CDAC MUMBAI

Concepts of Operating System

Assignment 2

Name : Ankita Balasaheb Dhumal
Corse :DAC(Online)Feb25

Part A

What will the following commands do?

- echo "Hello, World!"

Ans : This Command Prints the argument Hello world!

- name="Productive"

Ans : it stores value : Productive in variable : name

- touch file.txt

Ans : it will create new empty file : file.txt .

- ls -a

Ans : lists all files & directories and hidden files and directories from the current directory

- rm file.txt

Ans : **It will remove file.txt from directory.**

- cp file1.txt file2.txt

Ans: It will copy entire data of file1.txt into file2.txt if file2.txt is not present it will create it.

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

Ans : it will move file into given directory.

- chmod 755 script.sh

Ans : It changes the file permissions. Allows to owner to read, write, and execute (rwx=7). allows the group and others to read and execute (r-x=5) 755.

- grep "pattern" file.txt

Ans : It will search for pattern word in file.txt.

- kill PID

Ans : It will send termination signal to the process id.

- mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt □ ls -l | grep ".txt"

Ans : this command did multiple task at a time.first its create mkdir directory,then direct into it, then created empty file name as file.txt then print hello world into it and displayed contain of file.txt.

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

Ans : It will combines two files, removes duplicate data also arrange it alphabetically.

- ls -l | grep "^d"

Ans : ls -l will displays list of files and directories with permission information, it will filter the output and grep d find and shows lines starts with d.

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

Ans : it will recursively search for pattern from directory.

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

Ans : display combine both files in alphabetic order and shows only data which prasends in both files.

- chmod 644 file.txt

Ans : it sets permissions of file.txt as sets file permissions for file.txt so that : owner can read and write, the group can only read, others can only read.

- cp -r source_directory destination_directory

Ans : it copies one directory and its entire content into another directory.

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

Ans : it searched for all file that ends with .txt in directory.

- chmod u+x file.txt

Ans : it adds execute permission to user(owner) for file.txt.

- echo $PATH

Ans :Display current System's environment 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 , cd 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. correct command is chmod to change file permission.

2. cpy is used to copy files and directories.
Ans : Incorrect. correct command is cp to copy files and directories

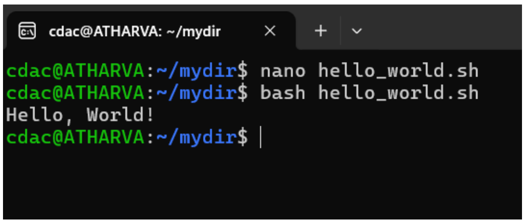
3. mkfile is used to create a new file.
Ans : Incorrect. there is no mkfile in Linux.

4. catx is used to concatenate files.
Ans : Incorrect. correct command is cat to concatenate file.

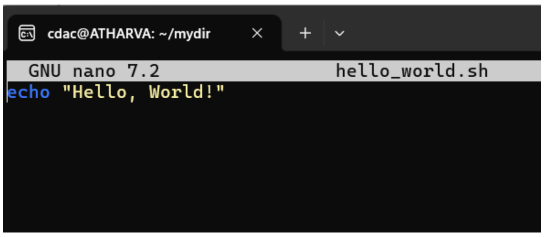
5. rn is used to rename
Ans : Incorrect. correct command is mv to rename.

Part C

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



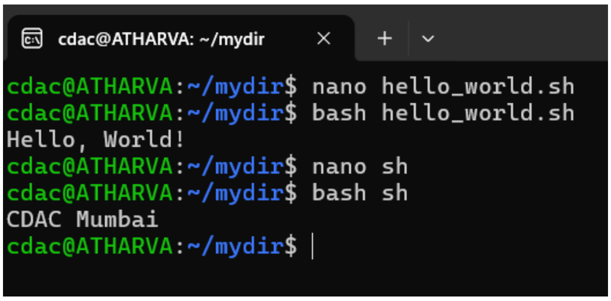
```
cdac@ATHARVA:~/mydir$ nano hello_world.sh
cdac@ATHARVA:~/mydir$ bash hello_world.sh
Hello, World!
cdac@ATHARVA:~/mydir$ |
```



