Assignment No.2

Part A

1. **Echo “Hello, World!”:**

   By Running the following command Hello, World! Will get printed on the screen.

2. **Name = “Productive”:**

   This Command will assign a string literal Productive to the shell variable Name.

3. **Touch file.txt:**

   This command will Create a new file with name as file.txt.

4. **Ls – a:**

   By Running the following command list of contents of current directory will be displayed including hidden files and directories.

5. **Rm file.txt:**

   This command is used to delete a file.txt.

6. **Cp file.txt file2.txt:**

   This Command is used to copy files and directories. This command will copy the contents of file1.txt creates a file named file2.txt and pastes the content in it.

7. **Mv file.txt/path/to/directory:**

   This command is used to rename or move a file. Mv command moves the file (file.txt) into the specified directory.

1. **Chmod 755 script.sh:**

This command is used to assign read, write, and execute permissions to owner, group and other users respectively.

2. **Grep “pattern” file.txt:**

Grep command is used to search for specific patterns or regular expression in text files and display the matching lines. Above command will search for string “pattern” From the file named file.txt.

3. **Kill PID:**

This Command will terminate the process whose PID is mentioned in the command. Since the above command doesn’t contain any process id.

4. **Mkdir mydir && mydir && touch file.txt && echo “Hello World!” file.txt && cat.file.txt.**

The above command produces a series of results where output of previous command acts as input for a next command. At first mkdir command creates a mydir directory in the current directory. Cd command is then used to change current directory to new created mydir directory. Touch file.txt creates an empty file named file.txt. echo command will display the message” Hello World!” This output of echo command is inserted into file.txt.

5. **Ls-l | grep ”.txt”:**

This command uses piping to combine the output of both ls and grep command. Ls -l is used to display the contents of current directory with details and grep “.txt” command is used to display all the files containing .txt pattern in their name.

6. **Cat file1.txt file2.txt | sort | uniq:**

The above command uses piping to combine output of cat sort and unique commands. Cat command is used to display the contents of file1.txt followed by contents of file2.txt. sort command is used to perform alphanumeric sort on the result of cat command. Contents of file.txt and file2.txt are sorted separately in the result.

7. **Ls -l | grep “^d”:**

Ls command lists the files and directories in long format. Grep ”^d” command filters the output to show only lines that start with “d” which in the ls -l output indicates directories.

8. **Grep -r “pattern”/path/to/directory:**

This command is used to recursively search for given pattern “pattern” in the directory /path/to/directory provided that such directory exists I first place.

9. **Cat file1.txt file2.txt | sort | uniq-d:**

Cat command displays the content of file1.txt followed by file2.txt. sort is used to perform alphanumeric sort on the result of cat command. Contents of file1.txt and duplicate lines the previous output.

10. **Chmod 644 file.txt:**

This command assigns read and write permissions to owner of the file.txt and read permissions to group users and other users respectively.

11. **Cp -r source_directory_destination_directory:**

This command is used to copy the source_directory to destination directory. This is done by using -r option so that all files in source_directory are copied recursively.

12. **Find/path/to/search-name”*.txt”:**

This 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.

13. **Chmod u+x file.txt:**

This command is used to grant execute permissions for file.txt file to user to the file.

14. **echo $PATH:**

this command displays value of system environment variable that stores directories where executable programs are located.

Part B

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, it is used to change the directory – False.
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 / directory 2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist – True.
8. Rm -rf file.txt deletes a file forcefully without confirmation -r (recursion option) is used for deleting directories, not files – False.

Identify incorrect commands:

- Chmodx is used to change file permisions.

Ans : Chmod command is used to change file permissions.

- Cpy is used to copy files and directories.

Ans: cp command is used to copy files and directories.

- Mkfile is used to create a new file.

Ans: touch command is used create a new file.

- Catx is used to concatenate files.

