

Name :Shivam Dolase_KH

CDAC MUMBAI

Concepts of Operating System

Assignment 2

Part A

What will the following commands do?

- `echo "Hello, World!"`

Answer: This command displays “Hello, World!”.

- `name="Productive"`

Answer: This command create new variable name and assigns it a String values.

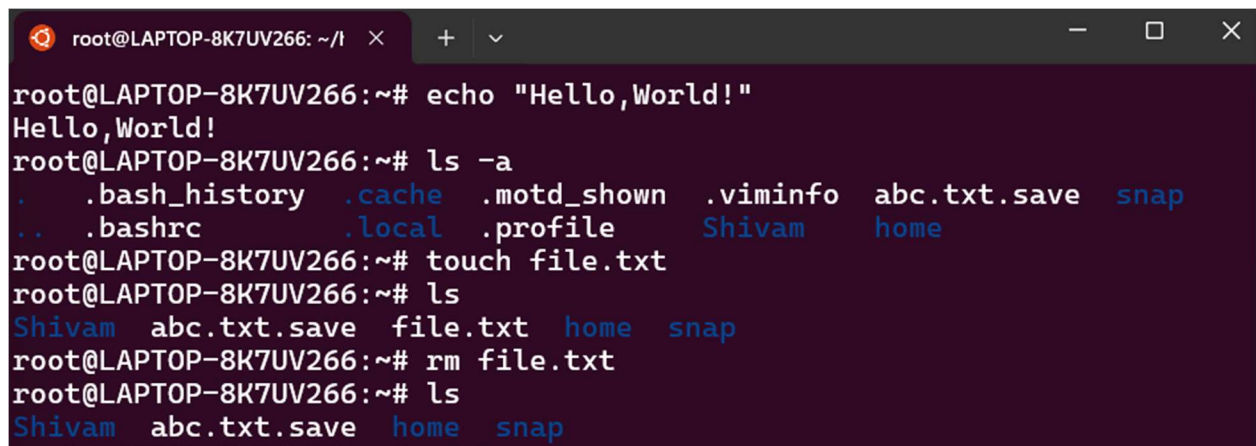
- `touch file.txt`

Answer: This command creates new file file.txt in current directory.

- `ls -a`

Answer: This command represent all files include hidden files and directories in the listing in current directory.

- `rm file.txt`



```
root@LAPTOP-8K7UV266: ~/l × + ▾
root@LAPTOP-8K7UV266:~# echo "Hello,World!"
Hello,World!
root@LAPTOP-8K7UV266:~# ls -a
.  .bash_history  .cache  .motd_shown  .viminfo  abc.txt.save  snap
.. .bashrc      .local  .profile     Shivam     home
root@LAPTOP-8K7UV266:~# touch file.txt
root@LAPTOP-8K7UV266:~# ls
Shivam  abc.txt.save  file.txt  home  snap
root@LAPTOP-8K7UV266:~# rm file.txt
root@LAPTOP-8K7UV266:~# ls
Shivam  abc.txt.save  home  snap
```

Answer: This command removes file.txt from current directory.

- `cp file1.txt file2.txt`

Answer: This command will copy file1.txt content to file2.txt.

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

Answer: This command moves file.txt to the directory.

- `chmod 755 script.sh`

Answer: This command gives execution rights to all owner-u, group-g and other-o for the file script.sh.

```
root@LAPTOP-8K7UV266:~/home# touch script.sh
root@LAPTOP-8K7UV266:~/home# chmod 755 script.sh
root@LAPTOP-8K7UV266:~/home# ls -l
total 0
-rwxr-xr-x 1 root root 0 Aug 30 23:48 script.sh
```

- `grep "pattern" file.txt`

Answer: This command highlights and displays the lines in file.txt containing the word “pattern”.

- `kill PID`

Answer:

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

Answer: Here first “mydir” directory is created using mkdir command and then moved into that directory using cd command. Then file.txt is created with touch command and with the help of echo command Hello,world! is written in file.txt and the cat command displays the file.txt .

```
root@LAPTOP-8K7UV266:~# mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt
Hello, World!
```

- `ls -l | grep ".txt"`

```
root@LAPTOP-8K7UV266:~/mydir# ls -l | grep ".txt"
-rw-r--r-- 1 root root 14 Aug 31 00:02 file.txt
```

Answer: It displays the controls of all the files with “.txt” extension in that directory.

