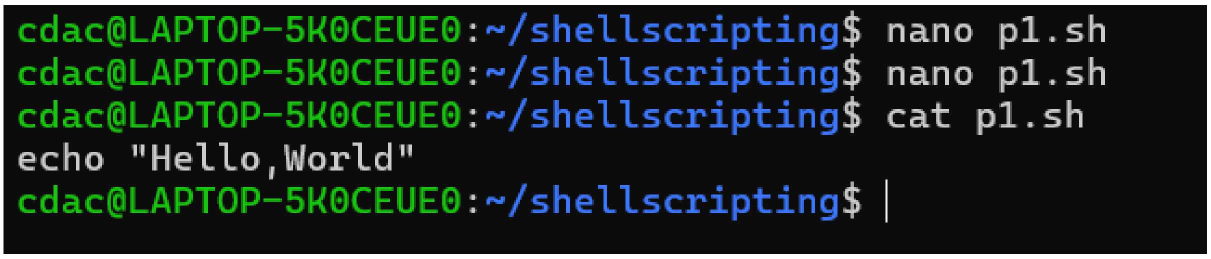
Concepts of Operating System Assignment 2

Part A

What will the following commands do?

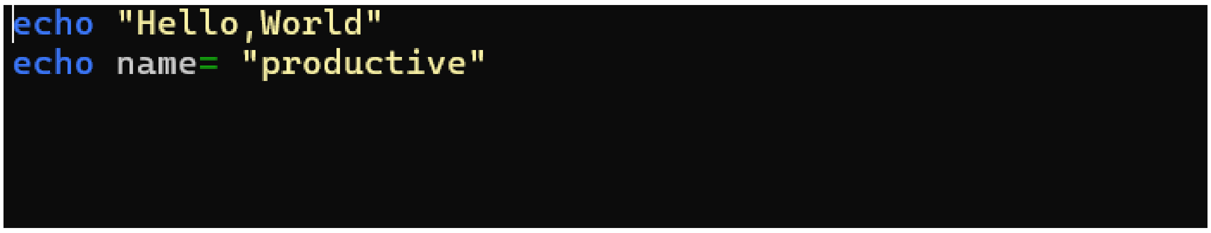
 echo "Hello, World!"



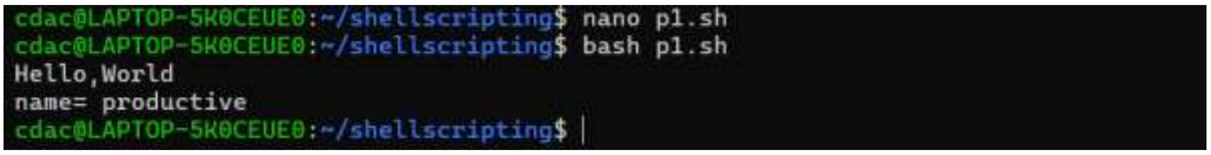
```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p1.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p1.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ cat p1.sh
echo "Hello,World"
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

Echo is used to print the given line as a output

 name="Productive"

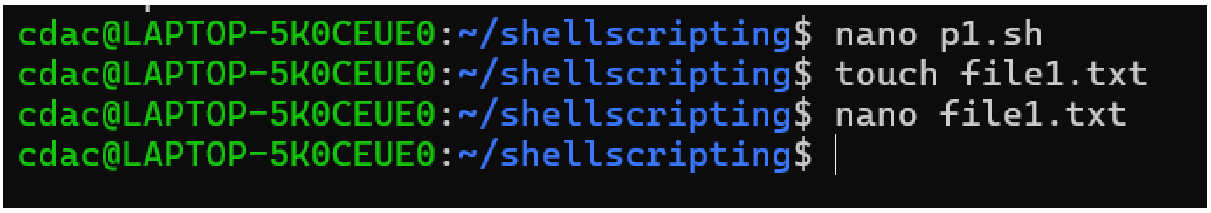


```
echo "Hello,World"
echo name= "productive"
```



```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p1.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p1.sh
Hello,World
name= productive
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

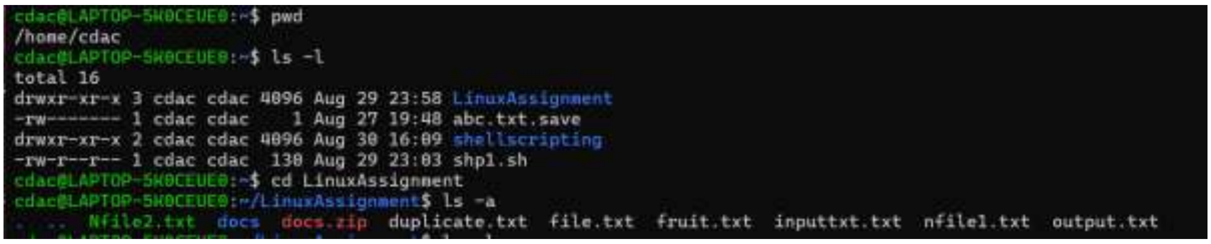
 touch file.txt



```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p1.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ touch file1.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano file1.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

Touch Command is use to create a new file

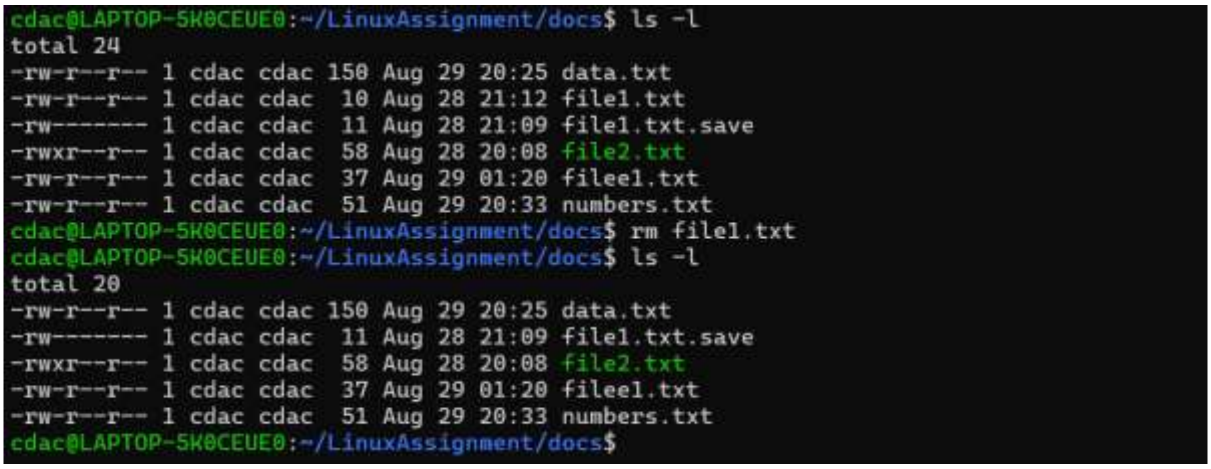
 ls -a



```
cdac@LAPTOP-5K0CEUE0:~$ pwd
/home/cdac
cdac@LAPTOP-5K0CEUE0:~$ ls -l
total 16
drwxr-xr-x 3 cdac cdac 4096 Aug 29 23:58 LinuxAssignment
-rw------- 1 cdac cdac    1 Aug 27 19:48 abc.txt.save
drwxr-xr-x 2 cdac cdac 4096 Aug 30 16:09 shellscripting
-rw-r--r-- 1 cdac cdac  130 Aug 29 23:03 shp1.sh
cdac@LAPTOP-5K0CEUE0:~$ cd LinuxAssignment
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ ls -a
.  ..  Nfile2.txt  docs  docs.zip  duplicate.txt  file.txt  fruit.txt  inputtxt.txt  nfile1.txt  output.txt
```

List of all files including the hidden file

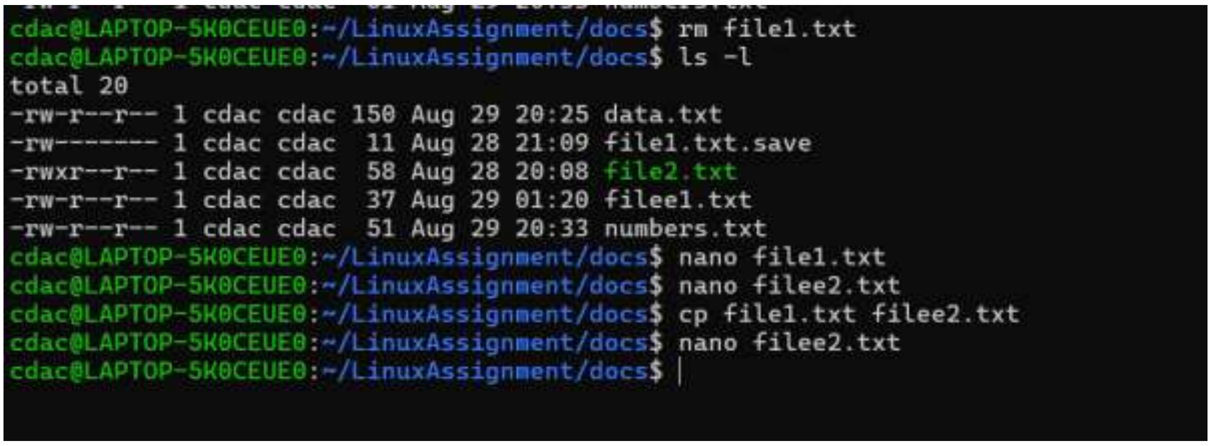
 rm file.txt



