Name- Aarya Sanjay Dange

Concepts of Operating System

<u>Assignment 2</u>

Part A

What will the following commands do?

- echo "Hello, World!"
  - ➔ This command will print the String present in double quotes “Hello, World!” . Providing double is not mandatory here unlike other programming languages.

- name="Productive"
  - ➔ This command will assign String value i.e. “Productive” to shell variable name

- touch file.txt
  - ➔ This command will create one file with the name file.txt which will be empty that has no content.

- ls -a
  - ➔ This command will list out i.e. print the list of all files and directories in present/current directory. -a option is for including all hidden files and directories also .

- rm file.txt
  - ➔ rm command will remove file or directory. In this case, this command will remove i.e. delete “file.txt” file.

- cp file1.txt file2.txt
  - ➔ cp command is used to copy files and contents of one file to another file. In above example, cp command is copying contents of file1.txt and pasting it into file2.txt after creating the file.

- mv file.txt /path/to/directory/
  - ➔ mv command used to rename or move the file into specific present directory. In above example, mv command is moving the file named “file.txt” into that specified directory.

- chmod 755 script.sh
  - ➔ chmod command is used to assign permissions such as read, write and execute to user, groups or others. chmod is abbreviation for change modifications.

Above command is giving read,write and execute all permissions to user . read and execute permission for groups and others.

- grep "pattern" file.txt
  - ➔ grep command is for searching patterns or strings in mentioned file. In above example, grep is searching for string “pattern” in file named file.txt. and this will return matching lines from file.txt .

- kill PID
  - ➔ As the name suggests kill command kills/terminate the process. But to make this happen instead of PID we have to provide actual process id of that process. Otherwise this above command will results into error.

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

  - ➢ mkdir : for making/creating new directory
  - ➢ && : to concatenate two commands
  - ➢ cd : change directory i.e. switch/go into specified directory
  - ➢ touch : creates new empty file
  - ➢ echo : prints given message/value

    - > : use output of one command as input for another command (redirection)
    - cat :prints content of the file

    - In above case,firstly directory with name “mydir” is created in current directory then user will go inside that directory and empty file will create with name “file.txt” .
    - After this , echo command will display the message “Hello World” on the terminal. This output of echo command is inserted into file.txt using (>) redirection operator. And finally, contents of file will be displayed using cat command.

- ls -l | grep ".txt"
    - ls -l command is to list out all files and directories along with its information inside current directory and grep is to search for a specific pattern. In above case, Among all the files present in current directory the files with .txt extension will be displayed along with their details.

- cat file1.txt file2.txt | sort | uniq
    - cat command is to display content of file.sort is to sort the data whether alphabetically or in asc/desc order.
    - uniq command displays only distinct content.

- ➔ In above example , fistly it will sort the content from both the files and all unique content will be displayed using cat command.

- ls -l | grep "^d"
  - ➔ ls command give the list of all files and directories in current directory and -l option will give details of that files and directories. grep command is to search for specific pattern or word in given file/directory. In above case, grep command is searching for all directories that listed out using ls command along with their details.

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

- cat file1.txt file2.txt | sort | uniq –d
  - ➔ In above example, sort command will use the output of cat command and display content of both the files i.e. file1.txt and file2.txt in sorted manner and

because of uniq -d command output will have only duplicate lines.

- chmod 644 file.txt
  - ➔ chmod command is for providing permissions to users. In this case, chmod command will give permission to read and write for owner of file.txt and only reading permission to group and others.

- cp -r source_directory destination_directory
  - ➔ Above command is used to copy the source_directory to destination directory. -r option will copy all files in source_directory .

- find /path/to/search -name "*.txt"
  - ➔ find command is used for searching the files and directories. Given command searches /path/to/search directory and its subdirectories for any file ending with .txt pattern.

- chmod u+x file.txt
  - ➔ This chmod command will give permission to execute the file to the user i.e. owner of file.txt.

- echo $PATH
    - ➔ This command displays the value of system environment variable that stores directories where executable programs are located.

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 used to change directory.

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 used to copy files and directories.

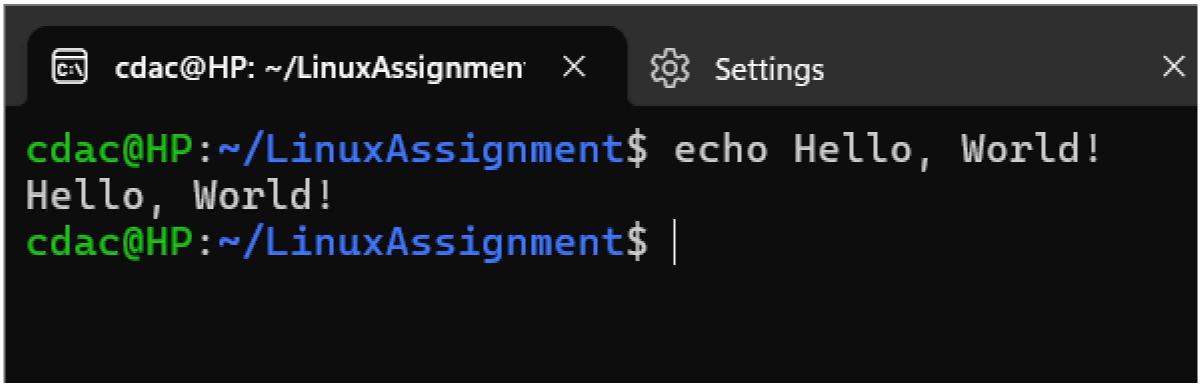
3. mkfile is used to create a new file.
mkdir is used to create a new file. touch is used to create new file.

4. catx is used to concatenate files.
cat command is used to concatenate files.

5. rn is used to rename files.
rm is used to remove files.