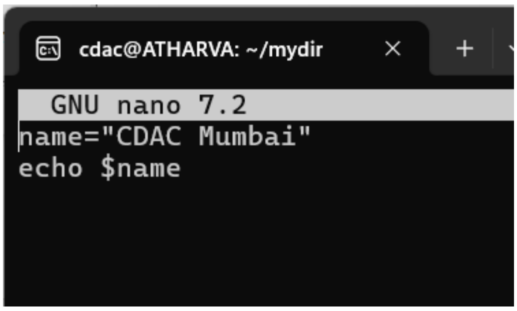
```
  GNU nano 7.2                    hello_world.sh
echo "Hello, World!"
```

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

Ans :



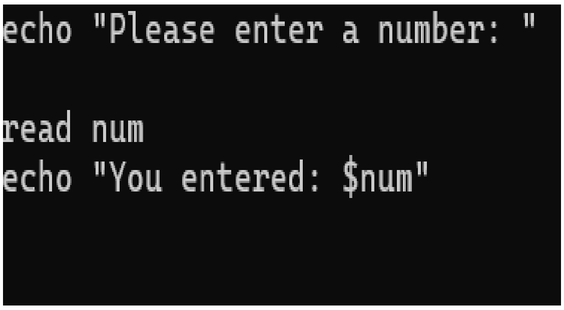
```
cdac@ATHARVA:~/mydir$ nano hello_world.sh
cdac@ATHARVA:~/mydir$ bash hello_world.sh
Hello, World!
cdac@ATHARVA:~/mydir$ nano sh
cdac@ATHARVA:~/mydir$ bash sh
CDAC Mumbai
cdac@ATHARVA:~/mydir$
```



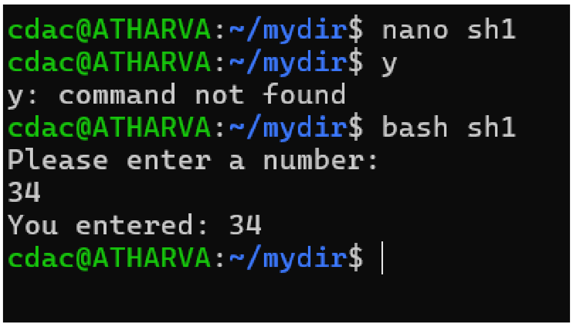
```
  GNU nano 7.2
name="CDAC Mumbai"
echo $name
```

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

Ans :



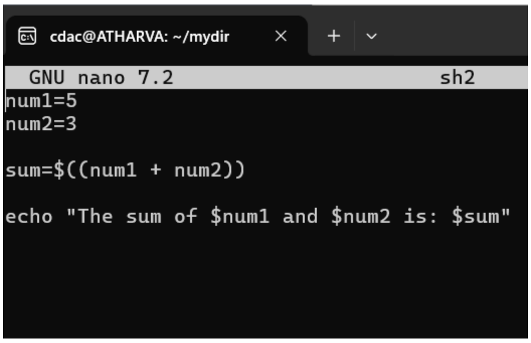
```
echo "Please enter a number: "

read num
echo "You entered: $num"
```



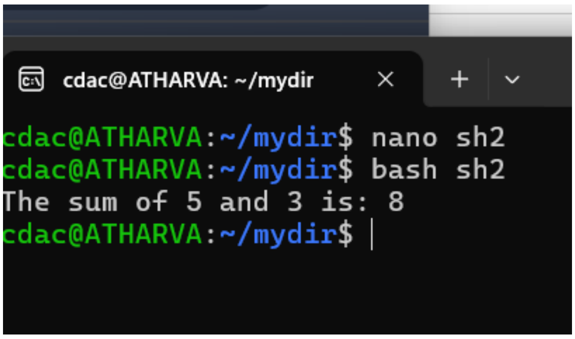
```
cdac@ATHARVA:~/mydir$ nano sh1
cdac@ATHARVA:~/mydir$ y
y: command not found
cdac@ATHARVA:~/mydir$ bash sh1
Please enter a number:
34
You entered: 34
cdac@ATHARVA:~/mydir$
```

