**Flowchart:**

1. You are working in a logistics company responsible for delivering packages. Design a flowchart to manage the process of receiving, sorting, and delivering packages. Include decision structures for handling fragile items and urgent deliveries.

Receive package

Register package in the system

Is the package fragile?

yes

Special handling

no

Is the package urgent?

yes

Priorities sorting and delivery

no

A1

A1

Sort package on regular basis

Load packages onto delivery vehicles

Deliver vehicle

Re-attempt delivery

Is the Package Delivered Successfully?

no

yes

Mark as delivered in the system

2. Imagine you are automating the process of a vending machine. Create a flowchart that includes decision points for user input, selecting products, accepting payment, and dispensing the correct item. Include error-handling for invalid inputs and insufficient funds.

read

Payment received valid input  
  
valid

va

Select product

price

calc

Payment received

**no no**

Valid input

**yes**

**yes**

print

**Pseudocode:**

1. Write pseudocode to find the smallest number among three given variables. Implement a decision-making structure to compare the variables.

Start

Input num1

Input num2

Input num3

IF num1<num2 and num1<num3 THEN

Print “The smallest number is num1”

ELSE IF num2<num1 and num2<num3 THEN

Print” The smallest number is num2”

ELSE num3<num1 and num3<num2 THEN

Print” The smallest number is num3”

End

1. Develop pseudocode for a basic calculator that performs multiplication and division. The pseudocode should prompt the user for two numbers and an operator, then display the result of the operation.

Start

Input n1,n2

SET product to 0

SET division to 0

SET product to n1\*n2

SET division to n1/n2

End

**Algorithms:**

1. Write an algorithm to determine whether a number is a prime number. The algorithm should iterate through possible divisors and determine if the number has any divisors other than 1 and itself.

Step1: Ask the user to input a number greater than 1.(n)

Step2: create a for loop to (i=2; i\*i<=n;i++)

Step3: if n%i is equal to 0 display it is not a prime number

Step4: if n%i is not equal to zero display it is a prime number.

1. Create an algorithm that asks the user for a day number (1-365) and outputs the corresponding day of the week, assuming that January 1st is a Monday.

Step1: Ask user to enter a day number from (1-365).

Step2: Adjust the “day\_number” by subtracting 1 to make it zero-based.

Step3: Calculate the day of the week using the formula: day\_of\_week = (day\_number - 1) % 7.

Step4: Create a list that maps the numbers 0 to 6 to their corresponding day names:

days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"].

step5: display days[day\_of\_week].

step6: print day name.

1. Develop an algorithm for a program that takes two numbers as input and finds the Greatest Common Divisor (GCD) of the two numbers using the Euclidean algorithm.

Step1: Ask user to enter number1.

Step2: Ask user to enter number2.

Step3: Number2 is not zero.