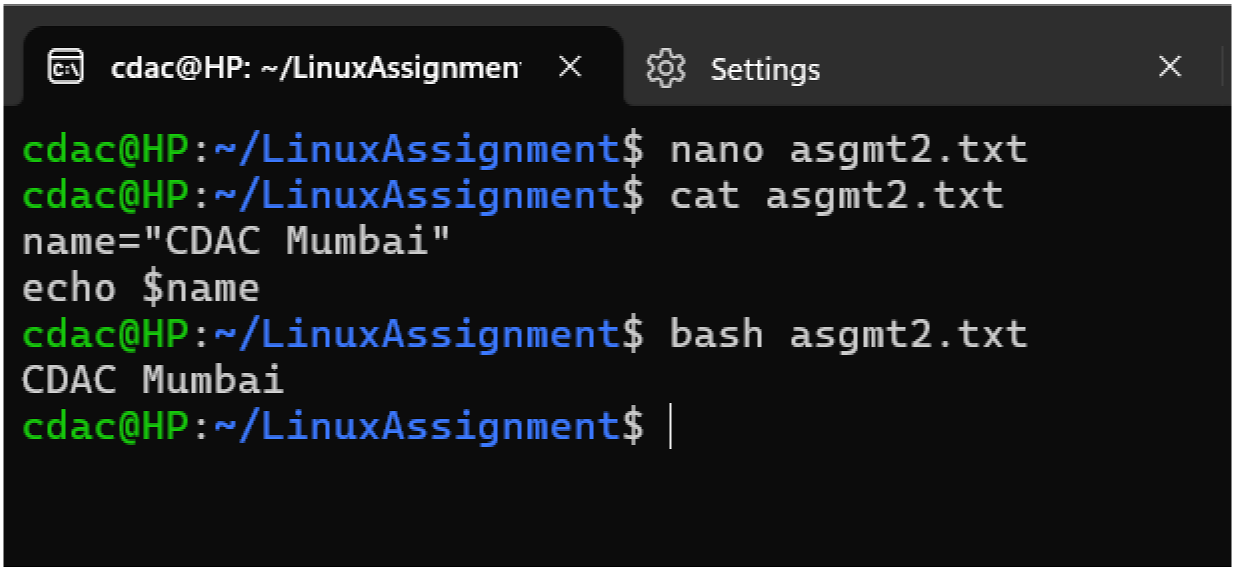
Part C

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



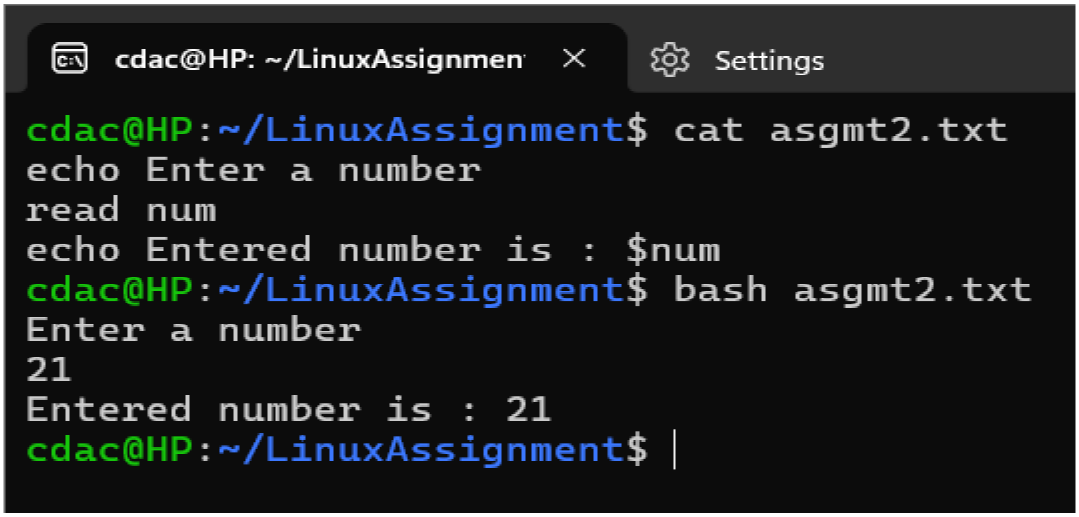
```
cdac@HP:~/LinuxAssignment$ echo Hello, World!
Hello, World!
cdac@HP:~/LinuxAssignment$
```