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

Answer: sort command first sorts the both file contents and uniq command displays unique contents in these files no duplicate contents

```
root@LAPTOP-8K7UV266:~/mydir# cat file1.txt file2.txt | sort | uniq
abc
ddd
ddd
shivam
```

- `ls -l | grep "^d"`

Answer: Lists the Permission list for all the sub-directories in the current directories.

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

Answer: This command return the search pattern in given directory recursively in all the files i.e. it dose the recursive search for the given pattern.

```
root@LAPTOP-8K7UV266:~# grep -r "pattern" home/
home/file1.txt:pattern hi patter
home/file1.txt:oh pattern my pattern
```

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

Answer: This command only outputs the repeated lines in file1.txt and file2.txt

- `chmod 644 file.txt`

Answer: This command gives read , write permission to user(owner) and read permission to group and other.

```
root@LAPTOP-8K7UV266:~/home# chmod 644 file1.txt
root@LAPTOP-8K7UV266:~/home# ls -l
ls-l: command not found
root@LAPTOP-8K7UV266:~/home# ls -l
total 8
-rw-r--r-- 1 root root 74 Aug 31 01:07 file1.txt
-rwxr-xr-x 1 root root 0 Aug 30 23:48 script.sh
-rwxr-xr-x 1 root root 0 Aug 30 23:52 script1.sh
drwxr-xr-x 2 root root 4096 Aug 31 01:08 torch
```

- `cp -r source_directory destination_directory`

Answer: This command recursively copies surce directory to the destination directory.

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

Answer: it searches for the ".txt" files in the given directory

- `chmod u+x file.txt`

Answer: it gives the user(owner) of the file permission to execute the file.

- `echo $PATH`

Answer: The command "echo \$PATH" in Linux displays the current value of the \$PATH variable. The \$PATH variable is an environment variable that contains a list of directories that the system checks before running a command.

```
root@LAPTOP-8K7UV266:~/home# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/usr/lib/wsl/lib:/mnt/c/Python312/Scripts:/mnt/c/Python312:/mnt/c/oracle/app/oracle/product/11.2.0/server/bin:/mnt/c/Program Files/Common Files/Oracle/Java/javapath:/mnt/c/Program Files/Python39/Scripts:/mnt/c/Program Files/Python39:/mnt/c/WINDOWS/system32:/mnt/c/WINDOWS:/mnt/c/WINDOWS/System32/Wbem:/mnt/c/WINDOWS/System32/WindowsPowerShell/v1.0:/mnt/c/WINDOWS/System32/OpenSSH:/mnt/c/MinGW/bin:/mnt/c/Program Files/Java/jdk-16.0.1/bin:/mnt/c/Program Files/Git/cmd:/mnt/c/Windows/System32/cmd.exe:/mnt/c/Program Files/Java/jdk-17.0.2/bin:/mnt/c/Program Files/MongoDB/Server/6.0/bin:/mnt/c/Users/hp/AppData/Roaming/Python/Python39/Scripts:/mnt/c/Program Files/nodejs:/mnt/c/ProgramData/chocolatey/bin:/mnt/c/Program Files/MySQL/MySQL Shell 8.0/bin:/mnt/c/Users/hp/AppData/Local/Microsoft/WindowsApps:/mnt/c/Users/hp/AppData/Local/Programs/Microsoft VS Code/bin:/mnt/c/Users/hp/AppData/Local/GitHubDesktop/bin:/mnt/c/Windows/System32/cmd.exe:/mnt/c/Program Files/MongoDB/Server/6.0/bin:/mnt/c/Program Files/JetBrains/IntelliJ IDEA Community Edition 2023.3.3/bin:/mnt/c/Users/hp/AppData/Roaming/npm:/snap/bin
```

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
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

Identify the Incorrect Commands:

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

Answer: “`chmod`” is the correct command

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

Answer: “`cp`” is the command is the correct command for coping the file.

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

Answer: “`mkdir`” is the correct command for creating directories and “`touch`” command is used to create a file , `mkfile` is not a correct command.

4. `catx` is used to concatenate files.

Answer: “`cat`” is the correct command for displaying the content of the required file.

5. `rn` is used to rename files.

Answer: “`mv`” command is used to rename files .Also “`rename`” command is used to rename the large group of files at once.

