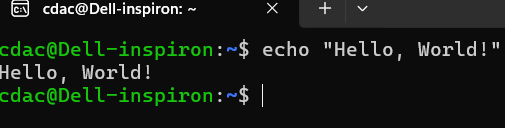
**Concepts of Operating System Assignment 2**

**Part A**

**What will the following commands do?**

1. echo "Hello, World!"

echo: This command is used to output text to the terminal.

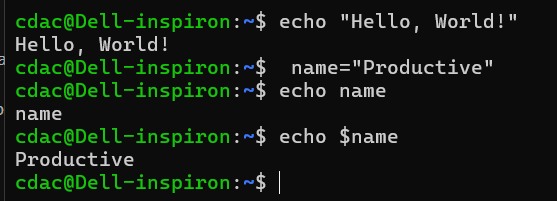


1. name="Productive"

name: This is the name of the variable.

=: This is the assignment operator, which assigns the value on the right to the variable on the left.

"Productive": This is the value assigned to the variable name.



1. touch file.txt

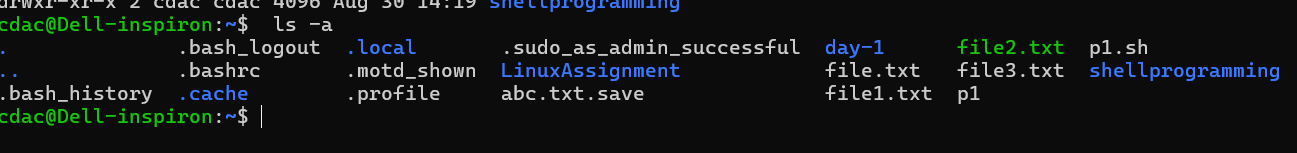
touch: This command is used to create a new, empty file or to change the access and modification times of an existing file.



1. ls -a

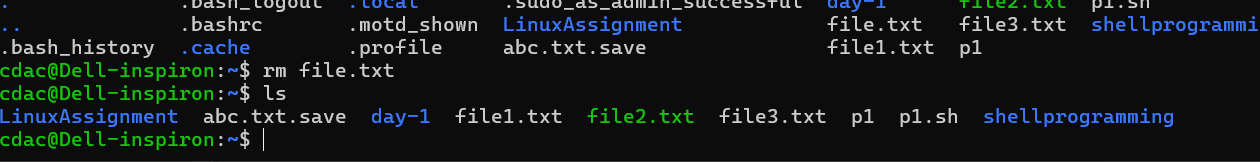
ls: This command lists files and directories in the current working directory.

-a: The -a option stands for "all," and it includes hidden files and directories in the listing.



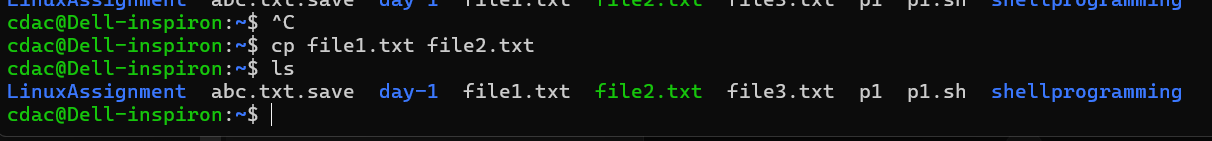
5)rm file.txt

The rm command is used to remove files or directories. When you run rm file.txt, it deletes the file named file.txt from the current directory.



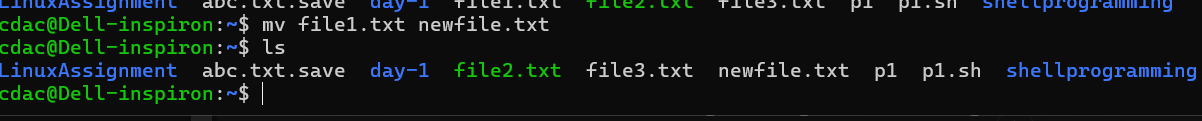
6) cp file1.txt file2.txt

it copies the contents of file1.txt into a new file named file2.txt



7)mv file.txt /path/to/directory/

It can also be used to rename files or directories. When you use mv file.txt /path/to/directory/, it moves the file file.txt to the specified directory.

