Assignment 2

Part A

What will the following commands do?

1)echo "Hello, World!"

->Prints "Hello, World!" to the terminal.

2)name="Productive"

-> String assign to ‘name ’ variable

3) touch file.txt

-> create file.txt file

4) ls -a

->Lists all files and directories, including hidden ones

5) rm file.txt

->delete file.txt file

5) cp file1.txt file2.txt

->copy file1.txt file in file2.txt

6) mv file.txt /path/to/directory/

->Moves file.txt to the specified directory

7) chmod 755 script.sh

->Gives the owner full (read, write, execute) permissions, while others get read and execute permissions on script.sh

8) grep "pattern" file.txt

->Searches for the word "pattern" in file.txt and prints matching lines.

9) kill PID

->Terminates the process with the specified PID (Process ID).

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

->Creates a directory mydir, enters it, creates file.txt, writes "Hello, World!" into it, and then displays its content.

11) ls -l | grep ".txt"

->Lists files in long format and filters only those containing .txt in their names

12) cat file1.txt file2.txt | sort | uniq

->Merges file1.txt and file2.txt, sorts the lines, and removes duplicates

13) ls -l | grep "^d"

->Lists only directories in long format

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

->Recursively searches for "pattern" in all files within the specified directory

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

->Merges file1.txt and file2.txt, sorts them, and displays only duplicate lines

16) chmod 644 file.txt

->Sets file permissions so the owner can read and write, while others can only read file.txt

17) cp -r source_directory destination_directory

->Recursively copies source_directory and its contents to destination_directory

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

->Searches for all .txt files in the specified path

19) chmod u+x file.txt

->Gives the user (owner) execute permission on file.txt

20) echo $PATH

->Displays the system's PATH variable

Part B

Identify True or False:

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

ANS - TRUE

2. mv is used to move files and directories.

ANS - TRUE

3. cd is used to copy files and directories.

ANS - FALSE

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

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:

ANS :

1. chmodx is used to change file permissions.

ANS – INCORRECT

CORRECT COMAND - chmod

2. cpy is used to copy files and directories.

ANS – its incorrect

Correct command - cp

3. mkfile is used to create a new file.

ANS – incorrect

Correct command – touch / echo

4. catx is used to concatenate files.

ANS – incorrect

Correct command - cat

5. rn is used to rename files.

ANS – incorrect

Correct command - mv

Part C

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

Ans:

```
echo "Hello, World!"
```

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

Ans:

```
name=”CDAC Mumbai”
echo "The value of the variable 'name' is: $name"
```

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

Ans:

```
echo “Enter a number:” :
read number :
echo “you entered: $number”:
read num2
```

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

Ans:

```
echo “Enter a number1:” :
read num1:
echo “Enter a number2:”:
read num2
sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is: $sum"
```

```
“Enter a number1:”
2
“Enter a number2:”
3
The sum of 2 and 3 is: 5
```

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

Ans:

```
echo “Enter a number1:”
read n
if [ $(( n % 2 )) -eq 0 ]
then
 echo $n is even
else
 echo $n is odd
fi
```

```
“Enter a number1:”
4
4 is even
```

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

Ans:

```
for (( i=1; i<=5; i++))
do
 echo $i
done
```

```
1
2
3
4
5
```

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

Ans:

```
i=1
while [ $i -le 5 ]
do
  echo $i
  i=$(( i + 1 ))
done
```

```
1
2
3
4
5
```

8) 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".

Ans:

```
filename="file.txt"

if [ -f "$filename" ]
then
    echo "File exists"
else
    echo "File does not exist"
fi
```

```
File exists
```

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

Ans:

```
cdac@Vivobook15:~/LinuxAssignment$ cat assign

echo "Enter a number:"
read num

if [ "$num" -gt 10 ]
then
  echo "The number is greater than 10."
else
  echo "The number is 10 or less."
fi
cdac@Vivobook15:~/LinuxAssignment$ bash assign
Enter a number:
5
The number is 10 or less.
```

10) 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.

Ans:

```
for i in {1..5}
do
  for j in {1..5}
  do
   printf "%d\t" $(( i * j ))
  done
  echo
done
cdac@Vivobook15:~/LinuxAssignment$ bash assign
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
```

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.

Ans:

```
while true
do
 read -p "Enter a Number :" num
 if [ $num -lt 0 ]
 then
    break
 fi
 echo "Square: $(( num * num ))"
done
echo "Loop exited"
cdac@Vivobook15:~/LinuxAssignment$ bash assign
Enter a Number :4
Square: 16
Enter a Number :-3
Loop exited
```

Part E

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

PID	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

Ans:

average waiting time is = 3.33

PID	Arrival Time	Burst Time	Respnce Time	Waiting Time	TAT			
P1	0	5	0	0	5			
P2	1	3	5	4	7			
P3	2	6	8	6	12			
			Avg RT=4.33	**Avg WT=3.33**	**Avg TT=8**			
		Gantt Chart	P1	P2	P4	P1	P3	
			0	1	5	7	12	19
		FCFS						

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

PID	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.

Ans:

Avg. Turnaround time = 5.5

PID	Arrival Time	Burst Time	Respnce Time	Waiting Time	TAT		
P1	0	3	0	0	3		
P2	1	5	8	7	12		
P3	2	1	3	1	2		
P4	3	4	4	1	5		
			Avg RT=3.75	**Avg WT=2.25**	**Avg TT=5.5**		
		Gantt Chart	P1	P3	P4	P2	
			0	3	4	8	13
		SJF					

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

PID	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.

Ans:

average waiting time = 4.5

PID	Arrival Time	Burst Time	Priority	Respnce Time	Waiting Time	TAT		
P1	0	6	3	0	6	12		
P2	1	4	1	1	0	4		
P3	2	7	4	12	10	17		
P4	3	2	2	7	2	4		
				Avg RT=5	**Avg WT=4.5**	**Avg TT=9.25**		
		Gantt Chart	P1	P2	P4	P1	P3	
			0	1	5	7	12	19
		Priority						

4. Consider the following processes with arrival times and burst times, and the time quantum for Round Robin scheduling is 2 units:

   Calculate the average turnaround time using Round Robin scheduling.
   Ans:
   average turnaround time=9.25

PID	Arrival Time	Burst Time	Respnce Time	Waiting Time	TAT	CT								
P1	0	4	0	6	10	10								
P2	1	5	2	8	13	14								
P3	2	2	4	2	4	6								
P4	3	3	6	7	10	13								time quantum=2
			Avg RT=3	**Avg WT=5.75**	**Avg TT=9.25**									
		Gantt Chart	P1	P2	P3	P4	P1	P2	P4	P3				
			0	2	4	6	8	10	12	13	14			
		Round Robin												