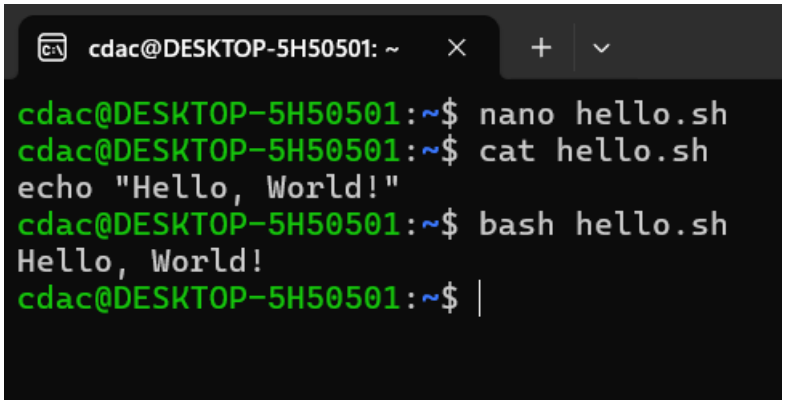
Ans: cat command is used to concatenate files.

- Rn is used to concatenate files.

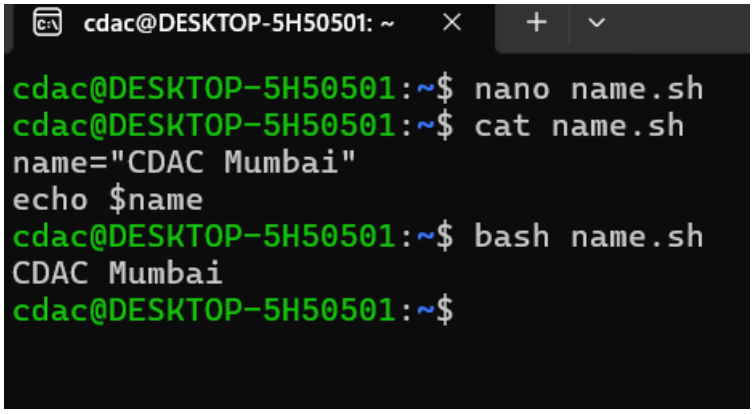
Ans: mv command is used to rename files when 2 files names are passed as arguments.

Part c

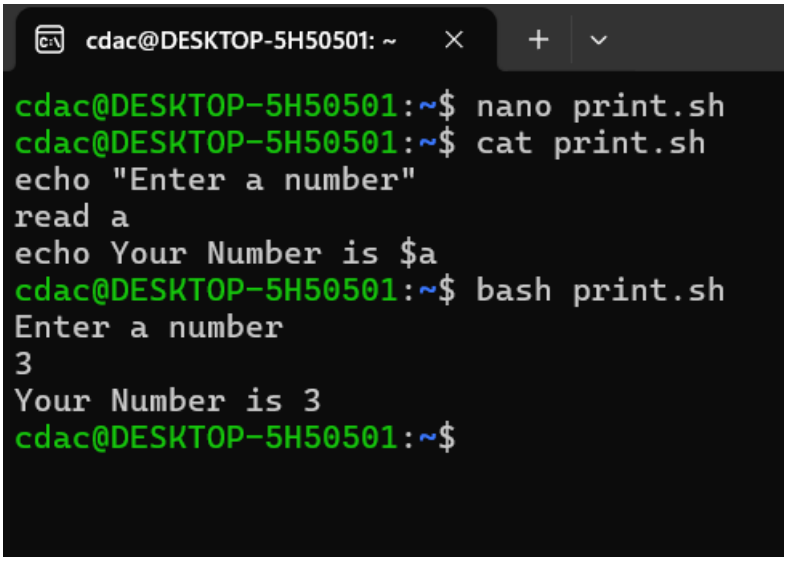
Q1.