```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ ls -l
total 24
-rw-r--r-- 1 cdac cdac 150 Aug 29 20:25 data.txt
-rw-r--r-- 1 cdac cdac  10 Aug 28 21:12 file1.txt
-rw------- 1 cdac cdac  11 Aug 28 21:09 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Aug 28 20:08 file2.txt
-rw-r--r-- 1 cdac cdac  37 Aug 29 01:20 fileel.txt
-rw-r--r-- 1 cdac cdac  51 Aug 29 20:33 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ rm file1.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ ls -l
total 20
-rw-r--r-- 1 cdac cdac 150 Aug 29 20:25 data.txt
-rw------- 1 cdac cdac  11 Aug 28 21:09 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Aug 28 20:08 file2.txt
-rw-r--r-- 1 cdac cdac  37 Aug 29 01:20 fileel.txt
-rw-r--r-- 1 cdac cdac  51 Aug 29 20:33 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$
```

Remove single file

 cp file1.txt file2.txt

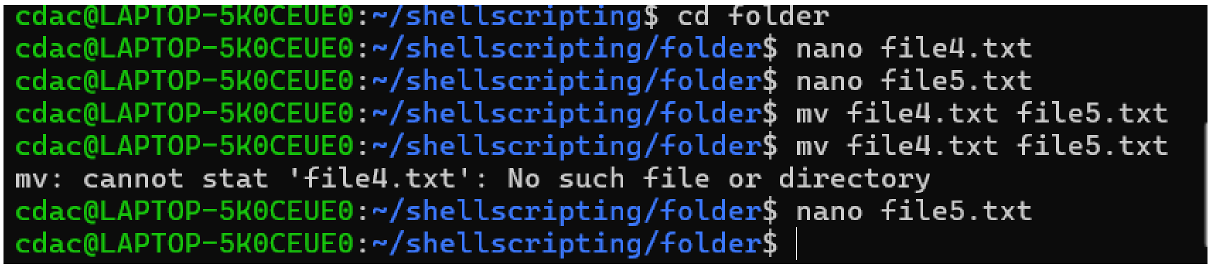


```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ rm file1.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ ls -l
total 20
-rw-r--r-- 1 cdac cdac 150 Aug 29 20:25 data.txt
-rw------- 1 cdac cdac  11 Aug 28 21:09 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Aug 28 20:08 file2.txt
-rw-r--r-- 1 cdac cdac  37 Aug 29 01:20 fileel.txt
-rw-r--r-- 1 cdac cdac  51 Aug 29 20:33 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ nano file1.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ nano filee2.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cp file1.txt filee2.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ nano filee2.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ |
```

Cp command is used to copy the content of file1 to file2

This command will copy the files and directories from the source path to the destination path

 mv file.txt /path/to/directory/

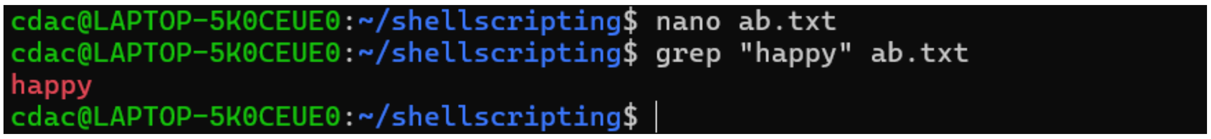

```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ cd folder
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$ nano file4.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$ nano file5.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$ mv file4.txt file5.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$ mv file4.txt file5.txt
mv: cannot stat 'file4.txt': No such file or directory
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$ nano file5.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting/folder$
```

mv command use to move the file.txt to the directory

 chmod 755 script.sh

chmod 755 script.sh sets the permissions of the file script.sh to 755.

 grep "pattern" file.txt

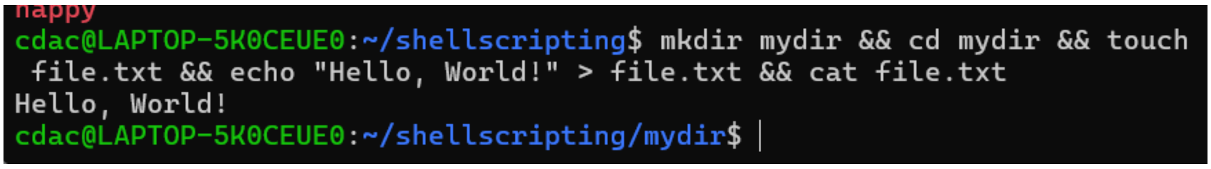

```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano ab.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ grep "happy" ab.txt
happy
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

grep "pattern" file.txt searches for the specified pattern within the file file.txt and displays lines that match the pattern.

 kill PID

The command kill PID is used to terminate a process in Unix-like operating systems, where PID stands for the Process ID of the process you want to stop.

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


```
happy
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ mkdir mydir && cd mydir && touch
 file.txt && echo "Hello, World!" > file.txt && cat file.txt
Hello, World!
cdac@LAPTOP-5K0CEUE0:~/shellscripting/mydir$
```

This sequence of commands:

Creates a directory named mydir.

Changes into that directory.

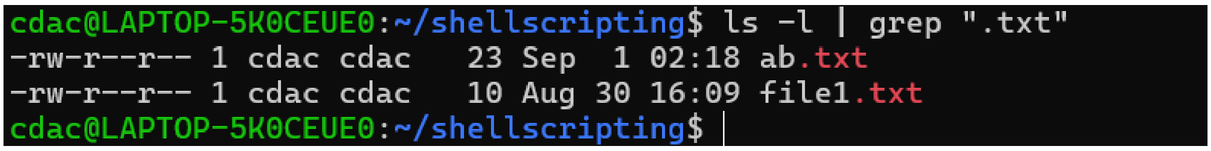
Creates an empty file named file.txt.

Writes Hello, World! to file.txt.

Displays the contents of file.txt, which will be Hello, World!.

If any command fails, the subsequent commands will not be executed.

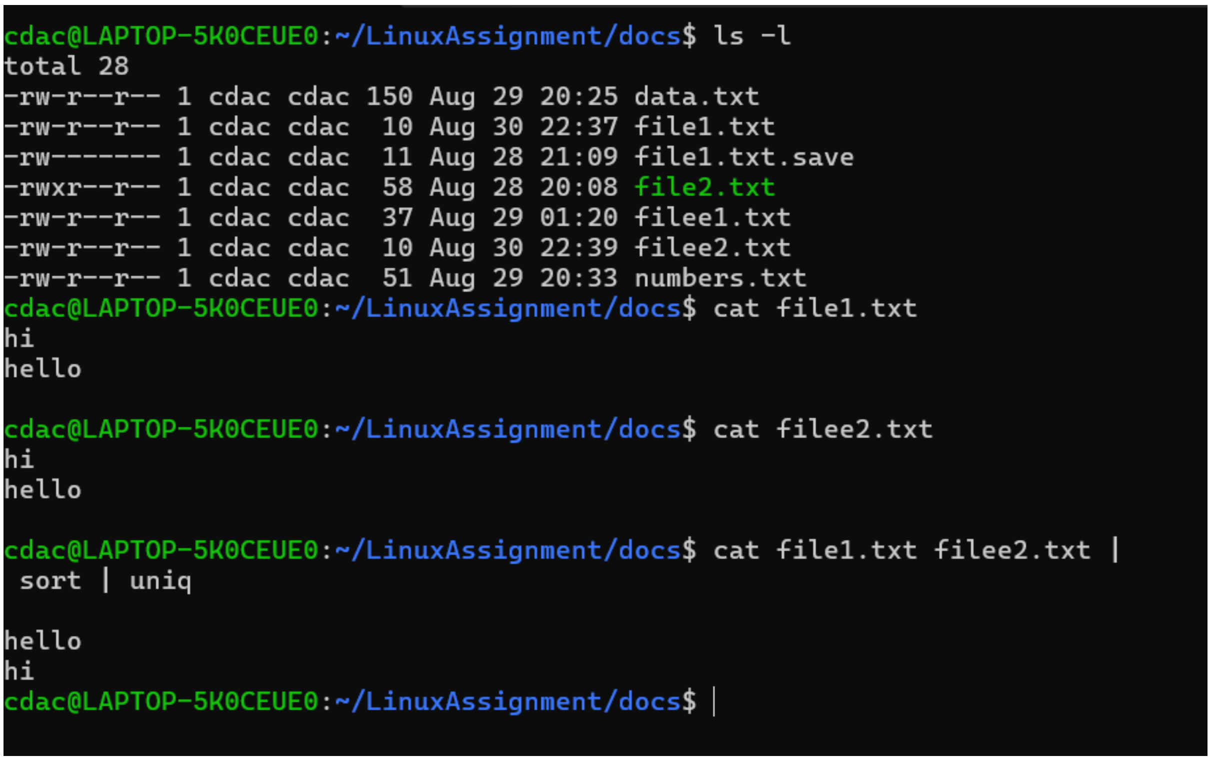
 ls -l | grep ".txt"



```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ ls -l | grep ".txt"
-rw-r--r-- 1 cdac cdac   23 Sep  1 02:18 ab.txt
-rw-r--r-- 1 cdac cdac   10 Aug 30 16:09 file1.txt
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ 
```

The command ls -l | grep ".txt" combines two commands using a pipe (|) to filter the output of ls -l with grep.

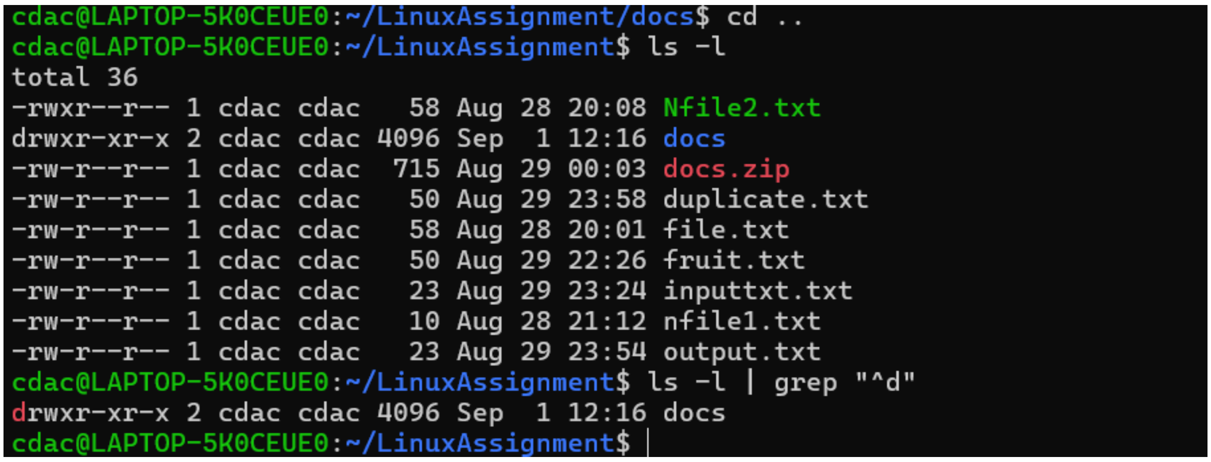
 cat file1.txt file2.txt | sort | uniq



```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ ls -l
total 28
-rw-r--r-- 1 cdac cdac 150 Aug 29 20:25 data.txt
-rw-r--r-- 1 cdac cdac  10 Aug 30 22:37 file1.txt
-rw------- 1 cdac cdac  11 Aug 28 21:09 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Aug 28 20:08 file2.txt
-rw-r--r-- 1 cdac cdac  37 Aug 29 01:20 filee1.txt
-rw-r--r-- 1 cdac cdac  10 Aug 30 22:39 filee2.txt
-rw-r--r-- 1 cdac cdac  51 Aug 29 20:33 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt
hi
hello

cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat filee2.txt
hi
hello

cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt filee2.txt |
 sort | uniq

hello
hi
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ 
```

this command combines the contents of file1.txt and file2.txt, sorts all the lines, and then removes any duplicate lines

 ls -l | grep "^d"