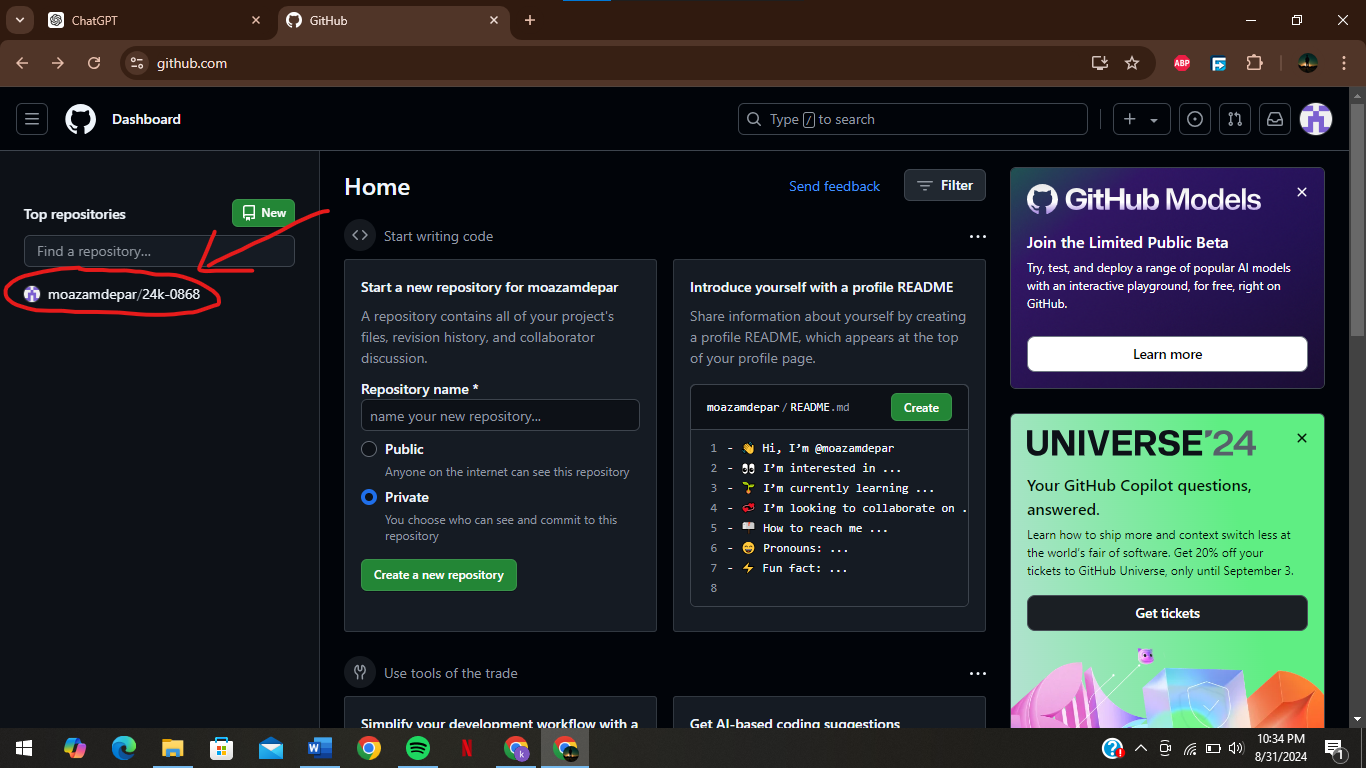
Step4: Calculate the reminder ‘r’ of ‘number1’ divided by ‘number2’

Step5: Assign ‘n2’ to ‘n1’ and ‘r’ to ‘n2’.

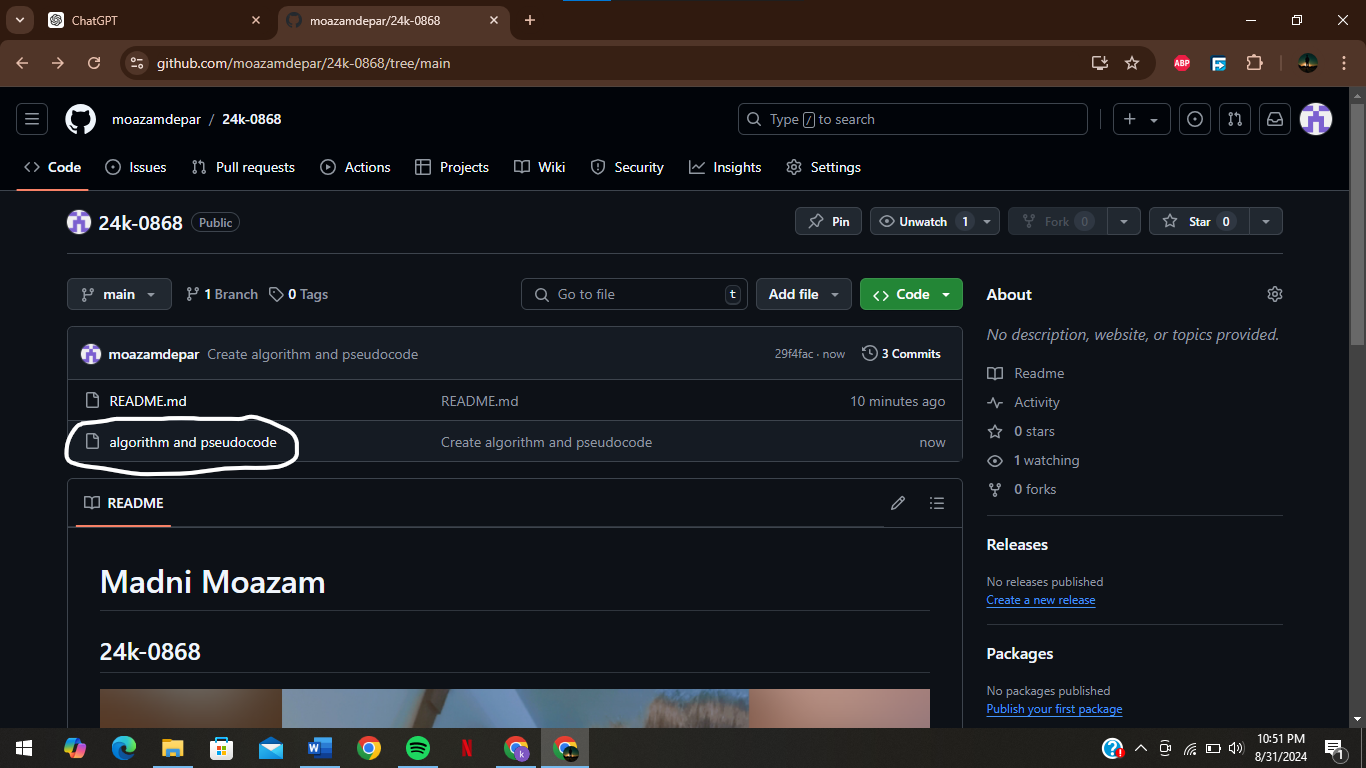
Step6: When ‘n2’ become zero, n1 contains the GCD.

