Name: Shubham C Thete

Assignment 2

Part A

What will the following commands do?

- echo "Hello, World!"

When we write this command on terminal it get print the string Hello World! On

Terminal

- name="Productive"

The string “Productive” is assign to the variable name

- touch file.txt

It will create the file name file.txt in the current directory touch command use to create

The file

- ls -a

It will list the all files and directories that present in that current directory

- rm file.txt

rm Command use for deleting the file, it will delete the file.txt

- cp file1.txt file2.txt

cp command use for the copy of content and from above command it will copy

the content of file1.txt and paste to file2.txt

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

mv command is used rename or move a file. In the above example, mv command moves the file (file.txt) into the specified directory (/path/to/directory/).

- chmod 755 script.sh

The above command gives read, write and execute permissions to the owner and read and execute permissions to group and other users respectively to script.sh file

- grep "pattern" file.txt

The grep command use to find particular word in given text file in that given word “pattern” it will search the word in file if matches it will print

- kill PID

The kill command use to terminate the process when the given process id is given

But in the above command it will give the error because there is no mention of process id

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

In the above command making directory name mydir and change the directory to the mydir In that directory it create the file name file.txt by using touch command and also prompt the message Hello World! By using echo command the message redirect to the file.txt means it save in the file.txt and last show the content of the file by using the cat command the && operator use here because we can use multiple command in single command

- ls -l | grep ".txt"

The above command uses piping to combine the output of both ls and grep command. ls -l is used to display the contents of current directory with details and grep “.txt” command is used to display all the files containing .txt pattern in their name.

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

The above command uses piping to combine the output of cat sort and uniq commands. First command i.e. cat command is used to display the contents of file1.txt followed by contents of file2.txt. sort command is used to perform alphanumeric sort on the result of cat command. Contents of file1.txt and file2.txt are sorted separately in the result. oFinally, uniq command is use to display only distinct lines in the result.

- ls -l | grep "^d"

ls -l command show the listing in details and in followed by pipelining grep command only show the directory because it started with d

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

Here grep command is used to recursively search for given pattern “pattern” in the directory /path/to/directory, provided that such directory exists in first place. The output will display the lines containing the “pattern” pattern in it.

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

cat command will show the contents of file1.txt and followed by file2.txt ans by pipelinling it use the output that and sort the contents according to the alphanumeric and at last uniq command is use it only show the uniq command but here -d is use that means all duplicate entry it not show

- chmod 644 file.txt

The above command assigns read and write permissions to owner of the file file.txt and read permission to group users and other users respectively.

- cp -r source_directory destination_directory

The above command is used to copy the source_directory to destination directory. This is done by using -r option so that all files in source_directory are copied recursively

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

find command is used for searching the files and directories. Given command searches /path/to/search directory and its subdirectories for any file ending with .txt pattern.

- chmod u+x file.txt

This command is used to grant execute permissions for file.txt file to the user(owner) of the file.

- echo $PATH

This command displays the value of system environment variable that stores directories where executable programs are located.

Part B

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 to change directory/folder

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.
   False

Identify the Incorrect Commands:

1. chmodx is used to change file permissions.
   chmod command is used to change file permissions.

2. cpy is used to copy files and directories
   cp command is used to copy files and directories.

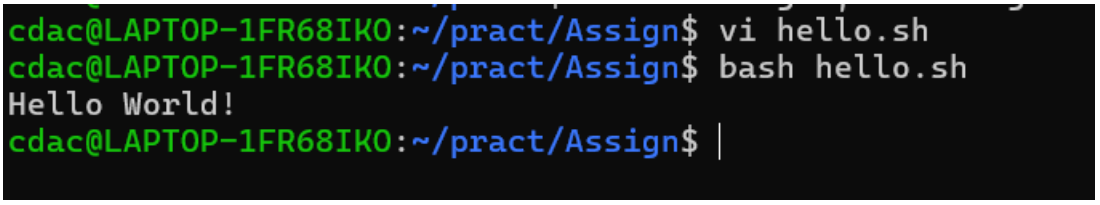
3. mkfile is used to create a new file.
   touch command is used to create a new file. mkdir command is used to create a new directory.