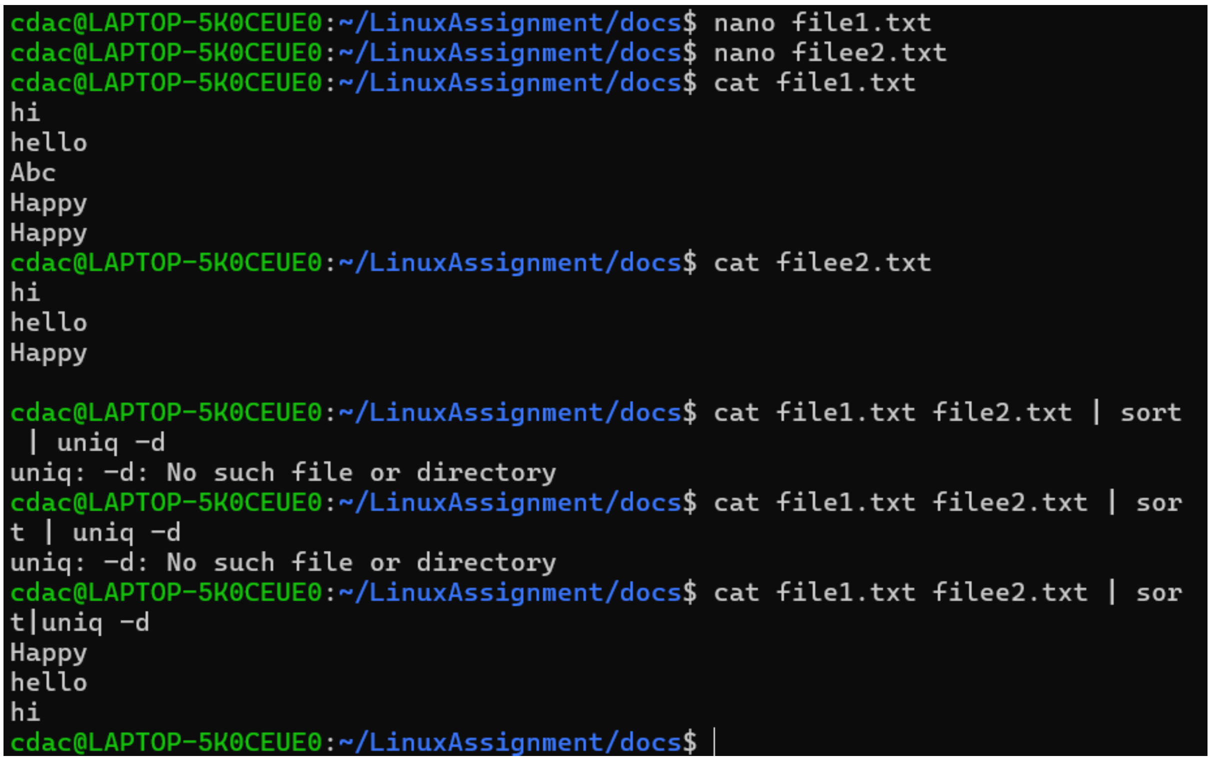
```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cd ..
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ ls -l
total 36
-rwxr--r-- 1 cdac cdac   58 Aug 28 20:08 Nfile2.txt
drwxr-xr-x 2 cdac cdac 4096 Sep  1 12:16 docs
-rw-r--r-- 1 cdac cdac  715 Aug 29 00:03 docs.zip
-rw-r--r-- 1 cdac cdac   50 Aug 29 23:58 duplicate.txt
-rw-r--r-- 1 cdac cdac   58 Aug 28 20:01 file.txt
-rw-r--r-- 1 cdac cdac   50 Aug 29 22:26 fruit.txt
-rw-r--r-- 1 cdac cdac   23 Aug 29 23:24 inputtxt.txt
-rw-r--r-- 1 cdac cdac   10 Aug 28 21:12 nfile1.txt
-rw-r--r-- 1 cdac cdac   23 Aug 29 23:54 output.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ ls -l | grep "^d"
drwxr-xr-x 2 cdac cdac 4096 Sep  1 12:16 docs
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ 
```

ls -l lists the files and directories in the current directory in long format.and grep "^d" filters the output of ls -l to include only lines that start with a d.

 grep -r "pattern" /path/to/directory/

the command grep -r "pattern" /path/to/directory/ searches for the specified pattern in all files within the given directory and its subdirectories, and prints out the lines from those files where the pattern is found.

 cat file1.txt file2.txt | sort | uniq –d



```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ nano file1.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ nano filee2.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt
hi
hello
Abc
Happy
Happy
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat filee2.txt
hi
hello
Happy

cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt file2.txt | sort
 | uniq -d
uniq: -d: No such file or directory
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt filee2.txt | sor
t | uniq -d
uniq: -d: No such file or directory
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ cat file1.txt filee2.txt | sor
t|uniq -d
Happy
hello
hi
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ 
```

the command cat file1.txt file2.txt | sort | uniq -d identifies and displays only the lines that appear more than once across both file1.txt and file2.txt, after sorting the combined contents.

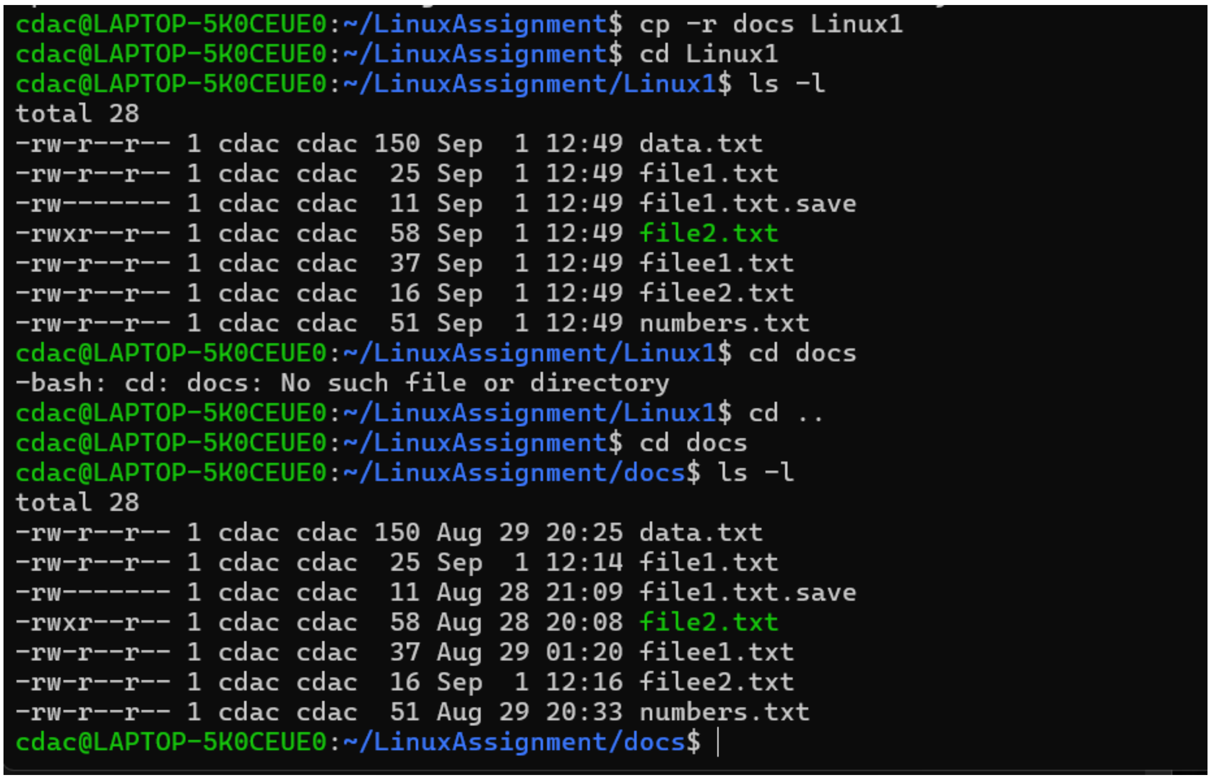
 chmod 644 file.txt

chmod 644 file.txt sets the permissions for file.txt such that:

The owner can read and write the file.

The group and others can only read the file.

 cp -r source_directory destination_directory



```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ cp -r docs Linux1
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ cd Linux1
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/Linux1$ ls -l
total 28
-rw-r--r-- 1 cdac cdac 150 Sep  1 12:49 data.txt
-rw-r--r-- 1 cdac cdac  25 Sep  1 12:49 file1.txt
-rw------- 1 cdac cdac  11 Sep  1 12:49 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Sep  1 12:49 file2.txt
-rw-r--r-- 1 cdac cdac  37 Sep  1 12:49 filee1.txt
-rw-r--r-- 1 cdac cdac  16 Sep  1 12:49 filee2.txt
-rw-r--r-- 1 cdac cdac  51 Sep  1 12:49 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/Linux1$ cd docs
-bash: cd: docs: No such file or directory
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/Linux1$ cd ..
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ cd docs
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$ ls -l
total 28
-rw-r--r-- 1 cdac cdac 150 Aug 29 20:25 data.txt
-rw-r--r-- 1 cdac cdac  25 Sep  1 12:14 file1.txt
-rw------- 1 cdac cdac  11 Aug 28 21:09 file1.txt.save
-rwxr--r-- 1 cdac cdac  58 Aug 28 20:08 file2.txt
-rw-r--r-- 1 cdac cdac  37 Aug 29 01:20 filee1.txt
-rw-r--r-- 1 cdac cdac  16 Sep  1 12:16 filee2.txt
-rw-r--r-- 1 cdac cdac  51 Aug 29 20:33 numbers.txt
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment/docs$
```

The command cp -r source_directory destination_directory copies the entire contents of source_directory to destination_directory.

 find /path/to/search -name "*.txt"

The command find /path/to/search -name "*.txt" searches for files with the .txt extension within a specified directory and its subdirectories

 chmod u+x file.txt

chmod: The command used to change file or directory permissions.

