CDAC MUMBAI

Concepts Of Operating System

Assignment No. 2

PART A

What will the following commands do ?

- echo “Hello World”

  echo command is used to just paste the content written inside it.

- name=“Productive”

  This command creates variable name and assigns string value “Productive”.

- touch file1.txt

  This command creates text file having name file1.

- ls -a

  This command is used to list all the directories and files.

- rm file.txt

  This command used to remove/delete the file.

- cp file1.txt file2.txt

  cp command creates new file2.txt and copies the content of file1.txt in it.

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

  This command moves the file.txt in specific directory whose path should be given by user.

- chmod 755 script.sh

  This command is used to change permission of file script.sh

Digits	Binary	Sequence	Permissions
7	111	Owner	Read, Write & Execute
5	101	Group	Read & Execute
5	101	Others	Read & Execute

- grep “pattren” file.txt

  grep command is used to find the specific word now in this command it used to find “pattern” word in file.txt and if he finds the word in file then print those lines.

- kill PID

  This command is used to terminate the running process by using

  PID – Process ID

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

  In this command set of instructions are there

  Step 1 - creating directory named mydir.

  Step 2 – moving inside mydir directory.

  Step 3 – creating file named file,txt

  Step 4 – saving content “Hello, World!” inside file.txt

  Step5 – printing the content in file.txt by cat command.

- ls -l | grep “.txt”

  This command gives the detailed information about files having extension .txt

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

  In this command cat file1.txt file2.txt concatenates both file line by line then sort command sort combined output alphabetically at last uniq eliminates repeated content and print the output.

- ls -l | grep "^d"

  This whole command print the detailed information of only directories due to grep "^d"

- chmod 644 file.txt

  This command used to change the permission of file .txt

Digits	Binary	Sequence	Permissions
6	110	Owner	Read & Write
4	100	Group	Read
4	100	Others	Read

- cp -r source_directory destination_directory

  This command copies all the content of source_directory including subdirectories and files in destination_directory.

- find /path/to/search – name ".txt"

  This command finds all ".txt" files in the defined path.

- chmod u+x file.txt

  This command gives permission to owner/user to execute the file.txt

- echo $PATH

  This command prints list of directories in which shell goes through to find executable commands.

Part B

Identify True or False:

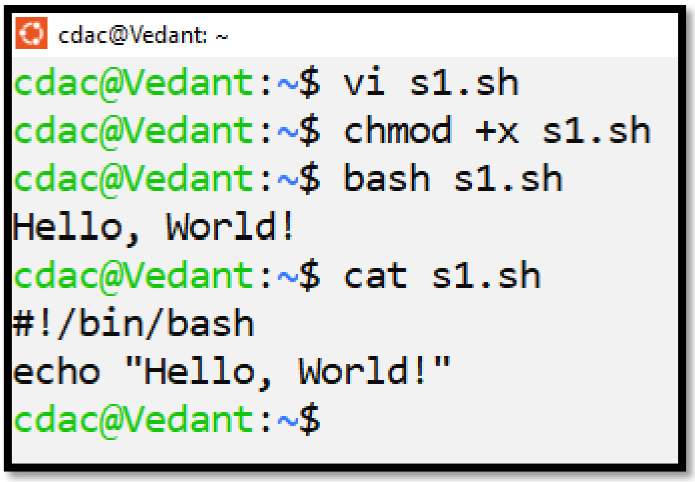
1. ls is used to list files and directories in a directory. {**True**}
2. mv is used to move files and directories. {**True**}
3. cd is used to copy files and directories. {**False**}
4. pwd stands for "print working directory" and displays the current directory. {**True**}
5. grep is used to search 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. {**Incorrect**}
2. cpy is used to copy files and directories. {**Incorrect**}
3. mkfile is used to create a new file. {**Incorrect**}
4. catx is used to concatenate files. {**Incorrect**}
5. rn is used to rename files. {**Incorrect**}