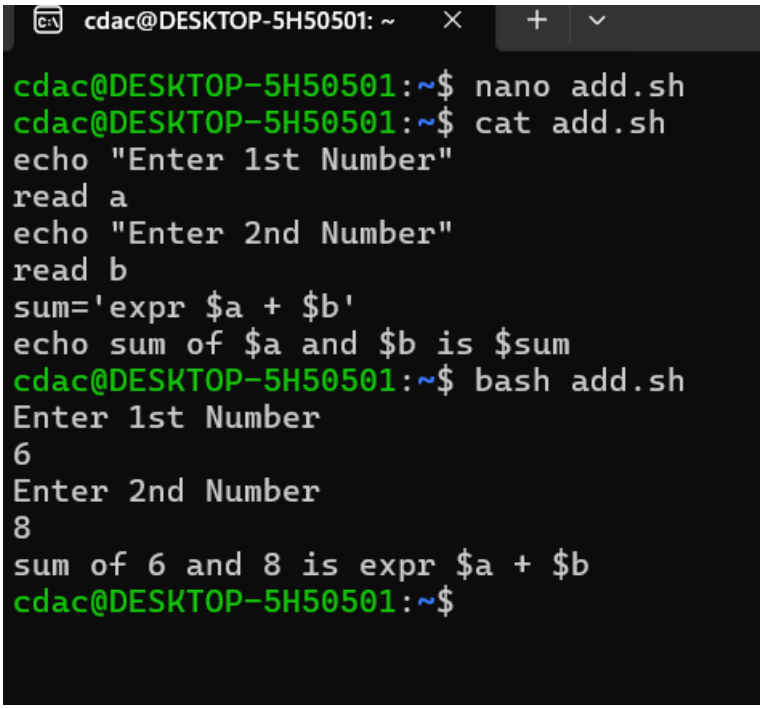
Q2.



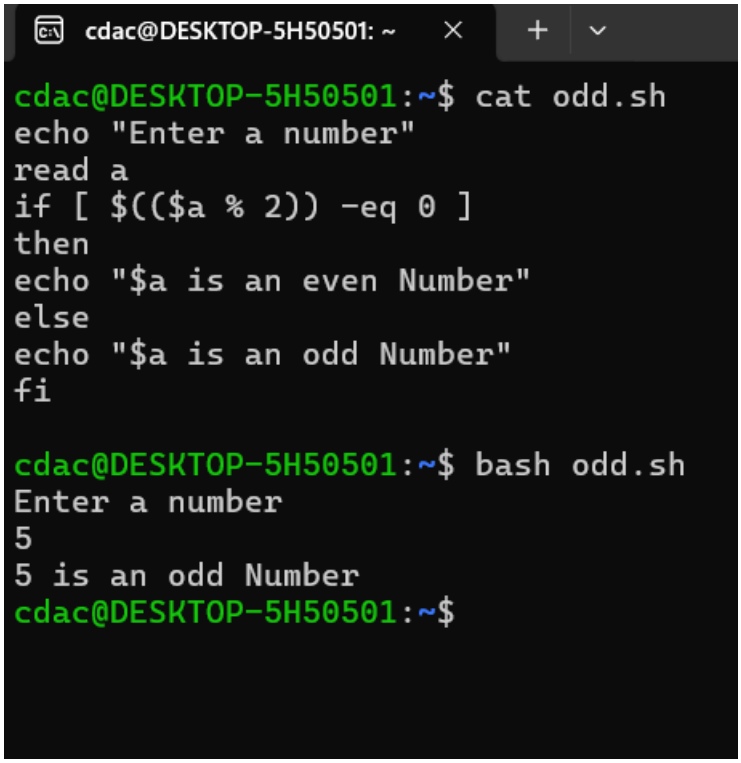
Q3.



