) mkfile is used to create a new file.

   Answer: Incorrect
   Correct command: touch

4) catx is used to concatenate files.

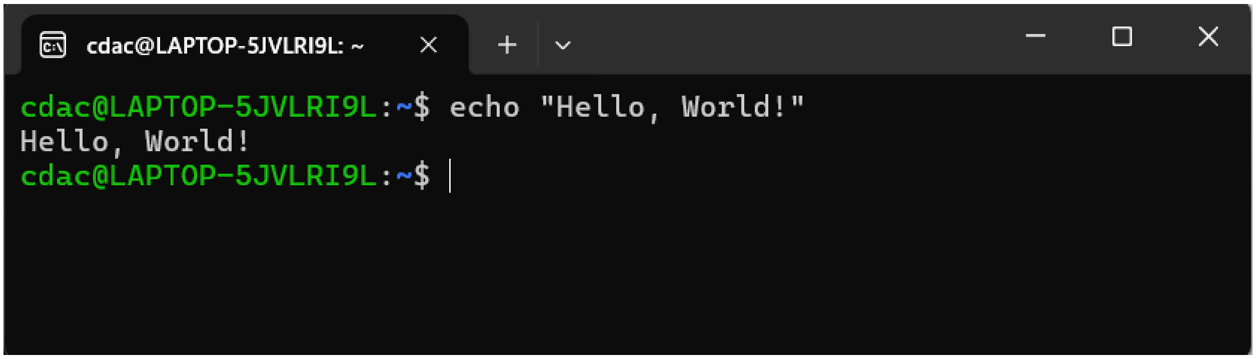
   Answer: Incorrect
   Correct Command: cat

5) rn is used to rename files.

   Answer: Incorrect
   Correct Command: mv

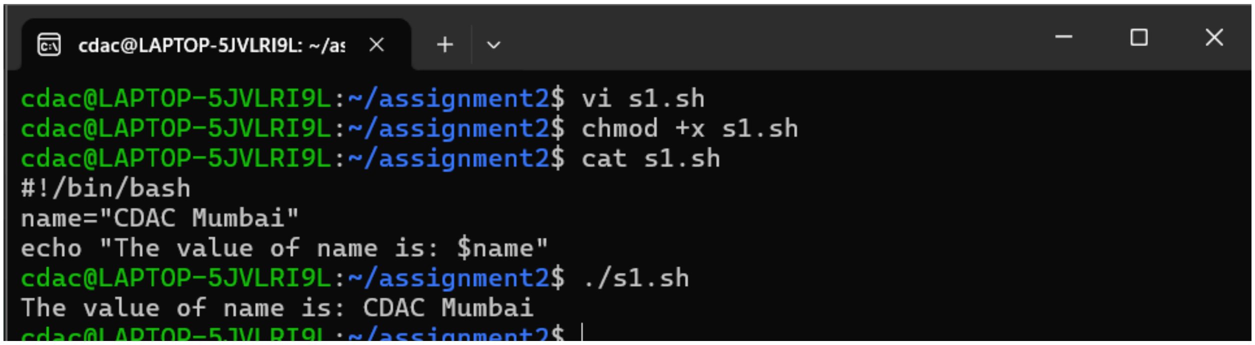
Shell Script Command

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



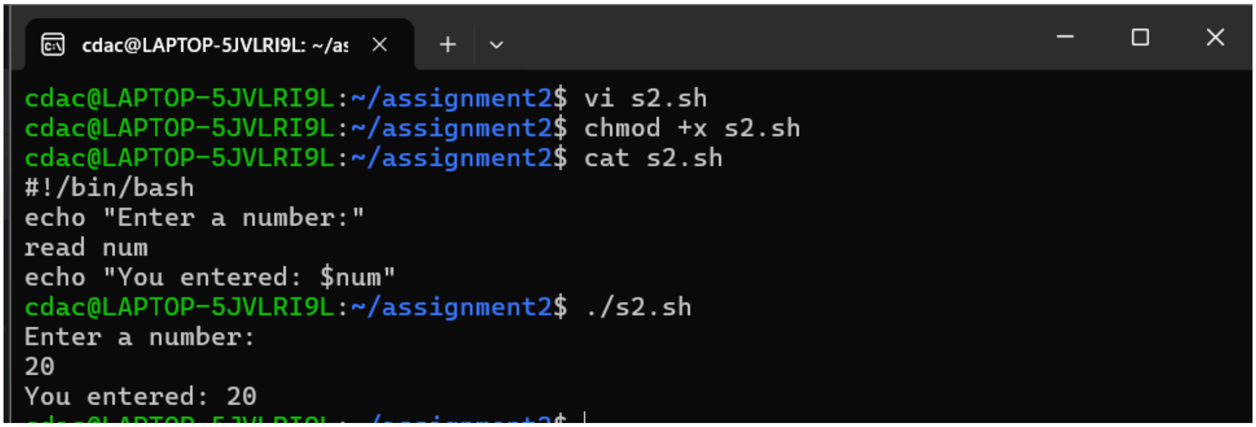
```
cdac@LAPTOP-5JVLRI9L:~$ echo "Hello, World!"
Hello, World!
cdac@LAPTOP-5JVLRI9L:~$
```