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

Ans :



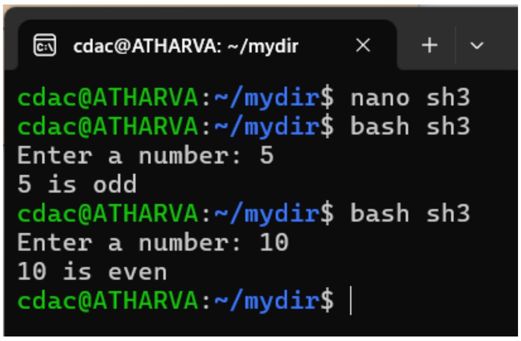
```
  GNU nano 7.2                          sh2
num1=5
num2=3

sum=$((num1 + num2))

echo "The sum of $num1 and $num2 is: $sum"
```



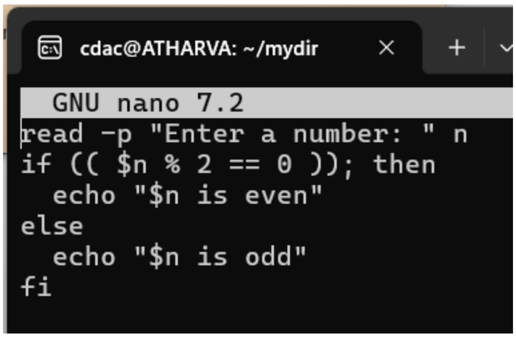
```
cdac@ATHARVA:~/mydir$ nano sh2
cdac@ATHARVA:~/mydir$ bash sh2
The sum of 5 and 3 is: 8
cdac@ATHARVA:~/mydir$
```

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

Ans :

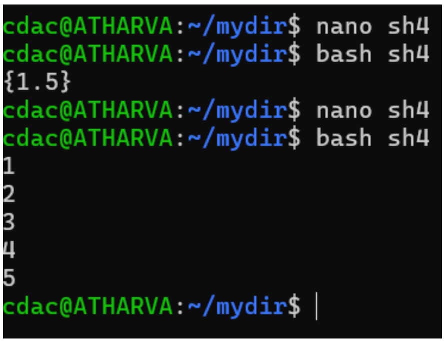


```
cdac@ATHARVA:~/mydir$ nano sh3
cdac@ATHARVA:~/mydir$ bash sh3
Enter a number: 5
5 is odd
cdac@ATHARVA:~/mydir$ bash sh3
Enter a number: 10
10 is even
cdac@ATHARVA:~/mydir$
```

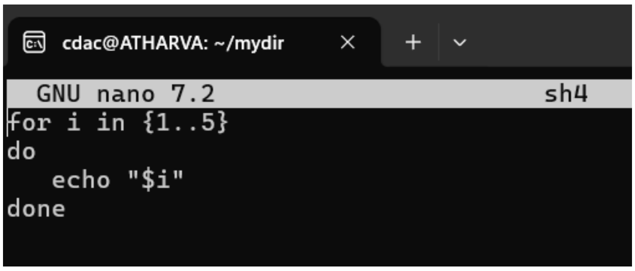


```
  GNU nano 7.2
read -p "Enter a number: " n
if (( $n % 2 == 0 )); then
  echo "$n is even"
else
  echo "$n is odd"
fi
```