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



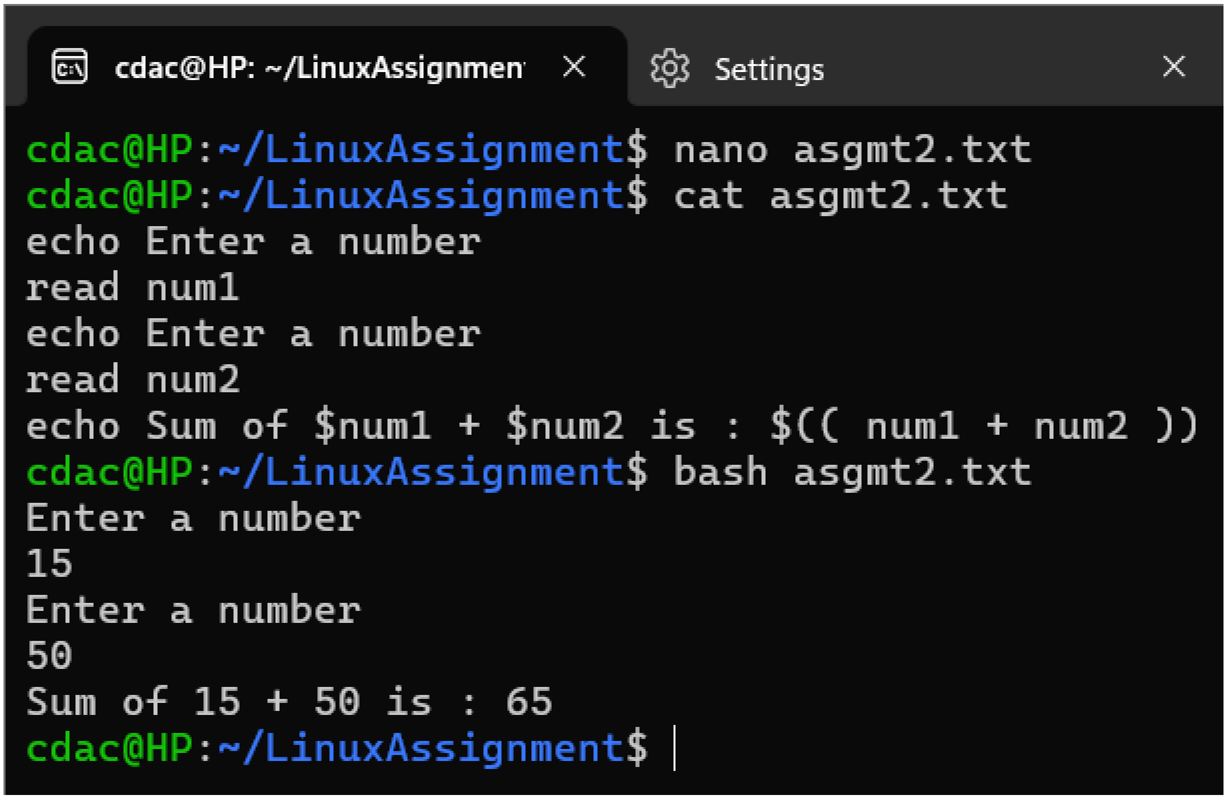
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
name="CDAC Mumbai"
echo $name
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
CDAC Mumbai
cdac@HP:~/LinuxAssignment$
```

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



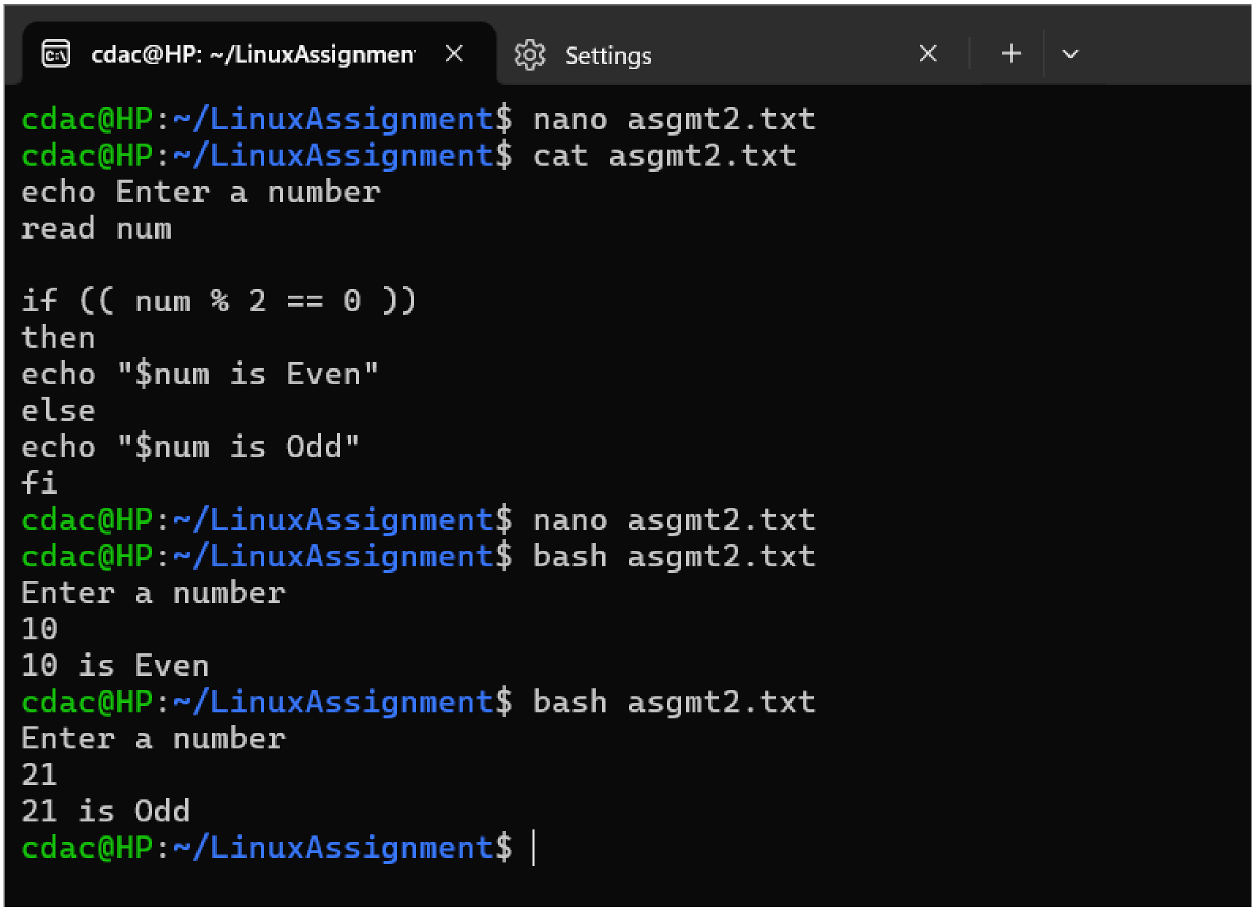
```
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
echo Enter a number
read num
echo Entered number is : $num
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number
21
Entered number is : 21
cdac@HP:~/LinuxAssignment$
```

Question 4: Write a shell script that performs addition of two numbers and prints the result.



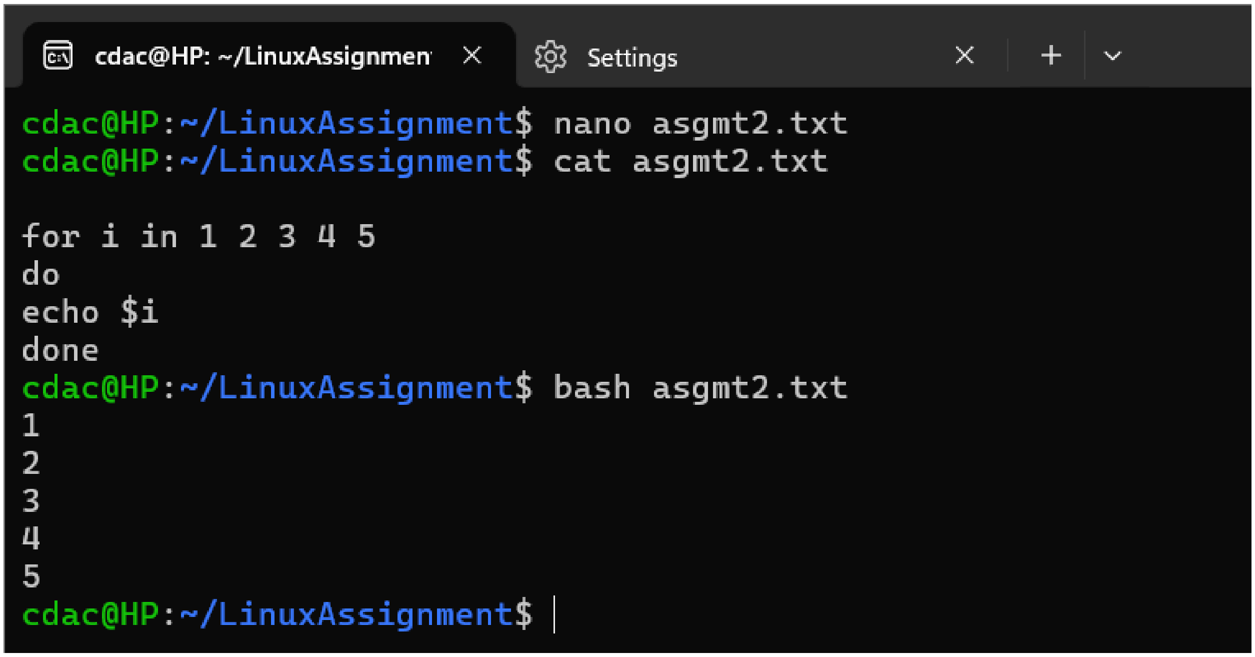
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
echo Enter a number
read num1
echo Enter a number
read num2
echo Sum of $num1 + $num2 is : $(( num1 + num2 ))
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number
15
Enter a number
50
Sum of 15 + 50 is : 65
cdac@HP:~/LinuxAssignment$
```

