MIHIR KUMAR TIJARE

JUHU CDAC

LINUX ASSIGNMENT -2

ASSIGNMENT 2 – PART A

1-print "Hello, World!" to the console:

2-It sets the variable name to "Productive" we caan access with echo "$name"

3- create a file

4 list all file in a directory

5 – rm remove the file.txt

6- Copies the contents of file1.txt to file2.txt.

7-Moves file.txt to the specified directory.

8 –755 means

7 read write exexcute The owner can read, write, and execute the script

5 read execute The group and others can only read and execute it.

5 read execute .

9 – it terminate the process mean stop

10 –

Explanation:

mkdir mydir → Creates a new directory named mydir.

cd mydir → Moves into the mydir directory.

touch file.txt → Creates an empty file named file.txt.

echo "Hello, World!" > file.txt → Writes "Hello, World!" into file.txt.

cat file.txt → Displays the contents of file.txt.

11 –

List directoris and greg will do find pattern “|” – will do piping

12

It will copy file into file 2 and uniq will find non duplicat and sort will a arrange in a manner

13

List all directories of pattern name start with d

14•

grep -r "pattern" /path/to/directory/ o Here grep command is used to recursively search for given pattern “pattern” in the directory /path/to/directory, provided that such directory exists in first place. The output will display the lines containing the “pattern” pattern in it.

15 –

It combine the file 1 and file 2 and sort the text and uniq -d display the duplicate file

16

644

6 – read ,write by owner

4 - read only by growup

4 – read only by other

17

Find the file from path or directory where to search begin with extension of .txt

18

Chmod – change file permission

U + x - execute (x) permission to the owner (u) of the file.

19

$ path – is a variable

----------xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx----------

ASSIGNMENT 2 – PART B

Identify True or False

• ls is used to list files and directories in a directory. – **True**

• mv is used to move files and directories. **– True**

• cd is used to copy files and directories. **– False,**

it is used to change the directory.

• pwd stands for "print working directory" and displays the

current directory. – **True**

• grep is used to search for patterns in files. – **True**

• chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and

execute permissions to group and others. **– True**

• mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1

if directory1 does not exist. **— True**

• rm -rf file.txt deletes a file forcefully without confirmation**. –False,**

-r (recursive option) is used for deleting directories, not files

----------XXXXXXXXXXXXXX----------

Identify the Incorrect Commands:

• chmodx is used to change file permissions.

chmod command is used to change file permissions.

• cpy is used to copy files and directories.

cp command is used to copy files and directories.

• mkfile is used to create a new file.

touch command is used to create a new file. mkdir command is used to create a new

directory.

• catx is used to concatenate files.

cat command is used to concatenate files.

• rn is used to rename files.

mv command is used to rename files when 2 files names are passed as arguments

----------xxxxxxxxxxxxxxx----------

ASSIGNMENT 2 – PART C

Q1. Write a shell script that prints "Hello, World!" to the terminal

```
cat: linuxAssignment: Is a directory
tarun@hp93:~$ cd linuxAssignment
tarun@hp93:~/linuxAssignment$ q1.sh
q1.sh: command not found
tarun@hp93:~/linuxAssignment$ touch q1.sh
tarun@hp93:~/linuxAssignment$ nano q1.sh
tarun@hp93:~/linuxAssignment$ cat q1.sh
hello world!
tarun@hp93:~/linuxAssignment$ touch q2.sh
```

• **Q2. Declare a variable named "name" and assign the value "CDAC Mumbai" to it .**

Print the value of the variable.

```
tarun@hp93:~/linuxAssignment$ touch q2.sh
tarun@hp93:~/linuxAssignment$ nano q2.sh
tarun@hp93:~/linuxAssignment$ nano name.sh
tarun@hp93:~/linuxAssignment$ cat name
cat: name: No such file or directory
tarun@hp93:~/linuxAssignment$ cat name.sh
name = cdac mumbai
tarun@hp93:~/linuxAssignment$ bash name.sh
```

• **Q3. Write a shell script that takes a number as input from the user and prints it.**

```
echo "You entered: $number"


tarun@hp93:~/linuxAssignment$ 56
56: command not found
tarun@hp93:~/linuxAssignment$ nano q3.sh
tarun@hp93:~/linuxAssignment$ chmod +x q3.sh
tarun@hp93:~/linuxAssignment$ bash q3.sh
Enter a number: 590
You entered: 590
tarun@hp93:~/linuxAssignment$ touch q4.sh
```

