Operating System

Assignment 2

Part A

What will the following commands do?

- echo "Hello, World!"

Ans: This command prints Hello, World! to the terminal.

- name="Productive"

Ans: Sets a variable named name to Productive.

- touch file.txt

Ans: Creates an empty file called file.txt, or updates its timestamp if it already exists.

- ls -a

Ans: Lists all files and folders in the current directory, including hidden ones.

- rm file.txt

Ans: Deletes the file named file.txt.

- cp file1.txt file2.txt

Ans: Copies the content of file1.txt to file2.txt.

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

Ans: Moves file.txt to the specified directory.

- chmod 755 script.sh

Ans: Sets permissions so that the owner can read, write, and execute script.sh, while others can only read and execute.

- grep "pattern" file.txt

Ans: Searches for and displays lines containing pattern in file.txt.

- kill PID

Ans: Terminates the process with the ID PID.

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

Ans: Creates a directory mydir.

Moves into mydir.

Creates a file file.txt.

Writes Hello, World! into file.txt.

Shows the content of file.txt.

- ls -l | grep ".txt"

Ans: Lists detailed information about files and shows only those with .txt in their names.

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

Ans: Combines file1.txt and file2.txt.

Sorts the combined content.

Shows only lines that appear more than once.

- ls -l | grep "^d"

Ans: Lists detailed information and shows only directories.

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

Ans: Searches for pattern in all files under /path/to/directory/.

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

Ans: Combines file1.txt and file2.txt.

Sorts the combined content.

Shows only lines that appear more than once.

- chmod 644 file.txt

Ans: Sets permissions so the owner can read and write file.txt, while others can only read it.

- cp -r source_directory destination_directory

Ans: Copies the entire source_directory and its contents to destination_directory.

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

Ans: Finds and lists all .txt files under /path/to/search.

- chmod u+x file.txt

Ans: Adds execute permission for the owner of file.txt.

- echo $PATH

Ans: Displays the list of directories where the system looks for executable files.

Part B

Identify True or False:

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

True. The ls command lists files and directories in the current directory or a specified directory.

2. mv is used to move files and directories.

True. The mv command moves files and directories from one location to another or renames them.

3. cd is used to copy files and directories.

False. The cd command is used to change the current directory. It does not copy files or directories.

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

True. The pwd command prints the path of the current working directory.

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

True. The grep command searches for specified patterns within files and displays matching lines.

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

True. The chmod 755 command sets permissions so that the owner has read, write, and execute permissions, while the group and others have read and execute permissions.

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

True. The mkdir -p command creates nested directories, and it will create parent directories as needed.

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

True. The rm -rf command removes files or directories forcefully and recursively without prompting for confirmation.

Part C

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.

Incorrect. The correct command is chmod, not chmodx.

2. cpy is used to copy files and directories.

Incorrect. The correct command is cp, not cpy.

3. mkfile is used to create a new file.

Incorrect. There is no standard mkfile command in Unix/Linux. The correct way to create a new file is using touch or redirecting output, e.g., > filename.

4. catx is used to concatenate files.

Incorrect. The correct command is cat, not catx.

5. rn is used to rename files.

Incorrect. There is no rn command. The correct command for renaming files is mv.

Part C

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

#!/bin/bash

echo "Hello,World!"

```
cdac@DESKTOP-NNPDEGO: ~
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p2.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p2.txt
Hello,World!
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$
```

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

#!/bin/bash

name="CDAC Mumbai"

echo $name

```
cdac@DESKTOP-NNPDEGO: ~
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p3.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p3.txt
CDAC Mumbai
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p3.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p3.txt
CDAC Mumbai
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$
```

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

#!/bin/bash

echo "Enter a number"

read number

echo "The Number is:" $number

```
cdac@DESKTOP-NNPDEGO: ~
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p4.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p4.txt
Enter a number
4
The Number is: 4
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$
```

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

```
#!bin/bash

echo Enter a number

read Num1

echo Enter a number

read Num2

res=$(expr $Num1 + $Num2)

echo "Result: $res"
```

```
cdac@DESKTOP-NNPDEGO: ~
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p1.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p1.txt
Enter a number
3
Enter a number
4
Result: 7
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$
```