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



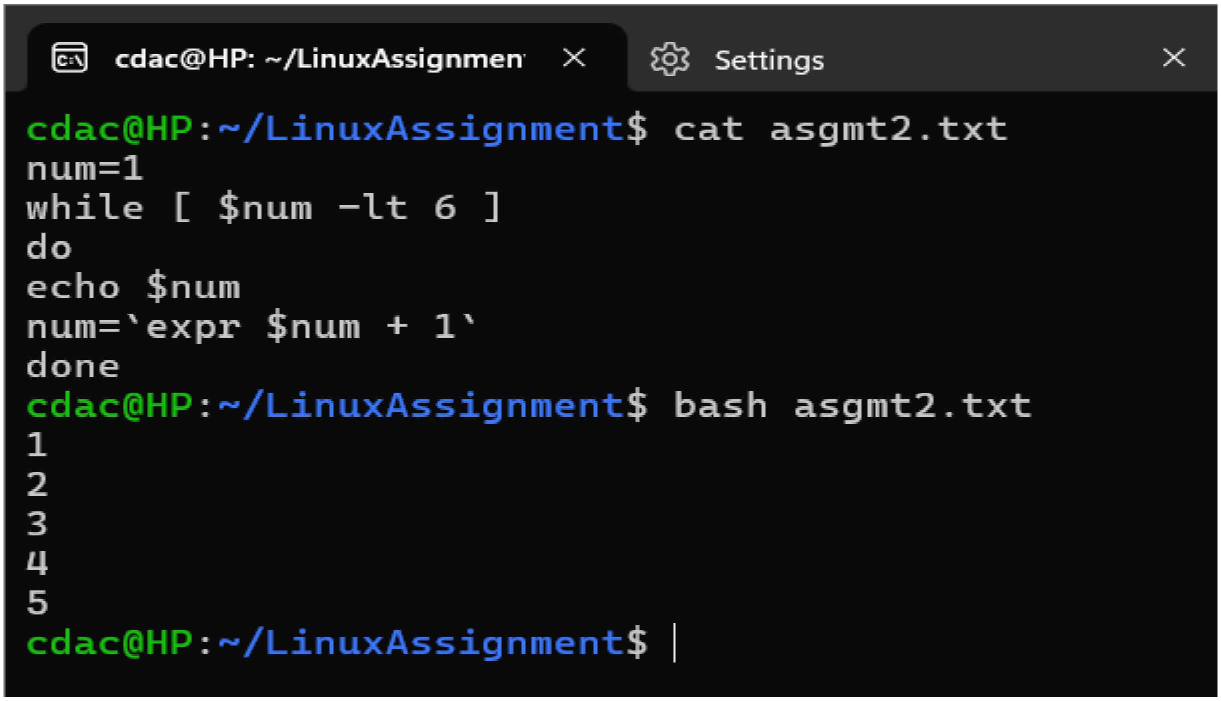
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
echo Enter a number
read num

if (( num % 2 == 0 ))
then
echo "$num is Even"
else
echo "$num is Odd"
fi
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number
10
10 is Even
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number
21
21 is Odd
cdac@HP:~/LinuxAssignment$
```

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



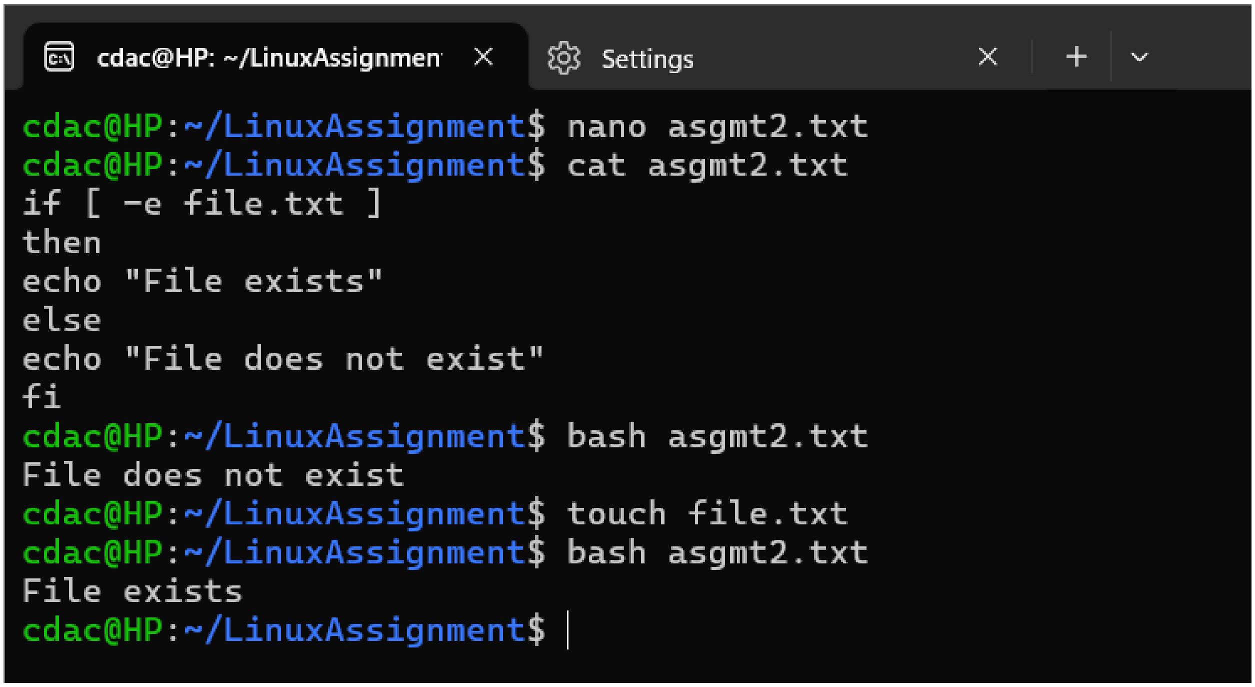
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt

for i in 1 2 3 4 5
do
echo $i
done
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
1
2
3
4
5
cdac@HP:~/LinuxAssignment$
```