- **Q4. Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result**

```
tarun@hp93: ~/linuxAs
tarun@hp93:~/linuxAssignment$ bash q4.sh
1st
5
2nd
3
sum:  ' $a + $b ': No such file or directory
sum of 5 and 3  is
tarun@hp93:~/linuxAssignment$ nano q4.sh
tarun@hp93:~/linuxAssignment$ y
y: command not found
tarun@hp93:~/linuxAssignment$
tarun@hp93:~/linuxAssignment$ chmod +x q4.sh
tarun@hp93:~/linuxAssignment$ cat q4.sh
echo "1st "
read a
echo "2nd"
read b
sum  ' $a + $b '
echo "sum of $a and $b  is $sum"

tarun@hp93:~/linuxAssignment$ nano q4.sh
tarun@hp93:~/linuxAssignment$ cat q4.sh
echo "1st "
read a
echo "2nd"
read b
sum = ' $a + $b '
echo "sum of $a and $b  is $sum"

tarun@hp93:~/linuxAssignment$ nano q4.sh
tarun@hp93:~/linuxAssignment$ bash q4.sh
Enter the first number:
5
Enter the second number:
3
The sum of 5 and 3 is: 8
tarun@hp93:~/linuxAssignment$
tarun@hp93:~/linuxAssignment$
tarun@hp93:~/linuxAssignment$ nano q5.sh
```

- **Q5. Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".**

```
5
q5.sh: line 3: [ expr $a % 2 : command not found
5 is odd
tarun@hp93:~/linuxAssignment$ nano q5.sh
tarun@hp93:~/linuxAssignment$ bash q5.sh
Enter a number:
6
6 is even
tarun@hp93:~/linuxAssignment$ nano q6.sh
```

• **Q6. Write a shell script that uses a for loop to print numbers from 1 to 5.**

```
for i in {1..5}
do
  echo $i
done
```

```
bash: q6.sh: No such file or directory
tarun@hp93:~/linuxAssignment$ nano q6.sh
tarun@hp93:~/linuxAssignment$ bash q6.sh
1
2
3
4
5
tarun@hp93:~/linuxAssignment$ nano q7.sh
```

• **Q7. Write a shell script that uses a while loop to print numbers from 1 to 5.**

```
tarun@hp93:~/linuxAssignment$ nano q7 .sh
tarun@hp93:~/linuxAssignment$ nano q7.sh
tarun@hp93:~/linuxAssignment$ a=1
while [ $a -lt 6 ]
do
        echo $a
a=$((a + 1))
done
1
2
3
4
5
```