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



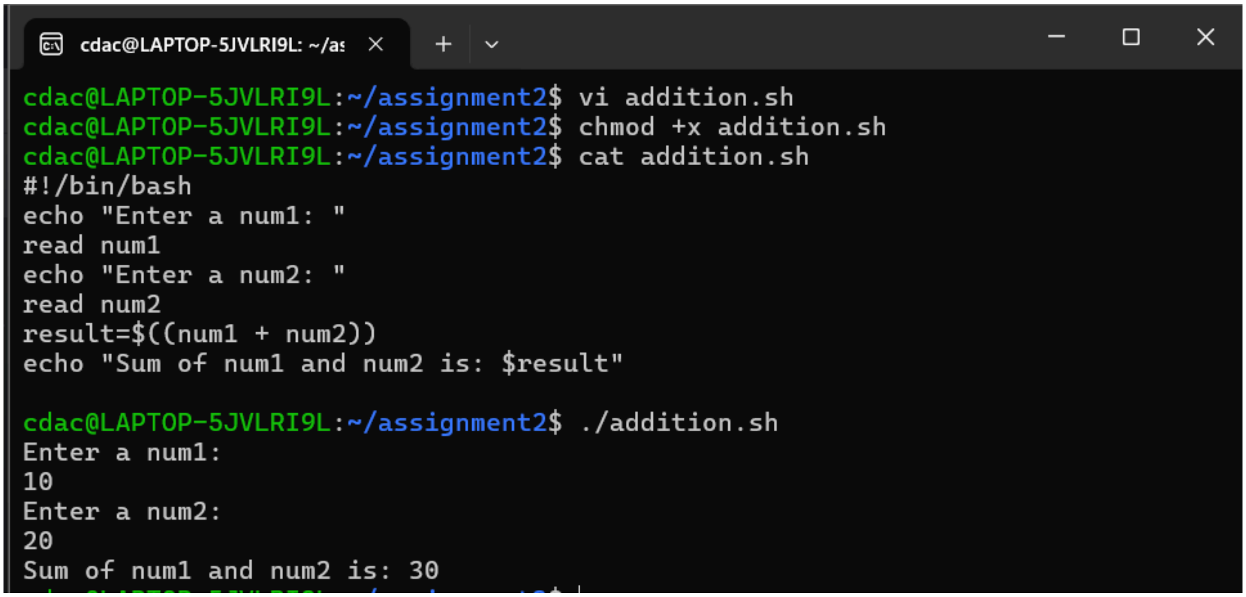
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi s1.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x s1.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat s1.sh
#!/bin/bash
name="CDAC Mumbai"
echo "The value of name is: $name"
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./s1.sh
The value of name is: CDAC Mumbai
cdac@LAPTOP-5JVLRI9L:~/assignment2$
```

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



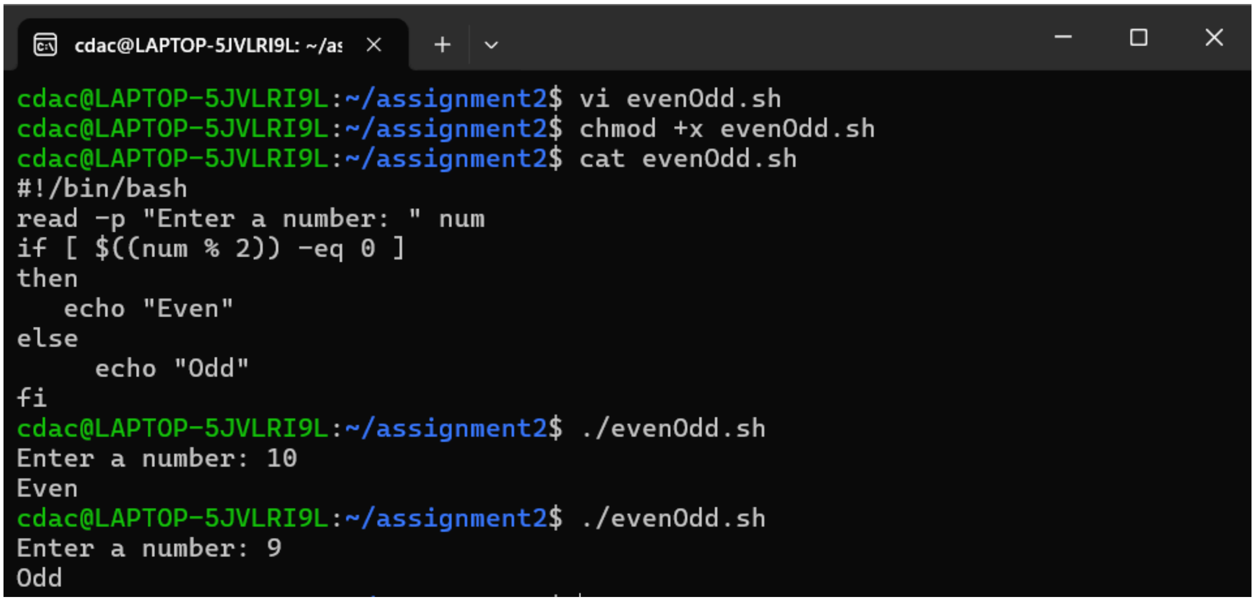
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi s2.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x s2.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat s2.sh
#!/bin/bash
echo "Enter a number:"
read num
echo "You entered: $num"
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./s2.sh
Enter a number:
20
You entered: 20
```

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



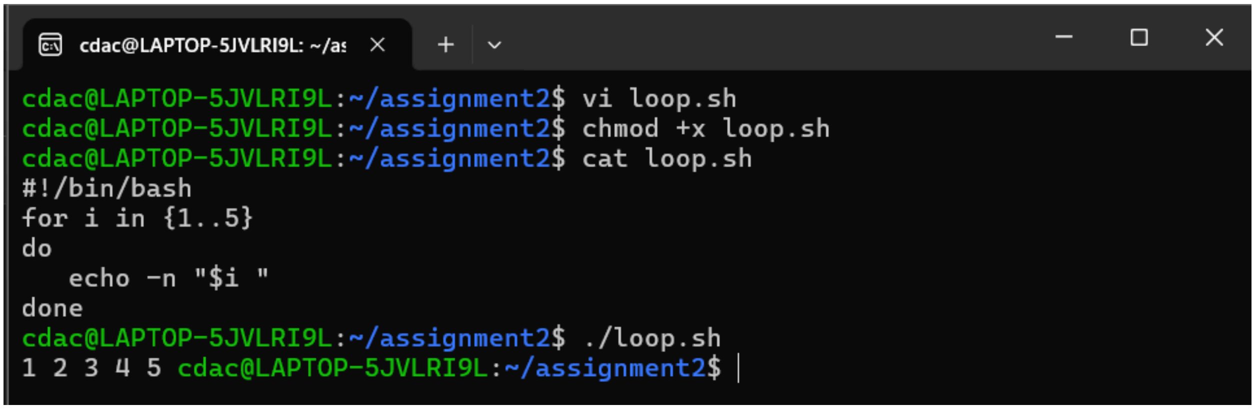
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi addition.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x addition.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat addition.sh
#!/bin/bash
echo "Enter a num1: "
read num1
echo "Enter a num2: "
read num2
result=$((num1 + num2))
echo "Sum of num1 and num2 is: $result"

cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./addition.sh
Enter a num1:
10
Enter a num2:
20
Sum of num1 and num2 is: 30
```

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



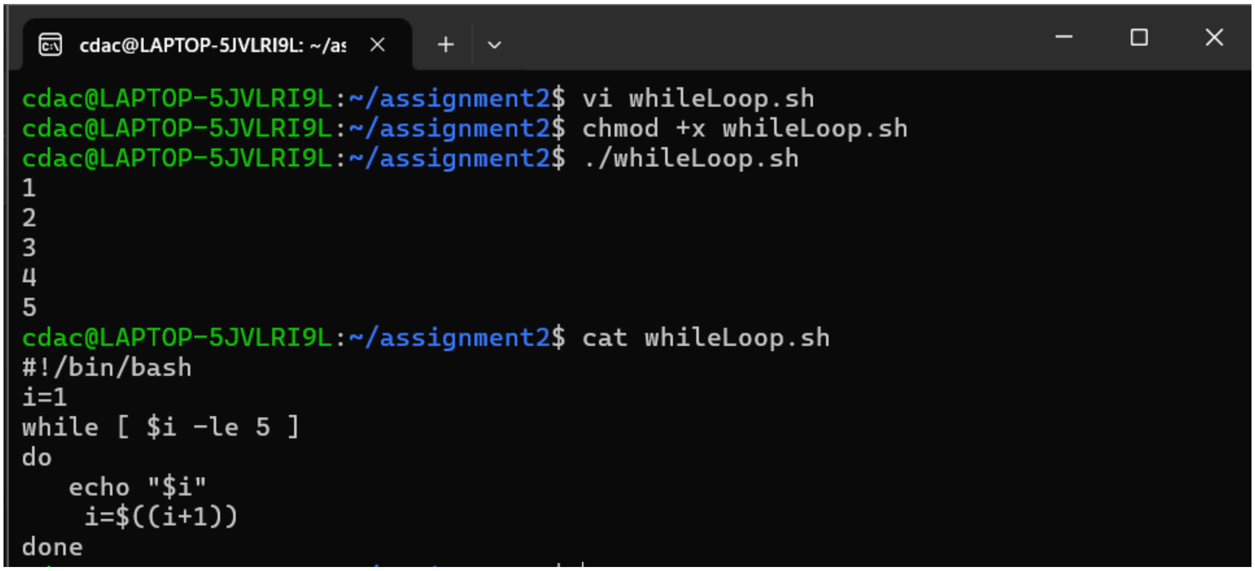
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi evenOdd.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x evenOdd.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat evenOdd.sh
#!/bin/bash
read -p "Enter a number: " num
if [ $((num % 2)) -eq 0 ]
then
   echo "Even"
else
     echo "Odd"
fi
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./evenOdd.sh
Enter a number: 10
Even
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./evenOdd.sh
Enter a number: 9
Odd
```

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



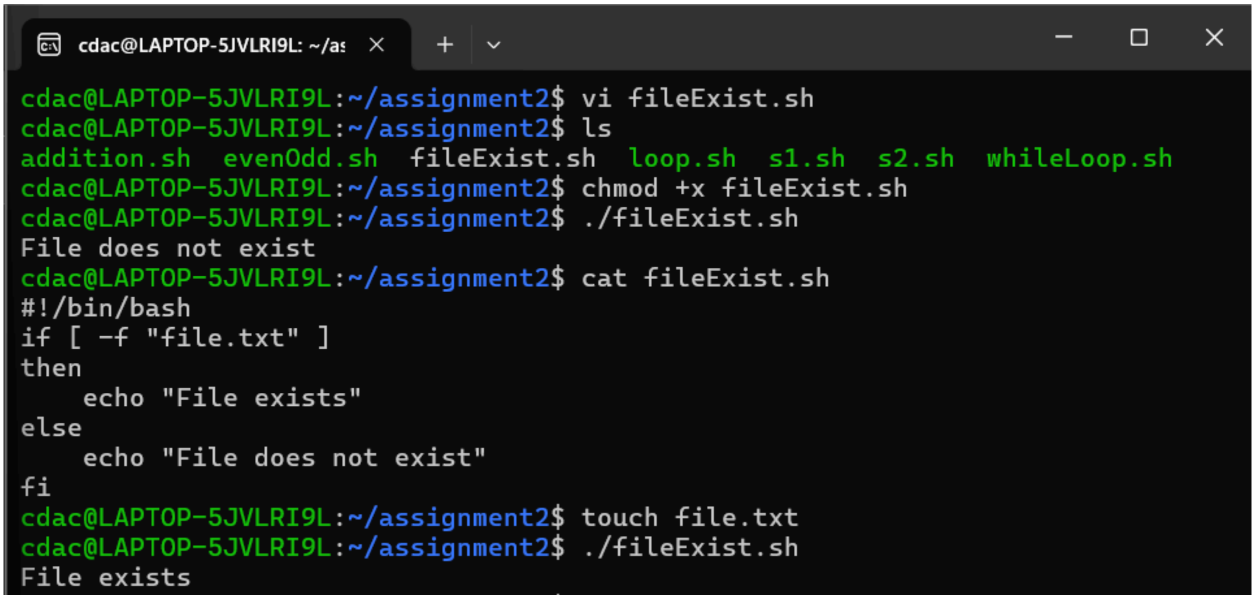
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi loop.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x loop.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat loop.sh
#!/bin/bash
for i in {1..5}
do
   echo -n "$i "
done
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./loop.sh
1 2 3 4 5 cdac@LAPTOP-5JVLRI9L:~/assignment2$ |
```

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



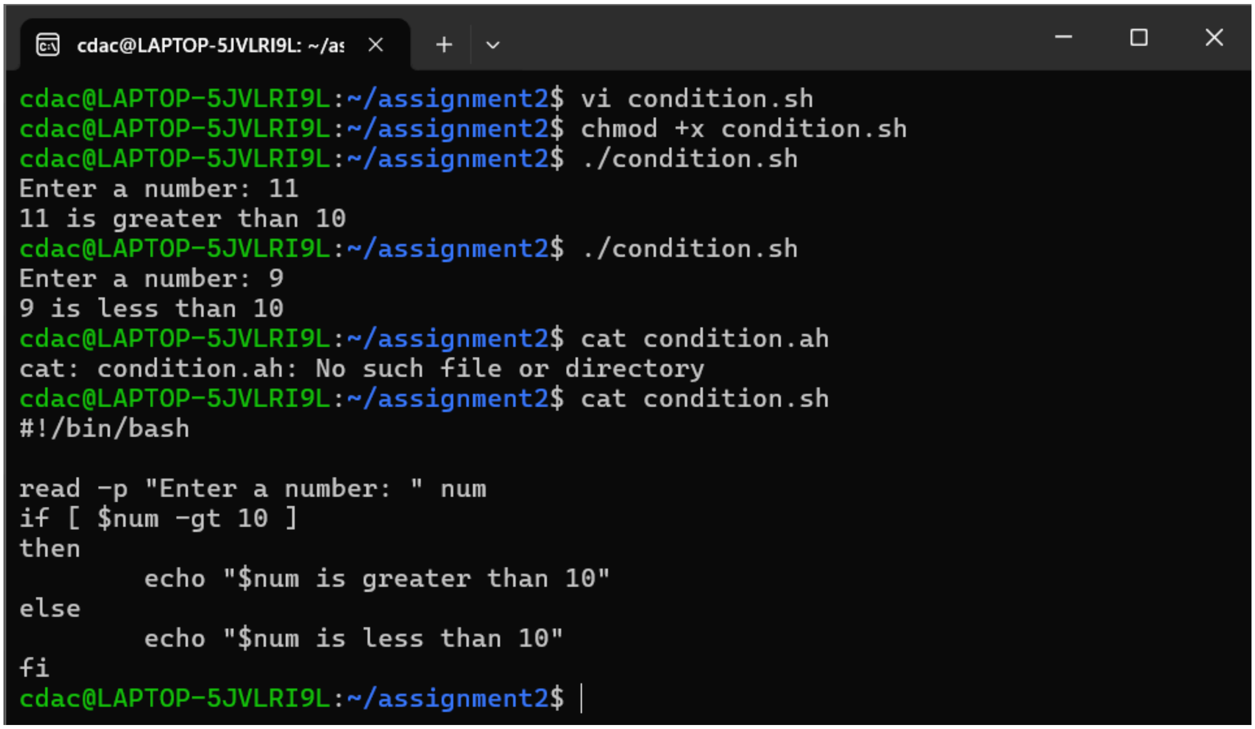
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi whileLoop.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x whileLoop.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./whileLoop.sh
1
2
3
4
5
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat whileLoop.sh
#!/bin/bash
i=1
while [ $i -le 5 ]
do
   echo "$i"
    i=$((i+1))
done
```