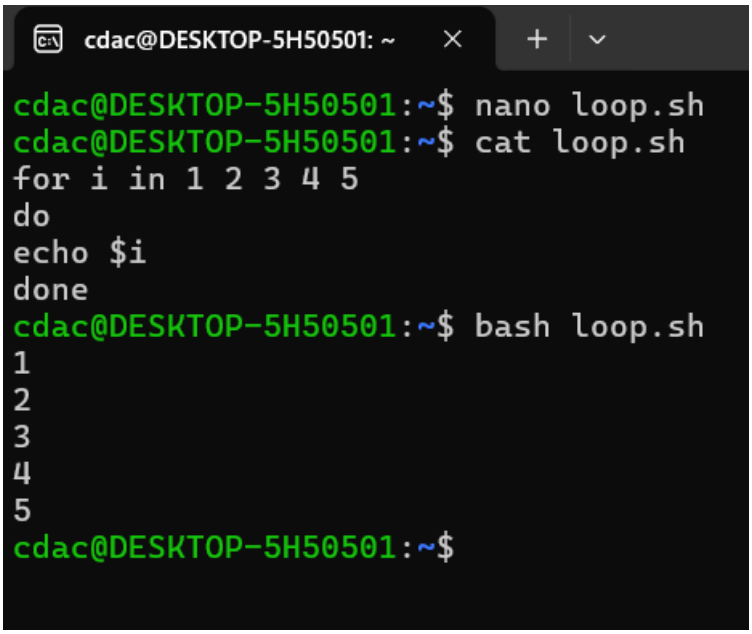
Q4.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ nano add.sh
cdac@DESKTOP-5H50501:~$ cat add.sh
echo "Enter 1st Number"
read a
echo "Enter 2nd Number"
read b
sum='expr $a + $b'
echo sum of $a and $b is $sum
cdac@DESKTOP-5H50501:~$ bash add.sh
Enter 1st Number
6
Enter 2nd Number
8
sum of 6 and 8 is expr $a + $b
cdac@DESKTOP-5H50501:~$

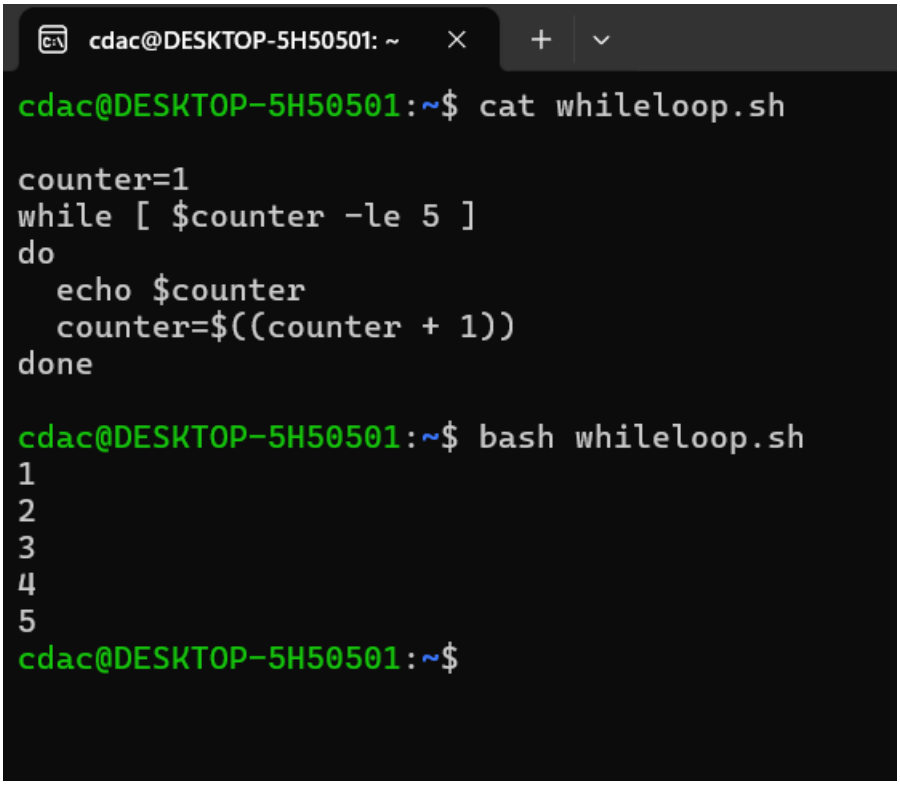
Q5.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ cat odd.sh
echo "Enter a number"
read a
if [$(($a % 2)) -eq 0]
then
echo "$a is an even Number"
else
echo "$a is an odd Number"
fi
cdac@DESKTOP-5H50501:~$ bash odd.sh
Enter a number
5
5 is an odd Number
cdac@DESKTOP-5H50501:~$

