Concepts of Operating System

Assignment 2

Name – Sanika Subhash Karade (DAC- Juhu)

Part A

What will the following commands do?

- echo "Hello, World!"

Ans - Prints Hello, World! to the terminal.

- name="Productive"

Ans – Creates a variable name and assigns it the value Productive.

- touch file.txt

Ans - Creates an empty file named file.txt or updates its timestamp if it already exists.

- ls -a

Ans- Lists all files and directories in the current directory, including hidden ones.

- rm file.txt

Ans- Removes the file file.txt permanently.

- cp file1.txt file2.txt

Ans - Copies file1.txt to file2.txt. If file2.txtexists, it will be overwritten.

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

Ans- Moves file.txt to the specified directory.

- chmod 755 script.sh

Ans- Grants the owner full permissions (read, write, execute) and gives others read and execute permissions on script.sh.

- grep "pattern" file.txt

Ans- Searches for occurrences of "pattern" in file.txt and prints matching lines.

- kill PID

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

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

Ans - make a mydir directory and then go mydir also create a file.txt file and save echo "Hello, World!" in file.txt and display the data of file.txt

• ls -l | grep ".txt"

Ans - Lists files in long format and filters only those containing ". Txt" in their names.

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

Ans- Concatenates file1.txtand file2.txt, sorts them, and removes duplicate lines.

• ls -l | grep "^d"

Ans - Lists directories (entries starting with d in long format output).

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

Ans - Searches for "pattern" recursively in all files under /path/to/directory/.

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

Ans - Concatenates file1.txt and file2.txt, sorts them, and displays only duplicate lines.

• chmod 644 file.txt

Ans- Grants the owner read and write permissions, while others get read-only access to file.txt.

• cp -r source directory destination directory

Ans- Recursively copies source directory to destination directory, preserving contents.

• find /path/to/search -name "*.txt"

Ans- Finds all .txt files in/path/to/search and its subdirectories.

• chmod u+x file.txt

Ans- Gives the owner (u) execute permission on file.txt.

• echo $PATH

Ans - Displays the system's PATH environment variable, listing directories where executable files are searched for.

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 – The cd command is used to change the directory.

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:

1. chmodx is used to change file permissions.
   Ans- Incorrect - chmodx is not a valid command. The correct command to change file permissions is chmod.

2. cpy is used to copy files and directories.
   Ans- Incorrect - cpy is not a valid command. The correct command to copy files and directories is cp.

3. mkfile is used to create a new file.
   Ans- Incorrect - mkfile is not a standard Linux command. To create a new file, use filename.

4. catx is used to concatenate files.
   Ans- Incorrect - touch catx is not a valid command. The correct command to concatenate files is cat.

5. rn is used to rename files.
   Ans- Incorrect - rn is not a valid command. To rename files, use the mv command.

Part C

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

cdac@DESKTOP-150G8G6: ~/LinuxAssignment2

```
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh1
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh1
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh1
echo Hello, World!
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh1
Hello, World!
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

cdac@DESKTOP-150G8G6: ~/LinuxAssignment2

```
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh2
name="CDAC Mumbai"
echo "name = $name"
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh2
name = CDAC Mumbai
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

```
cdac@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh3
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh3
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh3
echo "Enter a Number"
read a
echo The Number is $a
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh3
Enter a Number
15
The Number is 15
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

```
cdac@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh4
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh4
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh4
echo "Enter a Number"
read a
echo "Enter a Number"
read b
sum=`expr $a + $b`
echo sum of $a and $b is $sum
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh4
Enter a Number
10
Enter a Number
5
sum of 10 and 5 is 15
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

```
cdac@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh5
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh5
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh5
echo "Enter a Number"
read a
if [ `expr $a % 2` -eq 0 ]
then
      echo "$a is an even number"
else
      echo "$a is an odd number"
fi

cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh5
Enter a Number
3
3 is an odd number
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh5
Enter a Number
8
8 is an even number
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