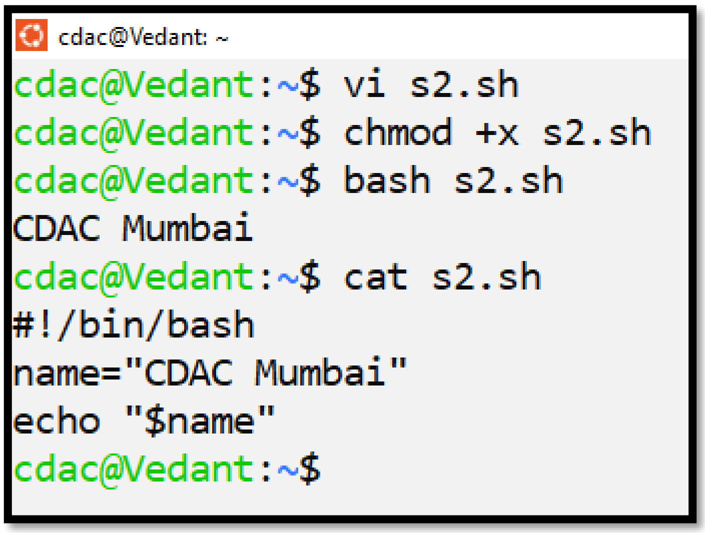
PART C

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



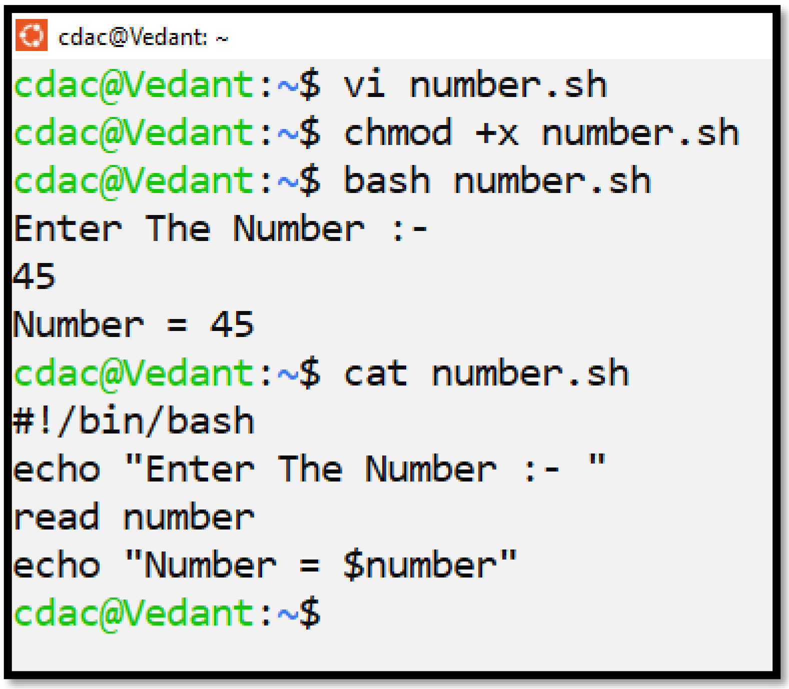
```
cdac@Vedant:~$ vi s1.sh
cdac@Vedant:~$ chmod +x s1.sh
cdac@Vedant:~$ bash s1.sh
Hello, World!
cdac@Vedant:~$ cat s1.sh
#!/bin/bash
echo "Hello, World!"
cdac@Vedant:~$
```

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



```
cdac@Vedant:~$ vi s2.sh
cdac@Vedant:~$ chmod +x s2.sh
cdac@Vedant:~$ bash s2.sh
CDAC Mumbai
cdac@Vedant:~$ cat s2.sh
#!/bin/bash
name="CDAC Mumbai"
echo "$name"
cdac@Vedant:~$
```

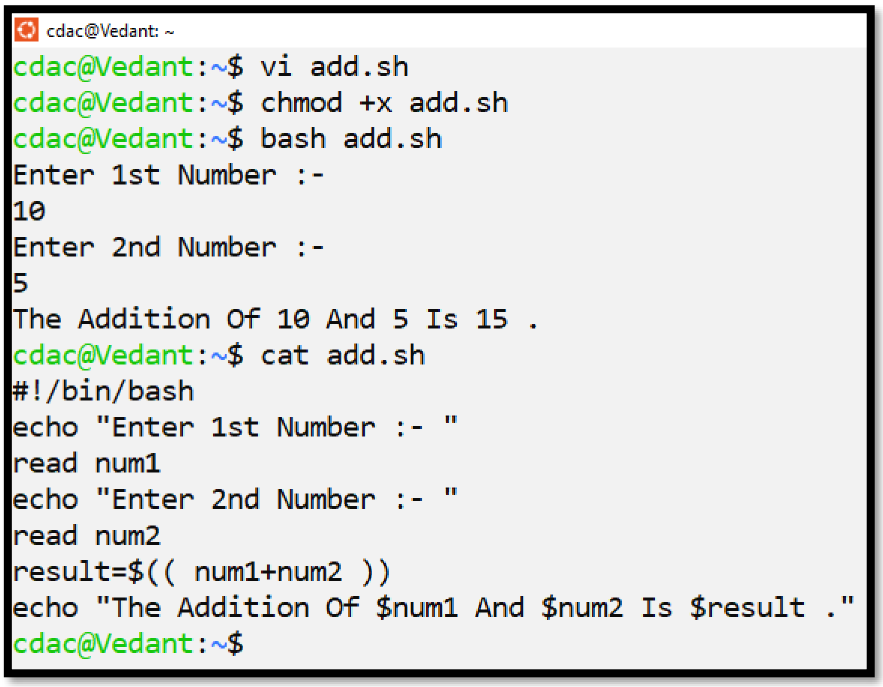
Question 3: Write a shell script that takes a number as input from the user and prints it.



```
cdac@Vedant:~$ vi number.sh
cdac@Vedant:~$ chmod +x number.sh
cdac@Vedant:~$ bash number.sh
Enter The Number :-
45
Number = 45
cdac@Vedant:~$ cat number.sh
#!/bin/bash
echo "Enter The Number :- "
read number
echo "Number = $number"
cdac@Vedant:~$
```