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



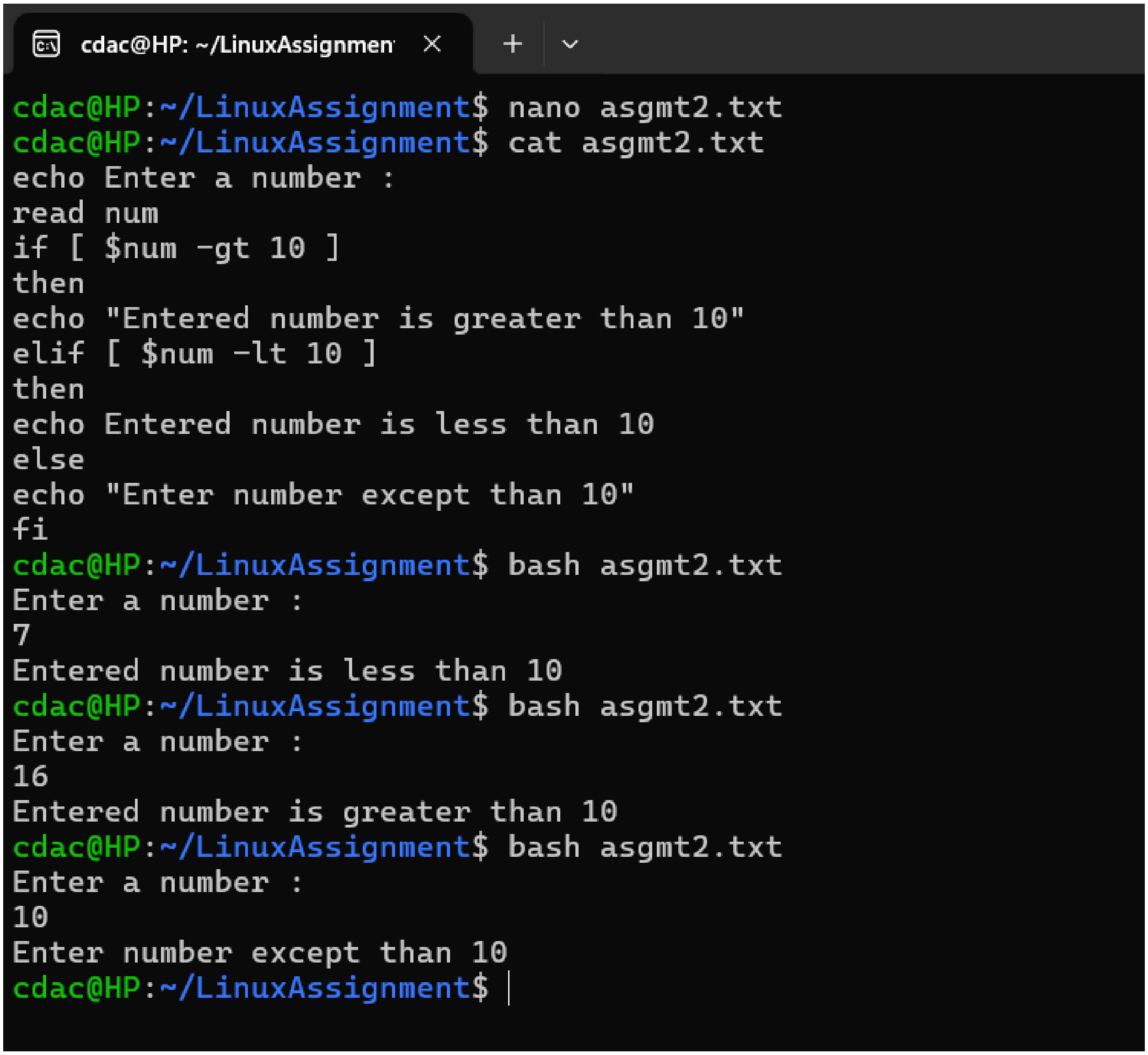
```
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
num=1
while [ $num -lt 6 ]
do
echo $num
num=`expr $num + 1`
done
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
1
2
3
4
5
cdac@HP:~/LinuxAssignment$
```

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".