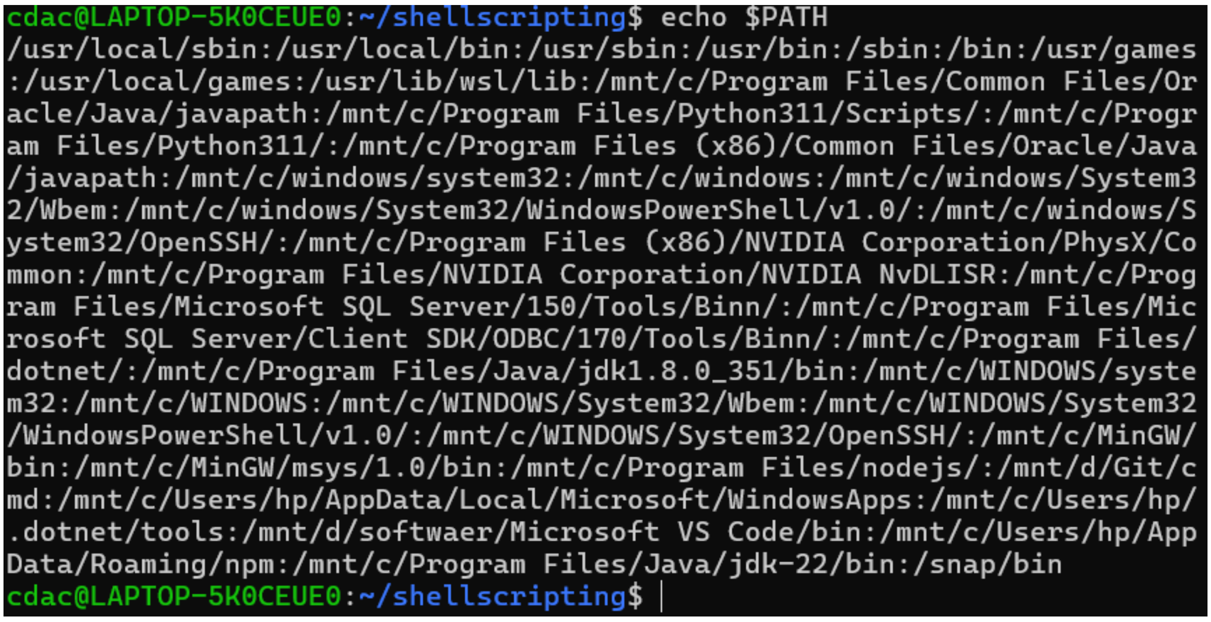
u+x: A permission modification option.

u: Stands for "user," referring to the owner of the file.

+: Indicates adding a permission.

x: Stands for "execute" permission.

 echo $PATH



```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
:/usr/local/games:/usr/lib/wsl/lib:/mnt/c/Program Files/Common Files/Or
acle/Java/javapath:/mnt/c/Program Files/Python311/Scripts/:/mnt/c/Progr
am Files/Python311/:/mnt/c/Program Files (x86)/Common Files/Oracle/Java
/javapath:/mnt/c/windows/system32:/mnt/c/windows:/mnt/c/windows/System3
2/Wbem:/mnt/c/windows/System32/WindowsPowerShell/v1.0/:/mnt/c/windows/S
ystem32/OpenSSH/:/mnt/c/Program Files (x86)/NVIDIA Corporation/PhysX/Co
mmon:/mnt/c/Program Files/NVIDIA Corporation/NVIDIA NvDLISR:/mnt/c/Prog
ram Files/Microsoft SQL Server/150/Tools/Binn/:/mnt/c/Program Files/Mic
rosoft SQL Server/Client SDK/ODBC/170/Tools/Binn/:/mnt/c/Program Files/
dotnet/:/mnt/c/Program Files/Java/jdk1.8.0_351/bin:/mnt/c/WINDOWS/syste
m32:/mnt/c/WINDOWS:/mnt/c/WINDOWS/System32/Wbem:/mnt/c/WINDOWS/System32
/WindowsPowerShell/v1.0/:/mnt/c/WINDOWS/System32/OpenSSH/:/mnt/c/MinGW/
bin:/mnt/c/MinGW/msys/1.0/bin:/mnt/c/Program Files/nodejs/:/mnt/d/Git/c
md:/mnt/c/Users/hp/AppData/Local/Microsoft/WindowsApps:/mnt/c/Users/hp/
.dotnet/tools:/mnt/d/softwaer/Microsoft VS Code/bin:/mnt/c/Users/hp/App
Data/Roaming/npm:/mnt/c/Program Files/Java/jdk-22/bin:/snap/bin
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ 
```

The command echo $PATH prints the current value of the PATH environment variable to the terminal.

echo: The command used to display text or variables to the terminal.

$PATH: The environment variable that stores a colon-separated list of directories where the shell looks for executable files

Part B

Identify True or False:

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

Ans: True "ls" is used to list files and directories in a directory

2. mv is used to move files and directories.

Ans : True

3. cd is used to copy files and directories.

Ans:True

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

Ans:True

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

Ans: true grep is used to search for patterns in files

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

Ans: True

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

Ans:true

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

Ans: True

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.

Ans : False chmod is used to change file permission

2. cpy is used to copy files and directories.

Ans : False. The correct command is `cp`.

3. mkfile is used to create a new file.

Ans: False touch,nano these commands are used to create new file

4. catx is used to concatenate files.

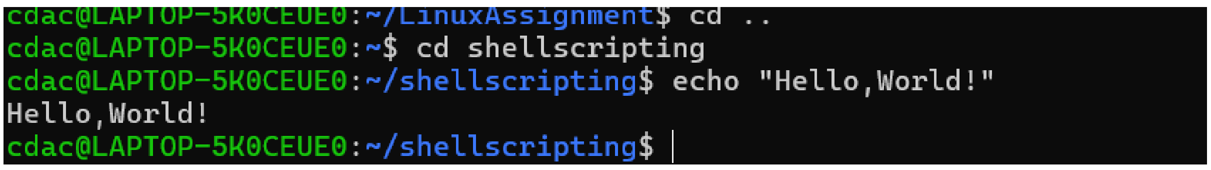
Ans: False catx is not used to concatenate files

5. rn is used to rename files.

Ans: False rn is not used to rename file mv is used to rename the file

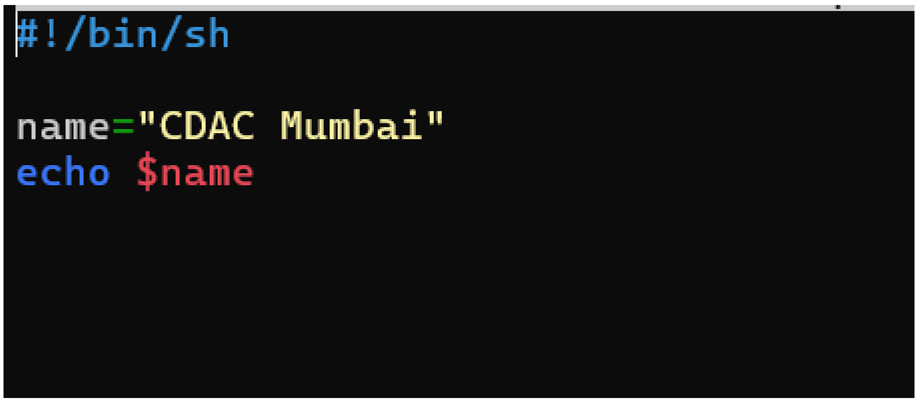
Part C

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

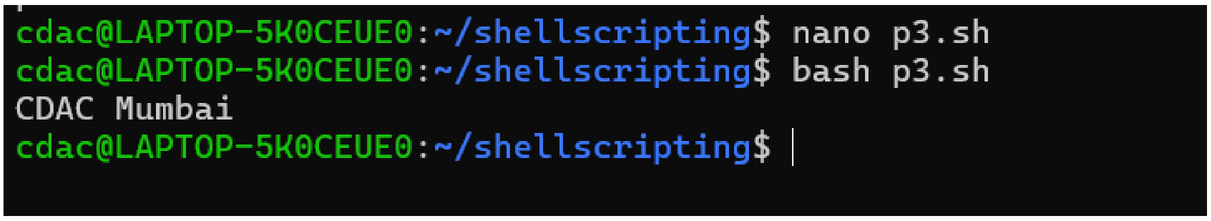


```
cdac@LAPTOP-5K0CEUE0:~/LinuxAssignment$ cd ..
cdac@LAPTOP-5K0CEUE0:~$ cd shellscripting
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ echo "Hello,World!"
Hello,World!
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ 
```

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

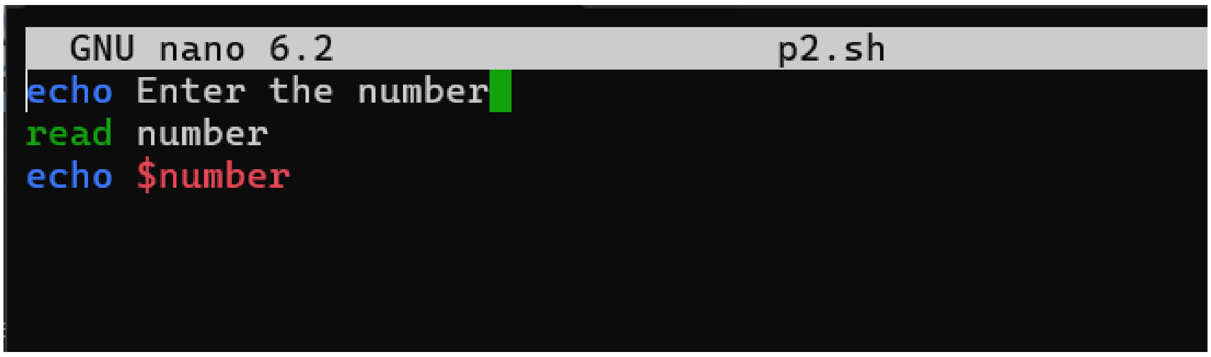


```
#!/bin/sh

name="CDAC Mumbai"
echo $name
```