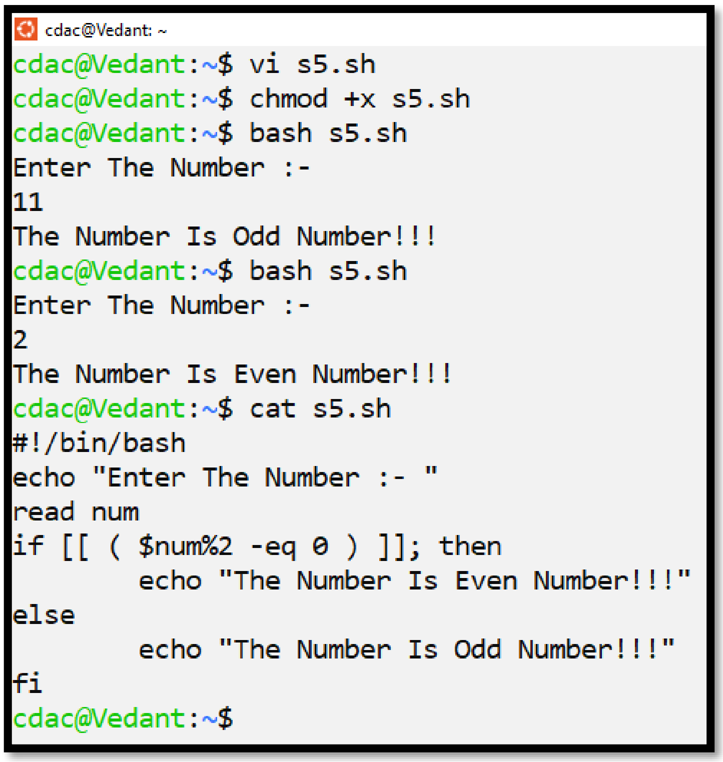
Question 4: Write a shell script that performs addition of two numbers

(e.g., 5 and 3) and prints the result.



```
cdac@Vedant:~$ vi add.sh
cdac@Vedant:~$ chmod +x add.sh
cdac@Vedant:~$ bash add.sh
Enter 1st Number :-
10
Enter 2nd Number :-
5
The Addition Of 10 And 5 Is 15 .
cdac@Vedant:~$ cat add.sh
#!/bin/bash
echo "Enter 1st Number :- "
read num1
echo "Enter 2nd Number :- "
read num2
result=$(( num1+num2 ))
echo "The Addition Of $num1 And $num2 Is $result ."
cdac@Vedant:~$
```

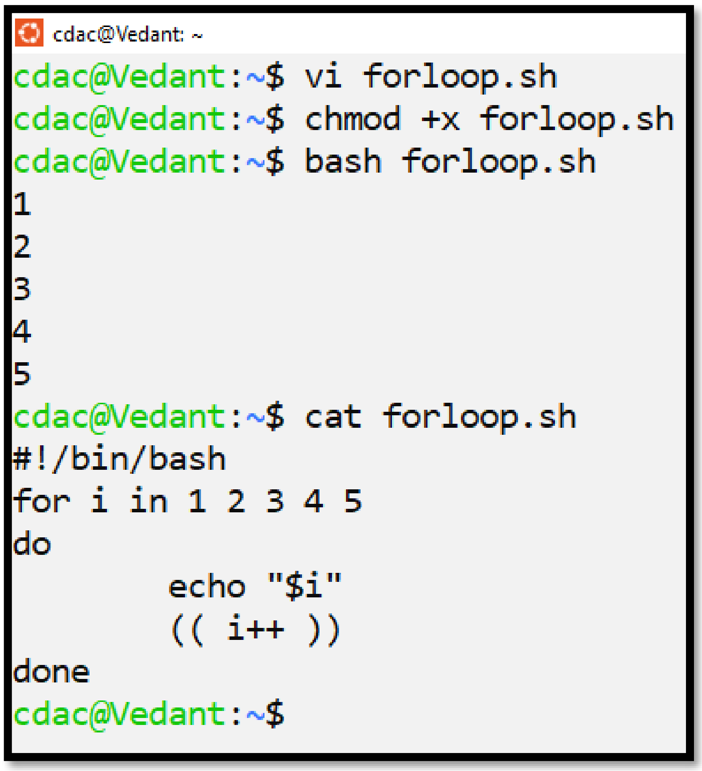
Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".



```
cdac@Vedant: ~
cdac@Vedant:~$ vi s5.sh
cdac@Vedant:~$ chmod +x s5.sh
cdac@Vedant:~$ bash s5.sh
Enter The Number :-
11
The Number Is Odd Number!!!
cdac@Vedant:~$ bash s5.sh
Enter The Number :-
2
The Number Is Even Number!!!
cdac@Vedant:~$ cat s5.sh
#!/bin/bash
echo "Enter The Number :- "
read num
if [[ ( $num%2 -eq 0 ) ]]; then
        echo "The Number Is Even Number!!!"
else
        echo "The Number Is Odd Number!!!"
fi
cdac@Vedant:~$
```

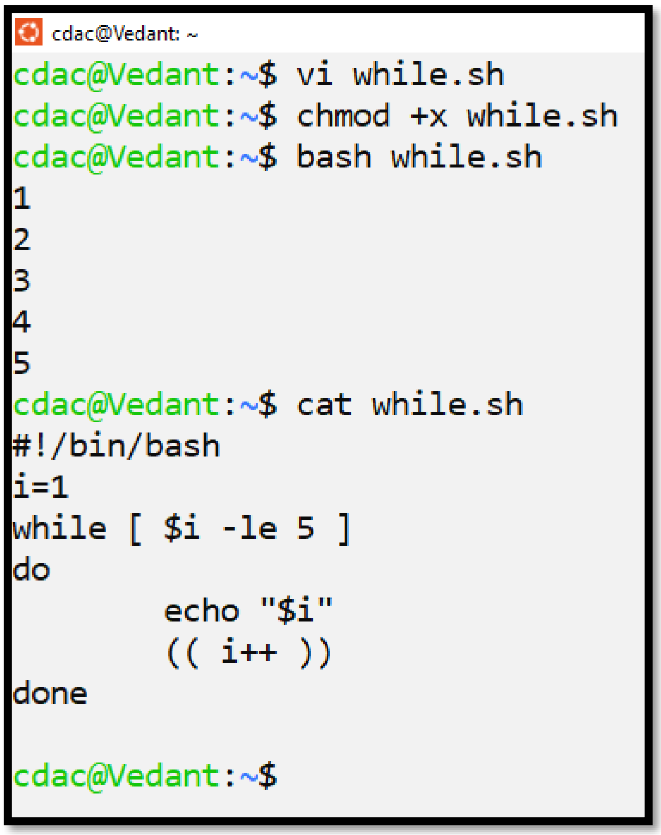
Question 6: Write a shell script that uses a for loop to print numbers

from 1 to 5.