Part C

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

Answer:

`echo "Hello, World!"`

```
cdac@LAPTOP-8K7UV266:~$ echo "Hello, World!"  
Hello, World!  
cdac@LAPTOP-8K7UV266:~$ |
```

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

Answer:

```
#!/bin/bash/  
name="CDAC Mumbai"  
echo $name  
LAPTOP-8K7UV266:~$ nano shell  
LAPTOP-8K7UV266:~$ bash shell  
Mumbai  
LAPTOP-8K7UV266:~$ |
```

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

Answer:

```
#!/bin/bash/  
  
echo "Enter a number:"  
read number  
echo "You entered: $number"  
cdac@LAPTOP-8K7UV266:~$ bash shell  
Enter a number:  
5  
You entered: 5  
cdac@LAPTOP-8K7UV266:~$ cat shell  
#!/bin/bash/  
echo "Enter a number:"  
read number  
echo "You entered: $number"  
cdac@LAPTOP-8K7UV266:~$ |
```

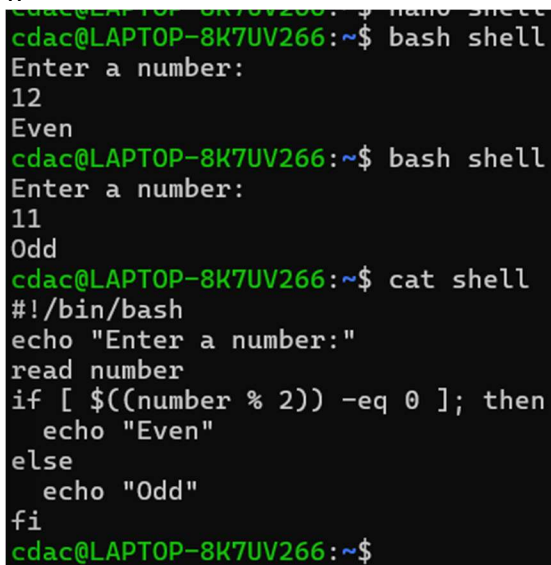
Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.

```
#!/bin/bash/  
num1=5  
num2=3  
sum=$((num1 + num2))  
echo "The sum of $num1 and $num2 is: $sum"  
cdac@LAPTOP-8K7UV266:~$ bash shell  
The sum of 5 and 3 is: 8  
cdac@LAPTOP-8K7UV266:~$ cat shell  
#!/bin/bash/  
num1=5  
num2=3  
sum=$((num1 + num2))  
echo "The sum of $num1 and $num2 is: $sum"  
cdac@LAPTOP-8K7UV266:~$
```

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

Answer:

```
#!/bin/bash
echo "Enter a number:"
read number
if [  $$(number \% 2)$  -eq 0 ]; then
    echo "Even"
else
    echo "Odd"
fi
```

A terminal window with a black background and green text. It shows the execution of a shell script. The prompt is 'cdac@LAPTOP-8K7UV266:~\$'. The user enters 'bash shell'. The prompt changes to 'cdac@LAPTOP-8K7UV266:~\$'. The script prompts 'Enter a number:'. The user enters '12'. The script outputs 'Even'. The prompt changes back to 'cdac@LAPTOP-8K7UV266:~\$'. The user enters 'bash shell'. The prompt changes to 'cdac@LAPTOP-8K7UV266:~\$'. The script prompts 'Enter a number:'. The user enters '11'. The script outputs 'Odd'. The prompt changes back to 'cdac@LAPTOP-8K7UV266:~\$'. The user enters 'cat shell'. The script content is displayed: '#!/bin/bash', 'echo "Enter a number:"', 'read number', 'if [$$(number \% 2)$ -eq 0]; then', ' echo "Even"', 'else', ' echo "Odd"', 'fi'. The prompt changes back to 'cdac@LAPTOP-8K7UV266:~\$'.

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

```
#!/bin/bash
for i in {1..5}
do
    echo $i
done
```

```
cdac@LAPTOP-8K7UV266:~$ bash shell
1
2
3
4
5
cdac@LAPTOP-8K7UV266:~$ cat shell
#!/bin/bash
for i in {1..5}
do
    echo $i
done
cdac@LAPTOP-8K7UV266:~$
```

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

```
#!/bin/bash/
```

```
n=1
while [ $n -lt 6 ]
do
    echo $n
    ((n++))
done
```

```
cdac@LAPTOP-8K7UV266:~$ nano shell
cdac@LAPTOP-8K7UV266:~$ bash shell
1
2
3
4
5
cdac@LAPTOP-8K7UV266:~$ cat shell
#!/bin/bash/

n=1
while [ $n -lt 6 ]
do
    echo $n
    ((n++))
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".