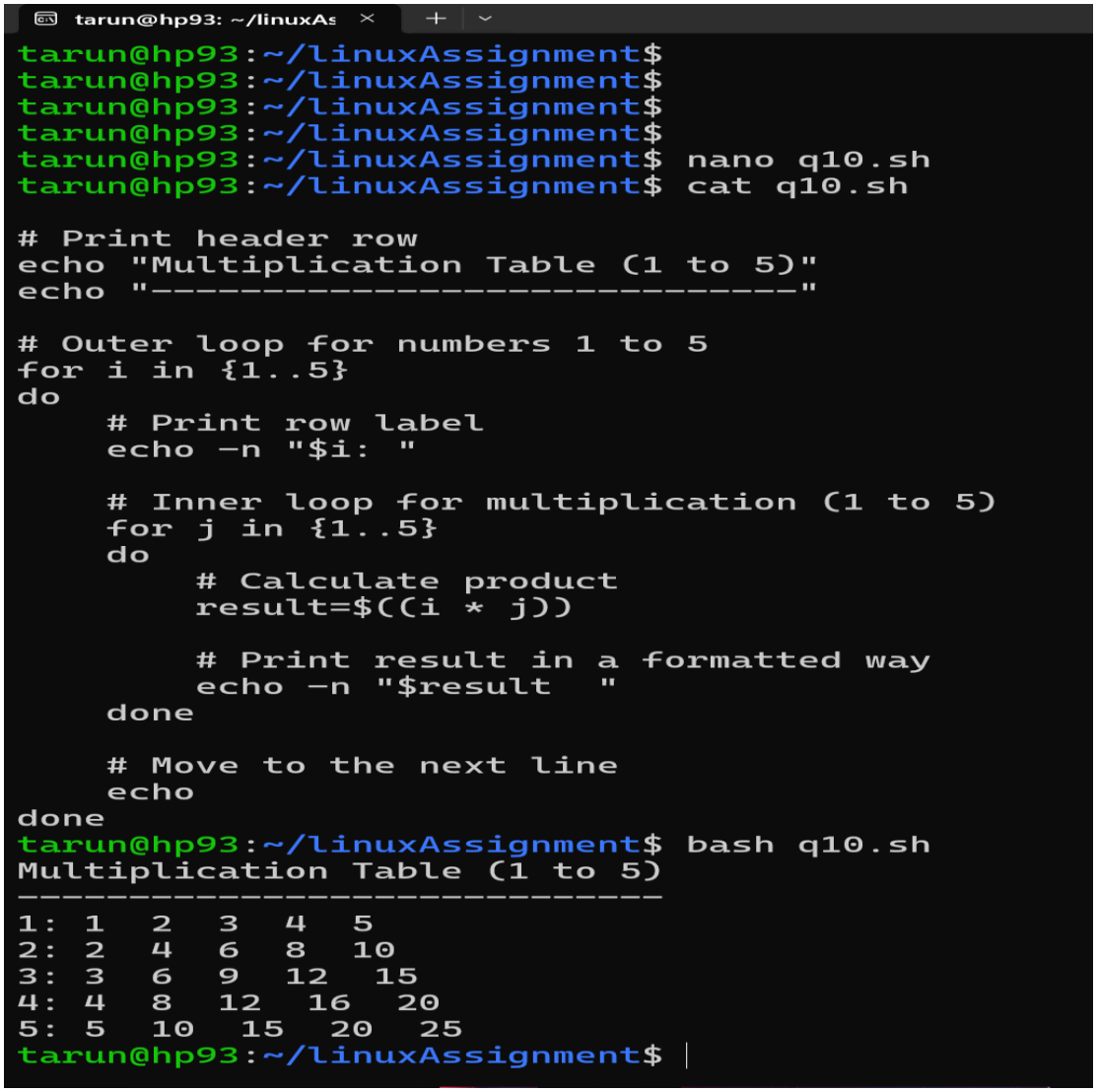
• Q8. Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".

```
arun@hp93:~/linuxAssignment$ nano q8.sh
arun@hp93:~/linuxAssignment$   GNU nano 7.2         q8.sh *
f [ -e file.txt ]
hen
    echo "file exist"
lse
     echo "file doesn't exist"
i




NU: command not found
ile doesn't exist
arun@hp93:~/linuxAssignment$
arun@hp93:~/linuxAssignment$ bash q8.sh
ile doesn't exist
arun@hp93:~/linuxAssignment$ nano q8 .sh
arun@hp93:~/linuxAssignment$
arun@hp93:~/linuxAssignment$ nano q8.sh
arun@hp93:~/linuxAssignment$ bash q8.sh
ile doesn't exist
arun@hp93:~/linuxAssignment$ touch file.txt
arun@hp93:~/linuxAssignment$ bash q8.sh
ile exist
arun@hp93:~/linuxAssignment$
arun@hp93:~/linuxAssignment$
```