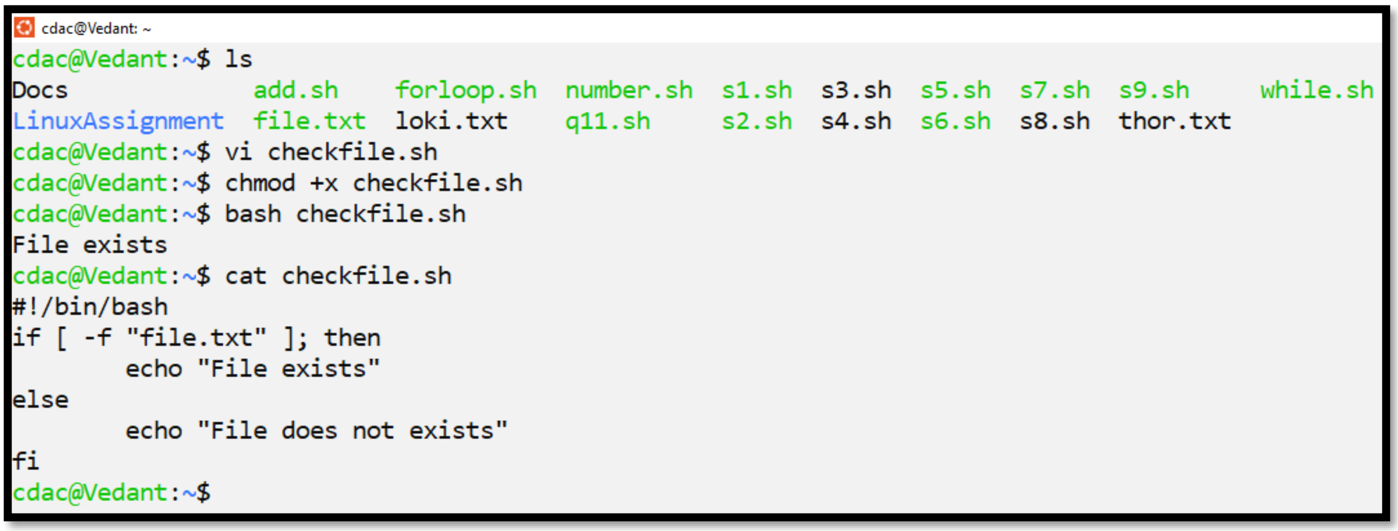
```
cdac@Vedant: ~
cdac@Vedant:~$ vi forloop.sh
cdac@Vedant:~$ chmod +x forloop.sh
cdac@Vedant:~$ bash forloop.sh
1
2
3
4
5
cdac@Vedant:~$ cat forloop.sh
#!/bin/bash
for i in 1 2 3 4 5
do
        echo "$i"
        (( i++ ))
done
cdac@Vedant:~$
```

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



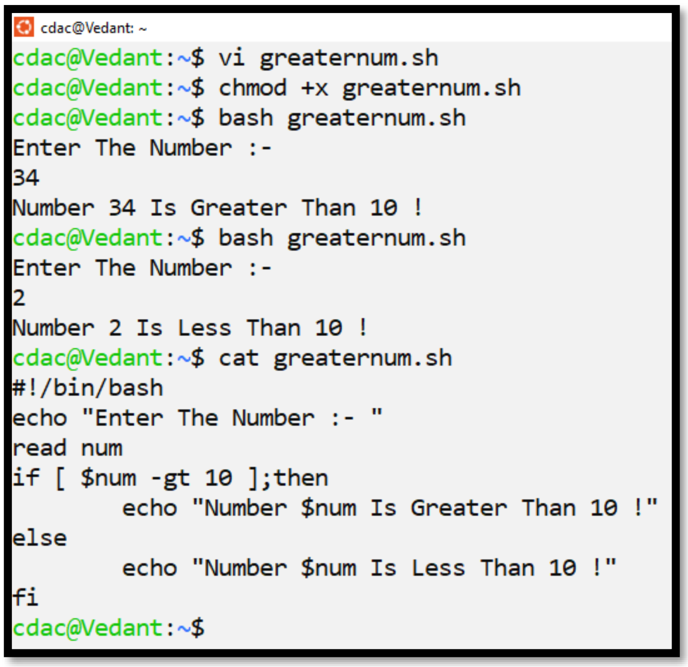
```
cdac@Vedant:~$ vi while.sh
cdac@Vedant:~$ chmod +x while.sh
cdac@Vedant:~$ bash while.sh
1
2
3
4
5
cdac@Vedant:~$ cat while.sh
#!/bin/bash
i=1
while [ $i -le 5 ]
do
        echo "$i"
        (( i++ ))
done

cdac@Vedant:~$
```