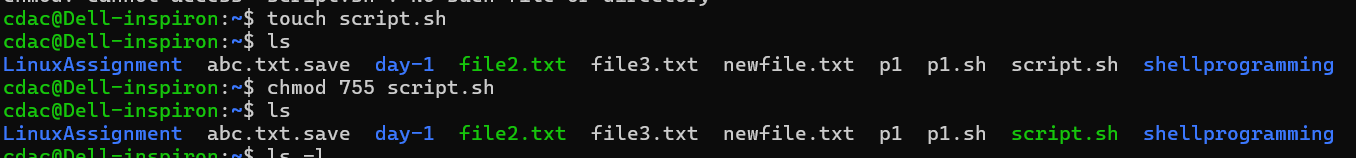


8)chmod 755 script.sh

chmod: Stands for "change mode." This command is used to change the access permissions of a file or directory.

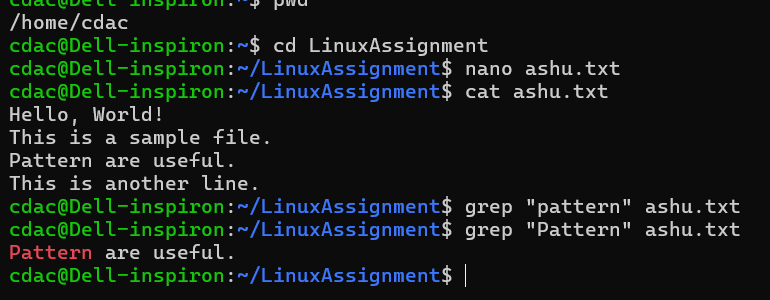
755: This is a permission setting that defines who can read, write, and execute the file.

script.sh: The name of the file whose permissions you are changing.



9) grep "pattern" file.txt

The grep command is used to search for a specific pattern within a file or multiple files



10) kill PID

It is used to terminate processes manually. *kill* command sends a signal to a process that terminates the process.

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

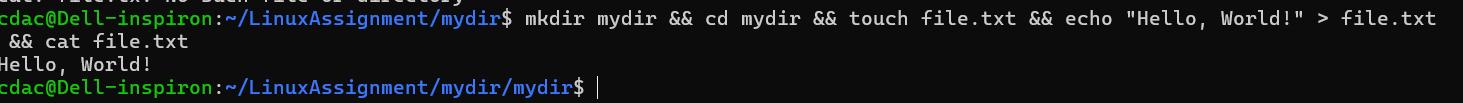
mkdir mydir: Creates the directory mydir.

cd mydir: Changes to the mydir directory.

touch file.txt: Creates an empty file.txt.

echo "Hello, World!" > file.txt: Writes "Hello, World!" to file.txt.

cat file.txt: Displays the contents of file.txt.

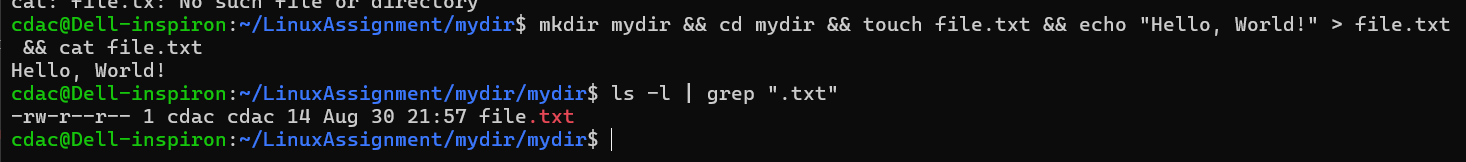


12) ls -l | grep ".txt"

ls -l: Lists files and directories in the current directory in long format, which includes permissions, number of links, owner, group, size, modification date, and name.

| pipe: Passes the output of the ls -l command to the next command grep)

grep ".txt": Filters the output to include only lines that contain .txt. This will show files with the .txt extension.