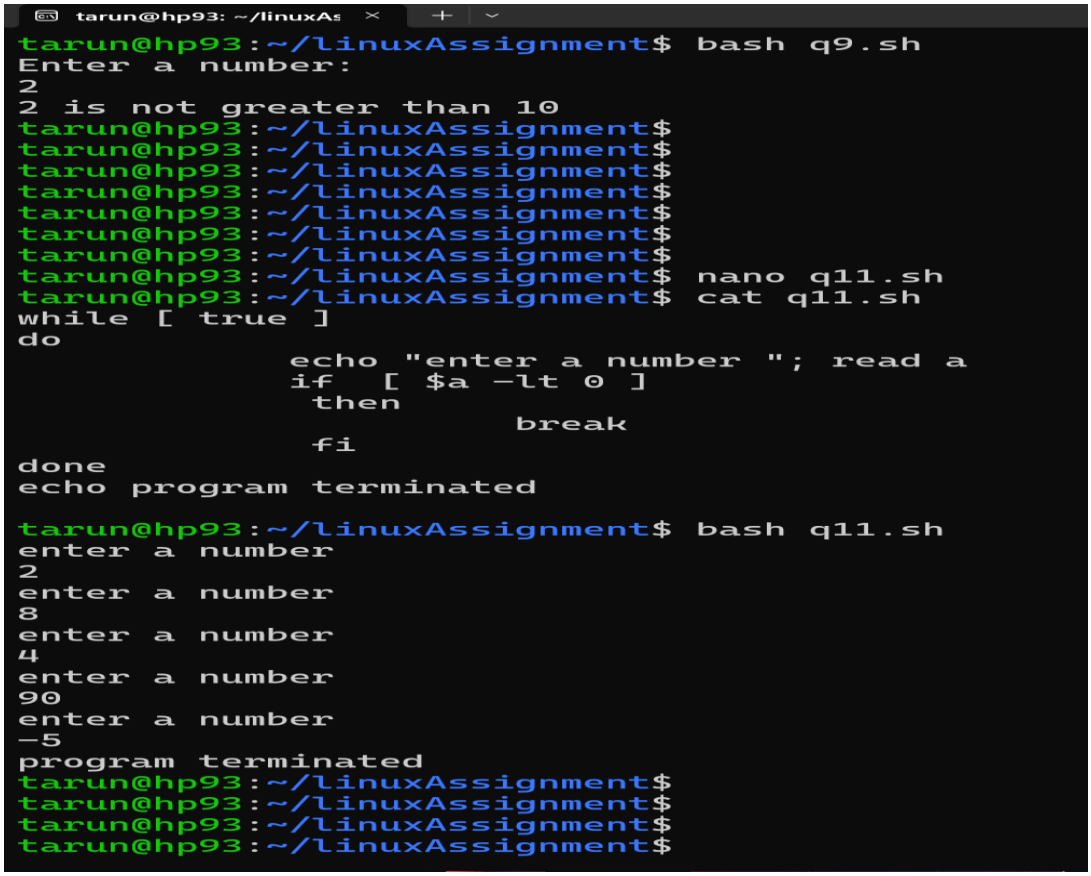
• Q9. Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.

```
tarun@hp93:~/linuxAssignment$ nano q9.sh
tarun@hp93:~/linuxAssignment$ cat q9.sh
#!/bin/bash

# Ask the user to enter a number
echo "Enter a number:"
read num  # Read user input

# Check if the number is greater than 10
if [ $num -gt 10 ]; then
    echo "$num is greater than 10"
else
    echo "$num is not greater than 10"
fi
tarun@hp93:~/linuxAssignment$ bash q9.sh
Enter a number:
500
500 is greater than 10
tarun@hp93:~/linuxAssignment$ bash q9.sh
Enter a number:
2
2 is not greater than 10
```

• Q10. 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.