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@Vedant:~$ ls
Docs             add.sh    forloop.sh  number.sh  s1.sh  s3.sh  s5.sh  s7.sh  s9.sh     while.sh
LinuxAssignment  file.txt  loki.txt    q11.sh     s2.sh  s4.sh  s6.sh  s8.sh  thor.txt
cdac@Vedant:~$ vi checkfile.sh
cdac@Vedant:~$ chmod +x checkfile.sh
cdac@Vedant:~$ bash checkfile.sh
File exists
cdac@Vedant:~$ cat checkfile.sh
#!/bin/bash
if [ -f "file.txt" ]; then
        echo "File exists"
else
        echo "File does not exists"
fi
cdac@Vedant:~$
```

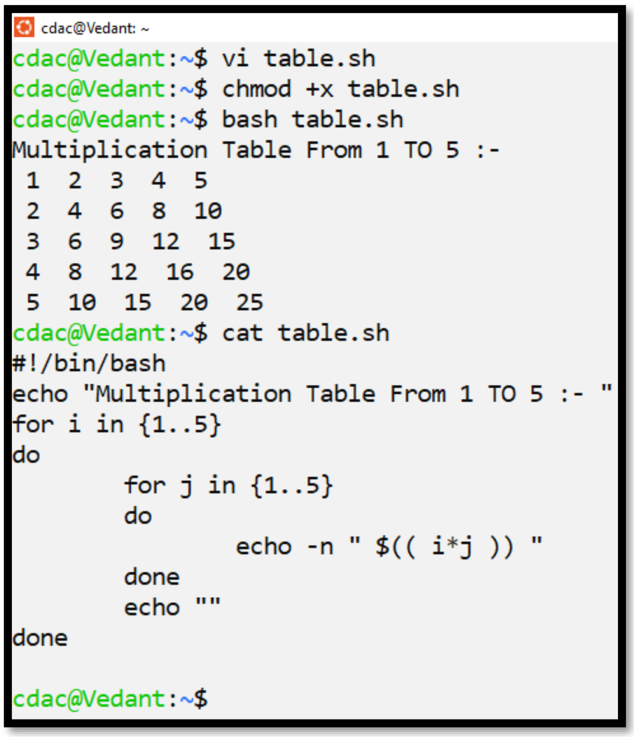
Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.



cdac@Vedant: ~