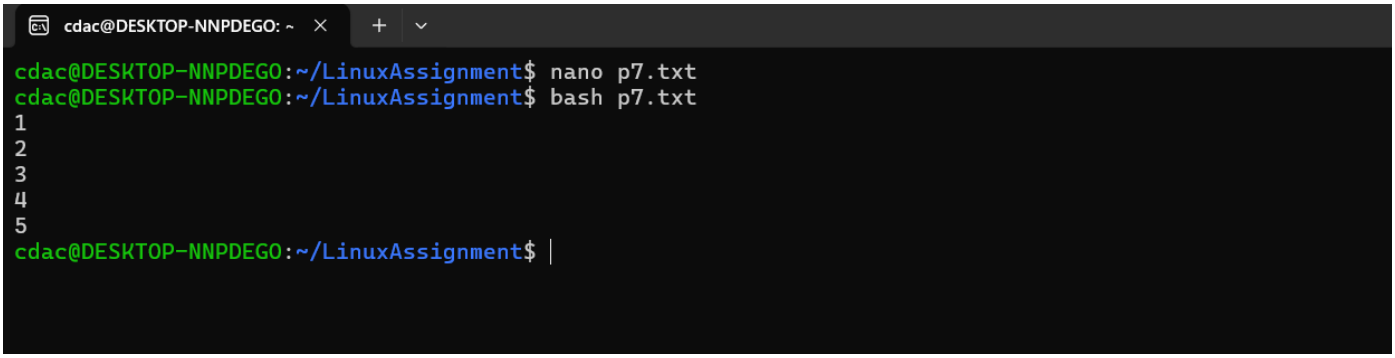
Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".

```
#!/bin/bash

echo "Enter a number:"

read number

if [ $((number % 2)) -eq 0 ]

then

   echo "Even"

else

   echo "Odd"

fi
```

```
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p6.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p6.txt
Enter a number:
3
Odd
```

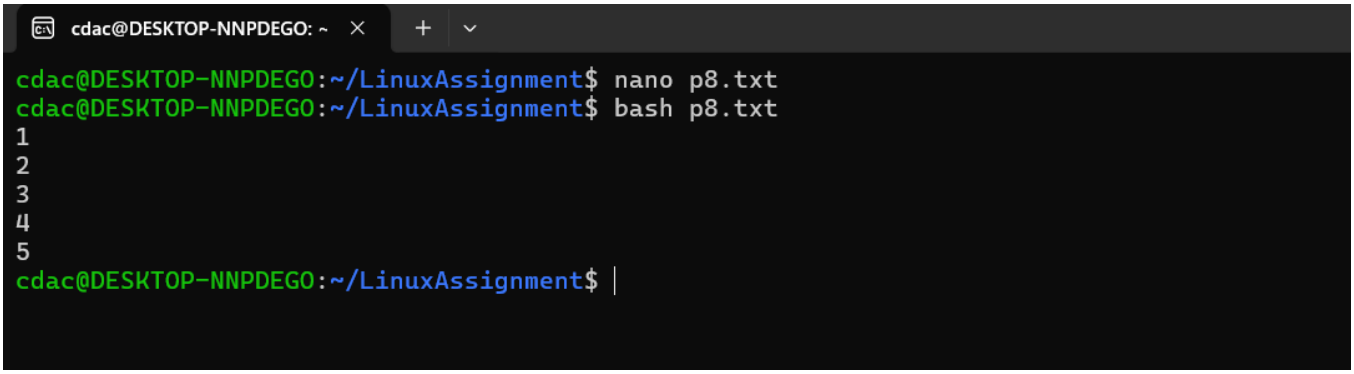
Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.

```
#!/bin/bash
for i in {1,2,3,4,5}
do
echo "$i"
done
```



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 $i
   i=$((i + 1))
done
```



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".

```
#!/bin/bash
if [ -f "file.txt" ];
then
    echo "File exists"
else
    echo "File does not exist"
fi
```

```
cdac@DESKTOP-NNPDEGO: ~
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ ls
data.txt  duplicate.txt  file1.txt  input.txt    output.txt  p2.txt  p4.txt  p6.txt  p8.txt
docs      file.txt       fruit.txt  numbers.txt  p1.txt      p3.txt  p5.txt  p7.txt  p9.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p9.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p9.txt
File Exist
cdac@DESKTOP-NNPDEGO:~/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.

```
#!/bin/bash
echo "Enter a Number:"
read Num
if [ $Num -gt 10 ]; then
    echo "The number is greater than 10."
else
    echo "The number is not greater than 10."
fi
```

```
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p10.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p10.txt
Enter a Number:
6
The number is not greater.
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p10.txt
Enter a Number:
19
The number is greater.
cdac@DESKTOP-NNPDEGO:~/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.