13) cat file1.txt file2.txt | sort | uniq

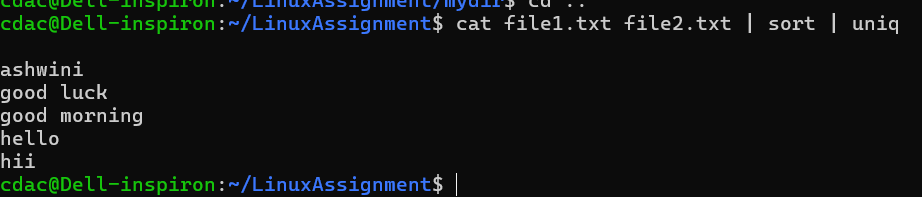
cat file1.txt file2.txt: Concatenates and displays the contents of file1.txt and file2.txt.

| (pipe): Passes the output of the cat command to the next command sort.

sort: Sorts the lines of the concatenated output in ascending order.

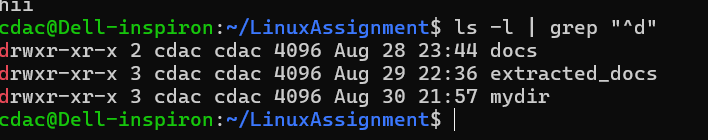
| (pipe): Passes the sorted output to the next command (uniq).

uniq: Removes duplicate lines from the sorted output, leaving only unique lines.



14) ls -l | grep "^d"

The command ls -l | grep "^d" lists only the directories in the current directory.

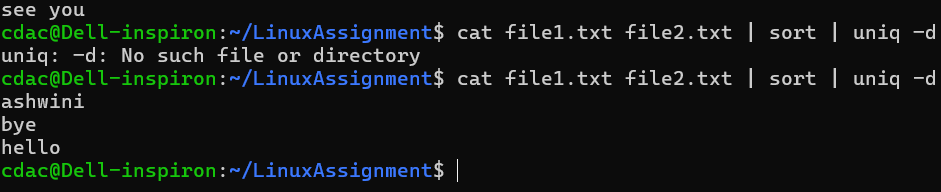


15) grep -r "pattern" /path/to/directory/

The command grep -r "pattern" /path/to/directory/ is used for searching within files for a specific pattern recursively in a directory.

16) cat file1.txt file2.txt | sort | uniq –d

The command cat file1.txt file2.txt | sort | uniq -d is used to find and display lines that are common between the two files