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".



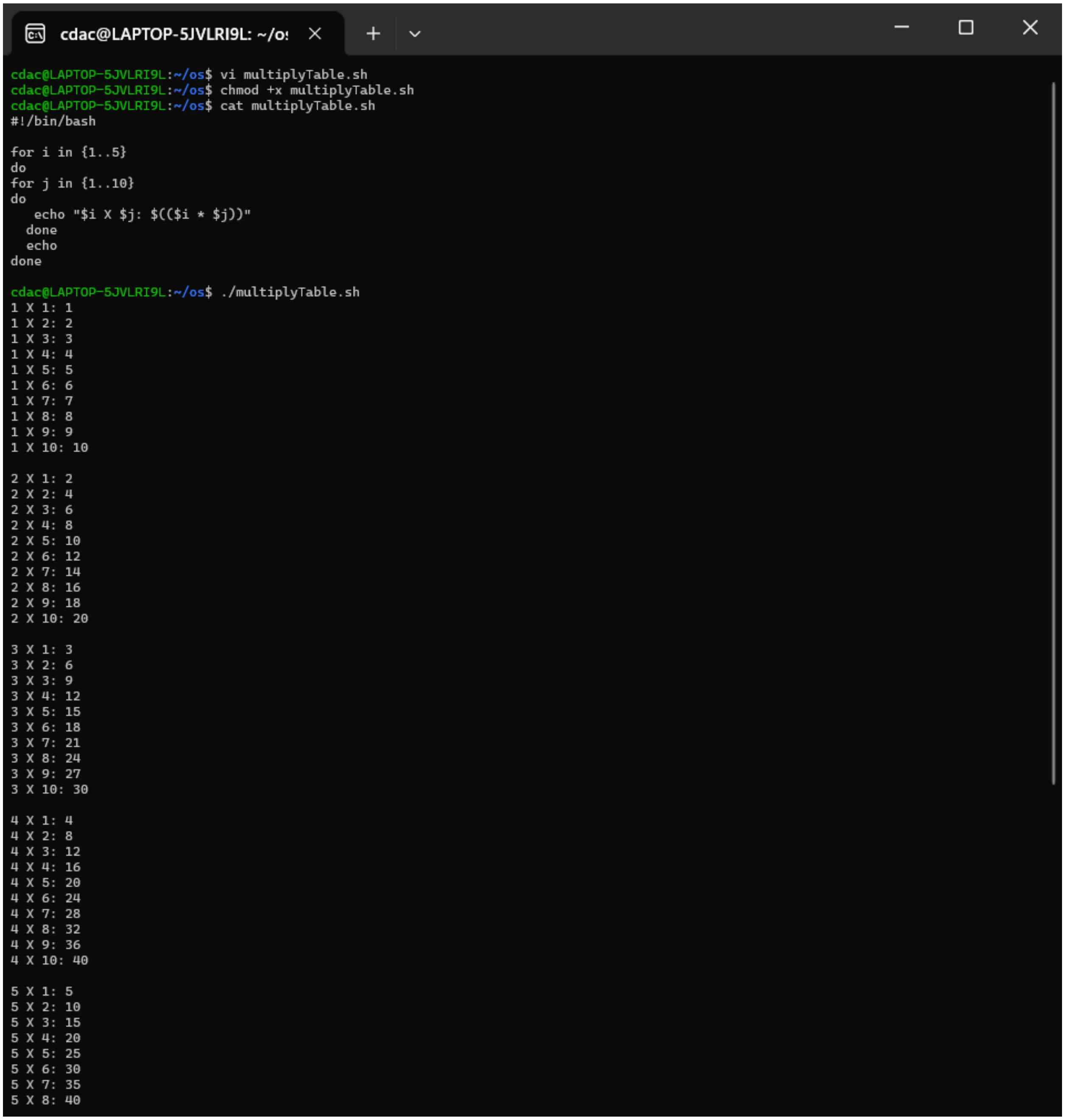
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi fileExist.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ls
addition.sh  evenOdd.sh  fileExist.sh  loop.sh  s1.sh  s2.sh  whileLoop.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x fileExist.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./fileExist.sh
File does not exist
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat fileExist.sh
#!/bin/bash
if [ -f "file.txt" ]
then
    echo "File exists"
else
    echo "File does not exist"
fi
cdac@LAPTOP-5JVLRI9L:~/assignment2$ touch file.txt
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./fileExist.sh
File exists
```

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.



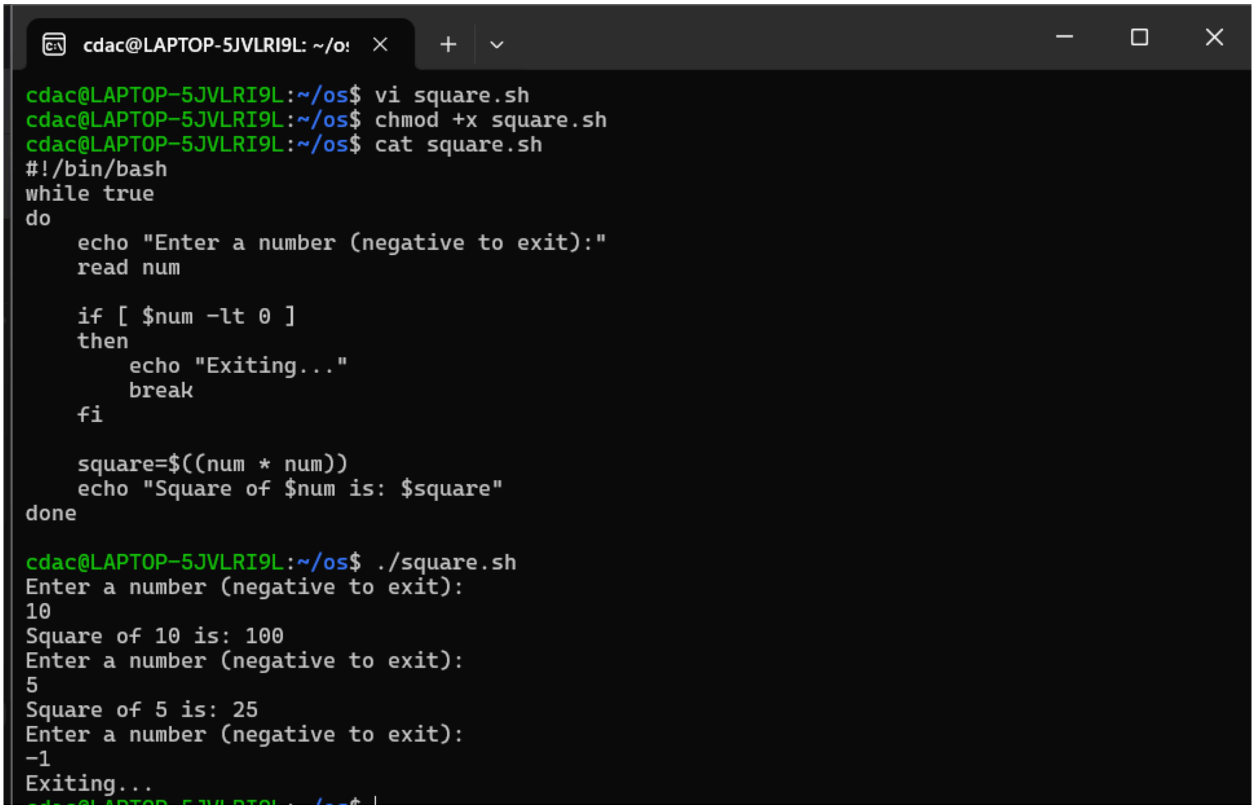
```
cdac@LAPTOP-5JVLRI9L:~/assignment2$ vi condition.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ chmod +x condition.sh
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./condition.sh
Enter a number: 11
11 is greater than 10
cdac@LAPTOP-5JVLRI9L:~/assignment2$ ./condition.sh
Enter a number: 9
9 is less than 10
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat condition.ah
cat: condition.ah: No such file or directory
cdac@LAPTOP-5JVLRI9L:~/assignment2$ cat condition.sh
#!/bin/bash