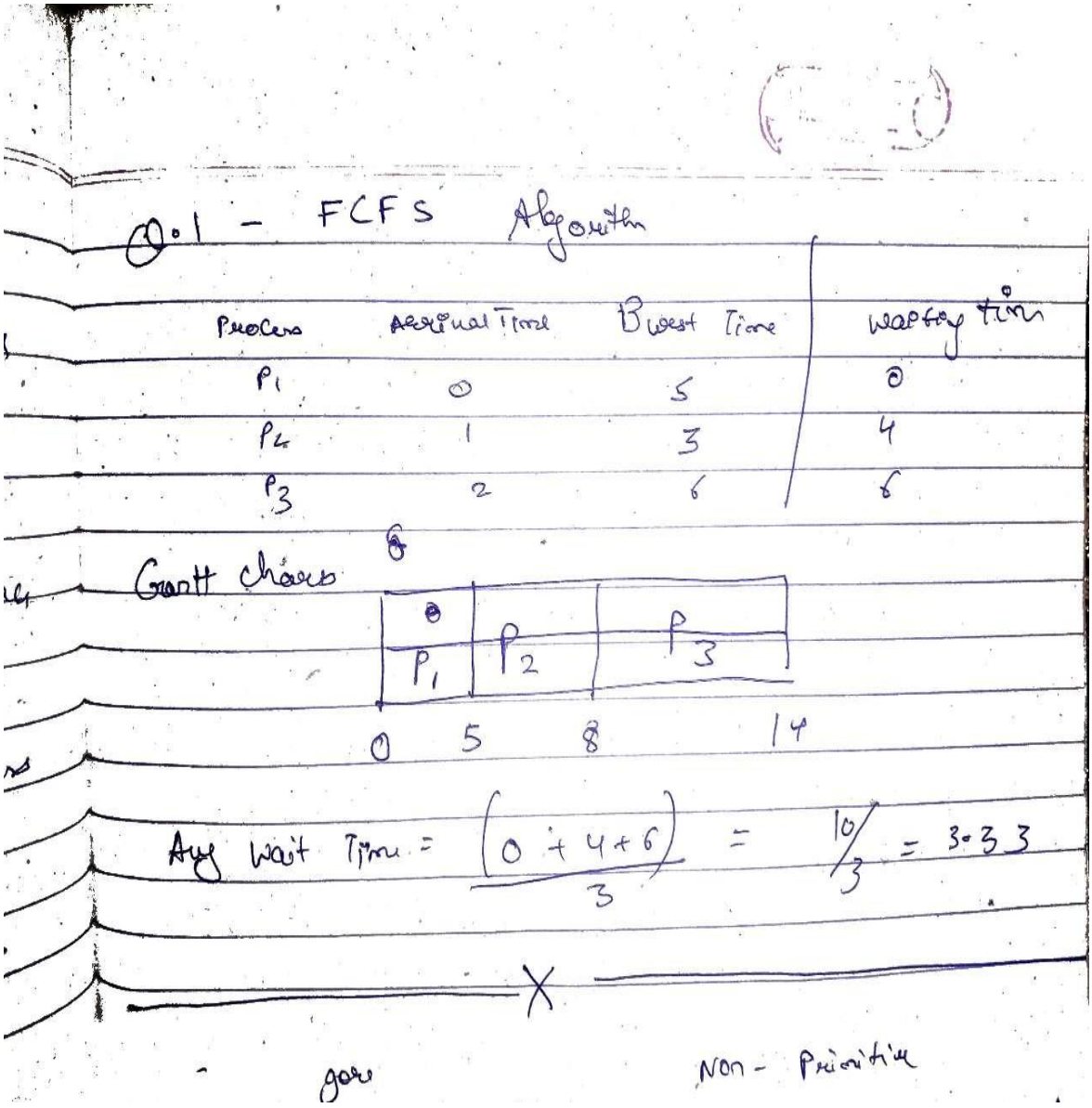
• Q11. Write a shell script that uses a while loop to read numbers from the user until the user enters a negative number. For each positive number entered, print its square. Use the break statement to exit the loop when a negative number is entered