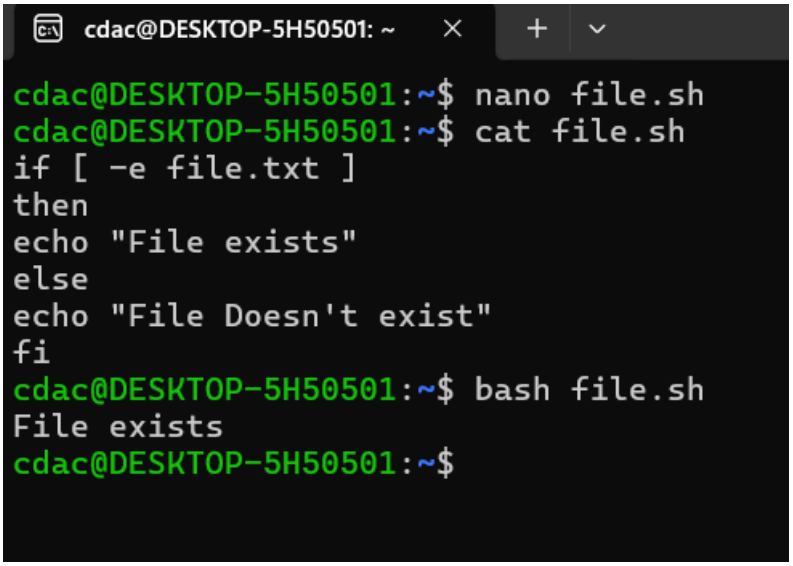
Q6.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ nano loop.sh
cdac@DESKTOP-5H50501:~$ cat loop.sh
for i in 1 2 3 4 5
do
echo $i
done
cdac@DESKTOP-5H50501:~$ bash loop.sh
1
2
3
4
5
cdac@DESKTOP-5H50501:~$

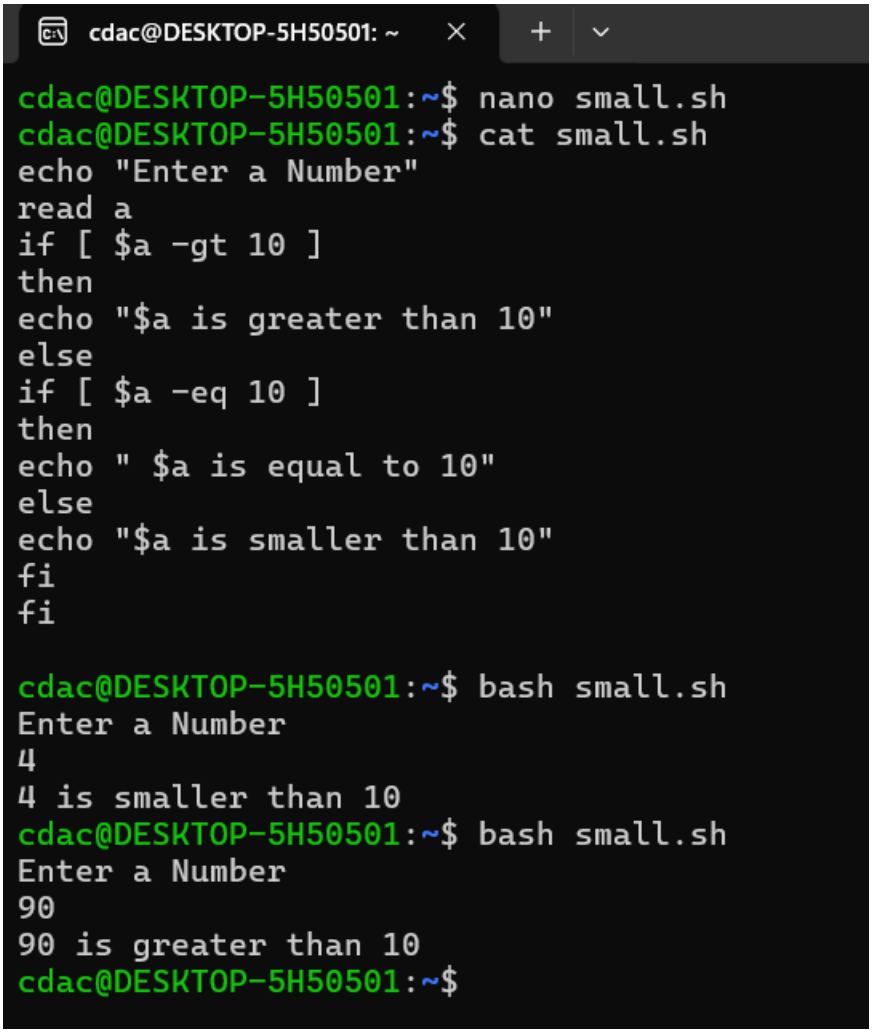
Q7.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ cat whileloop.sh
counter=1
while [$counter -le 5]
do
echo $counter
counter=$((counter + 1))
done
cdac@DESKTOP-5H50501:~$ bash whileloop.sh
1
2
3
4
5
cdac@DESKTOP-5H50501:~$