```
cdac@Vedant:~$ vi greaternum.sh
cdac@Vedant:~$ chmod +x greaternum.sh
cdac@Vedant:~$ bash greaternum.sh
Enter The Number :-
34
Number 34 Is Greater Than 10 !
cdac@Vedant:~$ bash greaternum.sh
Enter The Number :-
2
Number 2 Is Less Than 10 !
cdac@Vedant:~$ cat greaternum.sh
#!/bin/bash
echo "Enter The Number :- "
read num
if [ $num -gt 10 ];then
        echo "Number $num Is Greater Than 10 !"
else
        echo "Number $num Is Less Than 10 !"
fi
cdac@Vedant:~$
```

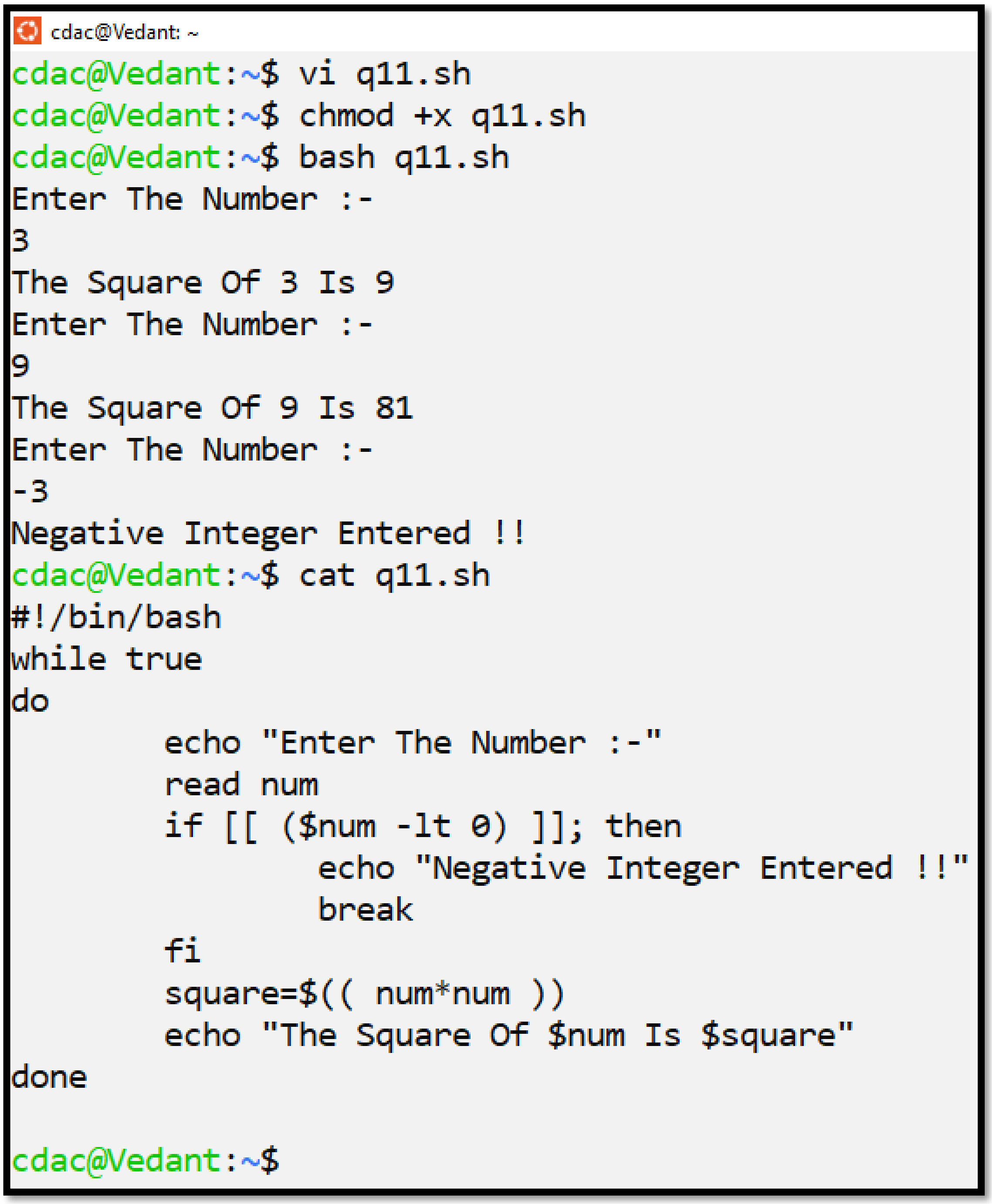
Question 10: Write a shell script that uses nested for loops to print a multiplication table for numbers from 1 to 5. The output should be formatted nicely, with each row representing a number and each column representing the multiplication result for that number.



cdac@Vedant: ~

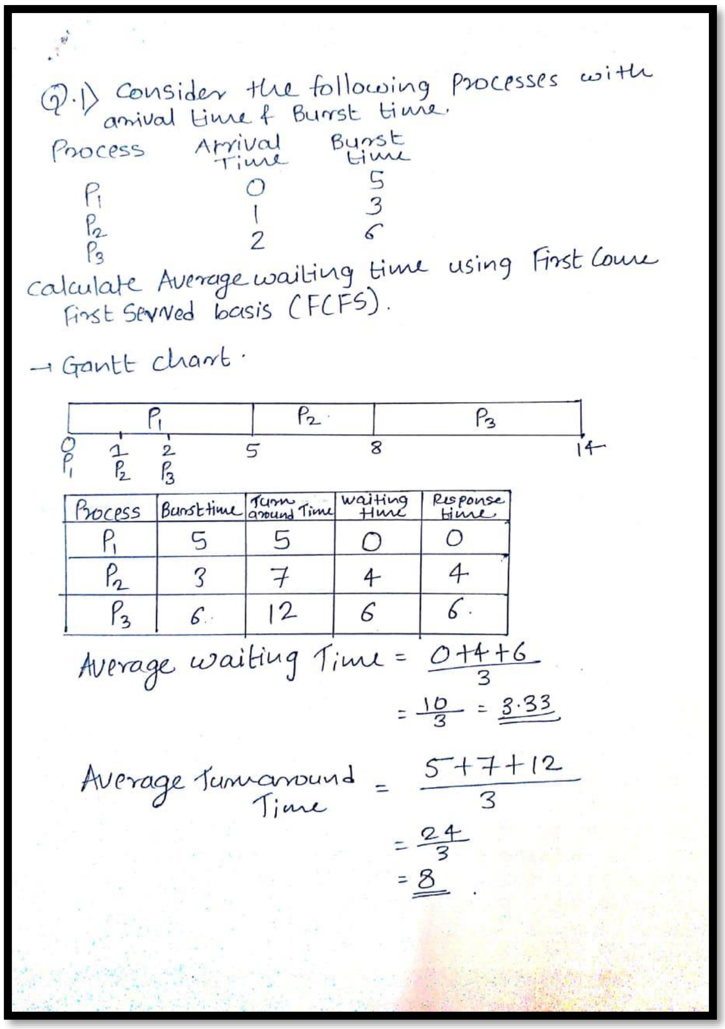
```
cdac@Vedant:~$ vi table.sh
cdac@Vedant:~$ chmod +x table.sh
cdac@Vedant:~$ bash table.sh
Multiplication Table From 1 TO 5 :-
 1  2  3  4  5
 2  4  6  8  10
 3  6  9  12  15
 4  8  12  16  20
 5  10  15  20  25
cdac@Vedant:~$ cat table.sh
#!/bin/bash
echo "Multiplication Table From 1 TO 5 :- "
for i in {1..5}
do
        for j in {1..5}
        do
                echo -n " $(( i*j )) "
        done
        echo ""
done

cdac@Vedant:~$
```

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@Vedant:~$ vi q11.sh
cdac@Vedant:~$ chmod +x q11.sh
cdac@Vedant:~$ bash q11.sh
Enter The Number :-
3
The Square Of 3 Is 9
Enter The Number :-
9
The Square Of 9 Is 81
Enter The Number :-
-3
Negative Integer Entered !!
cdac@Vedant:~$ cat q11.sh
#!/bin/bash
while true
do
        echo "Enter The Number :-"
        read num
        if [[ ($num -lt 0) ]]; then
                echo "Negative Integer Entered !!"
                break
        fi
        square=$(( num*num ))
        echo "The Square Of $num Is $square"
done

cdac@Vedant:~$
```

PART E



Q.1) Consider the following processes with arrival time & Burst time.

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

Calculate Average waiting time using First Come First Served basis (FCFS).

→ Gantt chart.