----------XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX----------

Assignment – 2 Part – E

Q1



Q.1 - FCFS Algorithm

Process	Arrival Time	Burst Time	waiting time
P_1	0	5	0
P_2	1	3	4
P_3	2	6	6

Gantt chart

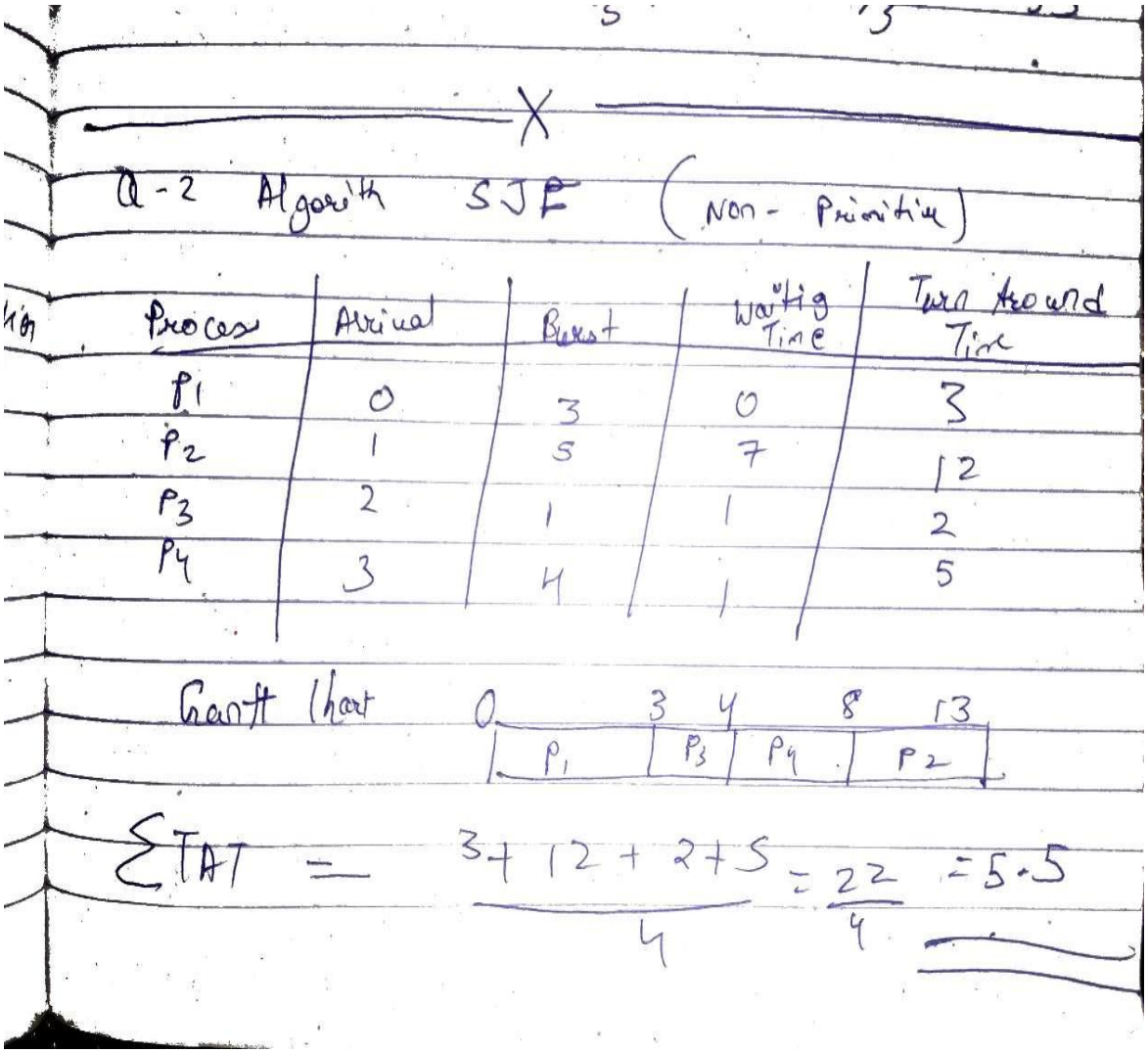
P_1	P_2	P_3
0	5	8	14

Avg wait Time = $\frac{(0+4+6)}{3} = \frac{10}{3} = 3.33$

gore

Non - Priority

Q2



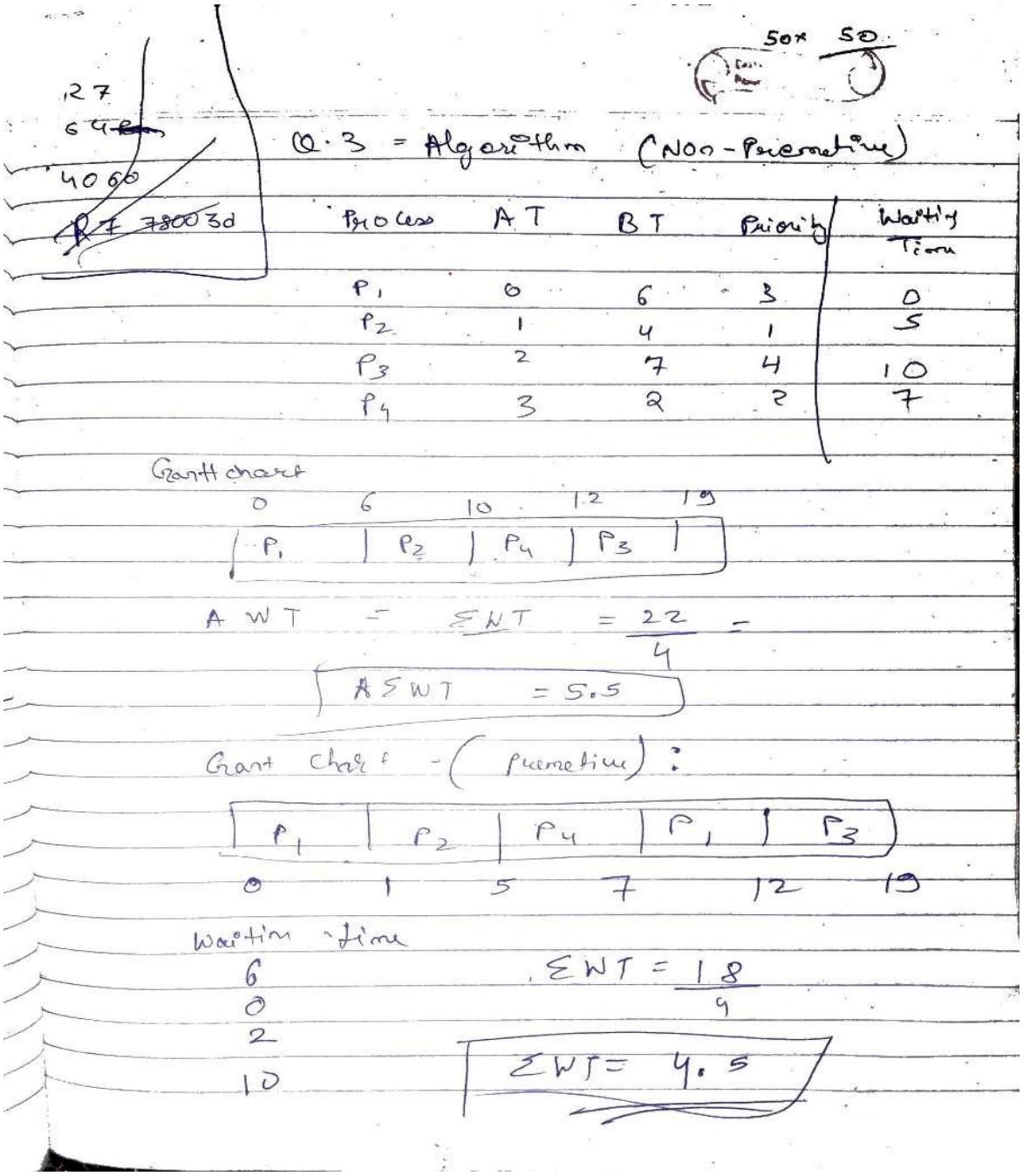
Q-2 Algorith SJF (Non-Primitive)

Process	Arrival	Burst	Waiting Time	Turn Around Time
P_1	0	3	0	3
P_2	1	5	7	12
P_3	2	1	1	2
P_4	3	4	1	5

Gantt Chart

$$\sum TAT = \frac{3+12+2+5}{4} = \frac{22}{4} = 5.5$$

Q3



Q.3 = Algorithm (Non-Preemptive)

Process	AT	BT	Priority	Waiting Time
P_1	0	6	3	0
P_2	1	4	1	5
P_3	2	7	4	10
P_4	3	2	2	7

Gantt chart

0	6	10	12	19
P_1	P_2	P_4	P_3	

AWT = $\frac{\Sigma WT}{} = \frac{22}{4} =$

