**Batch: C4-1 Roll No.: 16010123217**

**Experiment / assignment / tutorial No. 6**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE:** Write a program in C to implement user defined functions |

**AIM:**

1. Write a program to find the GCD of two numbers using recursion.
2. Write a program to find the LCM of two numbers by using a) above.

\_\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

Design modular programs using functions and the use of structure and union (CO4)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.
2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.
3. Introduction to programming and problem solving , G. Michael Schneider ,Wiley India edition.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Problem Definition:**

1. The program finds the GCD of two numbers using recursion

Example:

|  |  |
| --- | --- |
| Test case 1:  Input:  24,28  Output:  GCD: 4 | Test case 2:  Input:  24,25  Output:  GCD: 1 |

1. The program finds the LCM of two numbers using GCD.

Example:

|  |  |
| --- | --- |
| Test case 1:  Input:  6,12  Output:  LCM: 12 | Test case 2:  Input:  6,7  Output:  LCM: 42 |

**Algorithm:**

**1) Algorithm for GCD**

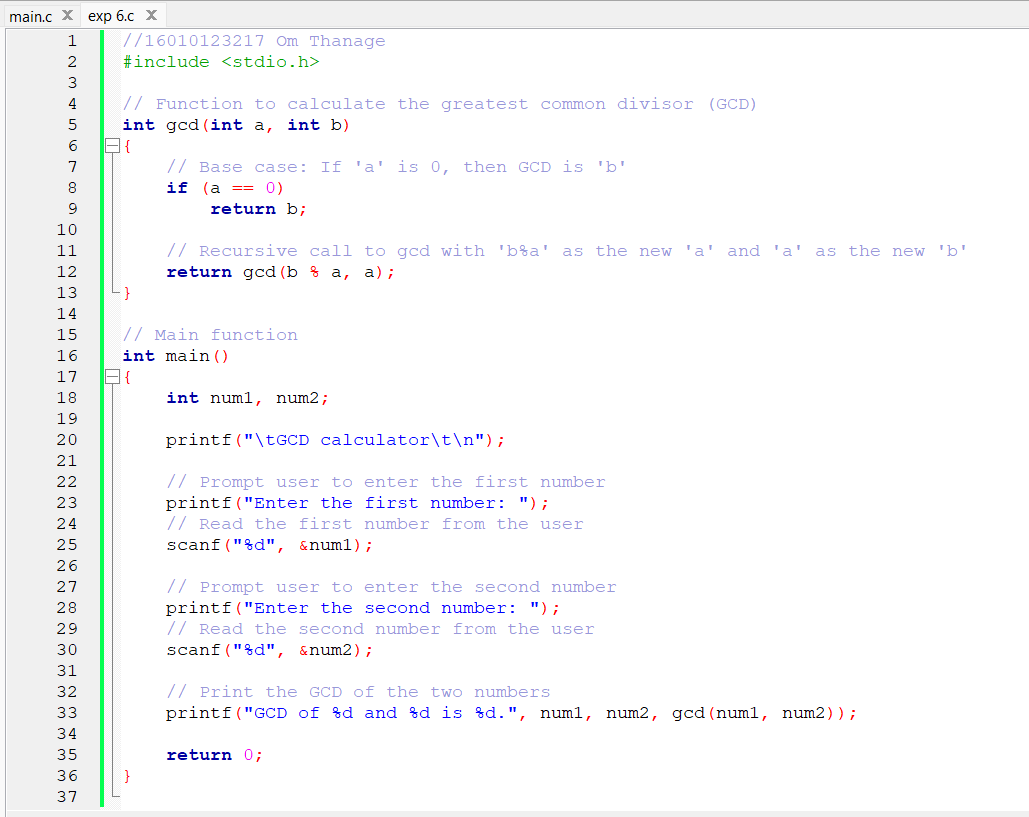
1. Start.
2. Declare the variables **num1** and **num2** to store the input numbers.
3. Display a header indicating that this program is a GCD calculator.
4. Prompt the user to enter the first number.
5. Read the first number from the user and store it in the variable **num1**.
6. Prompt the user to enter the second number.
7. Read the second number from the user and store it in the variable **num2**.
8. Calculate the GCD of **num1** and **num2** using the **gcd** function.
9. Print the result, displaying the GCD of **num1** and **num2**.
10. End.