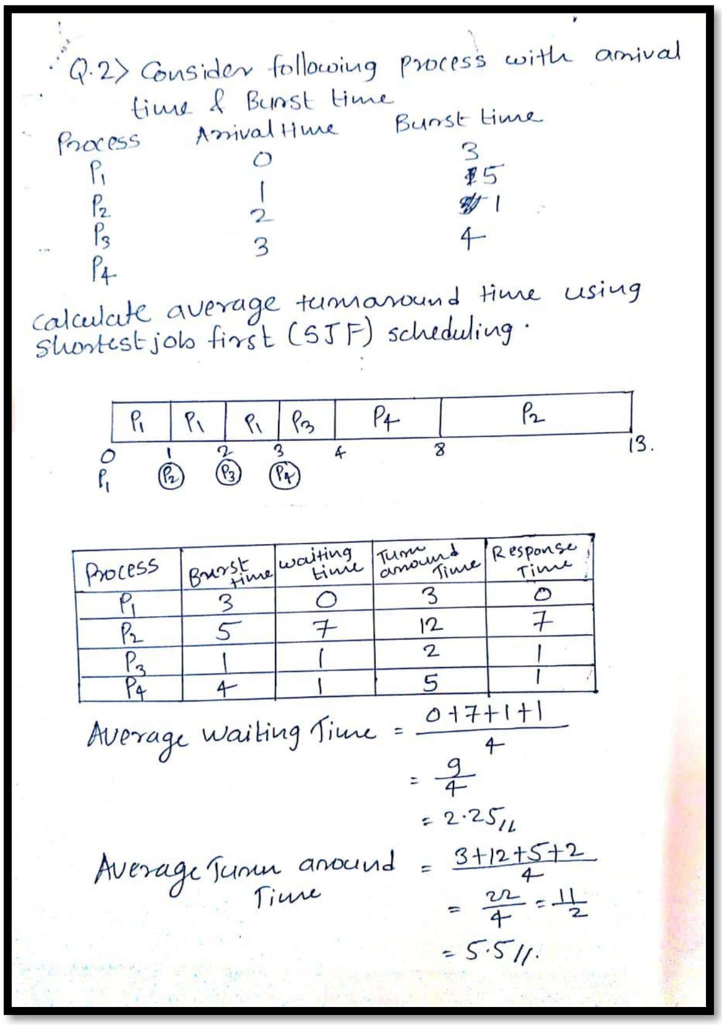
P_1	P_2	P_3

0 (P_1) 1 (P_2) 2 (P_3) 5 8 14

Process	Burst time	Turn around Time	waiting time	Response time
P_1	5	5	0	0
P_2	3	7	4	4
P_3	6	12	6	6

$$\text{Average waiting Time} = \frac{0+4+6}{3} = \frac{10}{3} = \underline{\underline{3.33}}$$

$$\text{Average Turnaround Time} = \frac{5+7+12}{3} = \frac{24}{3} = \underline{\underline{8}}$$



Q.2) Consider following process with arrival time & Burst time

Process	Arrival time	Burst time
P_1	0	3
P_2	1	~~4~~5
P_3	2	~~3~~1
P_4	3	4

calculate average turnaround time using shortest job first (SJF) scheduling.

P_1	P_1	P_1	P_3	P_4	P_2

0 (P_1) 1 (P_2) 2 (P_3) 3 (P_4) 4 8 13.

Process	Burst time	waiting time	Turn around Time	Response Time
P_1	3	0	3	0
P_2	5	7	12	7
P_3	1	1	2	1
P_4	4	1	5	1

$$\text{Average waiting Time} = \frac{0+7+1+1}{4} = \frac{9}{4} = 2.25//$$

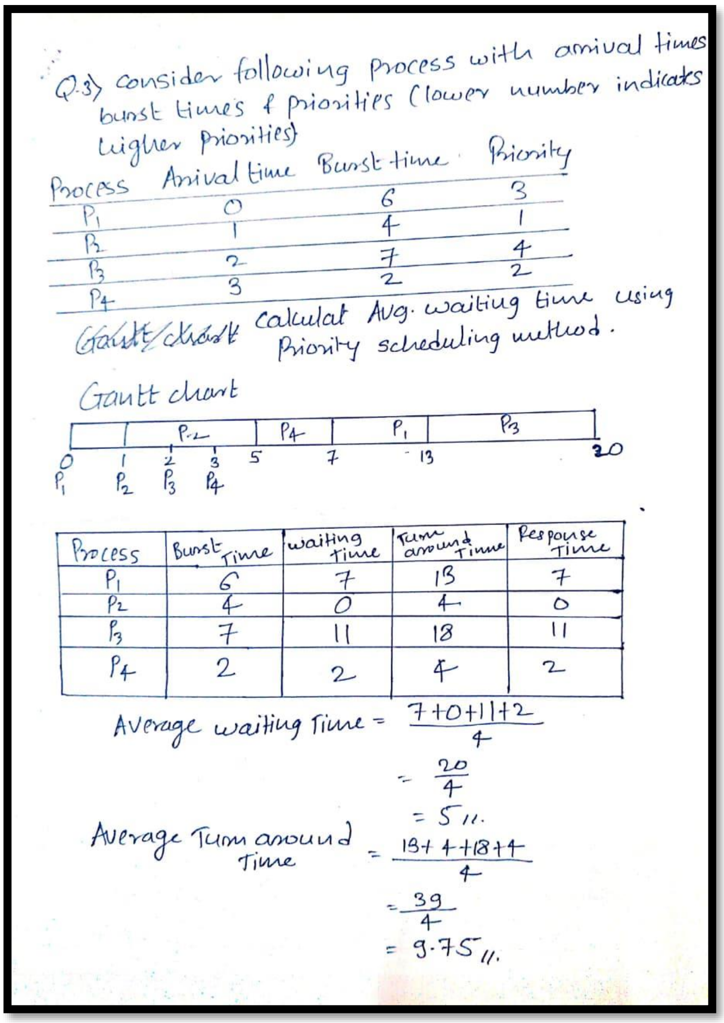
$$\text{Average Turn around Time} = \frac{3+12+5+2}{4} = \frac{22}{4} = \frac{11}{2} = 5.5//.$$