AWT = 5.5

Gant Chart - (Preemptive):

P_1	P_2	P_4	P_1	P_3
0	1	5	7	12	19

Waiting time

6
0
2
10

$\Sigma WT = \frac{18}{4}$

$\Sigma WT = 4.5$

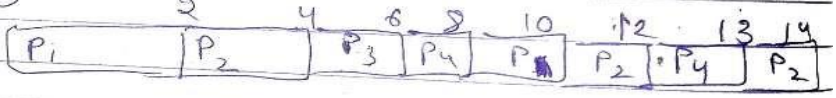
Q4

Q.4) Algo used Round Robin

Quantum – 2 unit

Process	A.T	B.T	W.T.	TAT
P_1	0	4	6	10
P_2	1	5	8	13
P_3	2	2	2	4
P_4	3	3	7	10

Gantt chart



ATAT = $\frac{(10+13+4+10)}{4}$

$= \frac{37}{4}$

$= 9.25$

- **Q5.**

o When the fork() system call is used, it creates a child process that has its own copy of the parent's memory. o Before forking, the parent has a variable x = 5. After the fork, both the parent and child have separate copies of x, still equal to 5. o Each process then increments x by 1, so both the parent and child have x = 6, but in their own separate memory. o In parent process, x=6. In child process, x=6