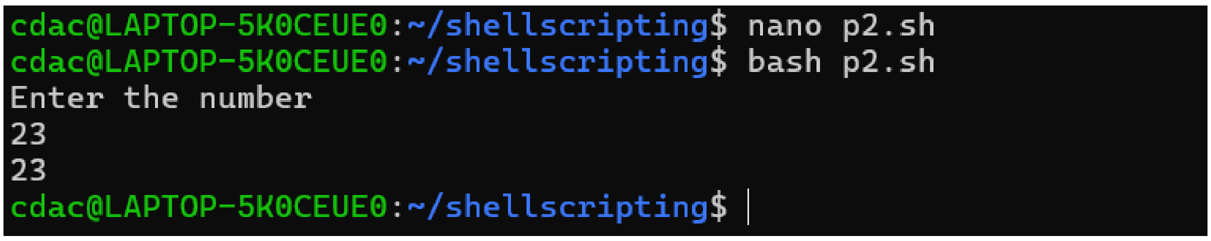


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p3.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p3.sh
CDAC Mumbai
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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

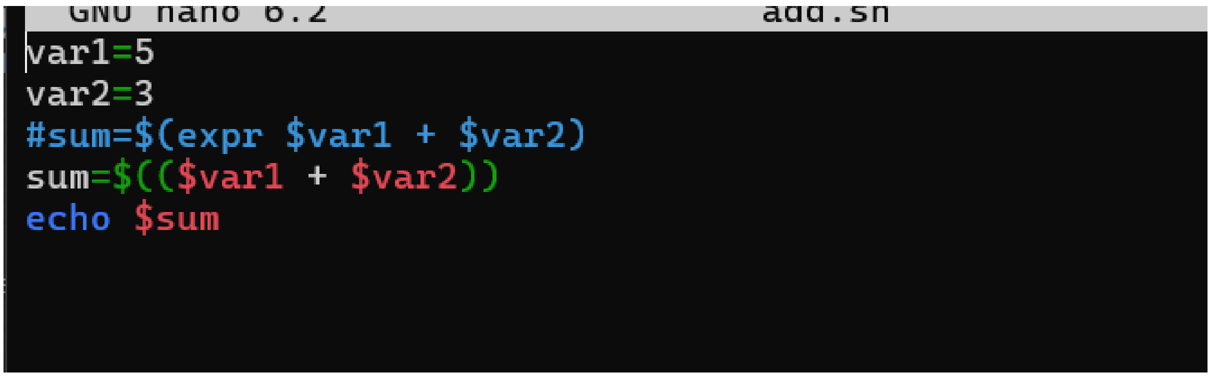


```
  GNU nano 6.2                        p2.sh
echo Enter the number
read number
echo $number
```

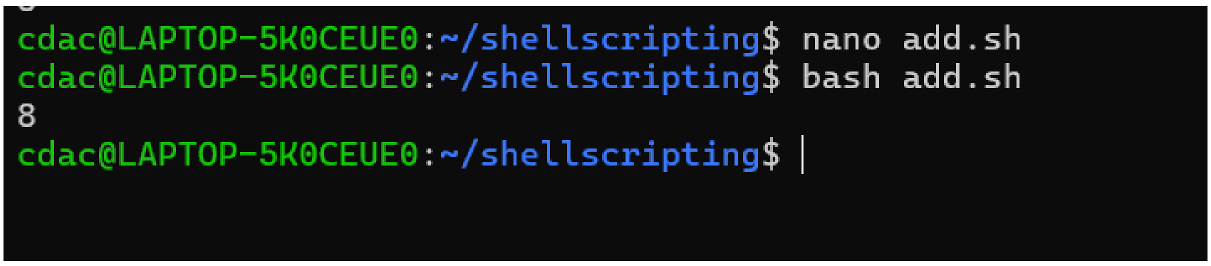


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p2.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p2.sh
Enter the number
23
23
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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

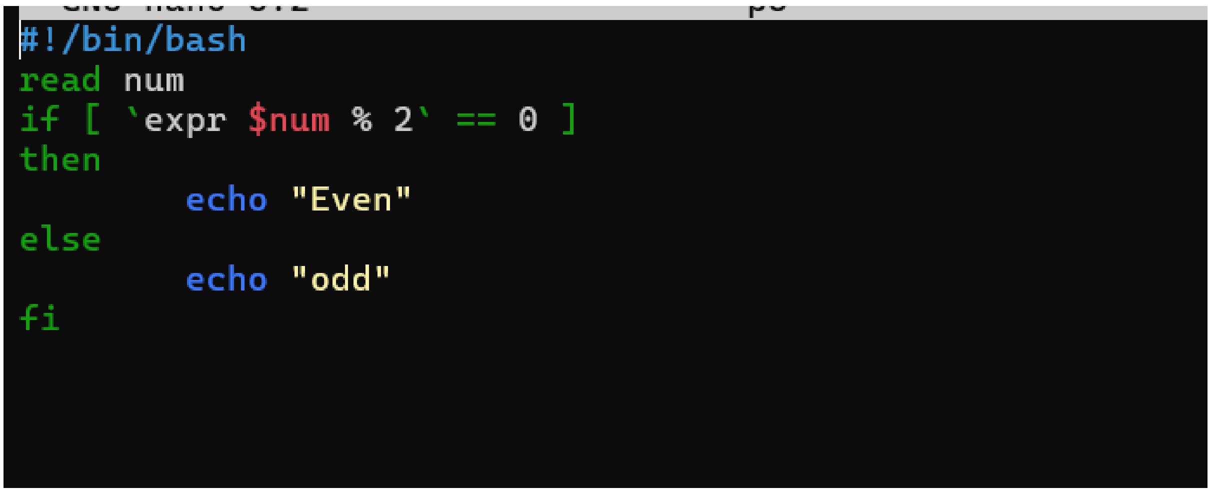


```
  GNU nano 6.2                           add.sh
var1=5
var2=3
#sum=$(expr $var1 + $var2)
sum=$(($var1 + $var2))
echo $sum
```

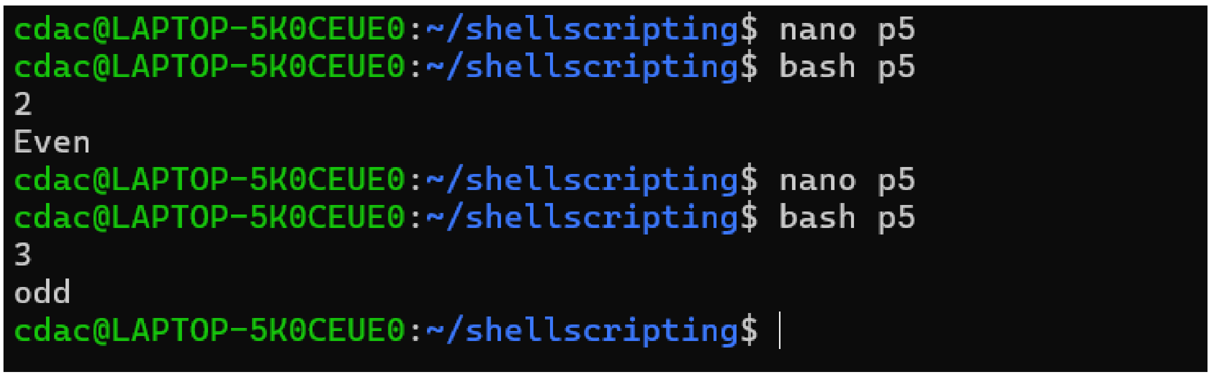


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano add.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash add.sh
8
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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

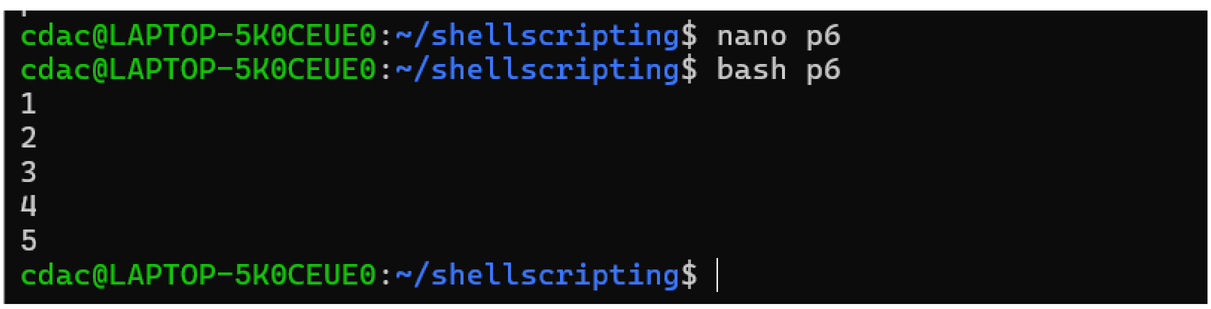


```
#!/bin/bash
read num
if [ `expr $num % 2` == 0 ]
then
        echo "Even"
else
        echo "odd"
fi
```