4. catx is used to concatenate files.
   cat command is used to concatenate files.

5. rn is used to rename files
   mv command is used to rename files when 2 files names are passed as arguments

PART_C

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



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

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ name="CDAC Mumbai"
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ name
Command 'name' not found, did you mean:
  command 'mame' from snap mame (mame0270)
  command 'nam' from deb nam (1.15-6)
  command 'namei' from deb util-linux (2.39.3-9ubuntu6.1)
  command 'lame' from deb lame (3.100-6)
  command 'nvme' from deb nvme-cli (2.8-1ubuntu0.1)
  command 'nama' from deb nama (1.216-2)
  command 'named' from deb bind9 (1:9.18.28-0ubuntu0.24.04.1)
  command 'uname' from deb coreutils (9.4-2ubuntu2)
  command 'mame' from deb mame (0.261+dfsg.1-1)
See 'snap info <snapname>' for additional versions.
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ ^C
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ $name
CDAC: command not found
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ echo name
name
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ echo $nam

cdac@LAPTOP-1FR68IKO:~/pract/Assign$ echo $name
CDAC Mumbai
```

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

```
3
3 is the number enter by you
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat number.sh
echo "Enter a number"
read num

echo $num is the number enter by you
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash number.sh
Enter a number
4
4 is the number enter by you
cdac@LAPTOP-1FR68IKO:~/pract/Assign$
```

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

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi add.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat add.sh
echo "Enter num1"
read num1
echo "Enter num2"
read num2

c=`expr $num1 + $num2`
echo "Addition of $num1 and $num2 is = $c"
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash add.sh
Enter num1
3
Enter num2
4
Addition of 3 and 4 is = 7
```

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

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi evenOdd.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash evenOdd.sh
Enter a Number:
3
3 is Odd
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash evenOdd.sh
Enter a Number:
6
6 is Even
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat evenOdd.sh
echo "Enter a Number:"
read num

if ((num%2==0))
then
        echo "$num is Even"
else
        echo "$num is Odd"

fi
cdac@LAPTOP-1FR68IKO:~/pract/Assign$
```

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

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi forloop.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash forloop.sh
1
2
3
4
5
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat forloop.sh
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.

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi whilelopp.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat whilelopp.sh
a=1
while [ $a -le 5 ]
do
        echo $a
        a=`expr $a + 1`
done

cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash whilelopp.sh
1
2
3
4
5
```

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-1FR68IKO:~/pract/Assign$ vi q-8.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat q-8.sh
if [ -e file.txt ]
then
        echo "File exists"
else
        echo "File doesn't exist"
fi
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-8.sh
File doesn't exist
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ touch file.txt
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-8.sh
File exists
```

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-1FR68IKO:~/pract/Assign$ vi q-9.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat q-9.sh
echo "Enter Number"
read num

if [ $num -gt 10 ]
then
        echo "$num is greter than 10"
else
        if [ $num -eq 10 ]
        then
                echo "$num is equal to 10"
        else
                echo "$num is smaller than 10"
        fi
fi
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-9.sh
Enter Number
4
4 is smaller than 10
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-9.sh
Enter Number
`9
q-9.sh: line 4: [: `9: integer expression expected
q-9.sh: line 8: [: `9: integer expression expected
`9 is smaller than 10
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-9.sh
Enter Number
10
10 is equal to 10
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-9.sh
Enter Number
111
111 is greter than 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.