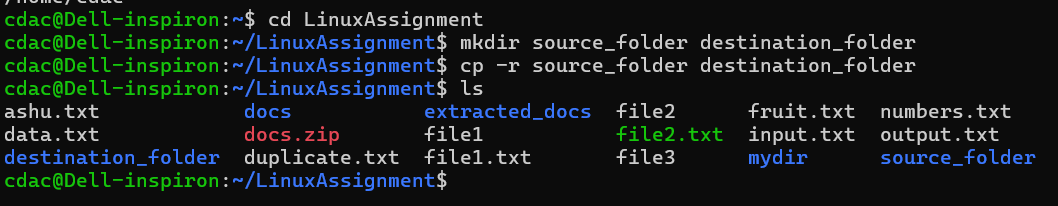


17) chmod 644 file.txt

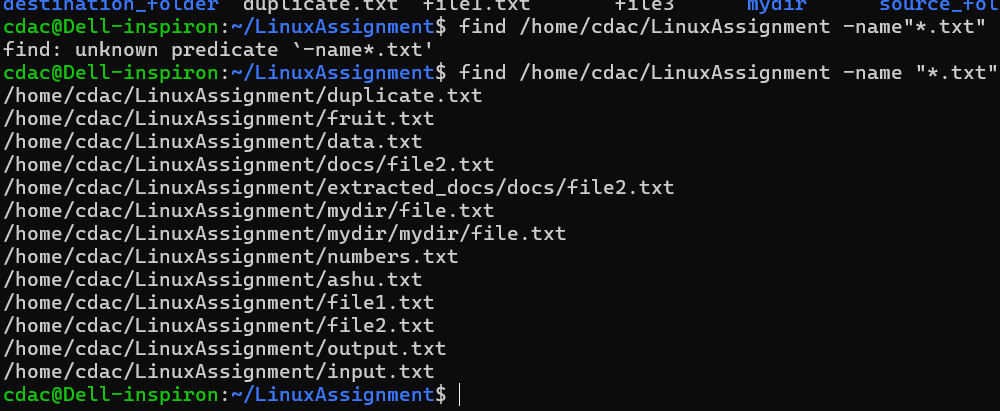
The command chmod 644 file.txt is used to change the permissions of the file file.txt

15) cp -r source\_directory destination\_directory

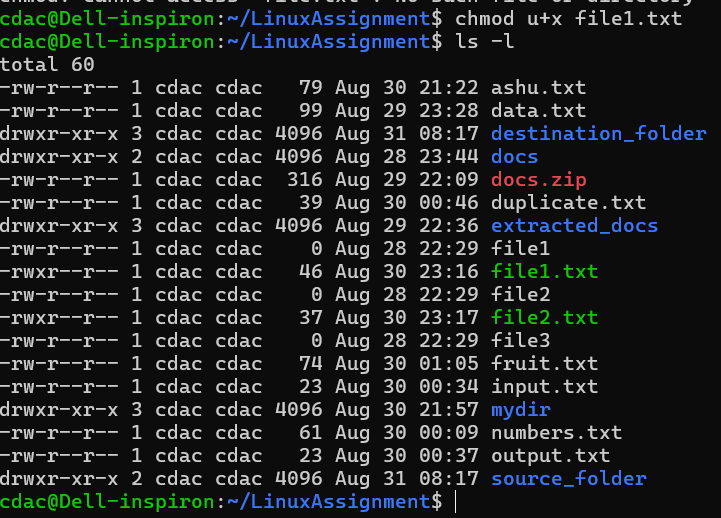
The command cp -r source\_directory destination\_directory is used to copy a directory and its contents recursively from one location to another.



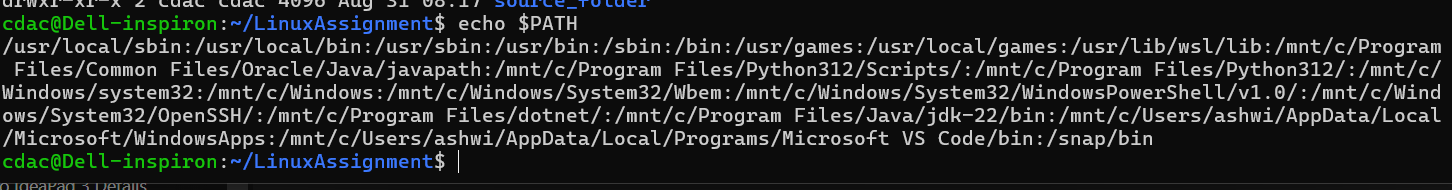
16) find /path/to/search -name "\*.txt"



17)chmod u+x file.txt



18) echo $PATH



**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

1. cd is used to copy files and directories.

🡪False ; cd is used to change the current directory, not to copy files. The correct command for copying is cp.

1. pwd stands for "print working directory" and displays the current directory.

🡪True

1. grep is used to search for patterns in files.

🡪True

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

🡪True

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

🡪True

1. rm -rf file.txt deletes a file forcefully without confirmation

🡪True

**Identify the Incorrect Commands:**

1. chmodx is used to change file permissions.

🡪chmodx is incorrect.The correct command to change file permissions is chmod.

2. cpy is used to copy files and directories.

🡪cpy is incorrect.The correct command to copy files and directories is cp.

3.mkfile is used to create a new file.

🡪 mkfile is incorrect command .in linux to create new file touch command is use.

4.catx is used to concatenate files.