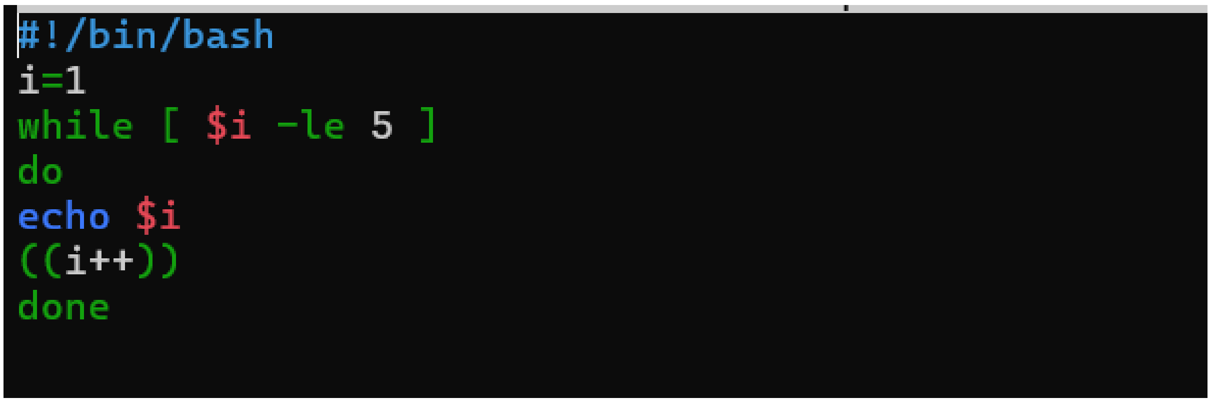
```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p5
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p5
2
Even
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p5
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p5
3
odd
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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

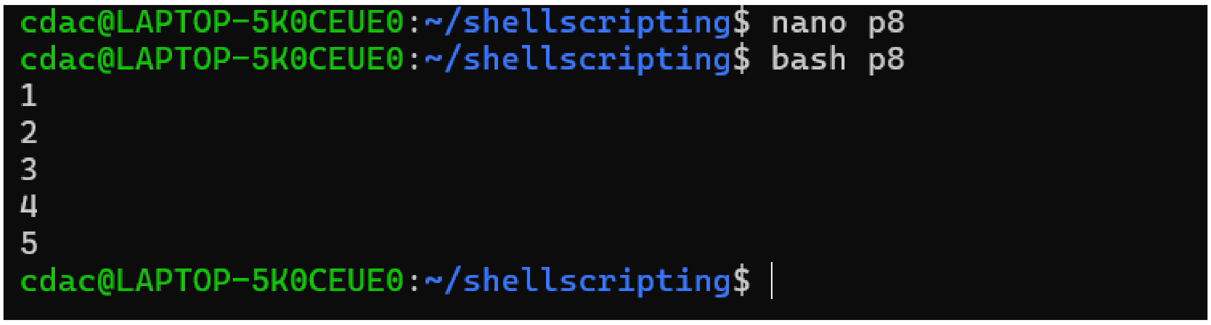


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p6
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p6
1
2
3
4
5
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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



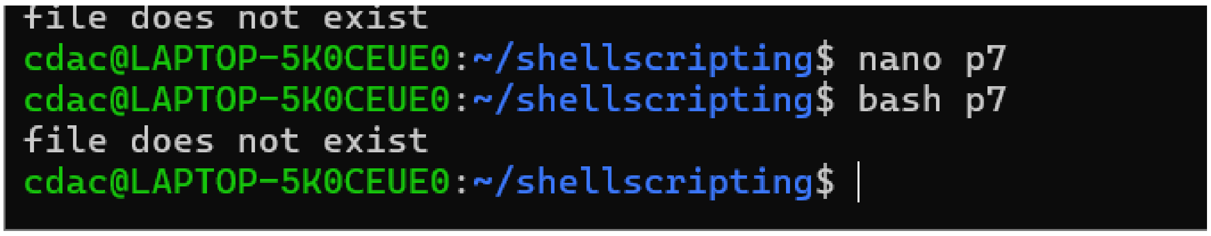
```
#!/bin/bash
i=1
while [ $i -le 5 ]
do
echo $i
((i++))
done
```



```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p8
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p8
1
2
3
4
5
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

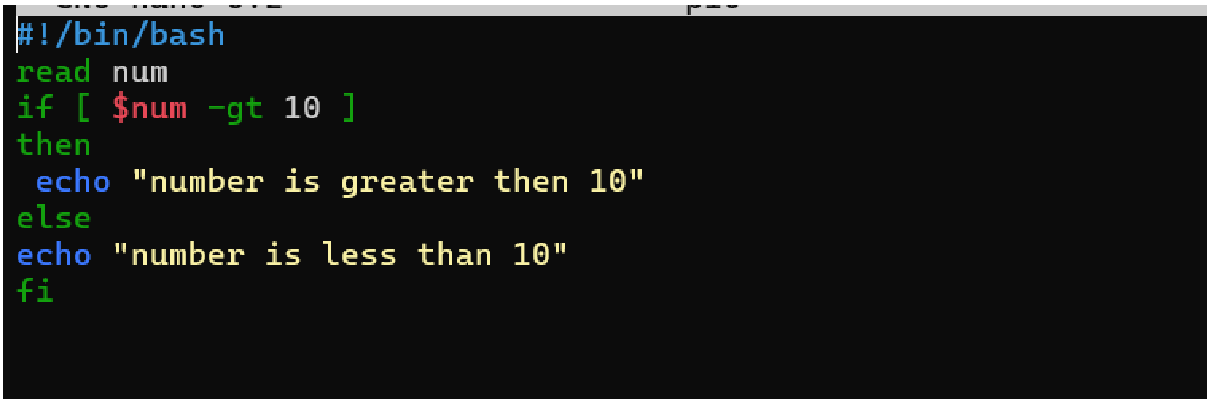
Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it

does, print "File exists", otherwise, print "File does not exist".

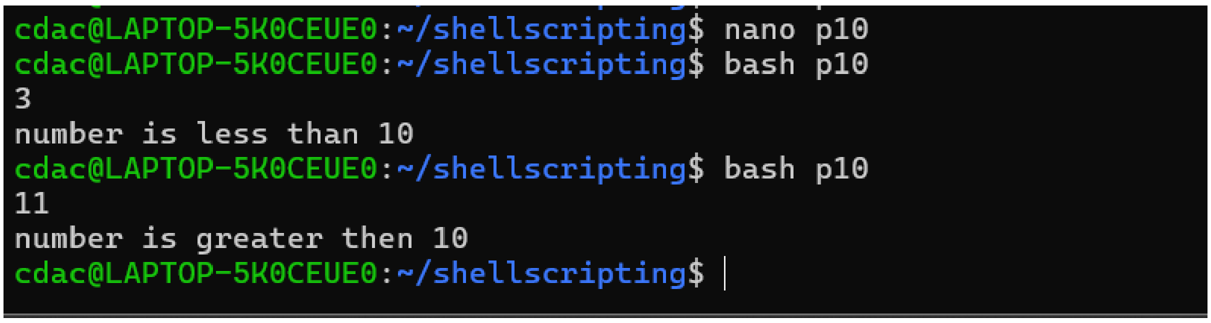


```
file does not exist
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p7
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p7
file does not exist
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

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.



```
#!/bin/bash
read num
if [ $num -gt 10 ]
then
 echo "number is greater then 10"
else
echo "number is less than 10"
fi
```

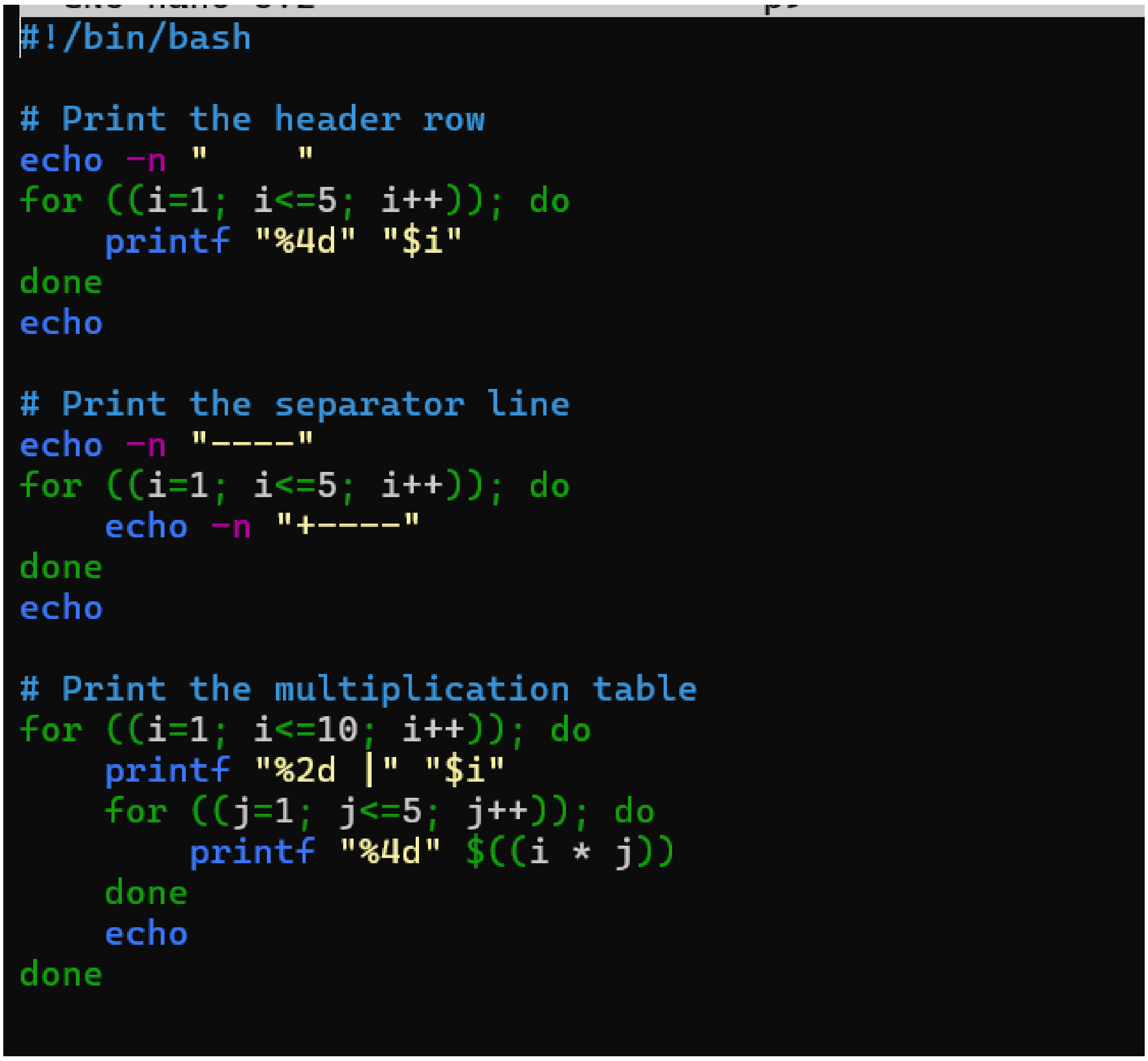


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p10
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p10
3
number is less than 10
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p10
11
number is greater then 10
cdac@LAPTOP-5K0CEUE0:~/shellscripting$
```