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

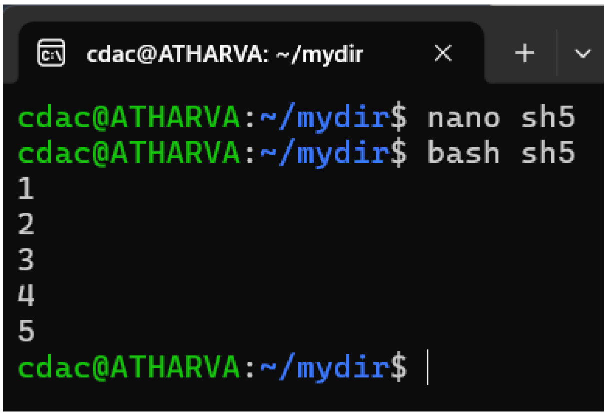


```
cdac@ATHARVA:~/mydir$ nano sh4
cdac@ATHARVA:~/mydir$ bash sh4
{1.5}
cdac@ATHARVA:~/mydir$ nano sh4
cdac@ATHARVA:~/mydir$ bash sh4
1
2
3
4
5
cdac@ATHARVA:~/mydir$
```

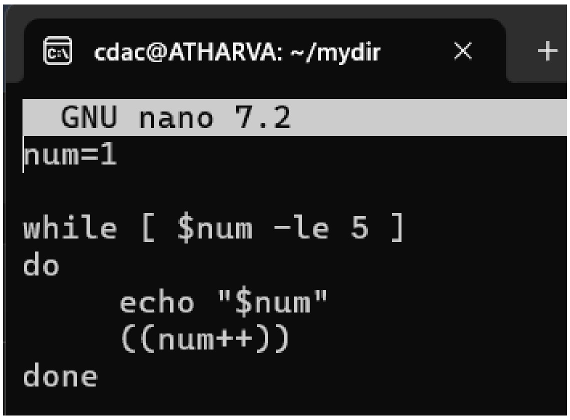


```
  GNU nano 7.2                                       sh4
for i in {1..5}
do
   echo "$i"
done
```

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

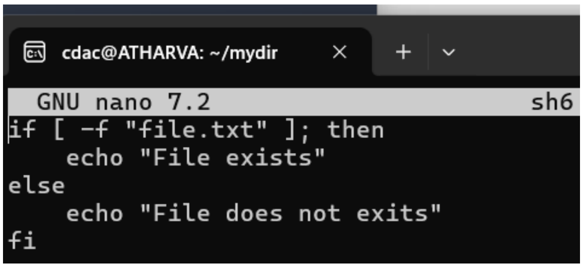


```
cdac@ATHARVA:~/mydir$ nano sh5
cdac@ATHARVA:~/mydir$ bash sh5
1
2
3
4
5
cdac@ATHARVA:~/mydir$
```