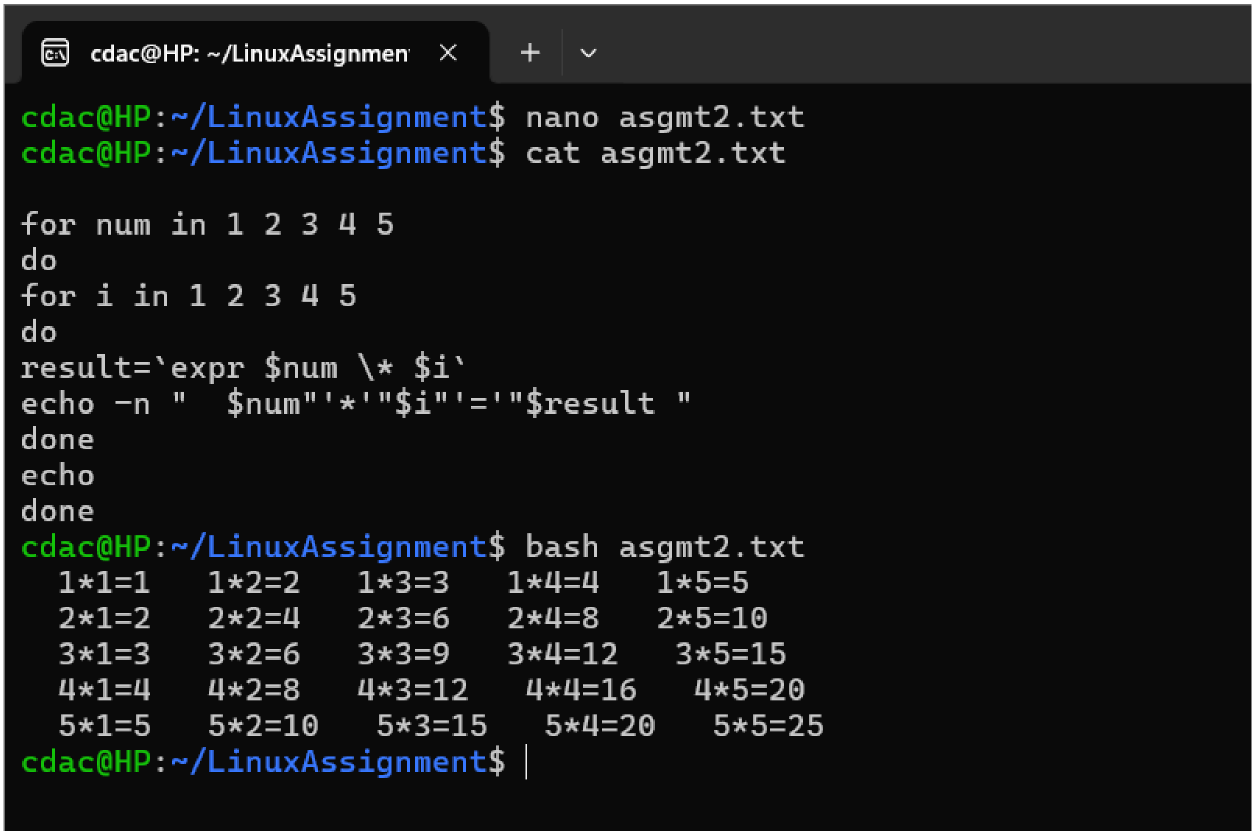
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
if [ -e file.txt ]
then
echo "File exists"
else
echo "File does not exist"
fi
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
File does not exist
cdac@HP:~/LinuxAssignment$ touch file.txt
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
File exists
cdac@HP:~/LinuxAssignment$
```

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.



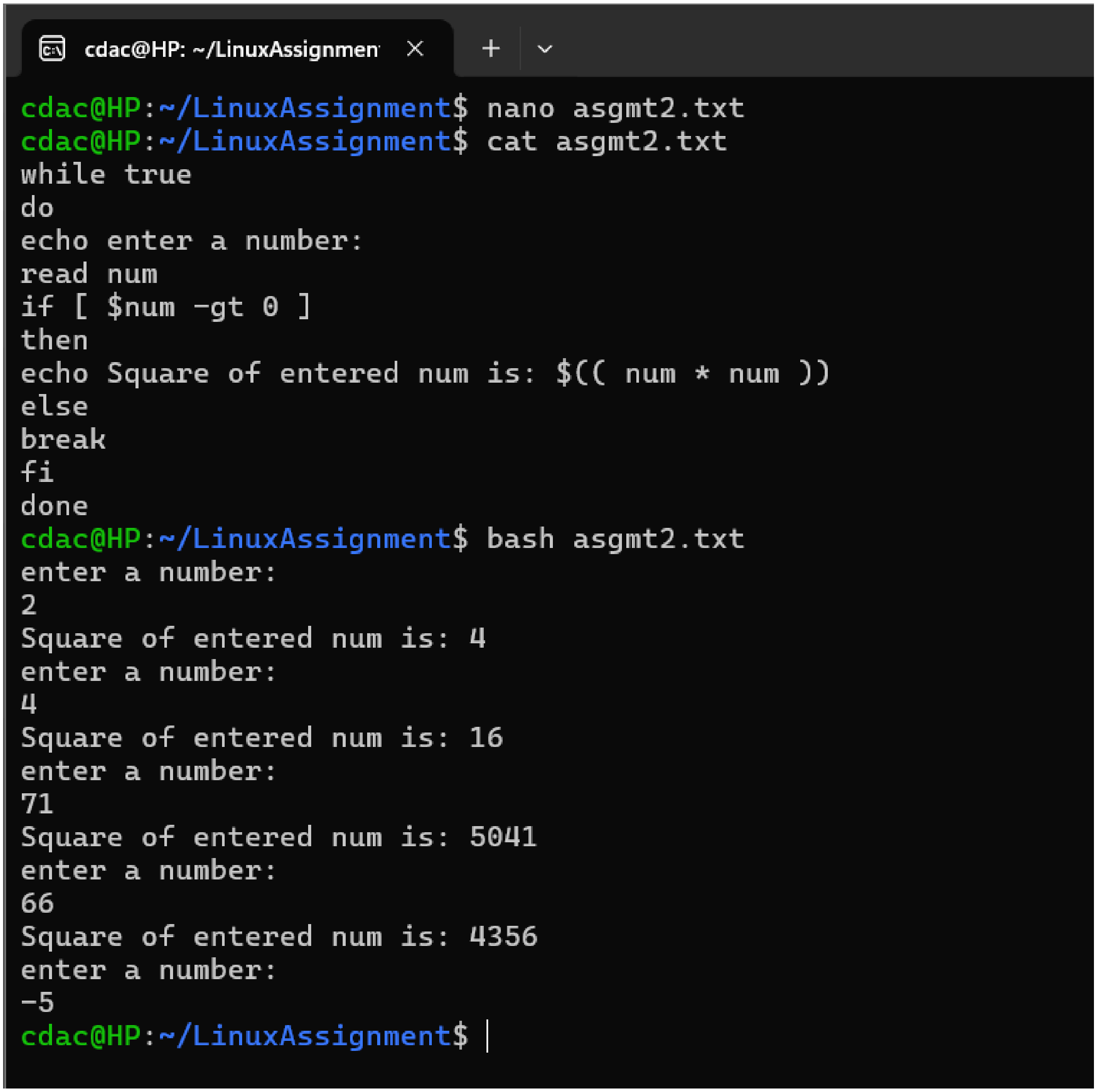
```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
echo Enter a number :
read num
if [ $num -gt 10 ]
then
echo "Entered number is greater than 10"
elif [ $num -lt 10 ]
then
echo Entered number is less than 10
else
echo "Enter number except than 10"
fi
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number :
7
Entered number is less than 10
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number :
16
Entered number is greater than 10
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
Enter a number :
10
Enter number except than 10
cdac@HP:~/LinuxAssignment$
```

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@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt

for num in 1 2 3 4 5
do
for i in 1 2 3 4 5
do
result=`expr $num \* $i`
echo -n "  $num"'*'"$i"'='"$result "
done
echo
done
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
  1*1=1   1*2=2   1*3=3   1*4=4   1*5=5
  2*1=2   2*2=4   2*3=6   2*4=8   2*5=10
  3*1=3   3*2=6   3*3=9   3*4=12   3*5=15
  4*1=4   4*2=8   4*3=12   4*4=16   4*5=20
  5*1=5   5*2=10   5*3=15   5*4=20   5*5=25
cdac@HP:~/LinuxAssignment$
```

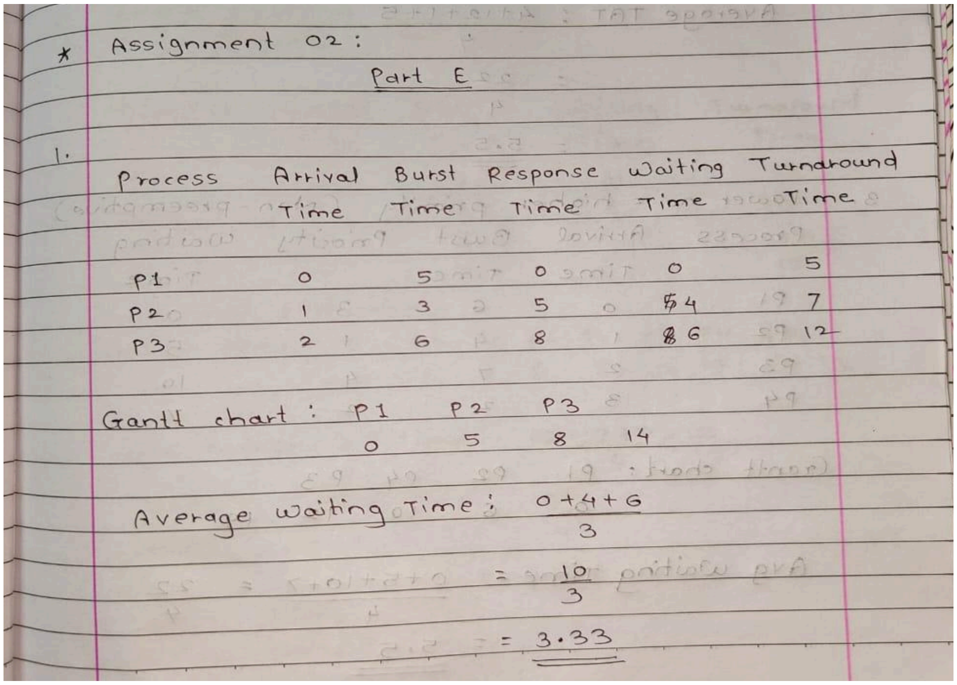
Question 11: Write a shell script that uses a while loop to read numbers from the user until the user enters a negative number. For each positive number entered, print its square. Use the break statement to exit the loop when a negative number is entered.



```
cdac@HP:~/LinuxAssignment$ nano asgmt2.txt
cdac@HP:~/LinuxAssignment$ cat asgmt2.txt
while true
do
echo enter a number:
read num
if [ $num -gt 0 ]
then
echo Square of entered num is: $(( num * num ))
else
break
fi
done
cdac@HP:~/LinuxAssignment$ bash asgmt2.txt
enter a number:
2
Square of entered num is: 4
enter a number:
4
Square of entered num is: 16
enter a number:
71
Square of entered num is: 5041
enter a number:
66
Square of entered num is: 4356
enter a number:
-5
cdac@HP:~/LinuxAssignment$
```