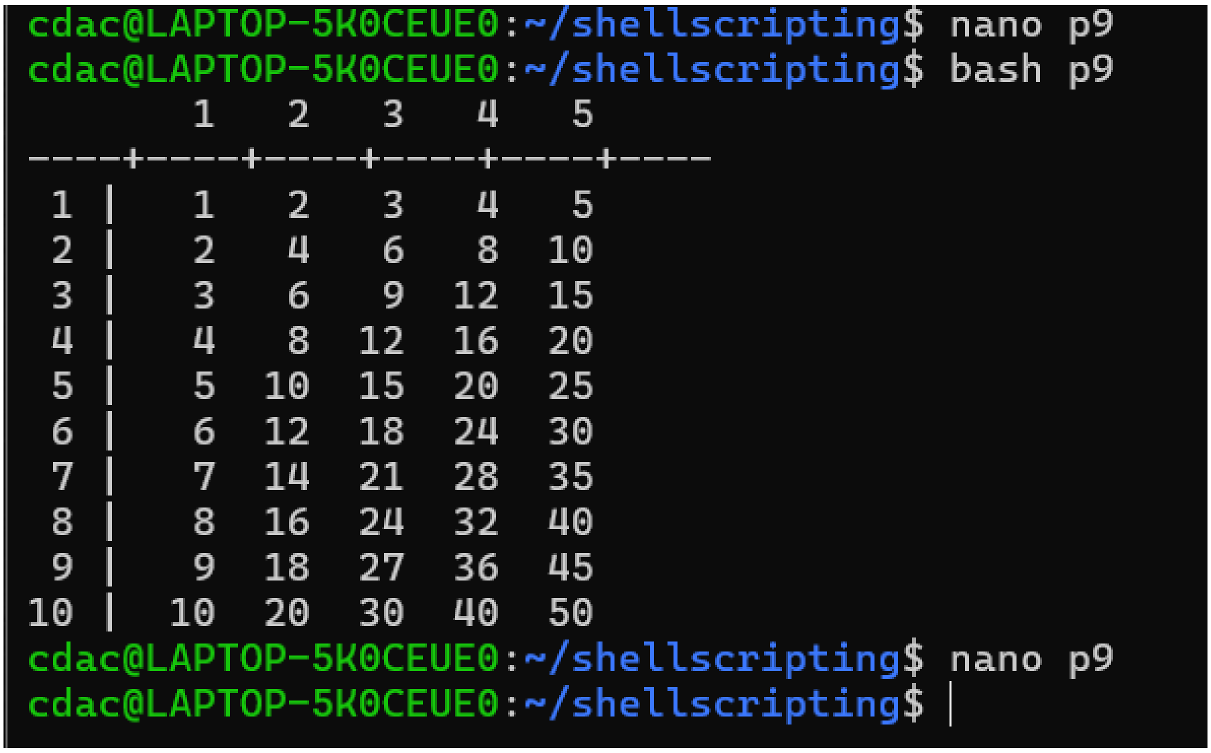
Question 10: Write a shell script that uses nested for loops to print a multiplication table for numbers

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.



```bash
#!/bin/bash

# Print the header row
echo -n "    "
for ((i=1; i<=5; i++)); do
    printf "%4d" "$i"
done
echo

# Print the separator line
echo -n "----"
for ((i=1; i<=5; i++)); do
    echo -n "+----"
done
echo

# Print the multiplication table
for ((i=1; i<=10; i++)); do
    printf "%2d |" "$i"
    for ((j=1; j<=5; j++)); do
        printf "%4d" $((i * j))
    done
    echo
done
```

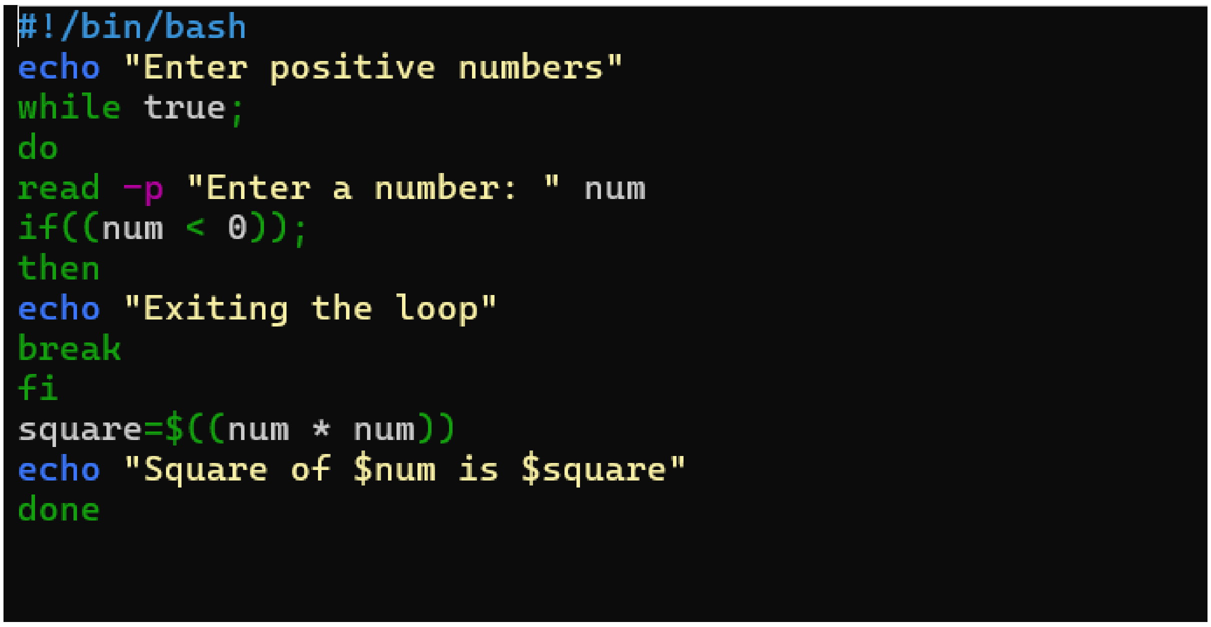


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p9
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p9
         1    2    3    4    5
----+----+----+----+----+----
  1 |    1    2    3    4    5
  2 |    2    4    6    8   10
  3 |    3    6    9   12   15
  4 |    4    8   12   16   20
  5 |    5   10   15   20   25
  6 |    6   12   18   24   30
  7 |    7   14   21   28   35
  8 |    8   16   24   32   40
  9 |    9   18   27   36   45
 10 |   10   20   30   40   50
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p9
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ 
```

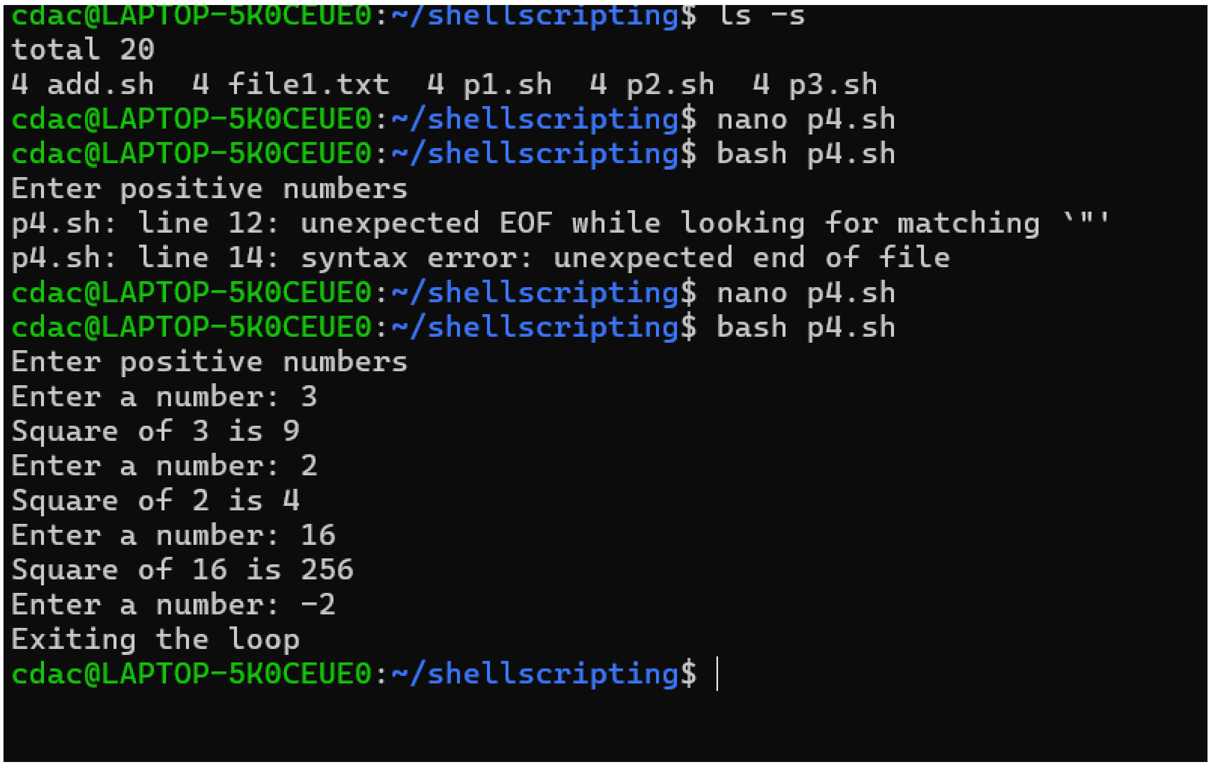
Question 11: Write a shell script that uses a while loop to read numbers from the user until the user enters

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.