```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi q10.sh
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat q10.sh
for i in {1..5}
do
        for j in {1..10}
        do
                a=$((i*j))
                echo  -n $a  ' '
        done
        echo -e
done
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q10.sh
1  2  3  4  5  6  7  8  9  10
2  4  6  8  10  12  14  16  18  20
3  6  9  12  15  18  21  24  27  30
4  8  12  16  20  24  28  32  36  40
5  10  15  20  25  30  35  40  45  50
```

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.

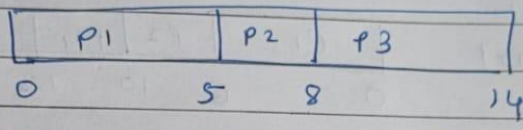
```
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ vi q-11.sh

cdac@LAPTOP-1FR68IKO:~/pract/Assign$ cat q-11.sh
while [ true ]
do
        echo "Enter number"
        read num
        if [ $num -lt 0 ]
        then
                break
        fi
done

echo "You enter negative number program end"
cdac@LAPTOP-1FR68IKO:~/pract/Assign$ bash q-11.sh
Enter number
2
Enter number
3
Enter number
4
Enter number
-1
You enter negative number program end
```

Part E

Q.1 FCFS

process	Arrival Time	Burst time	waiting time
P1	0	5	0
P2	1	3	4
P3	2	6	6

Gantt chart:

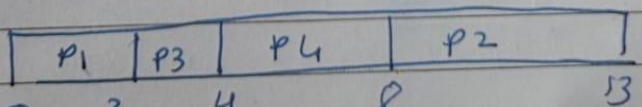


Avg. waiting time = (0+4+6)/3
= 10/3
= 3.33

Q.2] SJF (non-preemptive)

process	Arrival time	Burst time	waiting time	TAT
P1	0	3	0	3