**2)** **Algorithm for LCM**

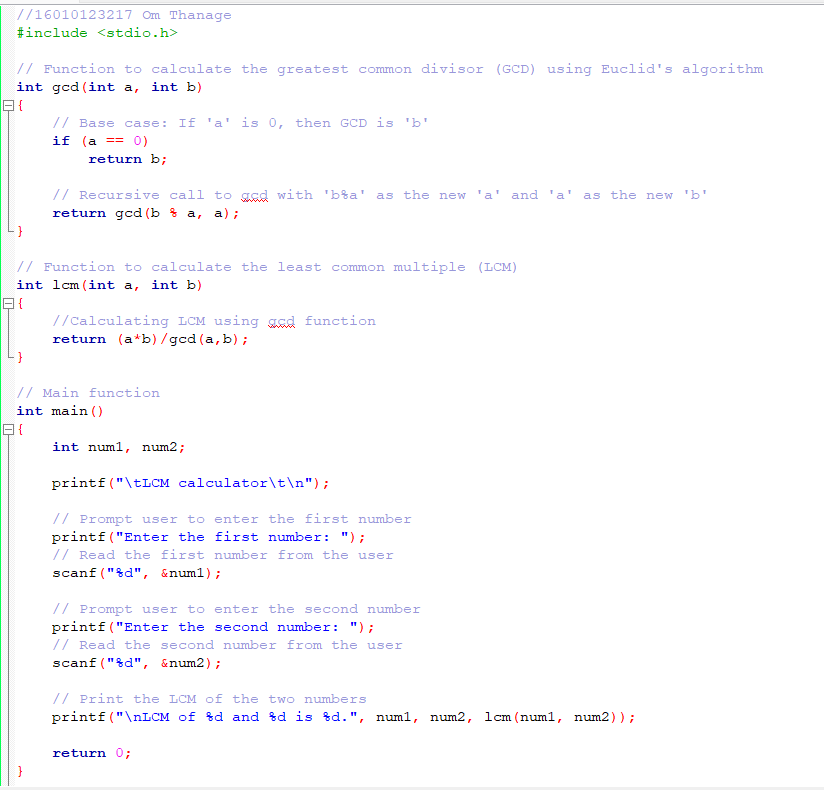
1. Start.
2. Declare the necessary variables **num1** and **num2** to store the input numbers.
   * Define the **gcd** function to calculate the greatest common divisor (GCD) of two numbers using Euclid's algorithm.
   * If one of the numbers is 0, return the other number.
   * Otherwise, recursively call **gcd** with the remainder of dividing **b** by **a** as the new **a** and **a** as the new **b**.
3. Define the **lcm** function to calculate the least common multiple (LCM) of two numbers.
   * Calculate LCM using the formula: **(a \* b) / gcd(a, b)**.
4. In the main function:
   * Display a header indicating that this program is an LCM calculator.
   * Prompt the user to enter the first number.
   * Read the first number from the user and store it in the variable **num1**.
   * Prompt the user to enter the second number.
   * Read the second number from the user and store it in the variable **num2**.
   * Calculate the LCM of **num1** and **num2** using the **lcm** function.
   * Print the result, displaying the LCM of **num1** and **num2**.
5. End.