Q8.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ nano file.sh
cdac@DESKTOP-5H50501:~$ cat file.sh
if [-e file.txt]
then
echo "File exists"
else
echo "File Doesn't exist"
fi
cdac@DESKTOP-5H50501:~$ bash file.sh
File exists
cdac@DESKTOP-5H50501:~$

Q9.


cdac@DESKTOP-5H50501: ~
cdac@DESKTOP-5H50501:~$ nano small.sh
cdac@DESKTOP-5H50501:~$ cat small.sh
echo "Enter a Number"
read a
if [$a -gt 10]
then
echo "$a is greater than 10"
else
if [$a -eq 10]
then
echo " $a is equal to 10"
else
echo "$a is smaller than 10"
fi
fi
cdac@DESKTOP-5H50501:~$ bash small.sh
Enter a Number
4
4 is smaller than 10
cdac@DESKTOP-5H50501:~$ bash small.sh
Enter a Number
90
90 is greater than 10
cdac@DESKTOP-5H50501:~$

Q10.

```
cdac@DESKTOP-5H50501:~$ cat mult.sh
for i in {1..5}
do
  echo "Multiplication table for $i:"

  for j in {1..10}
  do
    result=$((i * j))
    echo "$i * $j = $result"
  done
  echo ""
done

cdac@DESKTOP-5H50501:~$ bash mult.sh
Multiplication table for 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

Multiplication table for 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

Multiplication table for 3:
3 * 1 = 3
3 * 2 = 6
```