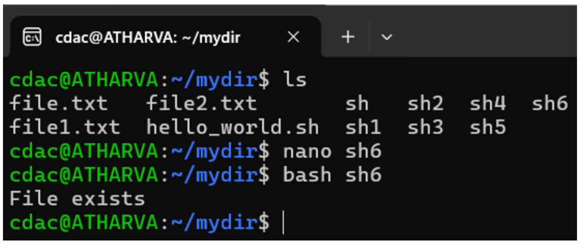


```
  GNU nano 7.2
num=1

while [ $num -le 5 ]
do
     echo "$num"
     ((num++))
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".
Ans :



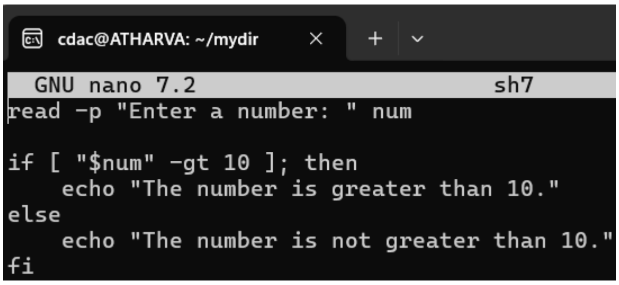
```
  GNU nano 7.2                          sh6
if [ -f "file.txt" ]; then
    echo "File exists"
else
    echo "File does not exits"
fi
```



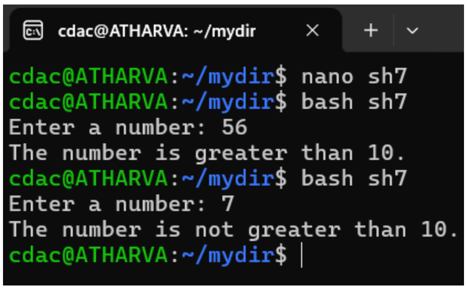
```
cdac@ATHARVA:~/mydir$ ls
file.txt   file2.txt       sh   sh2  sh4  sh6
file1.txt  hello_world.sh  sh1  sh3  sh5
cdac@ATHARVA:~/mydir$ nano sh6
cdac@ATHARVA:~/mydir$ bash sh6
File exists
cdac@ATHARVA:~/mydir$
```

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.

Ans :



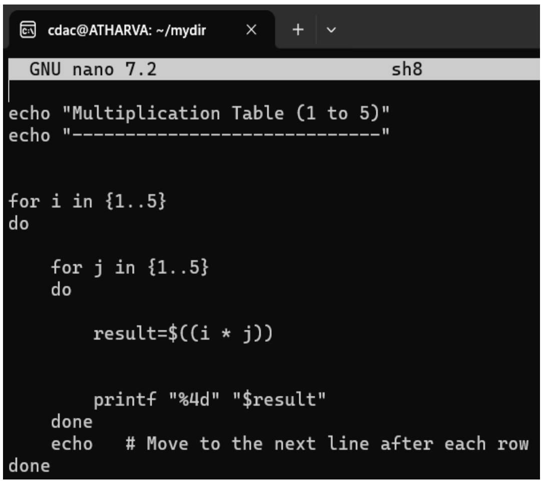
```
 GNU nano 7.2                                  sh7
read -p "Enter a number: " num

if [ "$num" -gt 10 ]; then
    echo "The number is greater than 10."
else
    echo "The number is not greater than 10."
fi
```



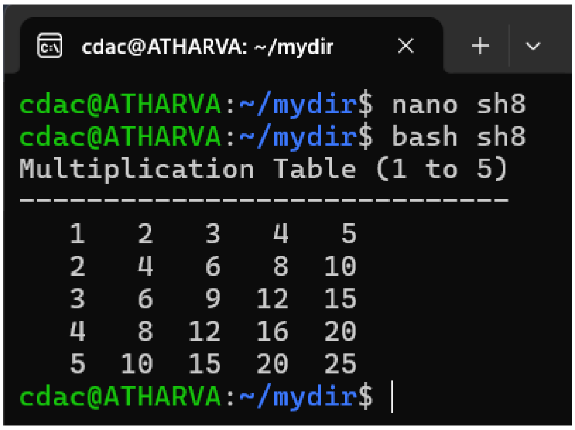
```
cdac@ATHARVA:~/mydir$ nano sh7
cdac@ATHARVA:~/mydir$ bash sh7
Enter a number: 56
The number is greater than 10.
cdac@ATHARVA:~/mydir$ bash sh7
Enter a number: 7
The number is not greater than 10.
cdac@ATHARVA:~/mydir$
```

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 :



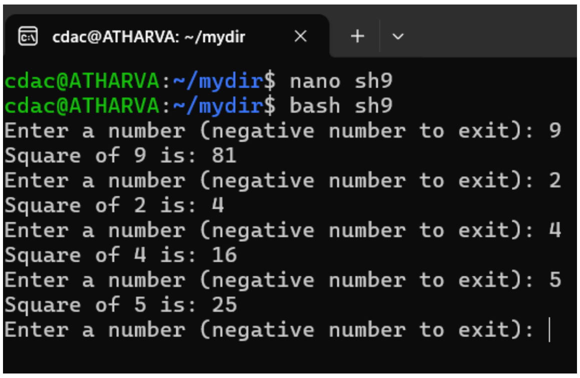
```
 GNU nano 7.2                                  sh8