FIRST COME FIRST SERVE SCHEDULING METHOD

(FCFS)

SHORTEST JOB FIRST SCHEDULING METHOD

(SJF)



Q.3) consider following process with arrival times burst times & priorities (lower number indicates higher priorities)

Process	Arrival time	Burst time	Priority
P_1	0	6	3
P_2	1	4	1
P_3	2	7	4
P_4	3	2	2

~~Gantt chart~~ calculat Avg. waiting time using Priority scheduling method.

Gantt chart

	P_2	P_4	P_1	P_3

0 1 2 3 5 7 13 20

P_1 P_2 P_3 P_4

Process	Burst Time	waiting time	Turn around Time	Response Time
P_1	6	7	13	7
P_2	4	0	4	0
P_3	7	11	18	11
P_4	2	2	4	2

Average waiting Time $= \frac{7+0+11+2}{4}$

$= \frac{20}{4}$

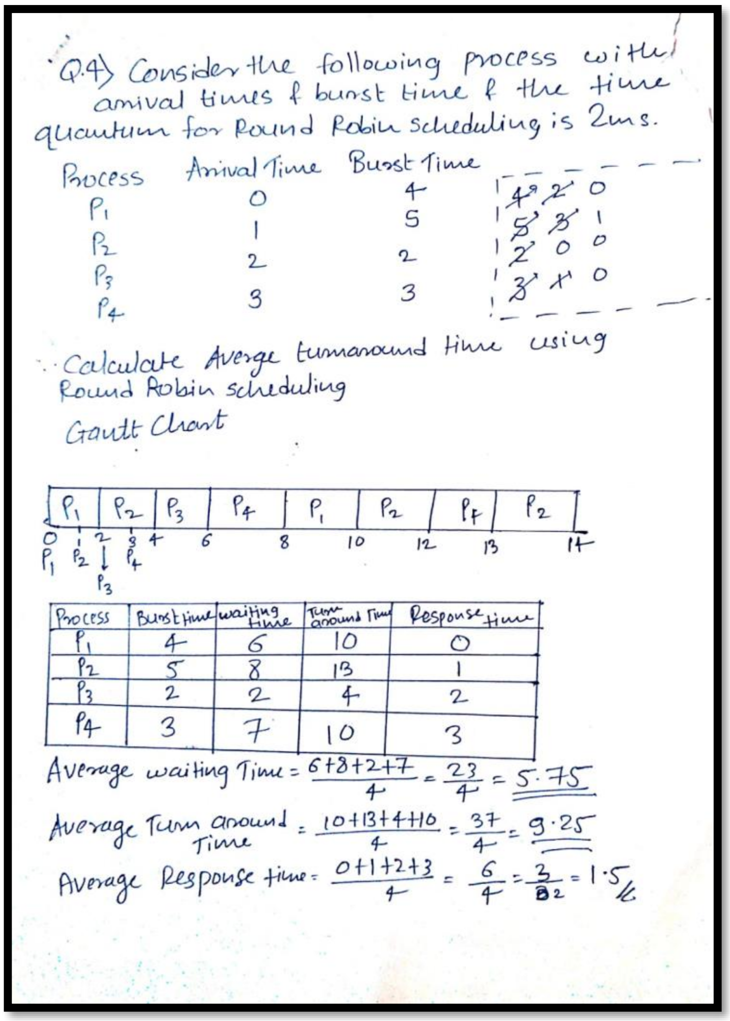
$= 5$ //.

Average Turn around Time $= \frac{13+4+18+4}{4}$

$= \frac{39}{4}$

$= 9.75$ //.

PRIORITY SCHEDULING METHOD



Q.4) Consider the following process with arrival times & burst time & the time quantum for Round Robin Scheduling is 2ms.

Process	Arrival Time	Burst Time			
P_1	0	4	~~4~~	~~2~~	0
P_2	1	5	~~5~~	~~3~~	1
P_3	2	2	~~2~~	0	0
P_4	3	3	~~3~~	~~1~~	0

Calculate Average turnaround time using Round Robin scheduling

Gantt Chart

P_1	P_2	P_3	P_4	P_1	P_2	P_4	P_2

0 1 2 3 4 6 8 10 12 13 14

P_1 P_2 P_3 P_4

Process	Burst time	waiting time	Turn around Time	Response time
P_1	4	6	10	0
P_2	5	8	13	1
P_3	2	2	4	2
P_4	3	7	10	3

Average waiting Time $= \frac{6+8+2+7}{4} = \frac{23}{4} = \underline{\underline{5.75}}$

Average Turn around Time $= \frac{10+13+4+10}{4} = \frac{37}{4} = \underline{\underline{9.25}}$

Average Response time $= \frac{0+1+2+3}{4} = \frac{6}{4} = \frac{3}{2} = 1.5$ //

ROUND ROBIN SCHEDULING METHOD

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?

Answer :- fork() will create a separate copy of the process means, child process x with value 5. Both child and parent cases are individual and the final values after incrementing them will be child process – 6 and parent process - 6