```
#!/bin/bash

echo "    1  2  3  4  5  6  7  8  9  10"

echo "   ----------------------------------------"

  for i in {1..10}; do

    echo -n "$i |"

  for j in {1..10}; do

    prod=$((i * j))

  printf "%3d" $prod

  done

  echo

done
```

```
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p11.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p11.txt
   1  2  3  4  5  6  7  8  9  10
  ----------------------------------------
1 |  1  2  3  4  5  6  7  8  9 10
2 |  2  4  6  8 10 12 14 16 18 20
3 |  3  6  9 12 15 18 21 24 27 30
4 |  4  8 12 16 20 24 28 32 36 40
5 |  5 10 15 20 25 30 35 40 45 50
6 |  6 12 18 24 30 36 42 48 54 60
7 |  7 14 21 28 35 42 49 56 63 70
8 |  8 16 24 32 40 48 56 64 72 80
9 |  9 18 27 36 45 54 63 72 81 90
10 | 10 20 30 40 50 60 70 80 90100
cdac@DESKTOP-NNPDEGO:~/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.

```
 #!/bin/bash
while true; do
    echo "Enter a number:"
    read number

    if [ $number -lt 0 ]; then
        break
    fi

    square=$(( number * number ))
    echo "Square of $number is: $square"
done

echo "Negative number entered."
```

```
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ nano p12.txt
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$ bash p12.txt
Enter a number:
7
Square of 7 is: 49
Enter a number:
-1
Negative number entered.
cdac@DESKTOP-NNPDEGO:~/LinuxAssignment$
```

Part E

Part E

Q.1

Process	AT	WT (TAT-BT)	BT	CT	TAT (CT-AT)
P1	0	0	5	5	5
P2	1	6	3	8	7
P3	2	4	6	14	12

Gantt chart

P1	P2	P3

0 5 8 14

Avg. TurnAround time $= \frac{5+7+12}{3}$

$= 8$

Avg. Waiting time $= \frac{0+6+4}{3}$

$= 3.33$

Q.2

Process	Arrival Time	Burst Time	Completion Time	WT	TAT
P1	0	3	3	0	3
P2	1	5	13	7	12
P3	2	1	4	1	2
P4	3	4	8	1	5

Algo:- shortest job First.

Gantt Chart:-

~~P1~~	~~P2~~	~~P4~~	~~P1~~	~~P3~~	P1	P3	P4	P2

0 1 5 7 12 19 0 3 4 8 13

Avg. waiting time $= \frac{0+7+1+1}{4} = \frac{9}{4} =$

Avg. Turnaround time $= \frac{3+12+2+5}{4} = \frac{37}{4} = 9.25$

Q. 3

Priority	Process	AT	BT	CT	WT	TAT
3		0	6	12	6	12
1		1	4	5	0	4
4		2	7	19	10	17
2		3	2	7	2	4

Priority Schedulity

gantt chart

P1	P2	P4	P1	P3

0 1 5 7 12 19

$$\text{Avg. wt} = \frac{6+0+10+2}{4} = \frac{18}{4} = 4.5$$

$$\text{Avg. TAT} = \frac{12+4+17+4}{4} = \frac{37}{4} = 9.25$$

Q.4

Process	AT	BT	CT ~~Priority~~	WT	TAT
P1	0	4	10	6	10
P2	1	5	14	8	13
P3	2	2	6	2	4
P4	3	3	13	7	10

Round-Robin :- 2 units

Gantt chart

P1	P2	P3	P4	P1	P2	P4	P2

0 2 4 6 8 10 12 13 14

$$\text{Avg. WT} = \frac{6+8+2+7}{4} = \frac{23}{4} = 5.75$$

$$\text{Avg. TAT} = \frac{10+13+4+10}{4} = \frac{37}{4} = 9.25$$

Q.5