🡪 This command is incorrect. The correct command for concatenating files is cat

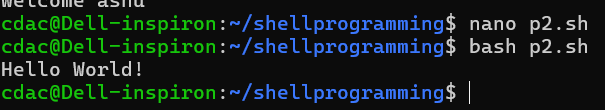
5. rn is used to rename files.

🡪 This command is incorrect. The correct command to rename files is mv.

**Part C**

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

echo "Hello World!"

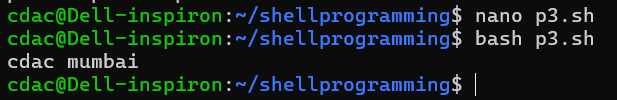


Question 2: Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

#!/bin/bash

name="cdac mumbai"

echo $name

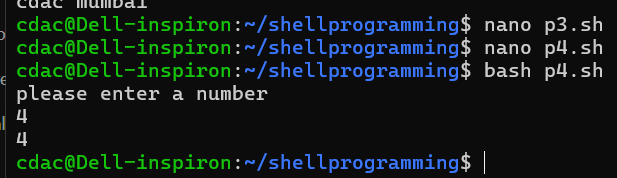


Question 3: Write a shell script that takes a number as input from the user and prints it.

echo "please enter a number"

read number

echo $number



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

echo enter a number

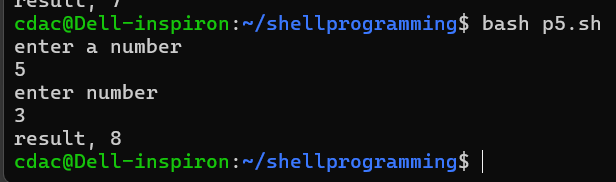
read Num1

echo enter number

read Num2

result=`expr $Num1 + $Num2`

echo result, $result



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

echo enter a number

read number

if [ $((number%2)) -eq 0 ]

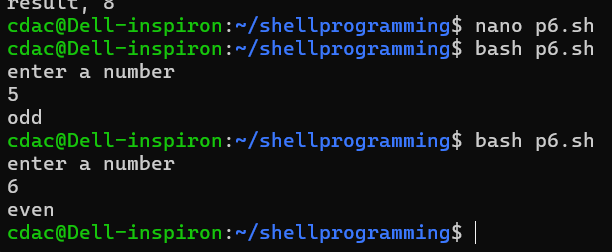
then

echo "even"

else

echo "odd"

fi



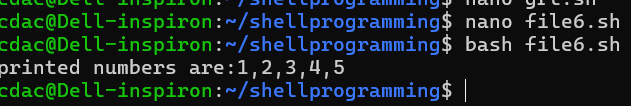
Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.

for i in 1,2,3,4,5

do

echo "printed numbers are:$i"

done



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

count=1

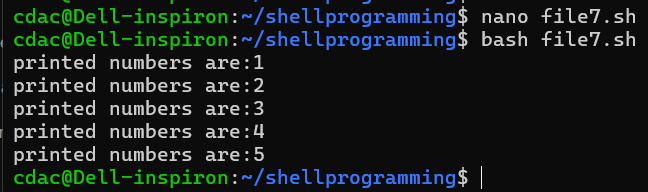
while [ $count -le 5 ]

do

echo "printed numbers are:$count"

count=$((count+1))

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" ];

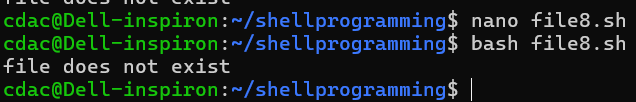
then

echo "file exist"

else

echo "file does not exist"

fi



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.

echo enter a number

read number

if [ $number -gt 10 ]

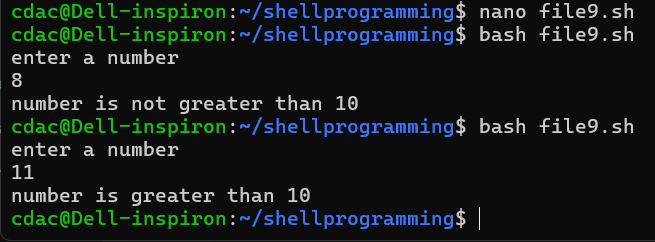
then

echo number is greater than 10

else

echo number 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.