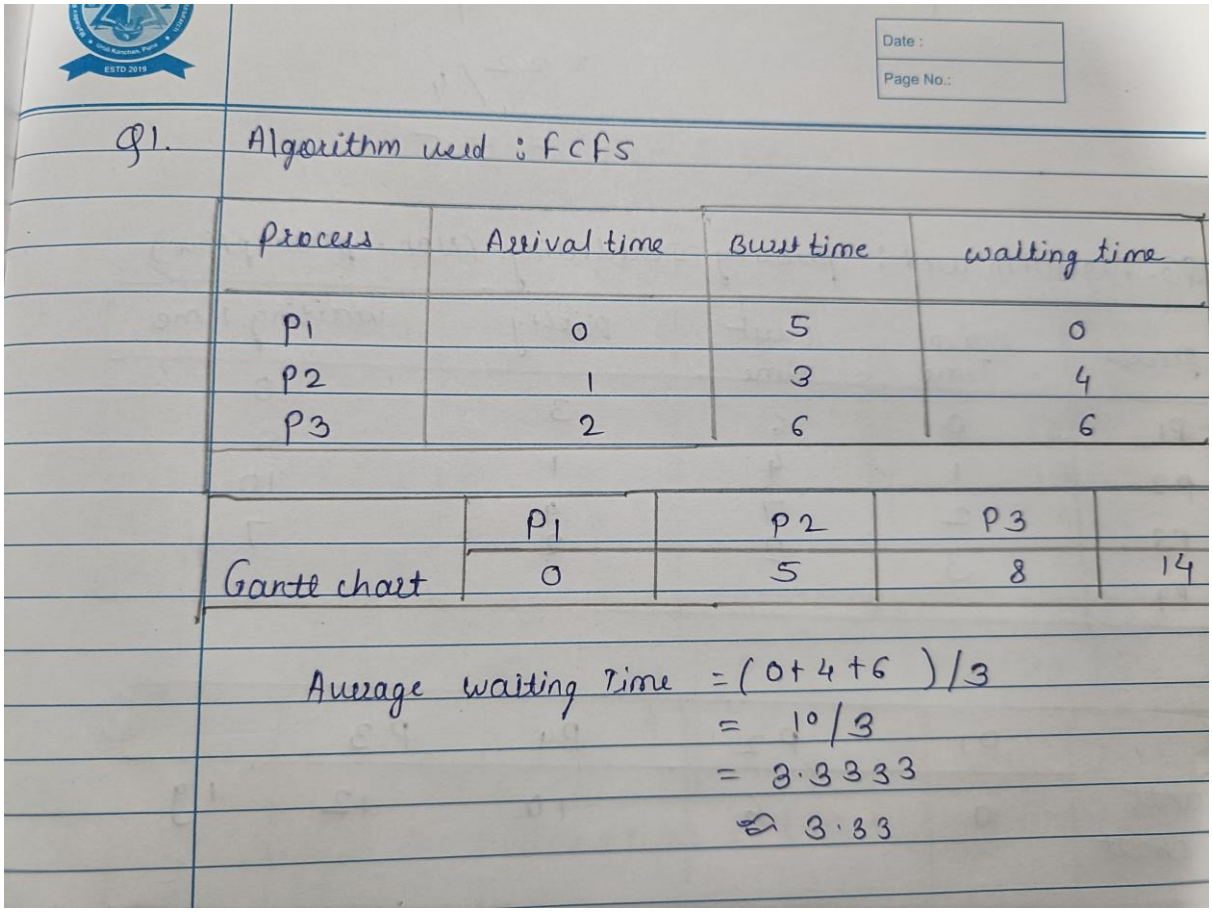
```
Multiplication table for 3:
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30

Multiplication table for 4:
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40

Multiplication table for 5:
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50

cdac@DESKTOP-5H50501:~$
```

Q11.

```
cdac@DESKTOP-5H50501:~$ cat print
num=0
while [ $num -ge 0 ]
do
echo "enter a number to exit:"
read num
if [ $num -lt 0 ]
then
break
fi
square=$((num*num))
echo "The square of sum is: $square"
done
echo "You Have Entered a negative number exiting..."
cdac@DESKTOP-5H50501:~$ bash print
enter a number to exit:
4
The square of sum is: 16
enter a number to exit:
5
The square of sum is: 25
enter a number to exit:
-2
You Have Entered a negative number exiting...
cdac@DESKTOP-5H50501:~$
```

Part E

Q1.



Q1. Algorithm used : FCFS

Process	Arrival time	Burst time	waiting time
P1	0	5	0
P2	1	3	4
P3	2	6	6

	P1	P2	P3	
Gantt chart	0	5	8	14

Average waiting time = (0+4+6)/3
= 10/3
= 3.3333
≈ 3.33

Q2.

$= 10/3$

$= 3.3333$

≈ 3.33

Q2. Algorithm used : SJF (Non-preemptive)

process	Arrival Time	Burst Time	Waiting Time	Turnaround Time
P1	0	3	0	3
P2	1	5	7	12
P3	2	1	1	2
P4	3	4	1	5

	P1	P3	P4	P2
Gantt chart	0	3	4	8

Average Turnaround Time $= \frac{3+12+2+5}{4}$

$= 22/4$

$= 5.5$

Q3.

