What will the following commands do?

Part A

- **echo "Hello, World!"**
  prints “Hello , World! ”
- **name="Productive"**
  setting variable with value
- **touch file.txt**
  Creating New File
- **ls -a**
  list all files including hidden
- **rm file.txt**
  removing/deleting file
- **cp file1.txt file2.txt**
  copy of one file to another file
- **mv file.txt /path/to/directory/**
  moving file to specific directory
- **chmod 755 script.sh**
  user =all permission group and others= read and executr
- **grep "pattern" file.txt**
  getting matching lines
- **kill PID**
  Terminating process with id

- **mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt**
  directory creates of name mydir doing to that firectory creates files printing Hello, World! And displays it

- **ls -l | grep ".txt"**
  listing and ,atching .txt files

- **cat file1.txt file2.txt | sort | uniq**
  conacanicating and then sorting it to the unique values

- **ls -l | grep "^d"**
  listing all directories and filters for directories

- **grep -r "pattern" /path/to/directory/**
  recursively searching for patterns in all files

- **cat file1.txt file2.txt | sort | uniq –d**
  concanates and shows only duplicate lines

- **chmod 644 file.txt**
  owner=rw group= r other=x

- **cp -r source_directory destination_directory**
  copying recursively sourve directory and its contents to destination directory

- **find /path/to/search -name "*.txt"**
  searches for all files ending with txt

- **chmod u+x file.txt**
  user having executive permission

- echo $PATH
  print value of path variable

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.it is used for changing directory**

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**

:

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

   True

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

11. **rm -rf file.txt** deletes a file forcefully without confirmation
    True.

Identify the Incorrect Commands:

1. **chmodx** is used to change file permissions.

   chmod is use to change file permission

2. **cpy** is used to copy files and directories.

   Cp is used to copy files and direcrories

3. **mkfile** is used to create a new file.

   Touch is use to create files

4. **catx** is used to concatenate files.

   cat is used to concatenate files

5. **rn** is used to rename files.

   mv is used to rename files

Part C

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

```
cdac@LAPTOP-6DTFV1AJ:~$ echo "Hello, World!"
Hello, World!
cdac@LAPTOP-6DTFV1AJ:~$
```

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

```
cdac@LAPTOP-6DTFV1AJ:~$ name="CDAC Mumbai"
cdac@LAPTOP-6DTFV1AJ:~$ echo $name
CDAC Mumbai
cdac@LAPTOP-6DTFV1AJ:~$
```

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

- **cdac@LAPTOP-6DTFV1AJ:~$ nano as.txt**
    - **echo enter what you want to print:**
    - **read sentence**
    - **echo $sentence**
- **cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt**
- **enter what you want to print:**
- **sanket sanjay kavanekar**
- **sanket sanjay kavanekar**
- **cdac@LAPTOP-6DTFV1AJ:~$**

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

- **cdac@LAPTOP-6DTFV1AJ:~$ echo $((5+3))**
- **8**
- **cdac@LAPTOP-6DTFV1AJ:~$**

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

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

- **cdac@LAPTOP-6DTFV1AJ:~$ nano as.txt**
    - **echo enter number**
    - **read number**
    - **if ((number % 2 == 0))**
    - **then**
    - **echo $number is even**
    - **else**
    - **echo $number is odd**
    - **fi**
- **cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt**
- **enter number**
- **13**
- **13 is odd**
- **cdac@LAPTOP-6DTFV1AJ:~$**

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

```
cdac@LAPTOP-6DTFV1AJ:~$ nano as.txt
        i=1
        while [ $i -le 5 ]
        do
        echo $i
        i=$((i+1))
        done
cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt
1
2
3
4
5
cdac@LAPTOP-6DTFV1AJ:~$
```

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@LAPTOP-6DTFV1AJ:~$ nano as.txt
    if [ -f "file.txt" ]
    then
    echo "File exists"
    else
    echo "file does not exist"
    fi
cdac@LAPTOP-6DTFV1AJ:~$ nano as.txt
cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt
file does not exist
cdac@LAPTOP-6DTFV1AJ:~$
```

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@LAPTOP-6DTFV1AJ:~$ nano as.txt
    echo enter number
    read number
    if [ $number -gt 10 ]
    then
    echo "$number is greater than 10"
    else
    echo "$number is not greater than 10"
    fi
cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt
enter number
13
13 is greater than 10
cdac@LAPTOP-6DTFV1AJ:~$
```

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@LAPTOP-6DTFV1AJ:~$ nano as.txt
    for i in {1..5}
    do
    for j in {1..5}
    do
    result=$((i*j))
    printf "%3d " $result
    done
    echo ""
    done
cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt
  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@LAPTOP-6DTFV1AJ:~$
```

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@LAPTOP-6DTFV1AJ:~$ nano as.txt**
    - **while true**
    - **do**
    - **read -p "Enter a number: " num**
    - **if [$num -lt 0]**
    - **then**
    - **break**
    - **fi**
        - **square=$((num*num))**
    - **echo "square of $num is $square"**
    - **done**
- **cdac@LAPTOP-6DTFV1AJ:~$ bash as.txt**
- **Enter a number: 12**
- **square of 12 is 144**
- **Enter a number: 13**
- **square of 13 is 169**

Part E

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.

Ans: **waiting time = 0 + 4+ 6=10**
Number of process=3
Avg wait = 10/3=3.33

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.

Tat Time = 3 + 12+ 2+ 5 = 22
No of Process = 4
Aveg Tat = 22 /4 = 5.5

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.

Waiting Time = 0 + 5+ 10 +7 = 22
No 0f process=4
Average Waiting Time = 22 / 4 = 5.5

---------- ---------------- --------------

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.

Ans: **10 + 13 + 4 + 10 = 37**
Process=4
Average tat =37/4=9.25

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?
Parent will be 6
Child will be 6

Submission Guidelines:

- Document each step of your solution and any challenges faced.
- Upload it on your GitHub repository

Additional Tips:

- Experiment with different options and parameters of each command to explore their functionalities.
- This assignment is tailored to align with interview expectations, CCEE standards, and industry demands.
- If you complete this then your preparation will be skyrocketed.