**Implementation details:**

1)Code:



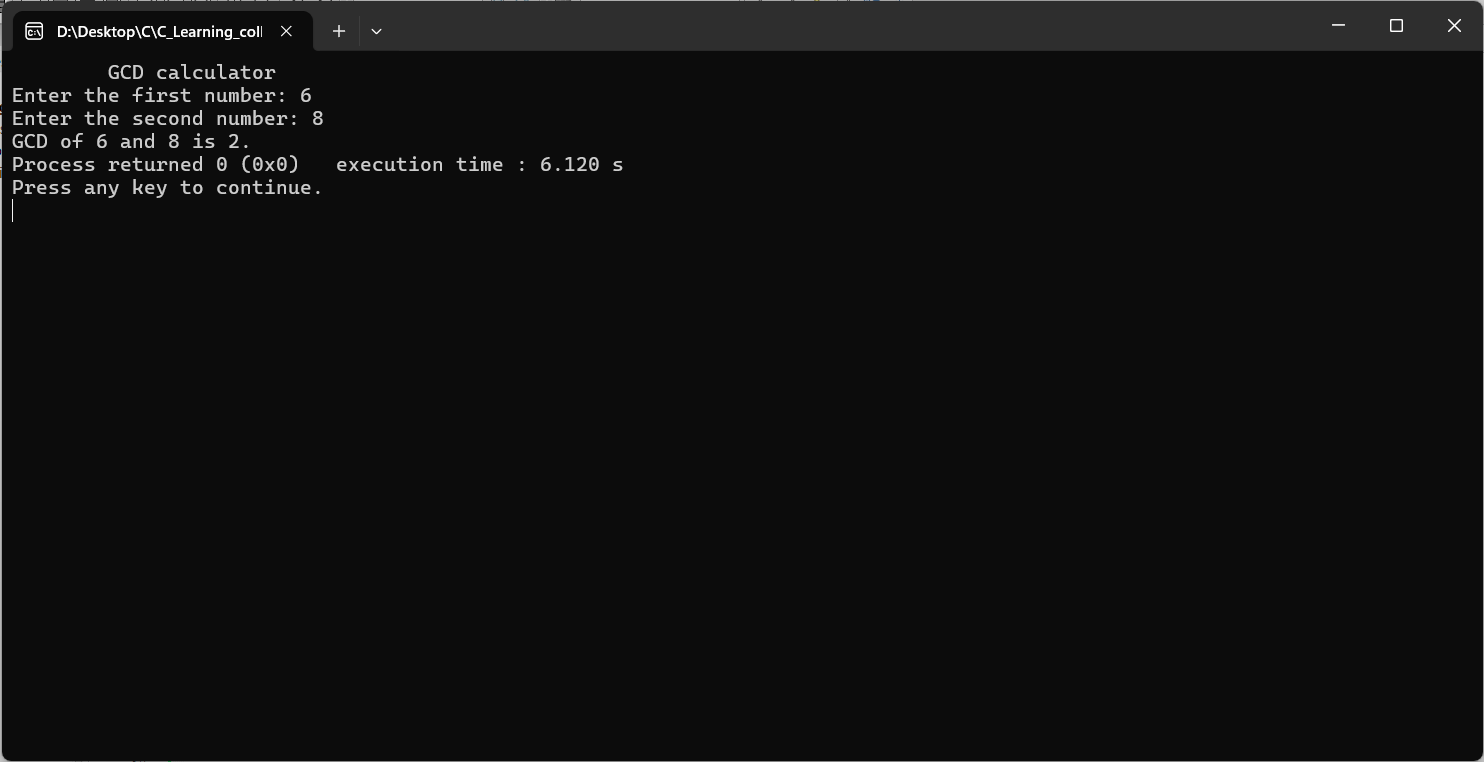
2) Code:



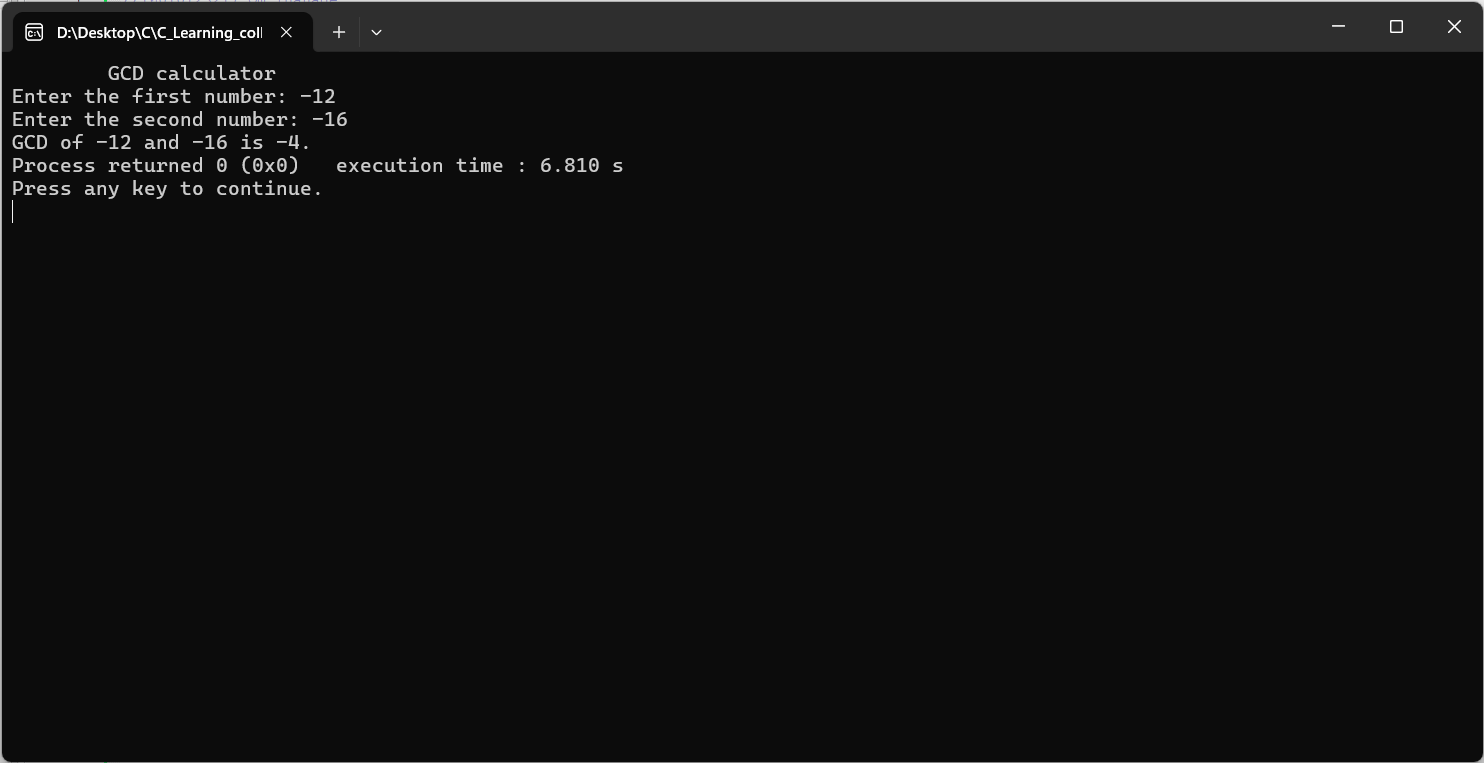
**Output(s):**

1) GCD

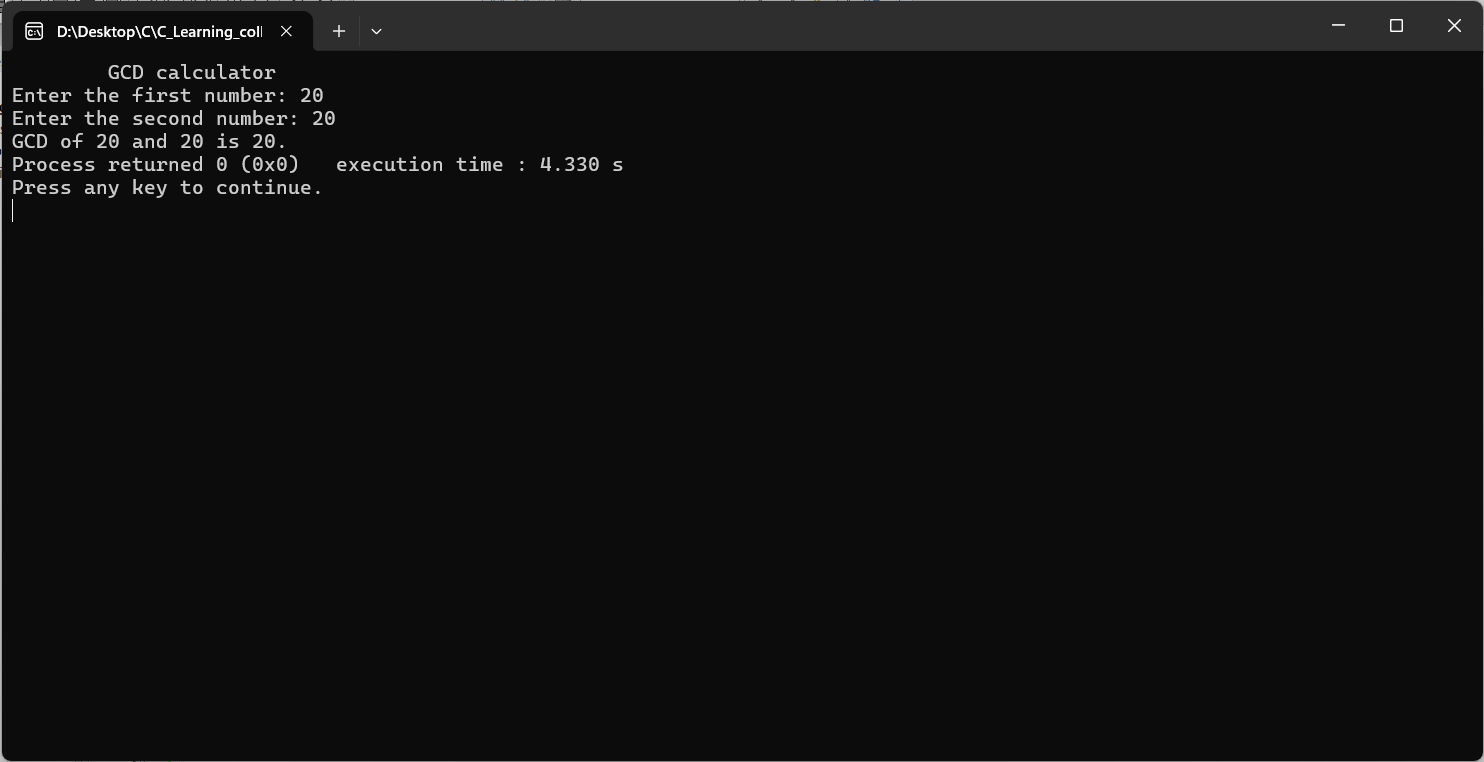
* Test case 1: Positive numbers