Part E

1. FCFS:



* Assignment 02 :

Part E

1.

Process	Arrival Time	Burst Time	Response Time	Waiting Time	Turnaround Time
P1	0	5	0	0	5
P2	1	3	5	~~5~~4	7
P3	2	6	8	~~8~~6	12

Gantt chart : P1 P2 P3

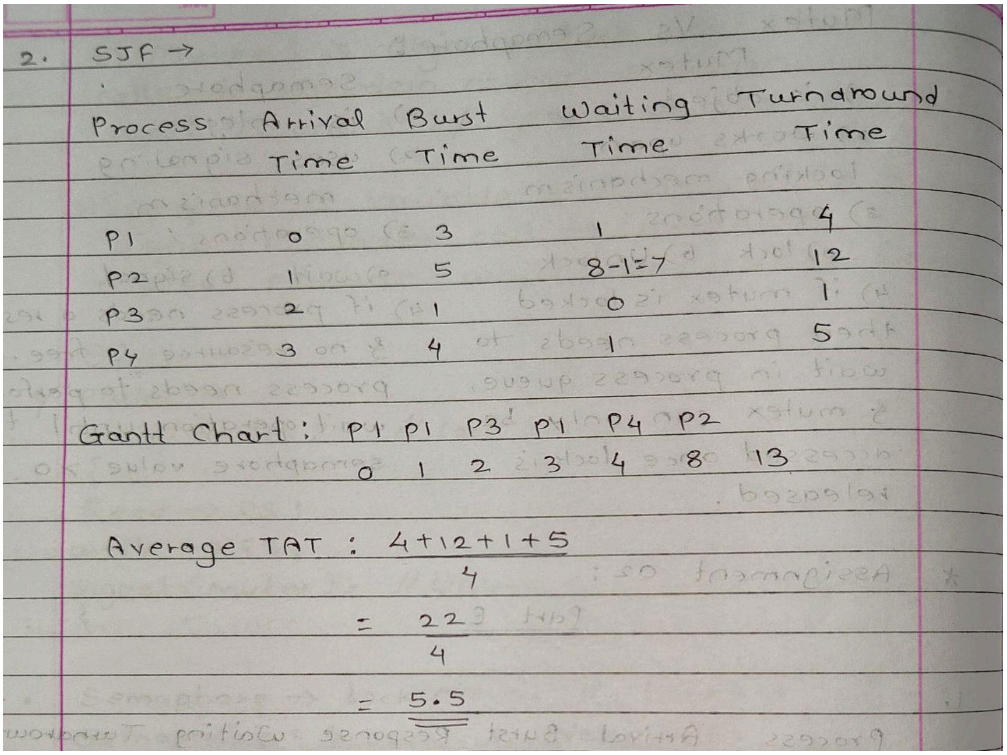
0 5 8 14

Average waiting Time : $\frac{0+4+6}{3}$

$= \frac{10}{3}$

$= 3.33$

2. SJF :



2. SJF →

Process	Arrival Time	Burst Time	Waiting Time	Turnaround Time
P1	0	3	1	4
P2	1	5	8-1=7	12
P3	2	1	0	1
P4	3	4	1	5

Gantt Chart : P1 P1 P3 P1 P4 P2

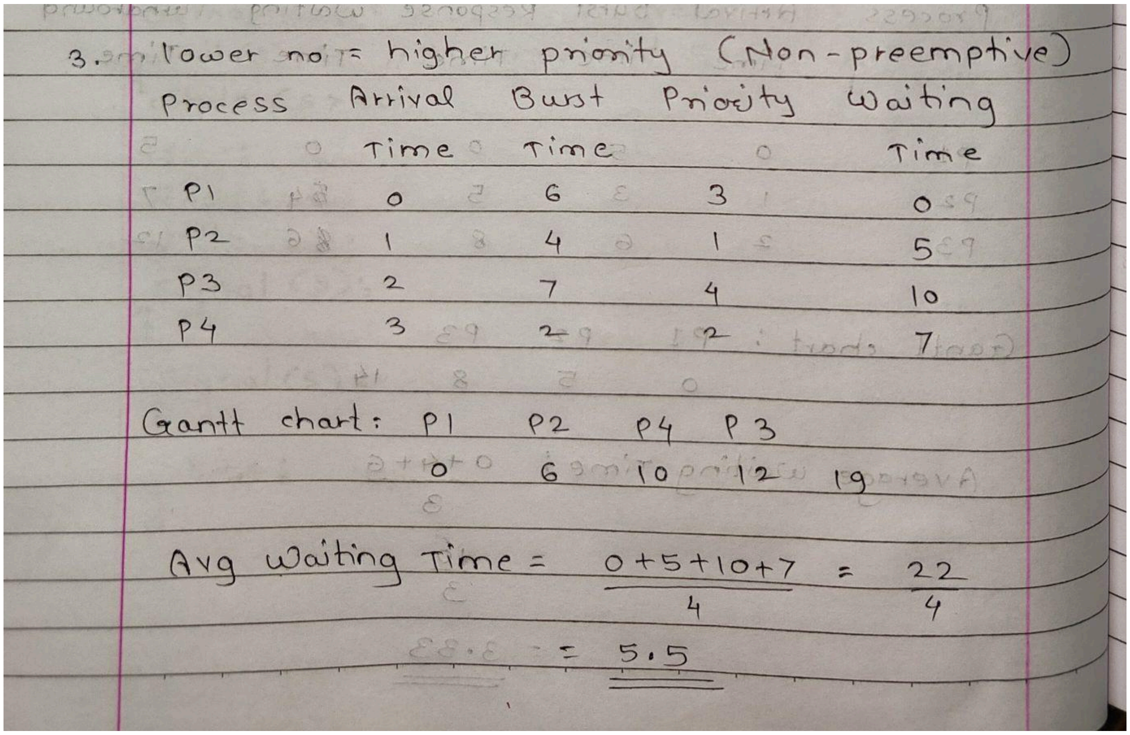
0 1 2 3 4 8 13

Average TAT : $\frac{4+12+1+5}{4}$

$= \frac{22}{4}$

$= 5.5$

3. Priority:



3. lower no. = higher priority (Non-preemptive)