Average turnaround time = $\frac{3+12+24+3}{4}$

$= 22/4$

$= 5.5$

Q3. Algorithm used: priority scheduling (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

Average waiting time = $\frac{22}{4}$

$= 5.5$

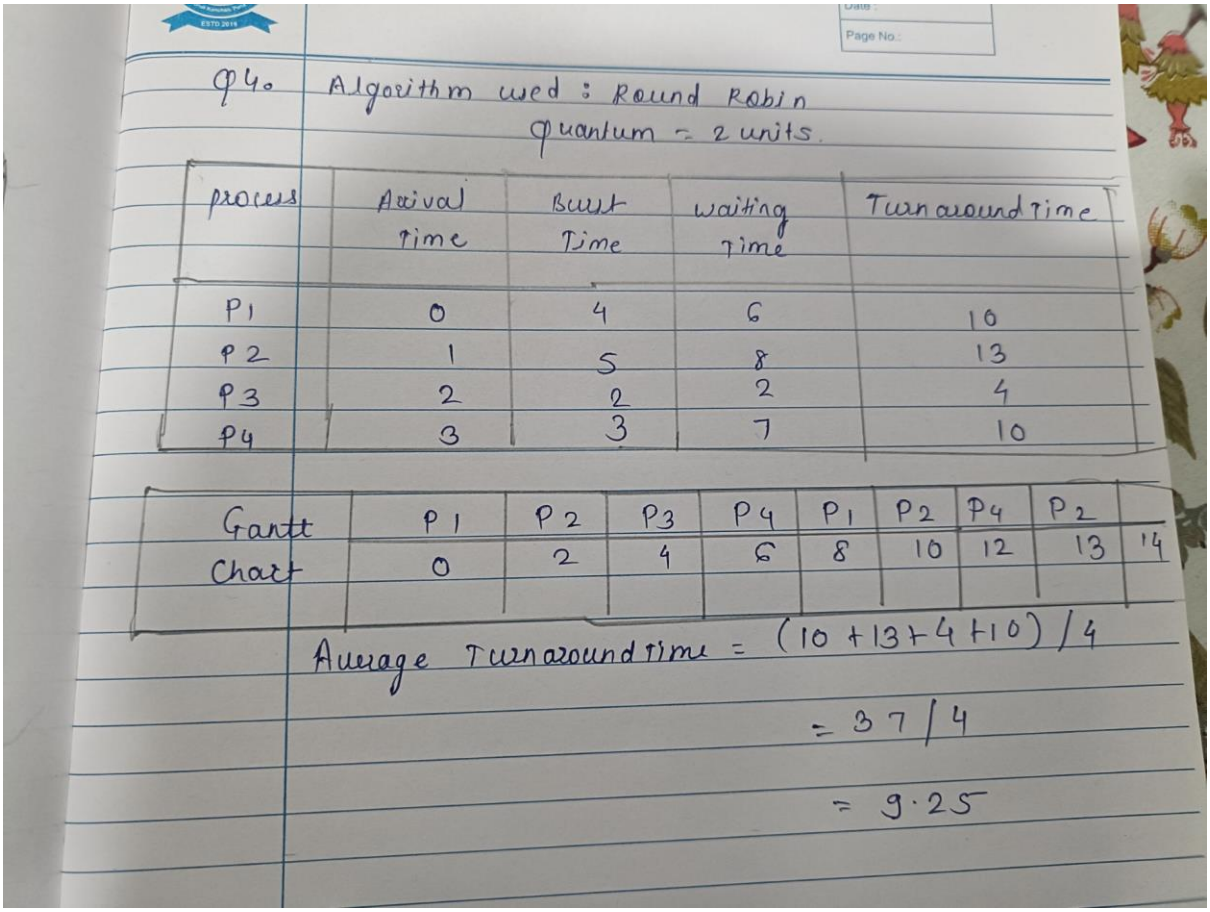
Gantt chart (Preemptive)

	P1	P2	P4	P1	P3	
	0	1	5	7	12	19

waiting time 6 0 2 10

Average WT = 18/4 = 4.5

Q4



Q4. Algorithm used : Round Robin
quantum = 2 units.

process	Arival 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

Gantt Chart	P1	P2	P3	P4	P1	P2	P4	P2	
	0	2	4	6	8	10	12	13	14

Average Turnaround time = (10 + 13 + 4 + 10) / 4

= 37 / 4

= 9.25