read -p "Enter a number: " num
if [ $num -gt 10 ]
then
        echo "$num is greater than 10"
else
        echo "$num is less than 10"
fi
cdac@LAPTOP-5JVLRI9L:~/assignment2$ 
```

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.

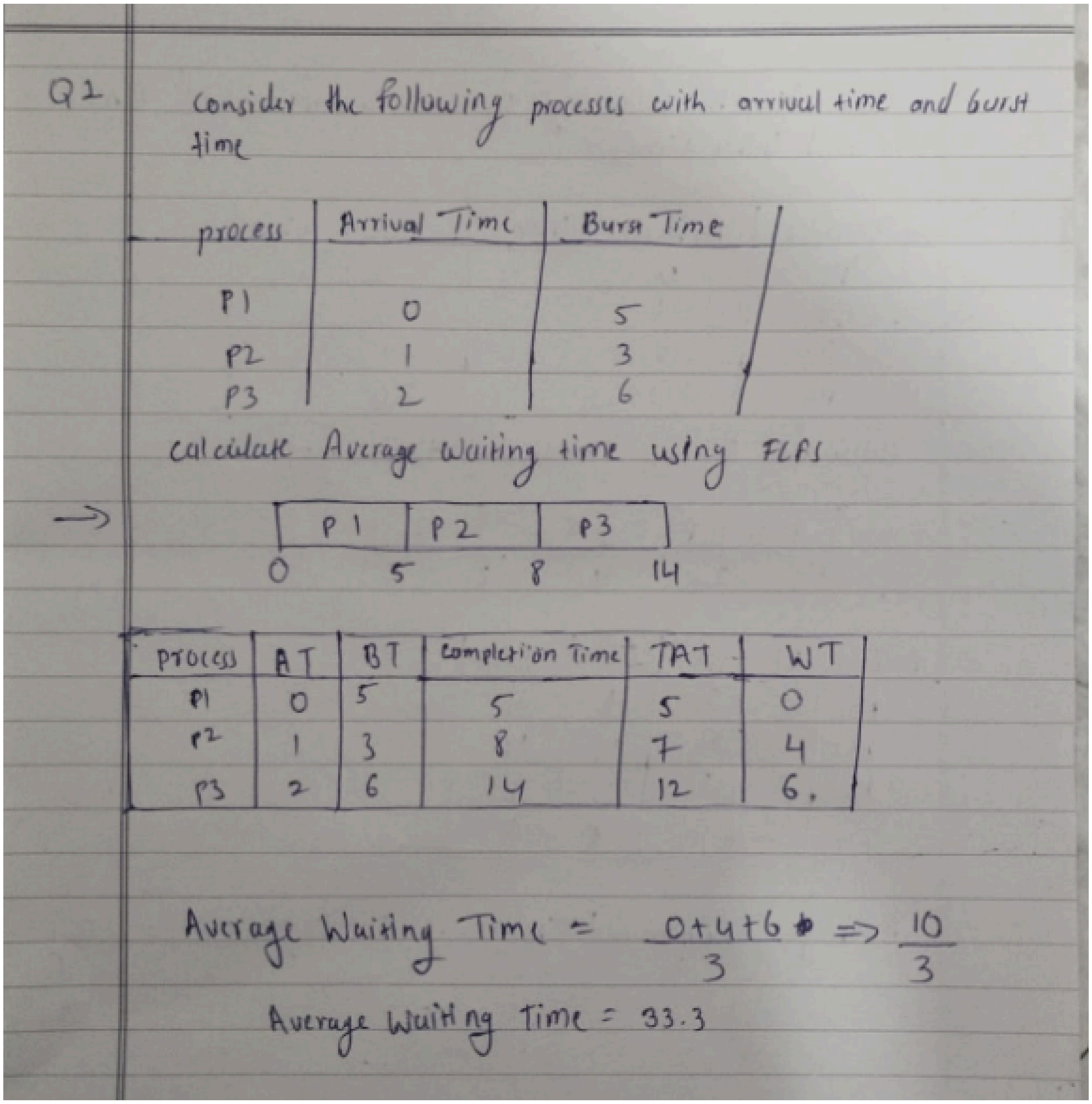


```
cdac@LAPTOP-5JVLRI9L:~/os$ vi multiplyTable.sh
cdac@LAPTOP-5JVLRI9L:~/os$ chmod +x multiplyTable.sh
cdac@LAPTOP-5JVLRI9L:~/os$ cat multiplyTable.sh
#!/bin/bash

for i in {1..5}
do
for j in {1..10}
do
   echo "$i X $j: $(($i * $j))"
  done
  echo
done

cdac@LAPTOP-5JVLRI9L:~/os$ ./multiplyTable.sh
1 X 1: 1
1 X 2: 2
1 X 3: 3
1 X 4: 4
1 X 5: 5
1 X 6: 6
1 X 7: 7
1 X 8: 8
1 X 9: 9
1 X 10: 10

2 X 1: 2
2 X 2: 4
2 X 3: 6
2 X 4: 8
2 X 5: 10
2 X 6: 12
2 X 7: 14
2 X 8: 16
2 X 9: 18
2 X 10: 20

3 X 1: 3
3 X 2: 6
3 X 3: 9
3 X 4: 12
3 X 5: 15
3 X 6: 18
3 X 7: 21
3 X 8: 24
3 X 9: 27
3 X 10: 30

4 X 1: 4
4 X 2: 8
4 X 3: 12
4 X 4: 16
4 X 5: 20
4 X 6: 24
4 X 7: 28
4 X 8: 32
4 X 9: 36
4 X 10: 40

5 X 1: 5
5 X 2: 10
5 X 3: 15
5 X 4: 20
5 X 5: 25
5 X 6: 30
5 X 7: 35
5 X 8: 40
```

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@LAPTOP-5JVLRI9L:~/os$ vi square.sh
cdac@LAPTOP-5JVLRI9L:~/os$ chmod +x square.sh
cdac@LAPTOP-5JVLRI9L:~/os$ cat square.sh
#!/bin/bash
while true
do
    echo "Enter a number (negative to exit):"
    read num

    if [ $num -lt 0 ]
    then
        echo "Exiting..."
        break
    fi

    square=$((num * num))
    echo "Square of $num is: $square"
done

cdac@LAPTOP-5JVLRI9L:~/os$ ./square.sh
Enter a number (negative to exit):
10
Square of 10 is: 100
Enter a number (negative to exit):
5
Square of 5 is: 25
Enter a number (negative to exit):
-1
Exiting...
cdac@LAPTOP-5JVLRI9L:~/os$
```