```
#!/bin/bash
echo "Enter positive numbers"
while true;
do
read -p "Enter a number: " num
if((num < 0));
then
echo "Exiting the loop"
break
fi
square=$((num * num))
echo "Square of $num is $square"
done
```

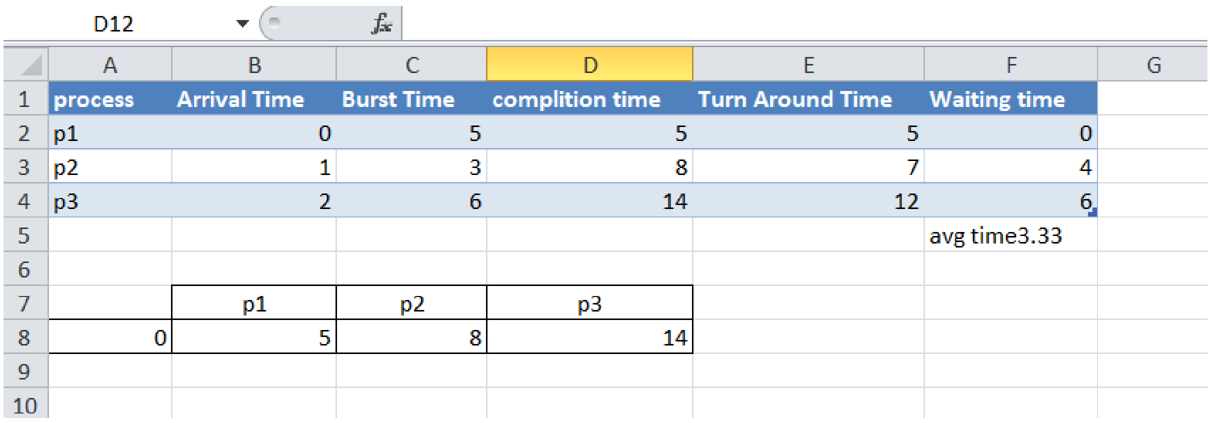


```
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ ls -s
total 20
4 add.sh  4 file1.txt  4 p1.sh  4 p2.sh  4 p3.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p4.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p4.sh
Enter positive numbers
p4.sh: line 12: unexpected EOF while looking for matching `"'
p4.sh: line 14: syntax error: unexpected end of file
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ nano p4.sh
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ bash p4.sh
Enter positive numbers
Enter a number: 3
Square of 3 is 9
Enter a number: 2
Square of 2 is 4
Enter a number: 16
Square of 16 is 256
Enter a number: -2
Exiting the loop
cdac@LAPTOP-5K0CEUE0:~/shellscripting$ |
```

1. Consider the following processes with arrival times and burst times:

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

Calculate the average waiting time using First-Come, First-Served (FCFS) scheduling.



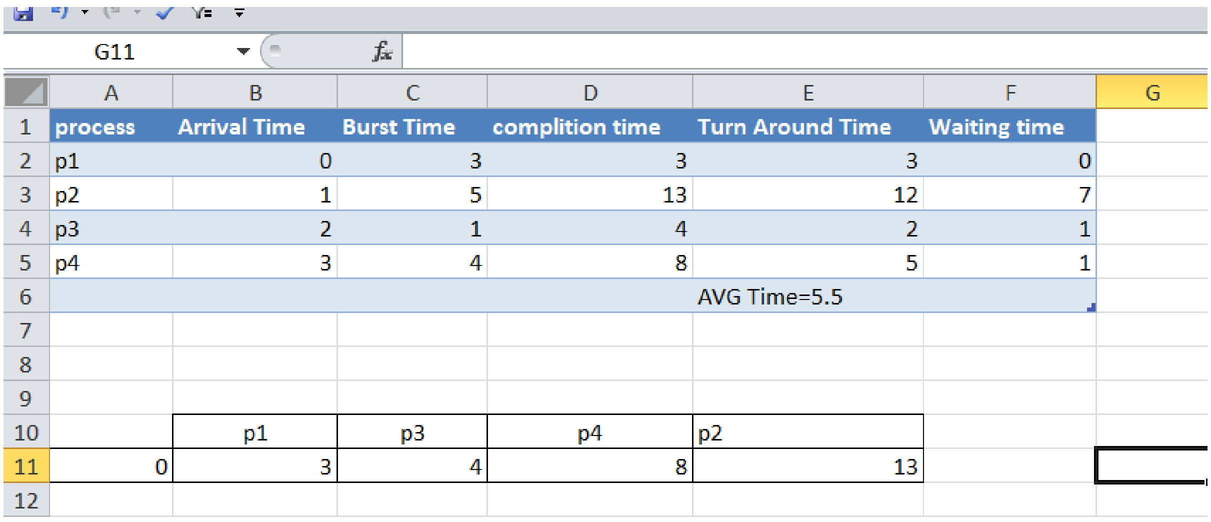
D12

	A	B	C	D	E	F	G
1	process	Arrival Time	Burst Time	complition time	Turn Around Time	Waiting time	
2	p1	0	5	5	5	0	
3	p2	1	3	8	7	4	
4	p3	2	6	14	12	6	
5						avg time3.33	
6							
7		p1	p2	p3			
8	0	5	8	14			
9							
10							

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 the average turnaround time using Shortest Job First (SJF) scheduling.



G11

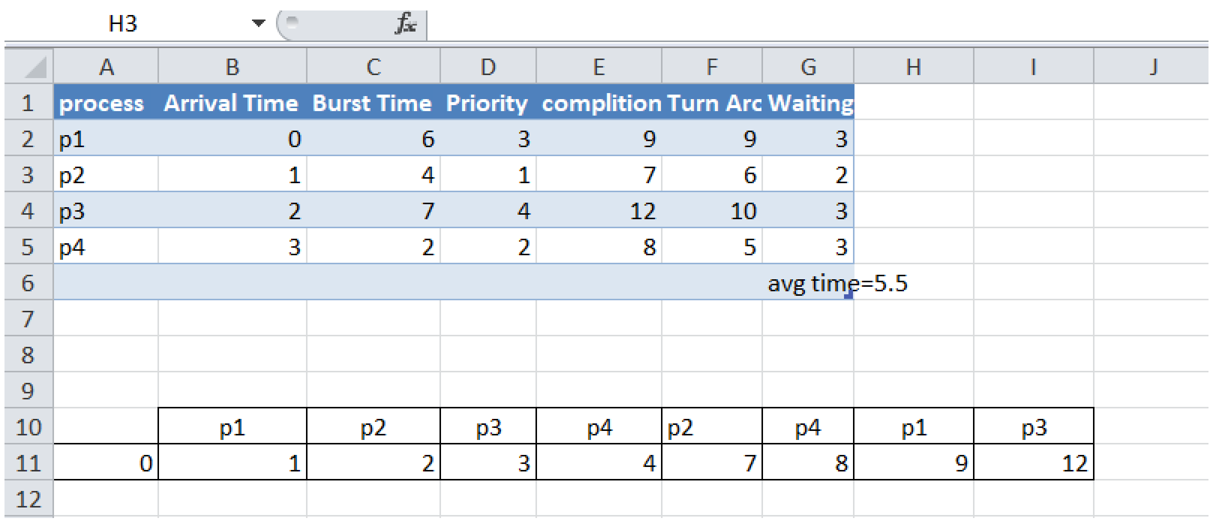
process	Arrival Time	Burst Time	complition time	Turn Around Time	Waiting time
p1	0	3	3	3	0
p2	1	5	13	12	7
p3	2	1	4	2	1
p4	3	4	8	5	1
				AVG Time=5.5	

	p1	p3	p4	p2
0	3	4	8	13

3. Consider the following processes with arrival times, burst times, and priorities (lower number indicates higher priority):

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

Calculate the average waiting time using Priority Scheduling.


H3

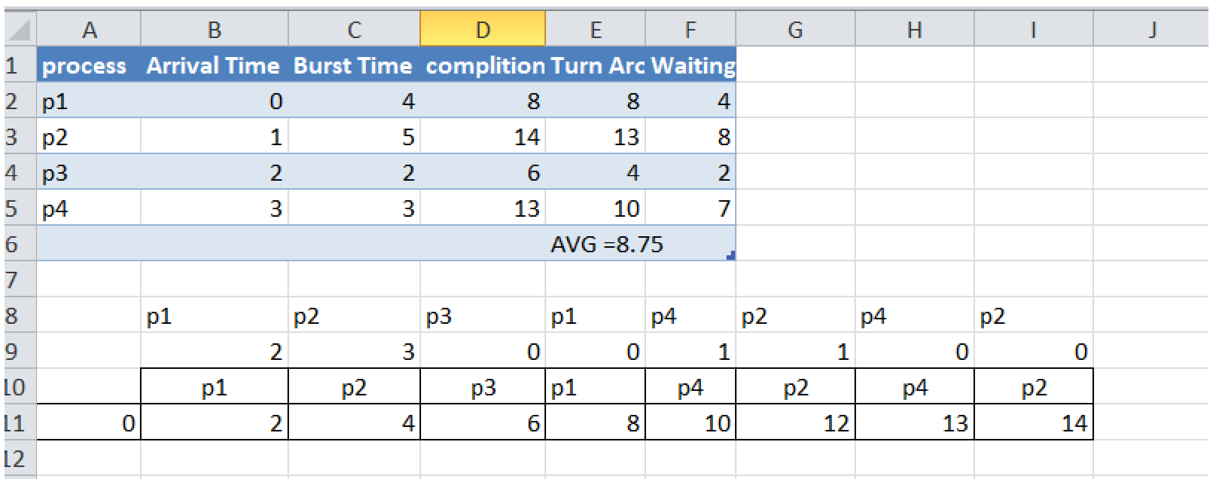
	A	B	C	D	E	F	G	H	I	J
1	process	Arrival Time	Burst Time	Priority	complition	Turn Arc	Waiting			
2	p1	0	6	3	9	9	3			
3	p2	1	4	1	7	6	2			
4	p3	2	7	4	12	10	3			
5	p4	3	2	2	8	5	3			
6							avg time=5.5			
7										
8										
9										
10		p1	p2	p3	p4	p2	p4	p1	p3	
11	0	1	2	3	4	7	8	9	12	
12										

4. Consider the following processes with arrival times and burst times, and the time quantum for

Round Robin scheduling is 2 units:

Process	Arrival Time	Burst Time
P1	0	4
P2	1	5
P3	2	2
P4	3	3

Calculate the average turnaround time using Round Robin scheduling.



	A	B	C	D	E	F	G	H	I	J
1	process	Arrival Time	Burst Time	complition	Turn Arc	Waiting				
2	p1	0	4	8	8	4				
3	p2	1	5	14	13	8				
4	p3	2	2	6	4	2				
5	p4	3	3	13	10	7				
6					AVG =8.75					
7										
8		p1	p2	p3	p1	p4	p2	p4	p2	
9		2	3	0	0	1	1	0	0	
10		p1	p2	p3	p1	p4	p2	p4	p2	
11	0	2	4	6	8	10	12	13	14	
12										