P2	1	5	7	12
P3	2	1	1	2
P4	3	4	1	5

Gantt chart

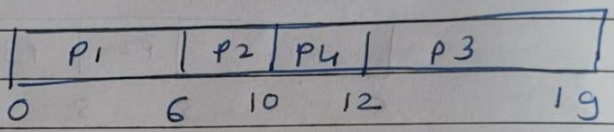


Avg. TAT = $\frac{3+12+2+5}{4}$
= 22/4
= 5.5

Q.3 Algorithm priority scheduling (non-preemptive)

process	Arrival time	Burst time	priority	waiting time
P1	0	6	3	0
P2	1	4	1	5
P3	2	7	4	10
P4	3	2	2	7

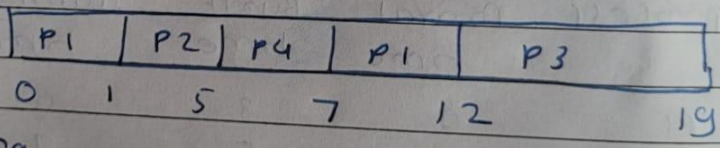
Gantt chart



Avg. WT = 22/4
= 5.5

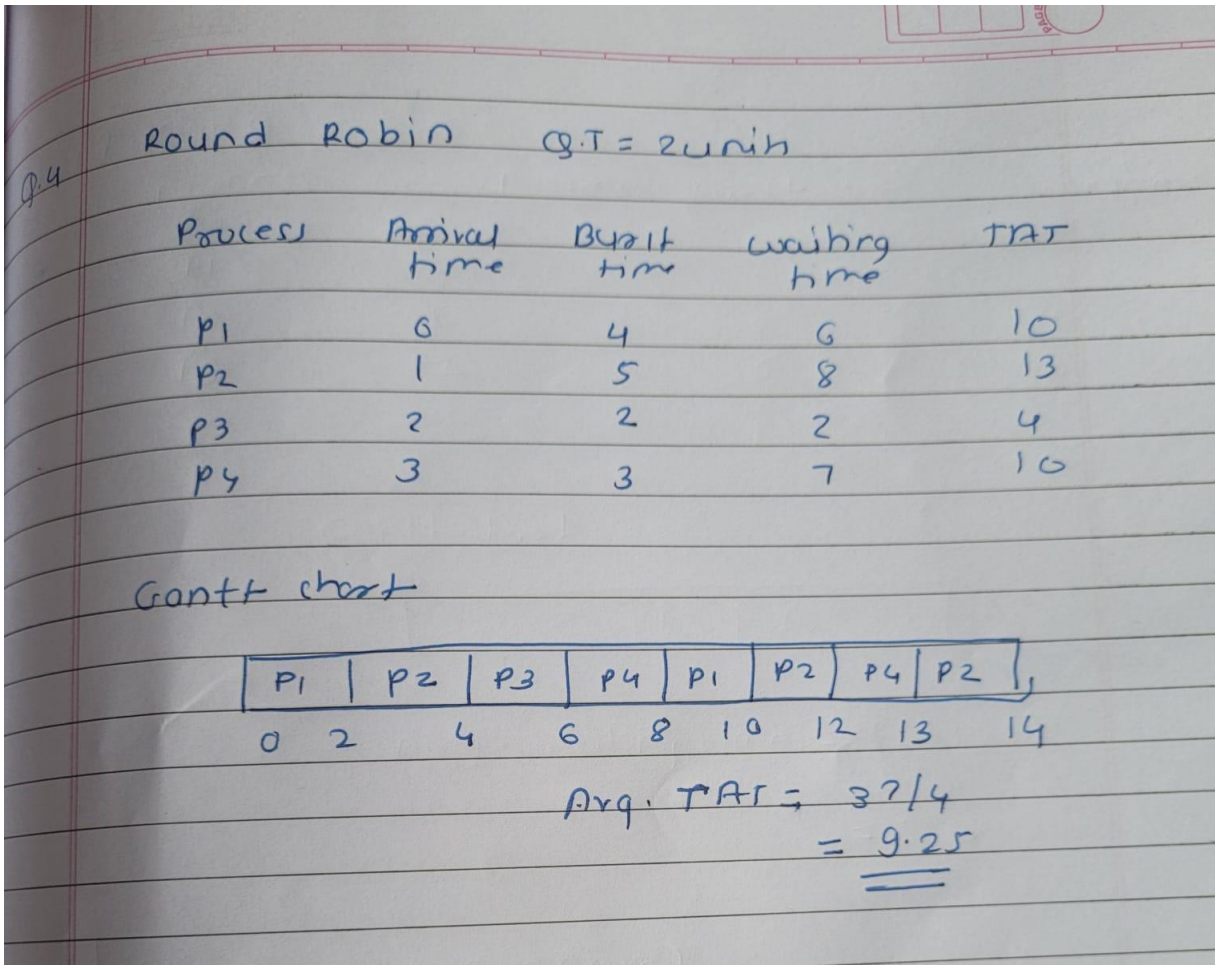
for preemptive.

Gantt chart (preemptive):



waiting time
6
0
2
10

Avg. WT = 18/4
= 4.5



Q.4

Round Robin Q.T = 2 unit

Process	Arrival time	Burst time	waiting time	TAT
P1	0	4	6	10
P2	1	5	8	13
P3	2	2	2	4
P4	3	3	7	10

Gantt chart

P1	P2	P3	P4	P1	P2	P4	P2

0 2 4 6 8 10 12 13 14

Avg. 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?

```
cdac@LAPTOP-1FR68IKO:~/pract$ cat sh.c
#include <stdio.h>

int main(){
        int x=5;
        fork();
        x=x+1;
        printf("%d\n",x);
        return 0;
}
cdac@LAPTOP-1FR68IKO:~/pract$ gcc sh.c
sh.c: In function ‘main’:
sh.c:5:9: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
    5 |         fork();
      |         ^~~~
cdac@LAPTOP-1FR68IKO:~/pract$ ./a.out
6
6
```

The value also become 6 when we increment the value

Each process then increments x by 1, so both the parent and child have x = 6, but in their own separate memory. In parent process, x=6. In child process, x=6