echo "Multiplication Table (1 to 5)"
echo "-----------------------------"


for i in {1..5}
do

    for j in {1..5}
    do

        result=$((i * j))


        printf "%4d" "$result"
    done
    echo    # Move to the next line after each row
done
```



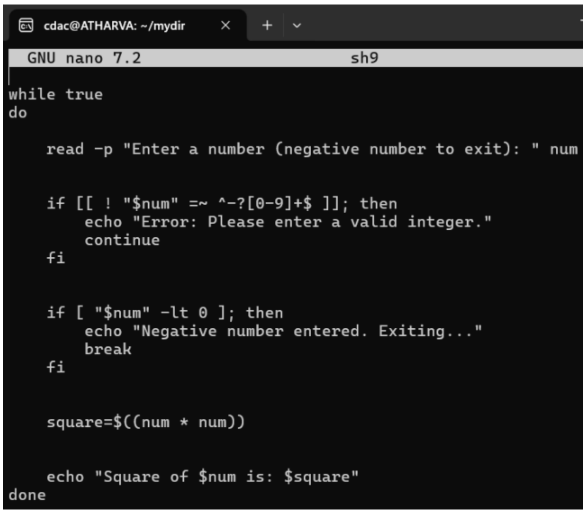
```
cdac@ATHARVA:~/mydir$ nano sh8
cdac@ATHARVA:~/mydir$ bash sh8
Multiplication Table (1 to 5)
-----------------------------
   1   2   3   4   5
   2   4   6   8  10
   3   6   9  12  15
   4   8  12  16  20
   5  10  15  20  25
cdac@ATHARVA:~/mydir$
```

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 :



```
cdac@ATHARVA:~/mydir$ nano sh9
cdac@ATHARVA:~/mydir$ bash sh9
Enter a number (negative number to exit): 9
Square of 9 is: 81
Enter a number (negative number to exit): 2
Square of 2 is: 4
Enter a number (negative number to exit): 4
Square of 4 is: 16
Enter a number (negative number to exit): 5
Square of 5 is: 25
Enter a number (negative number to exit):
```



```
 GNU nano 7.2                                  sh9

while true
do

    read -p "Enter a number (negative number to exit): " num


    if [[ ! "$num" =~ ^-?[0-9]+$ ]]; then
        echo "Error: Please enter a valid integer."
        continue
    fi


    if [ "$num" -lt 0 ]; then
        echo "Negative number entered. Exiting..."
        break
    fi


    square=$((num * num))


    echo "Square of $num is: $square"
done
```

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:

Process	Arrival Time	Burst Time	**Response Time**	**Turnaround Time**	**Wating Time**
P1	0	5	**5**	**5 -0 = 5**	**0**
P2	1	3	**8**	**8-1=7**	**4**
P3	2	6	**14**	**14-2=12**	**6**

Total Wating time=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.

Ans:

Process	Arrival Time	Burst Time	**Wating Time**	**Turnaround Time**	**Wating Time**
P1	0	3	**3**	**3**	**0**
P2	1	5	**4**	**2**	**1**
P3	2	1	**8**	**5**	**1**
P4	3	4	**13**	**12**	**17**

Average Turn Around Time = 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.

Ans :

Process	Arrival Time	Burst Time	Priority	**Response Time**	**Trun Around Time**	**Wating Time**
P1	0	6	3	**0**	**6**	**0**
P2	1	4	1	**5**	**9**	**5**
P3	2	7	4	**7**	**9**	7
P4	3	2	2	**10**	**17**	**10**

Average Wating Time : 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 :

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

Average Turnaround Time : 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 :

Child Process X = 6
Parent Process X =6