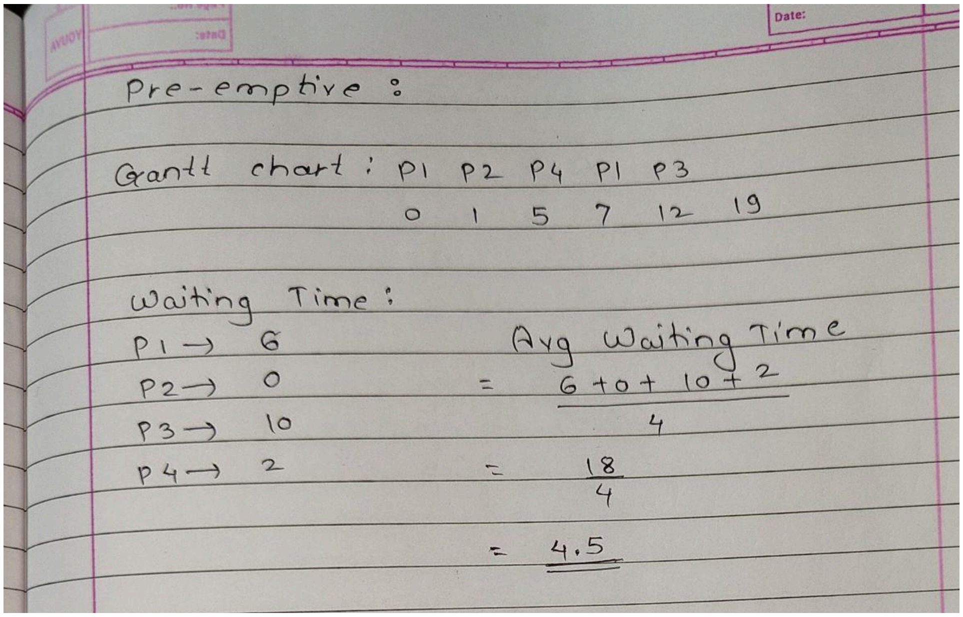
Process	Arrival Time	Burst Time	Priority	Waiting Time
P1	0	6	3	0
P2	1	4	1	5
P3	2	7	4	10
P4	3	2	2	7

Gantt chart: P1 P2 P4 P3

0 6 10 12 19

Avg waiting Time = $\frac{0+5+10+7}{4} = \frac{22}{4}$

$= 5.5$



Date:

Pre-emptive :

Gantt chart : P1 P2 P4 P1 P3

0 1 5 7 12 19

Waiting Time :

P1 → 6

P2 → 0

P3 → 10

P4 → 2

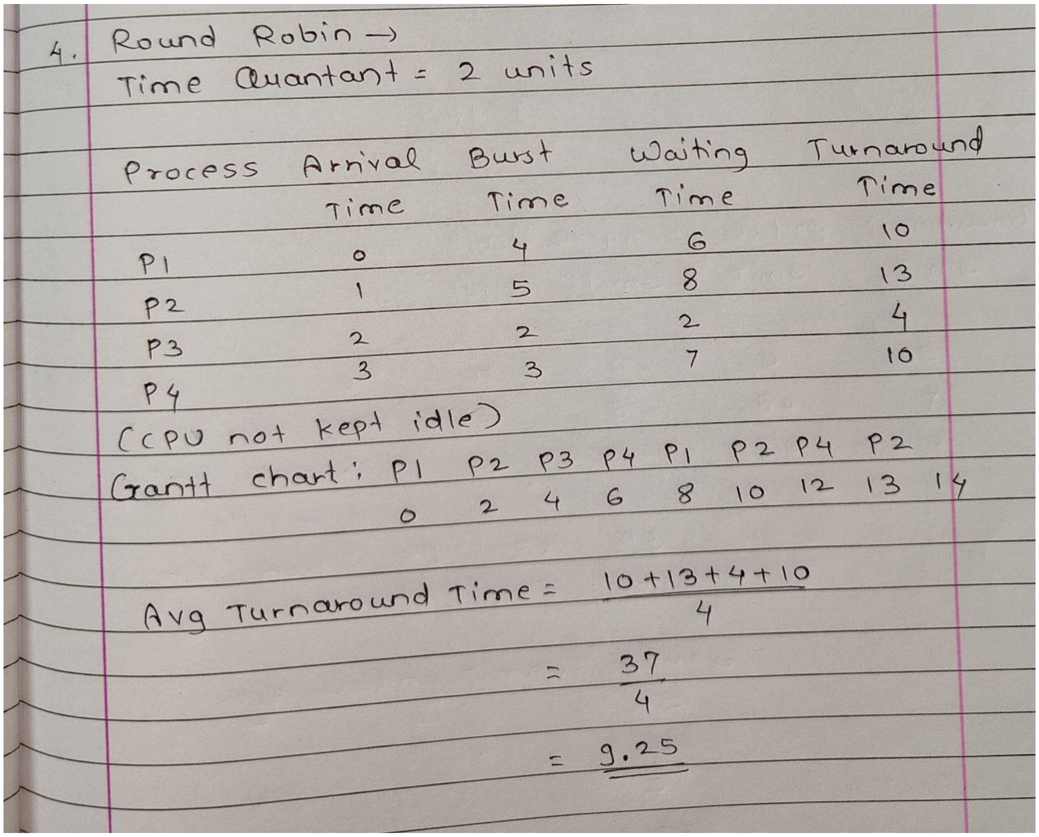
Avg Waiting Time

$= \frac{6+0+10+2}{4}$

$= \frac{18}{4}$

$= 4.5$

4. Round Robin:



4. Round Robin →

Time Quantant = 2 units

Process	Arrival Time	Burst Time	Waiting Time	Turnaround Time
P1	0	4	6	10
P2	1	5	8	13
P3	2	2	2	4
P4	3	3	7	10

(CPU not kept idle)

Gantt chart: P1 P2 P3 P4 P1 P2 P4 P2

0 2 4 6 8 10 12 13 14

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

$$= \frac{37}{4}$$

$$= \underline{\underline{9.25}}$$

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?

Ans.

I. When a program uses fork() system call then one child process gets created which will be exact replica of that process(i.e. Parent process) .
II. As parent process has variable x with the value 5 so child process will also have x variable having value 5.

After forking when both the processes increment value of x by 1 then , final value of x in child process and that of parent process will be 6 .