#!/bin/bash

for i in {1..5}

do

for j in {1..5}

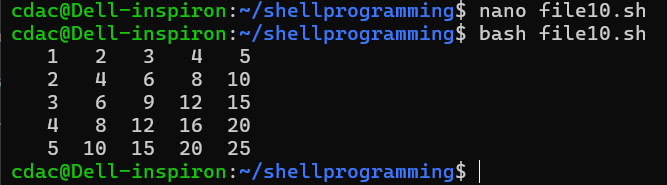
do

printf "%4d" $((i \* j))

done

echo

done



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.

#!/bin/bash

while true

do

echo "Enter a number (negative number to quit):"

read number

if [ $number -lt 0 ]

then

break

else

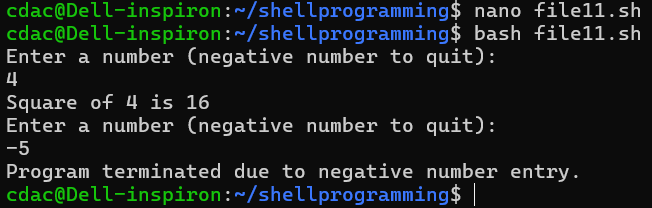
square=$((number \* number))

echo "Square of $number is $square"

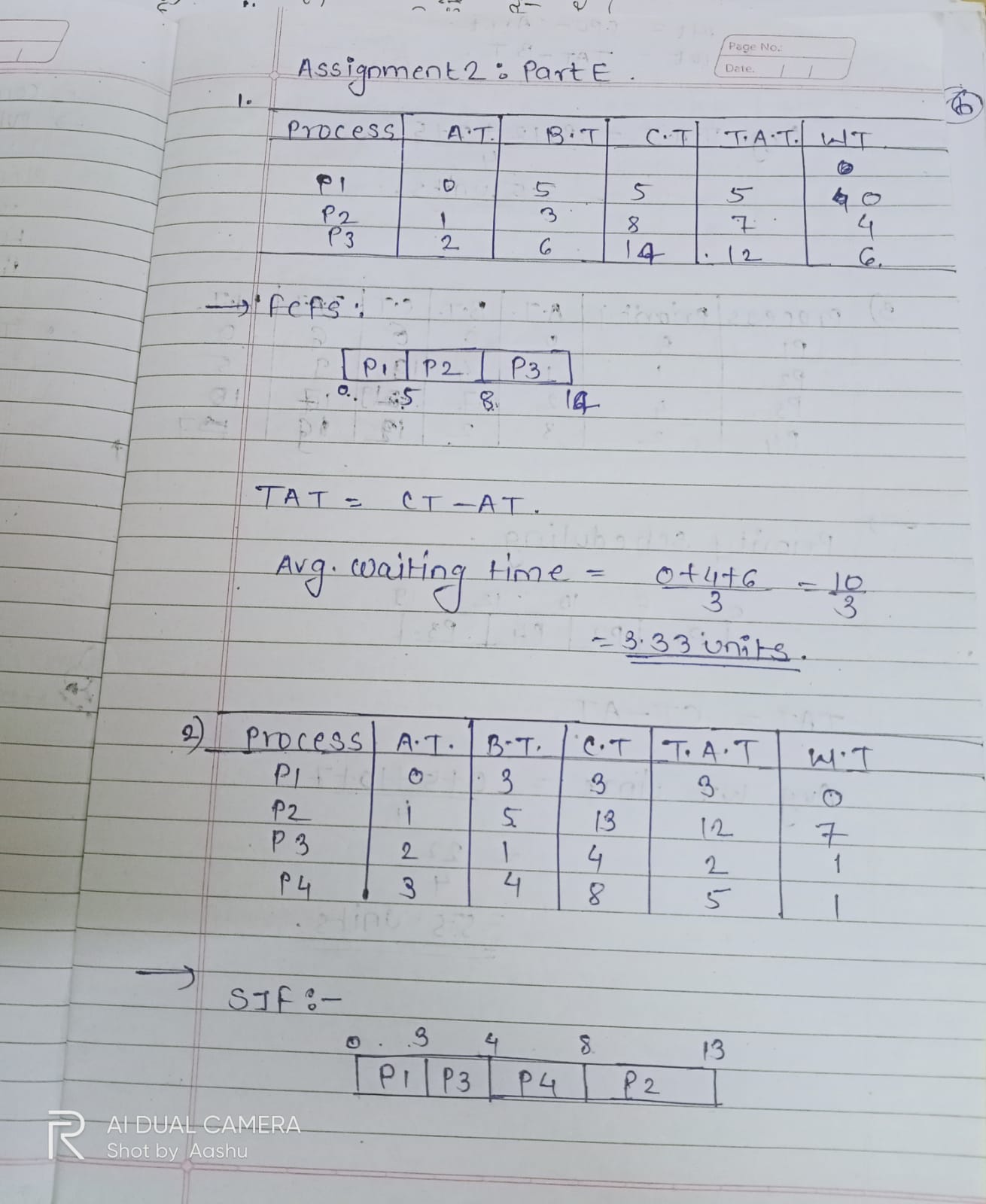
fi

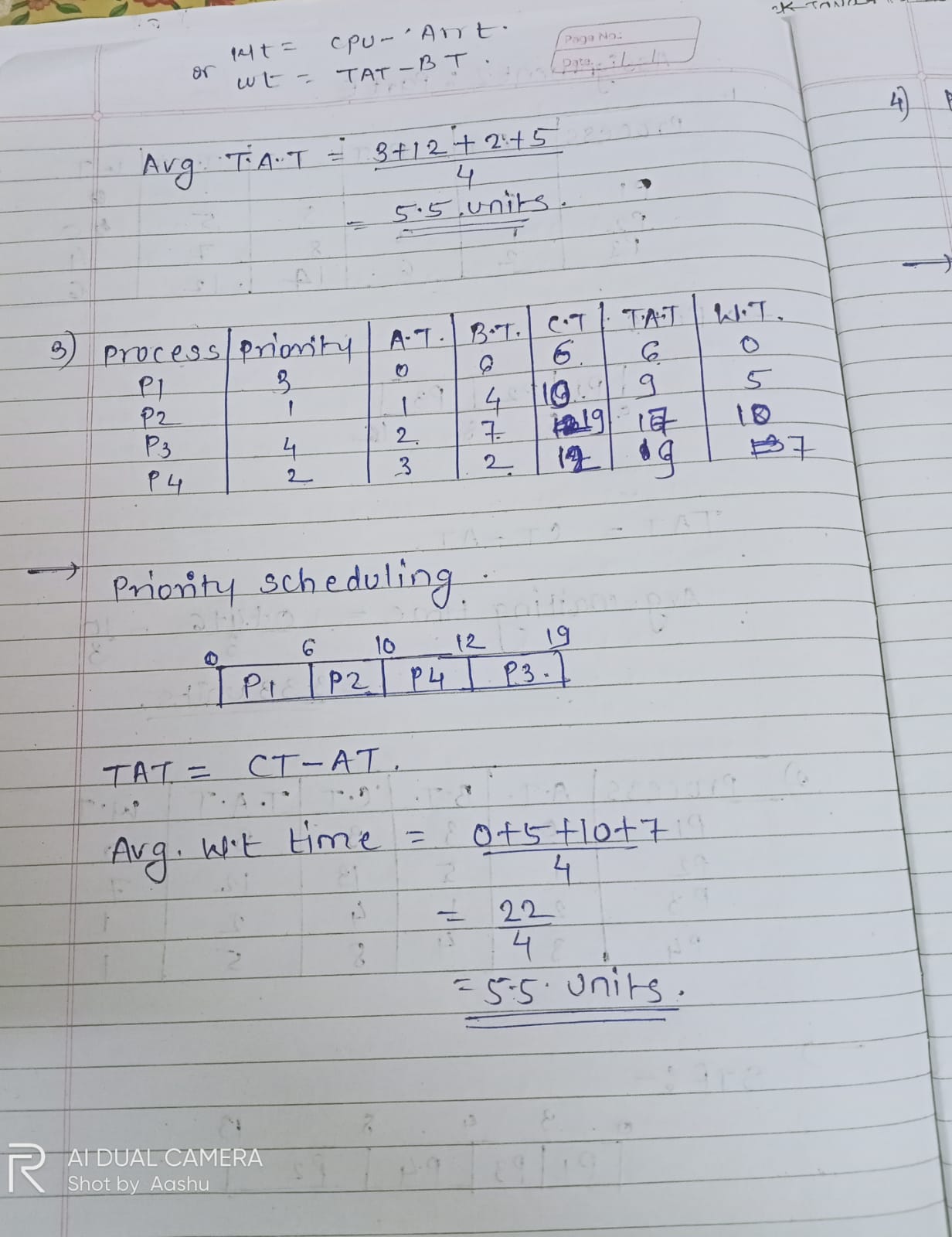
done

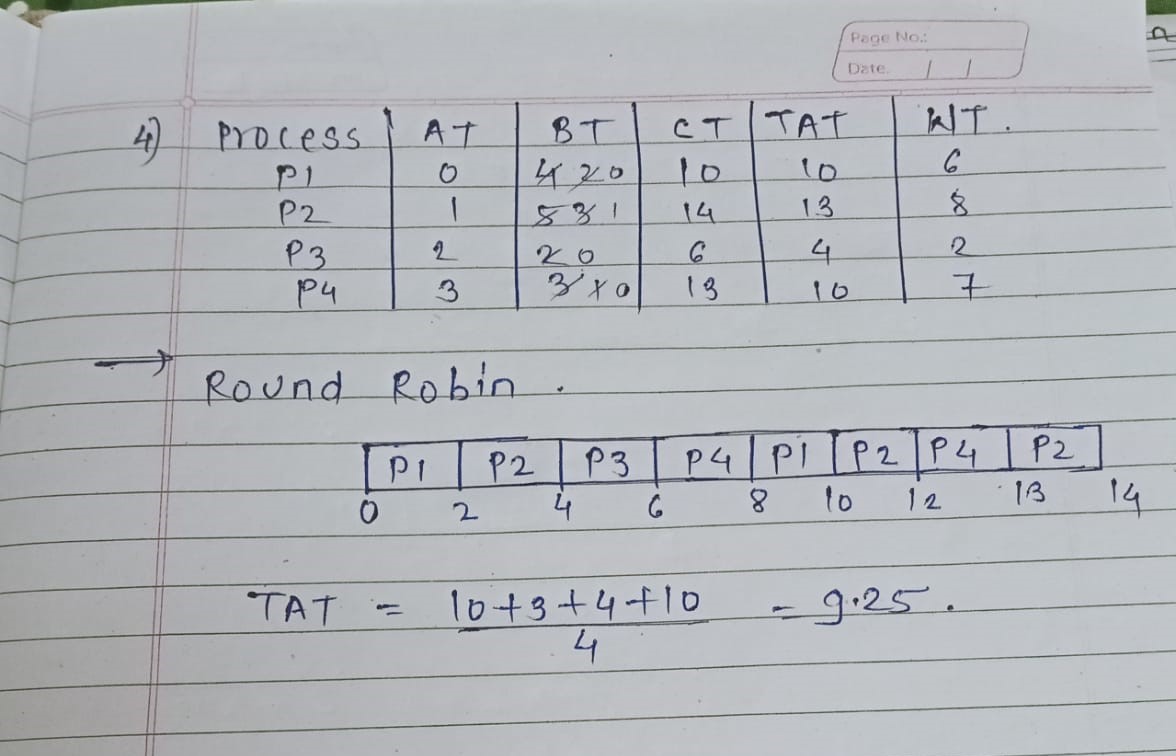
echo "Program terminated due to negative number entry."



**Part E**

****

****

****