```
if [ -f "file.txt" ]; then
```

```
    echo "File exists"
```

```
else
```

```
    # If the file does not exist, print "File does not exist"
```

```
    echo "File does not exist"
```

```
fi
```

```
cdac@LAPTOP-8K7UV266:~$ nano shell
cdac@LAPTOP-8K7UV266:~$ touch file.txt
cdac@LAPTOP-8K7UV266:~$ bash shell
File exists
cdac@LAPTOP-8K7UV266:~$ rm file.txt
cdac@LAPTOP-8K7UV266:~$ bash shell
File does not exist
cdac@LAPTOP-8K7UV266:~$ cat shell
#!/bin/bash/
if [ -f "file.txt" ]
then
    echo "File exists"
else
    echo "File does not exist"
fi

cdac@LAPTOP-8K7UV266:~$ |
```

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.

Answer:

```
read n
```

```
if [ $n -ge 10 ]
```

```
then
```

```
echo "$n is greater than 10"
```

```
else
```

```
echo "$n is not greater than 10"
```

```
fi
```

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.

```
for i in {1..5}
```

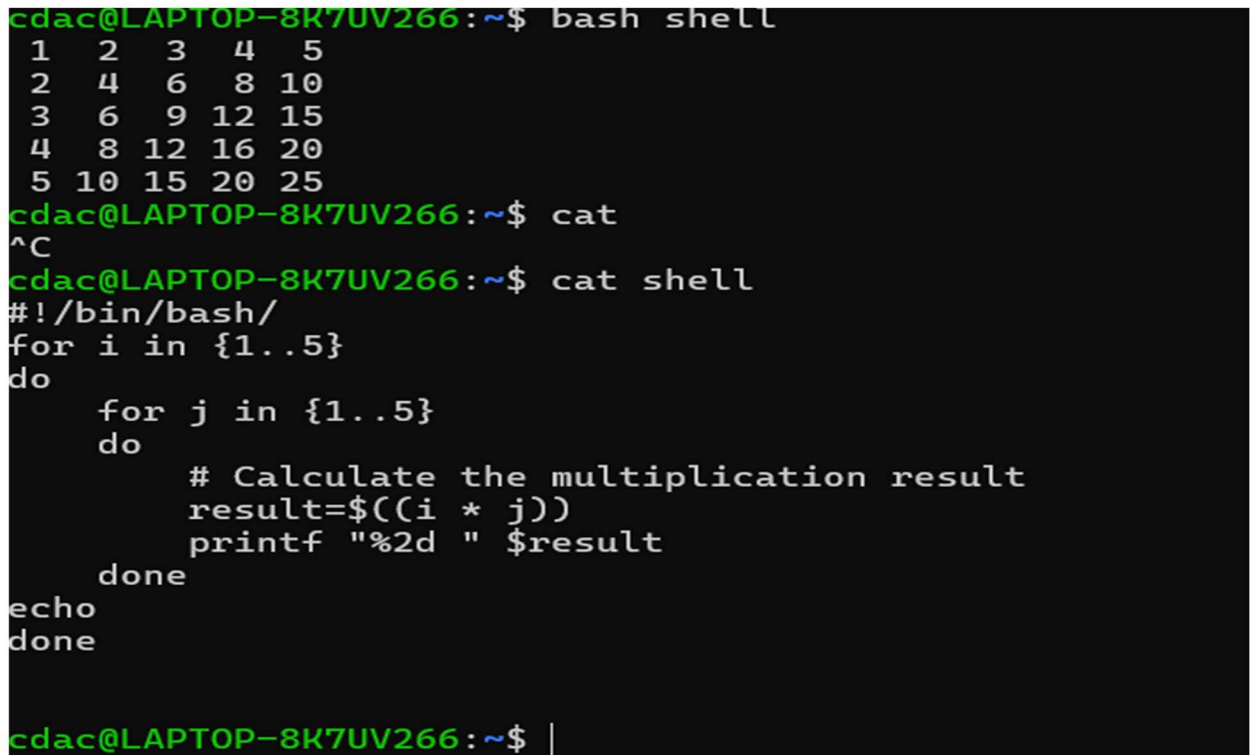
```
do
```

```

for j in {1..5}
do

    result=$((i * j))
    printf "%2d " $result
done
echo
done

```



```

cdac@LAPTOP-8K7UV266:~$ bash shell
 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-8K7UV266:~$ cat
^C
cdac@LAPTOP-8K7UV266:~$ cat shell
#!/bin/bash/
for i in {1..5}
do
    for j in {1..5}
    do
        # Calculate the multiplication result
        result=$((i * j))
        printf "%2d " $result
    done
done
echo
done

cdac@LAPTOP-8K7UV266:~$ |

```

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.