```
cdac@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh6
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh6
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh6
for i in 1 2 3 4 5
do
    echo $i
done
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh6
1
2
3
4
5
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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

```
Select cdac@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh7
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh7
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh7
a=1
while [ $a -le 5 ]
do
  echo $a
  a=$((a + 1))
done
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh7
1
2
3
4
5
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh8
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh8
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh8
if [ -e file.txt ]
then
     echo "File exists"
else
     echo "File doesn't exist"
fi
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh8
File doesn't exist
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh8
File doesn't exist
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch file.txt
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh8
File exists
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh9
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh9
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh9
echo "Enter a Number" ;
read a
if [ $a -gt 10 ]
then
     echo "$a is greater then 10"
else
     if [ $a -eq 10 ]
     then
          echo "$a is equal to 10"
     else
          echo "$a is smaller than 10"
     fi
fi
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh9
Enter a Number
3
3 is smaller than 10
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh10
bash: sh10: No such file or directory
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh9
Enter a Number
20
20 is greater then 10
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh9
Enter a Number
10
10 is equal to 10
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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@DESKTOP-150G8G6: ~/LinuxAssignment2

```
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh10
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh10
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh10
for num in {1..5}; do
          echo "Multiplication Table for $num"
          for i in {1..10};do
                    echo "$num x $i = $((num * i))"
          done
done
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh10
Multiplication Table for 1
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
1 x 10 = 10
Multiplication Table for 2
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
Multiplication Table for 3
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
```

```
Multiplication Table for 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
Multiplication Table for 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

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@DESKTOP-150G8G6: ~/LinuxAssignment2
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ touch sh11
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ nano sh11
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ cat sh11
num=0
while(( num >= 0))
do
   echo "Enter the Number"
   read num
   if ((num >=0 ))
   then
        s=$((num * num ))
        echo "The square of $num is $s"
   else
       break
   fi
done

cdac@DESKTOP-150G8G6:~/LinuxAssignment2$ bash sh11
Enter the Number
6
The square of 6 is 36
Enter the Number
-3
cdac@DESKTOP-150G8G6:~/LinuxAssignment2$
```

Part E

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

Q 1]

Process	Arrival Time	Burst Time	Waiting Time
P1	0	5	0
P2	1	3	4
P3	2	6	6

Gantt Chart using FCFS

P1	P2	P3	
0	5	8	14

$$\text{Average Waiting Time} = \frac{0+4+6}{3}$$

$$= 10/3$$

$$= 3.333$$

Average Waiting Time = 3.33

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

Q2]

Process	Arrival Time	Burst Time	Turn Around Time
P1	0	3	4
P2	1	5	12
P3	2	1	1
P4	3	4	5

Gantt Chart

P1	P1	P3	P1	P4	P2	
0	1	2	3	4	8	13

Average Turn Around Time $= (4+12+1+5)/4$

$= 5.5$

3. Calculate the average waiting time using Priority Scheduling.

Q3]

Process	Arrival Time	Burst Time	Priority	Waiting Time
P1	0	6	3	6
P2	1	4	1	0
P3	2	7	4	10
P4	3	2	2	2

Gantt Chart.

P1	P2	P2	P2	P2	P4	P4
0	1	2	3	4	5	6

P1	P3	
7	12	19

Average waiting Time $= (6+0+10+2)/4$

$= 4.5$

4. Calculate the average turnaround time using Round Robin scheduling.

Q4] Quantum = 2 units

Process	Arrival Time	Burst Time	Waiting Time	Turn Around Time
P1	0	4	6	10
P2	1	5	8	13
P3	2	2	2	4
P4	3	3	7	10

[Reduced Quantum – CPU not idle]

Gantt Chart using ~~Robi~~ Round Robin

P1	P2	P3	P4	P1	P2	P4
0	2	4	6	8	10	12

P2

13 14

Average TAT = (10+13+4+10)/4

Average TAT = 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?

Ans:

1. **Before the fork() call:**
    - The parent process has a variable x with a value of 5.
2. **After the fork() call:**
    - The parent process and the child process both have a variable x with an initial value of 5.
3. **Incrementing the value of x by 1 in both processes:**
    - In the parent process, x becomes 6.
    - In the child process, x also becomes 6.

Therefore, the final value of x in both the parent and child processes after the fork() call and the increment operations will be 6.