Q2 consider the following processes with arrival time and burst time

process	Arrival Time	Burst Time
P1	0	5
P2	1	3
P3	2	6

calculate Average waiting time using FCFS

→

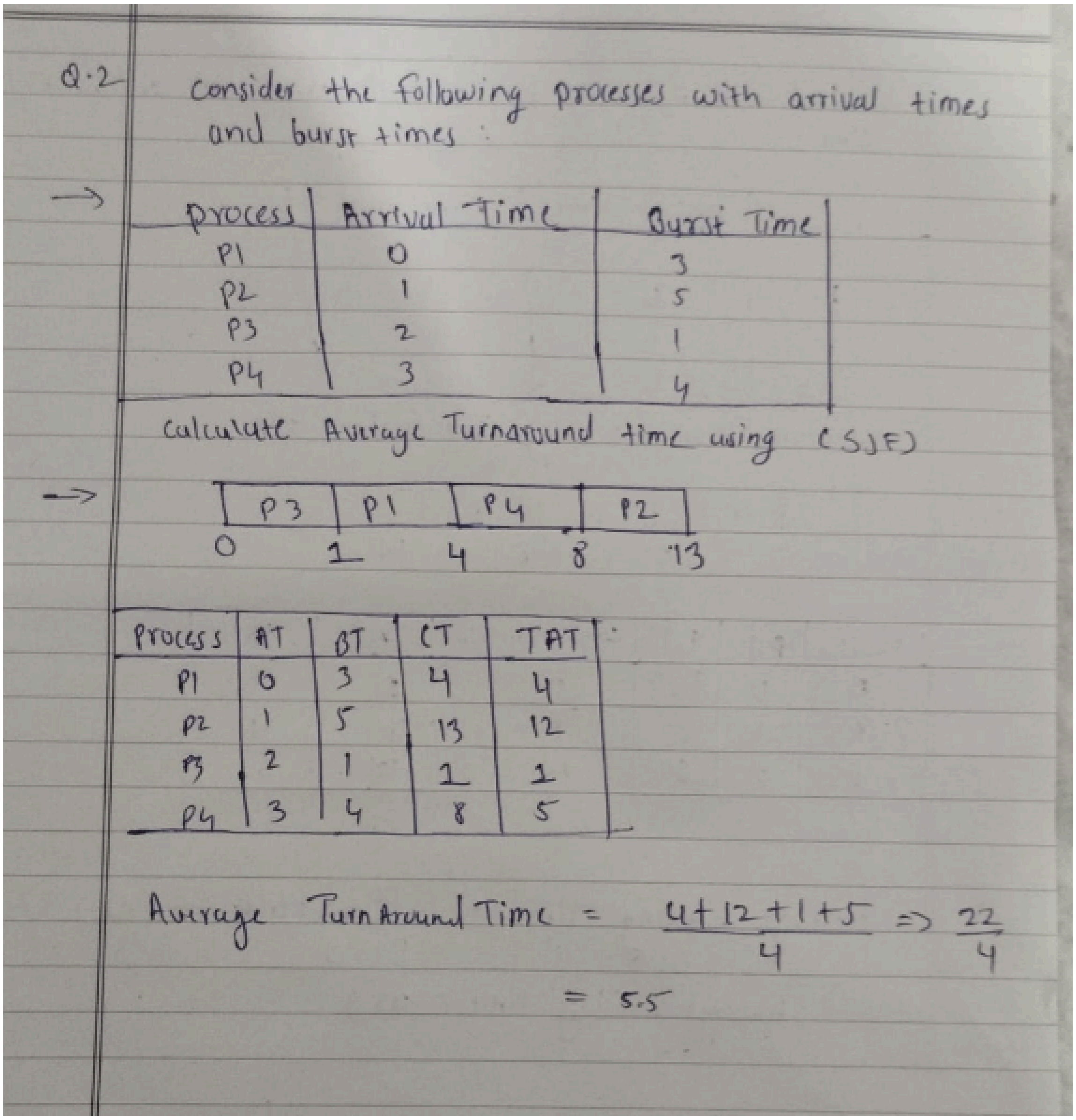
P1	P2	P3

0 5 8 14

process	AT	BT	completion Time	TAT	WT
P1	0	5	5	5	0
P2	1	3	8	7	4
P3	2	6	14	12	6.

Average Waiting Time = $\frac{0+4+6}{3} \Rightarrow \frac{10}{3}$

Average Waiting Time = 33.3



Q.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 Average Turnaround time using (SJF)