```

while true
do
    read num
    if [ $num -lt 0 ]; then
        break
    fi
    sq=$((num * num))
    echo $sq
done

```

```

cdac@LAPTOP-8K7UV266:~$ nano shell
cdac@LAPTOP-8K7UV266:~$ bash shell
12
144
6
36
-1
cdac@LAPTOP-8K7UV266:~$ cat shell
#!/bin/bash/
while true
do
    read num
    if [ $num -lt 0 ]; then
        break
    fi
    sq=$((num * num))
    echo $sq
done
cdac@LAPTOP-8K7UV266:~$ |

```

Part E

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

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

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 First-Come, First-Served (FCFS) scheduling.

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

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

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

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.

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.

Assignment 2 - OS

Part E:

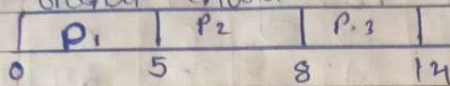
$$TAT = CT - AT$$

$$WT = TAT - BT$$

1) FCFS Scheduling:

Process	Arrival T	Burst T	Completion T	Turnaround T	WT
P ₁	0	5	5	5	0
P ₂	1	3	8	7	4
P ₃	2	6	14	12	6

Gantt Chart



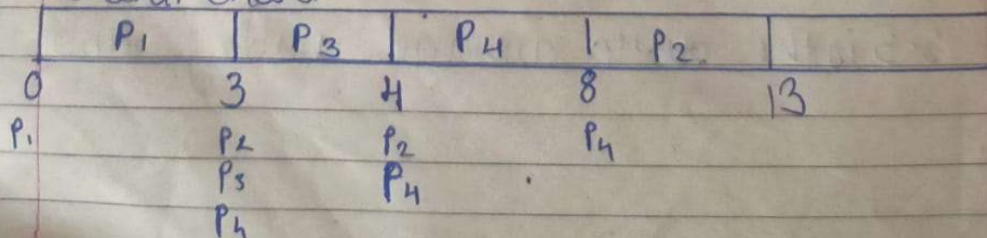
Average waiting time: $\frac{0 + 4 + 6}{3} = \frac{10}{3}$

[3.33 is the avg waiting time.]

2) SJF Scheduling

P _i	Arrival Time	Burst T	Completion T	Turnaround T	WT
P ₁	0	3	3	3	0
P ₂	1	5	13	12	7
P ₃	2	1	4	2	1
P ₄	3	4	8	5	1

Gantt Chart



Date: / /
Page No.:

$$\text{Average Waiting Time} = \frac{0 + 7 + 11 + 1}{4} = \frac{9}{4} = 2.25$$