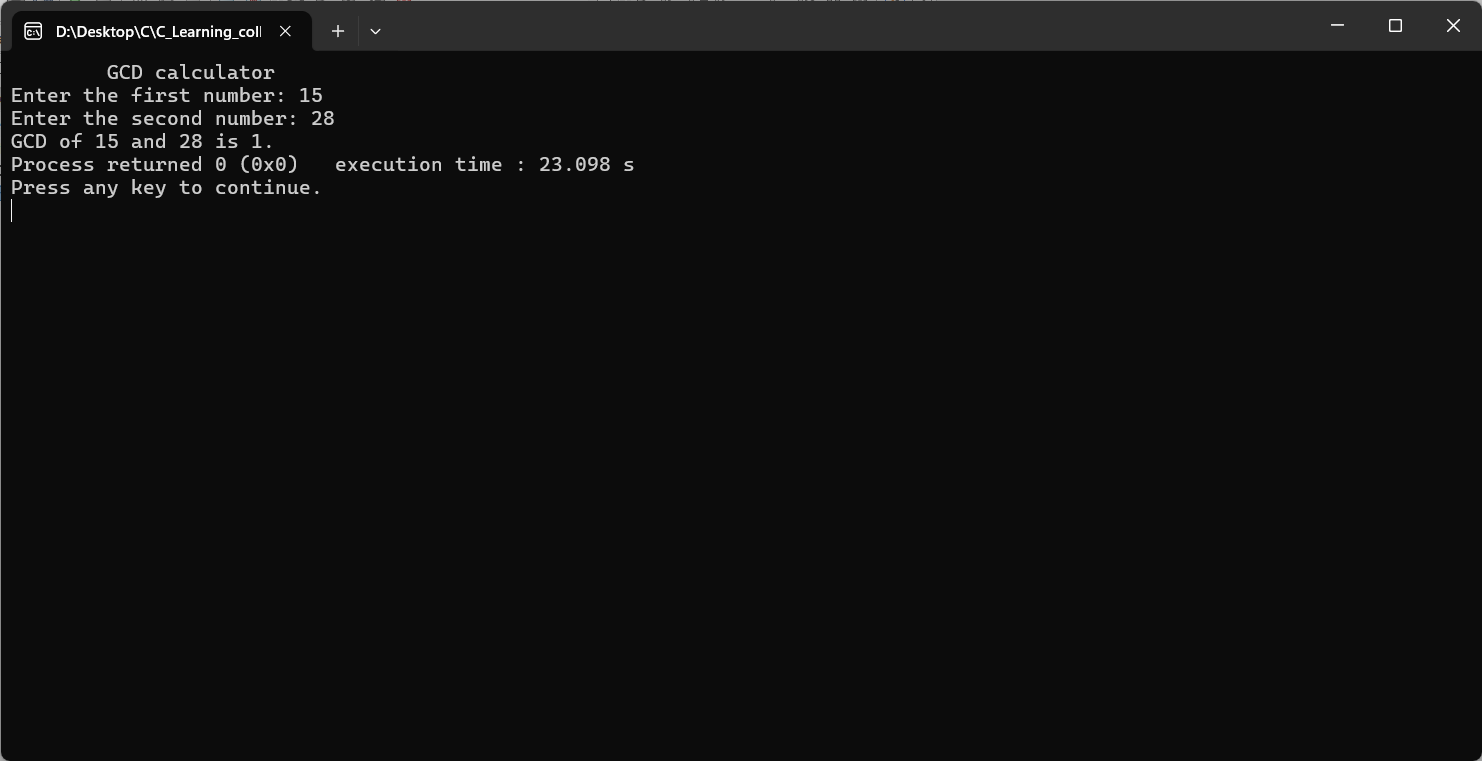
* Test case 2: Negative numbers



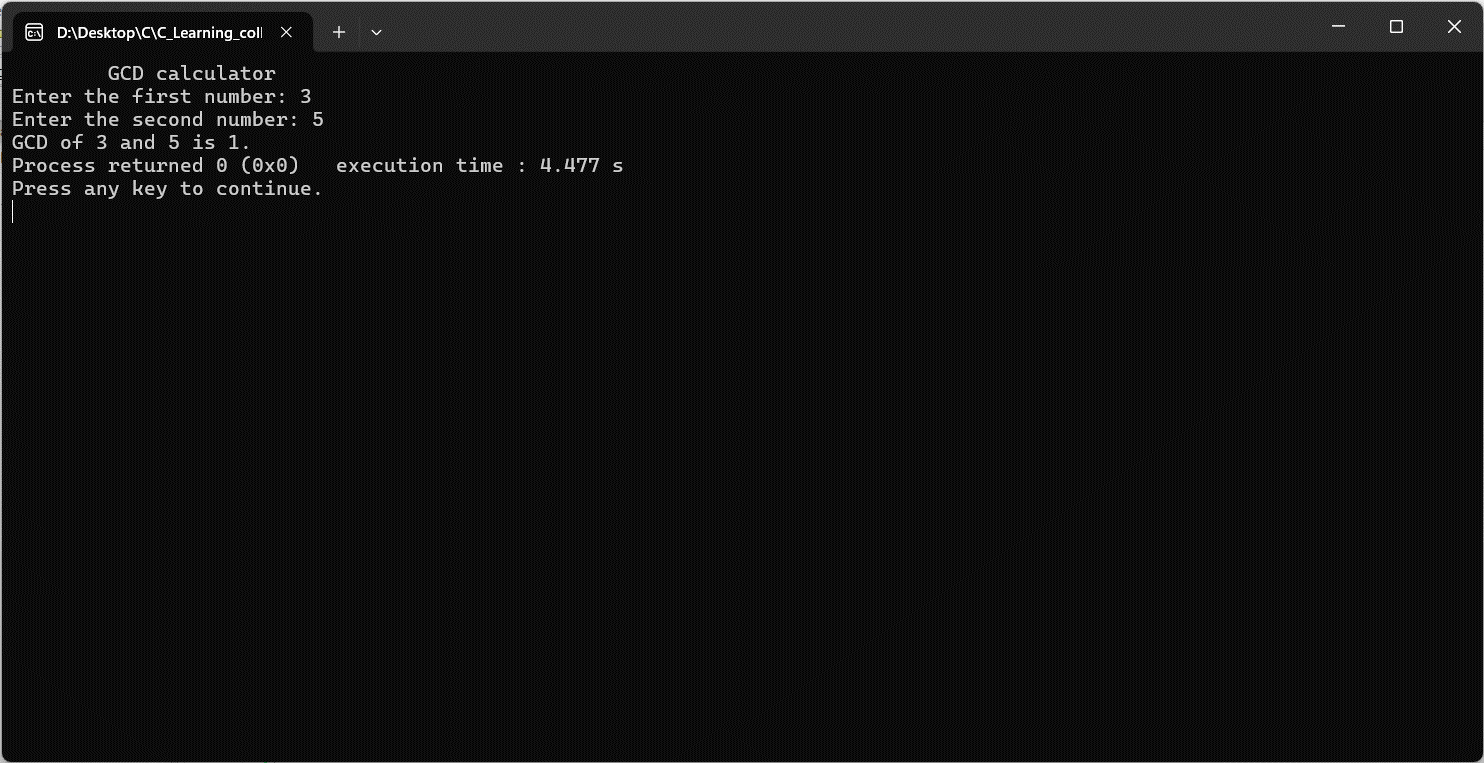