→

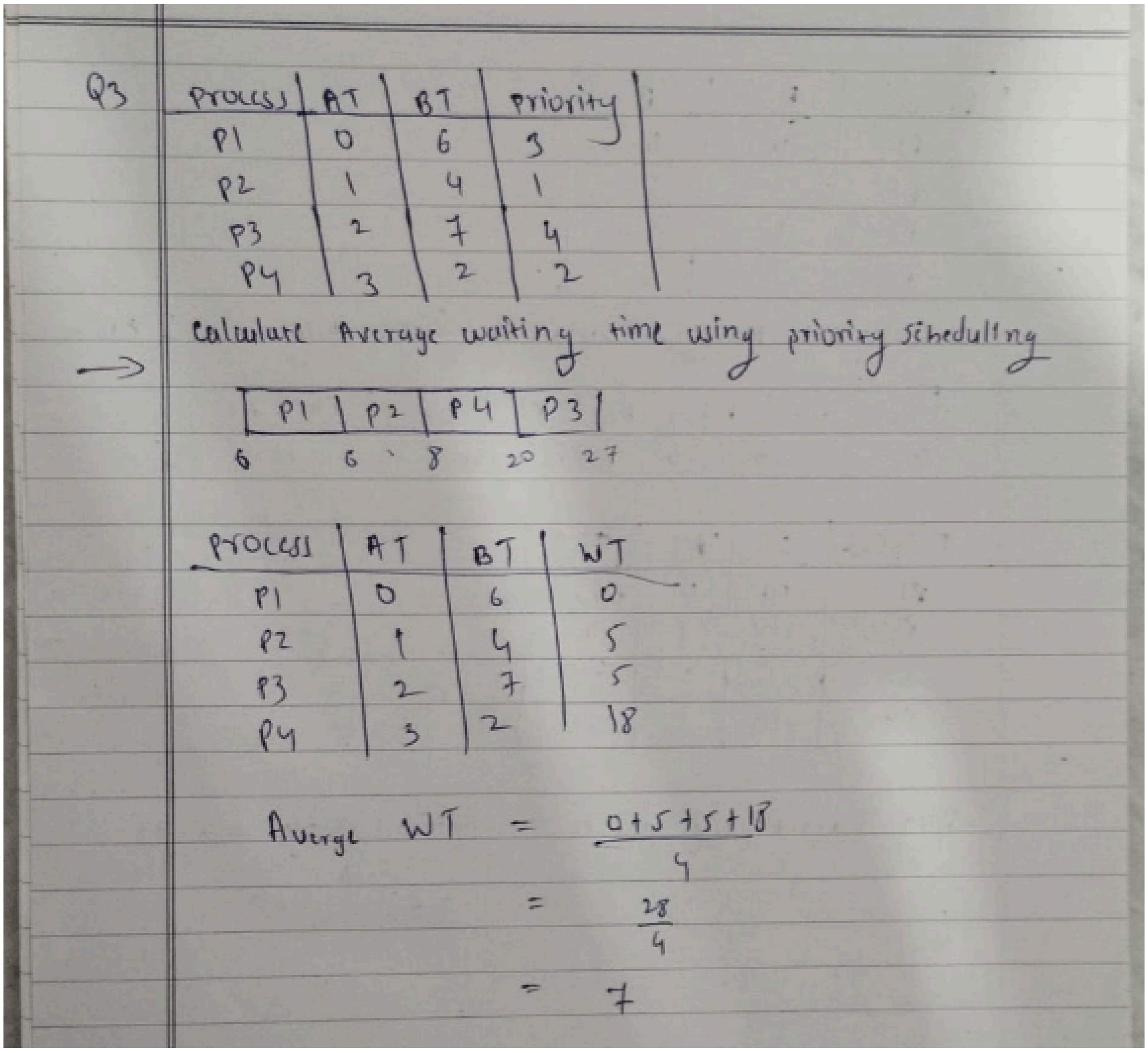
P3	P1	P4	P2

0 1 4 8 13

Process	AT	BT	CT	TAT
P1	0	3	4	4
P2	1	5	13	12
P3	2	1	1	1
P4	3	4	8	5

Average Turn Around Time = $\frac{4+12+1+5}{4} \Rightarrow \frac{22}{4}$

$= 5.5$



Q3

process	AT	BT	priority
P1	0	6	3
P2	1	4	1
P3	2	7	4
P4	3	2	2

calculate Average waiting time using priority scheduling

→

P1	P2	P4	P3

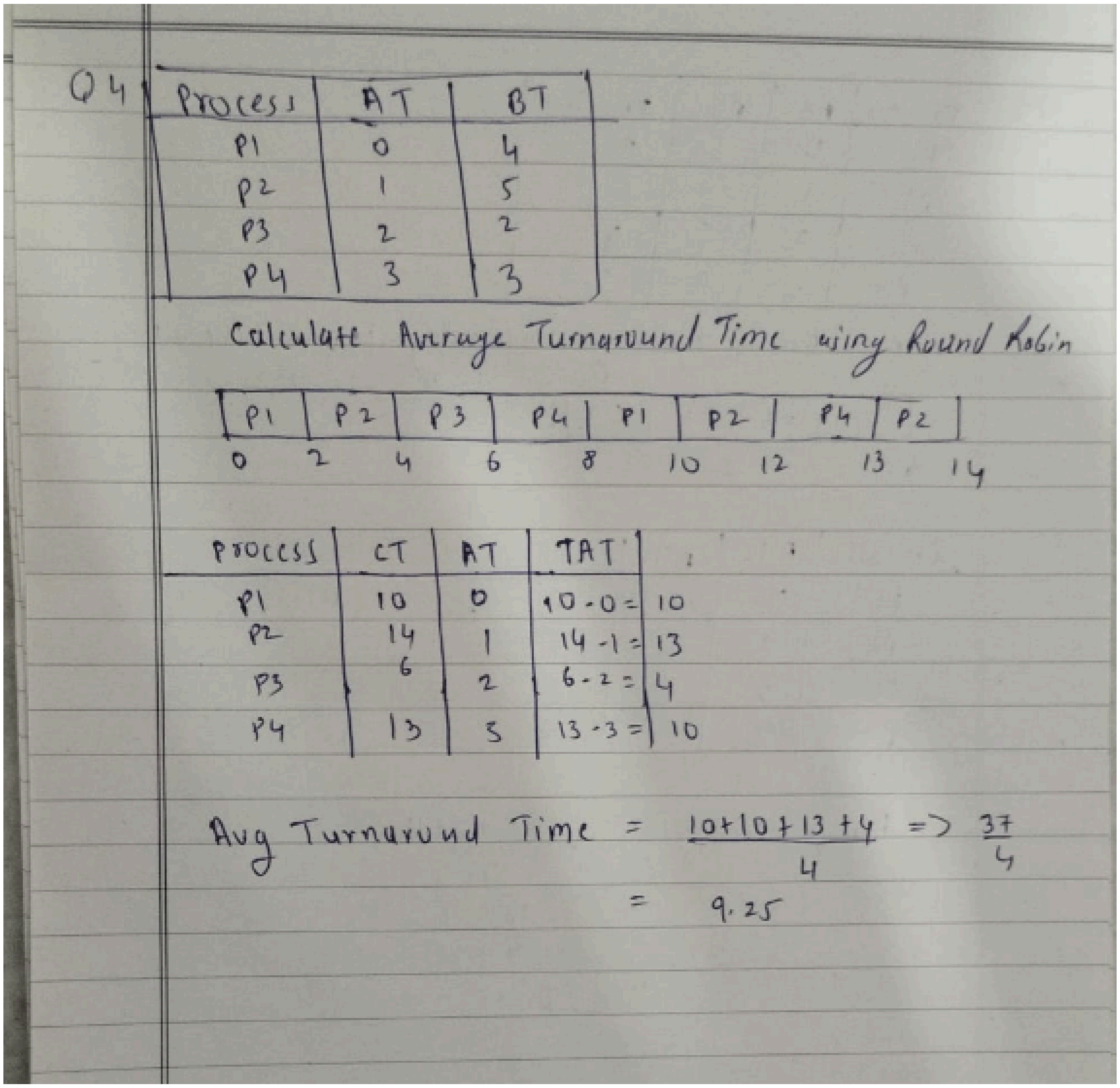
6 6 8 20 27

process	AT	BT	WT
P1	0	6	0
P2	1	4	5
P3	2	7	5
P4	3	2	18

Averge WT $= \frac{0+5+5+18}{4}$

$= \frac{28}{4}$

$= 7$



Q4

Process	AT	BT
P1	0	4
P2	1	5
P3	2	2
P4	3	3

Calculate Average Turnaround Time using Round Robin

P1	P2	P3	P4	P1	P2	P4	P2

0 2 4 6 8 10 12 13 14

Process	CT	AT	TAT	
P1	10	0	10 - 0 =	10
P2	14	1	14 - 1 =	13
P3	6	2	6 - 2 =	4
P4	13	3	13 - 3 =	10

$$\text{Avg Turnaround Time} = \frac{10+10+13+4}{4} \Rightarrow \frac{37}{4}$$

$$= 9.25$$