2.25 is the average waiting time.

3) a) Priorities scheduling (non-preemptive)

Pro	Arrival T	Burst T	Priority	CT	TAT	WT
P ₁	0	6	3	6	6	0
P ₂	1	4	1	10	9	5
P ₃	2	7	4	19	17	10
P ₄	3	2	2	12	9	7

[lower the number higher the priority]

Gantt Chart:

	P ₁	P ₂	P ₄	P ₃	
0	6	10	12	19	
P ₁	P ₂	P ₃	P ₃		
	P ₃	P ₄			
	P ₄				

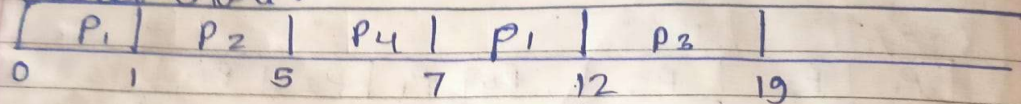
$$\text{Average Waiting Time} = \frac{0 + 5 + 10 + 7}{4} = \frac{22}{4} = 5.5$$

5.5 is the ~~out~~ average waiting time

b) Preemptive. Priorities scheduling.

Pro	Arrival T	Burst T	Priority	CT	TAT	WT
P ₁	0	6	3	12	12	6
P ₂	1	4	1	5	4	0
P ₃	2	7	4	19	17	10
P ₄	3	2	2	7	4	2

Gantt Chart:



Waiting time:

0: P₁

1: P₂

5: P₁ P₃ P₄

7: P₁ P₃

12: P₃

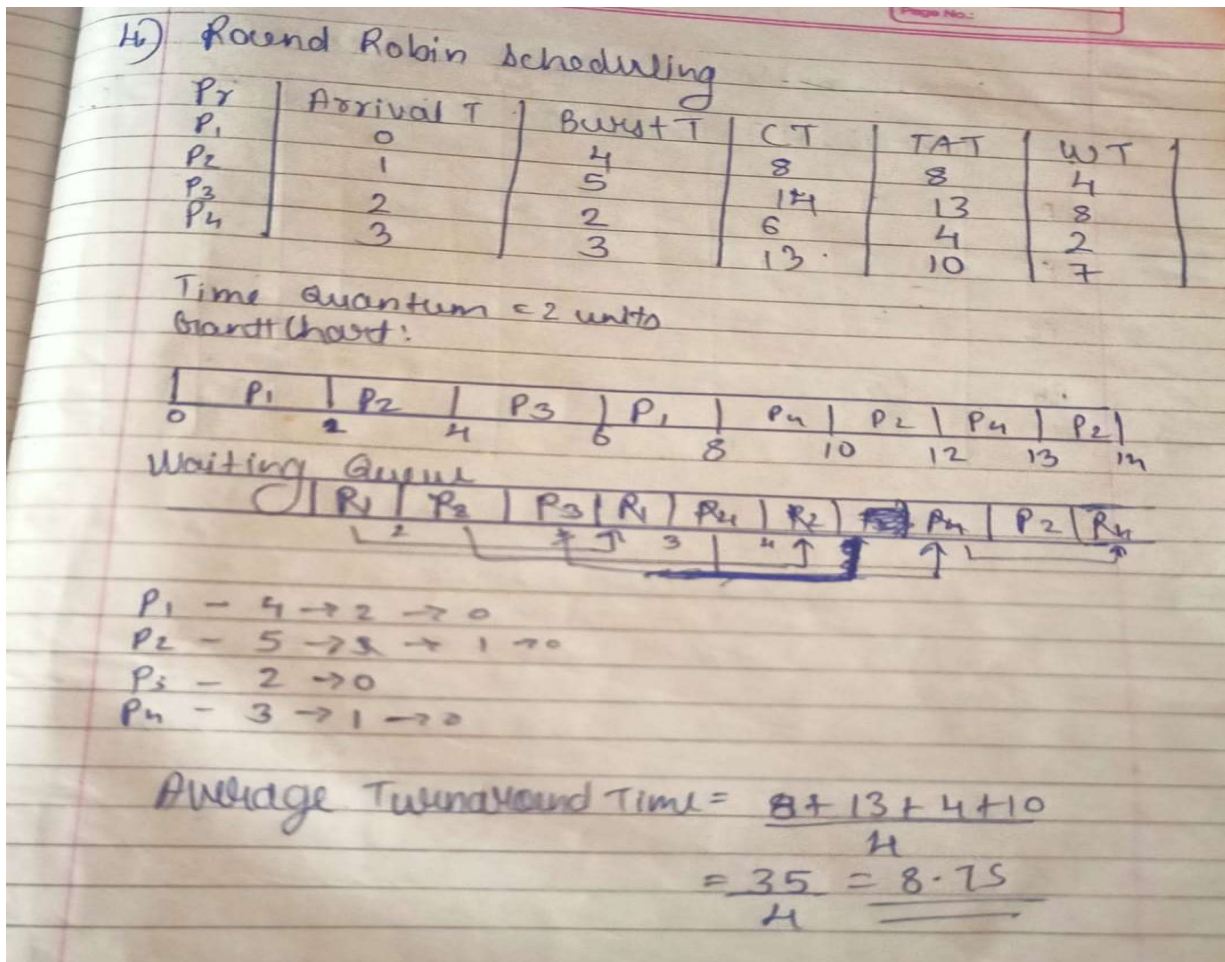
P₁ = 6 → 5

P₂ = 4 → 0

P₃ = 7

P₄ = 2 → 0

$$\text{Average Waiting Time: } \frac{6 + 0 + 10 + 2}{4} = \frac{18}{4} = 4.5$$



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?

Answer:

When we use fork() system call to create a child process the OS creates a new child process which is copy of the parent process .although the child process is copy of parent process , it has different memory space allocated to it .

Before 'fork()':

Parent process: x=5

After 'fork()':

Parent process : $x=5$

Child process : $x=5$

Now Both process apply increment operation.

Then,

$x++ \rightarrow x=6$ (parent process)

$x++ \rightarrow x=6$ (child process)

Therefore the final value of both parent and child processes is $x=6$.

Thus , changes made in one process do not affect the other.