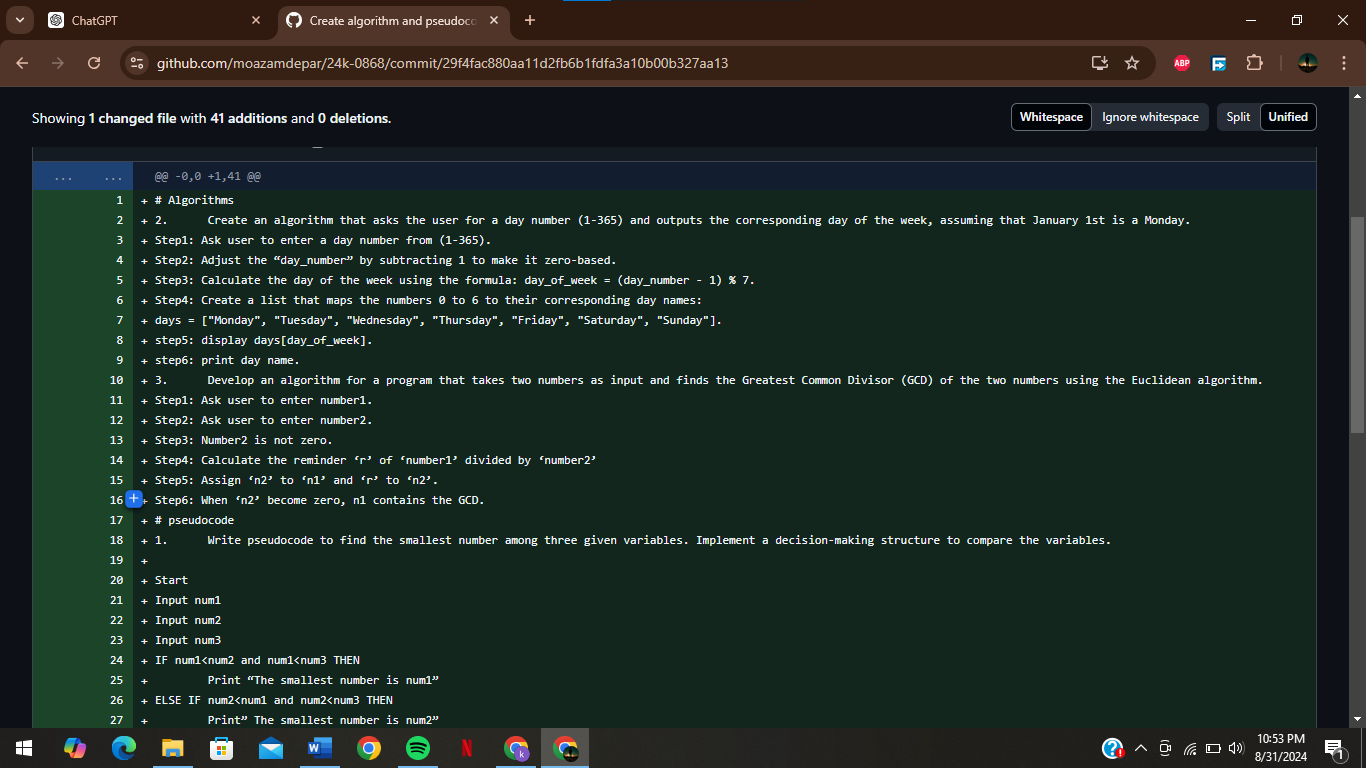
**Github:**

1. Create your repository with your roll number being your repo name.



1. Upload the algorithms and pseudo codes in your repository.





1. Create a small intro about yourself in the readme file with pictures and bullet points.