* Test case 3: Equal numbers



* Test case 4: Coprime numbers

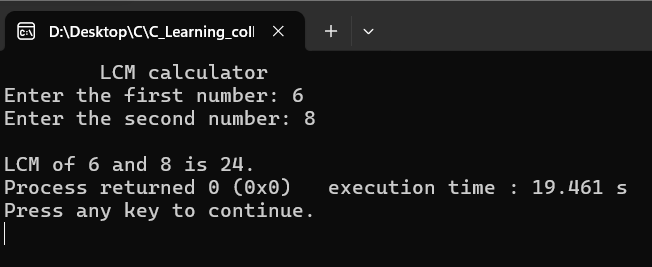


* Test case 5: Prime numbers

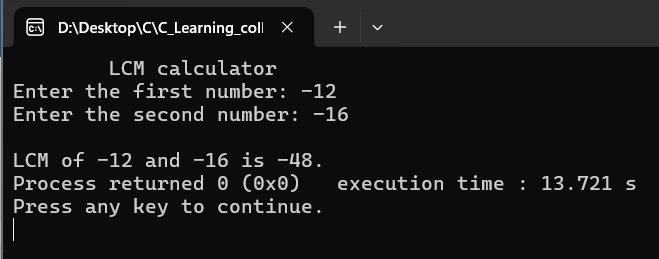


2)LCM

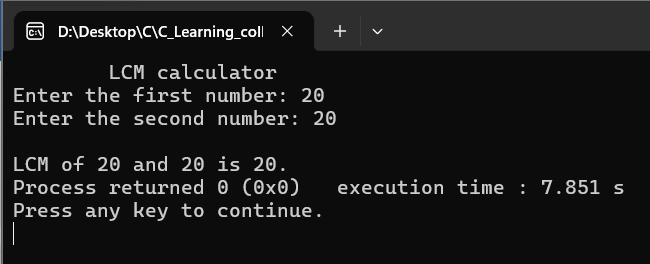
* Test case 1: Positive numbers



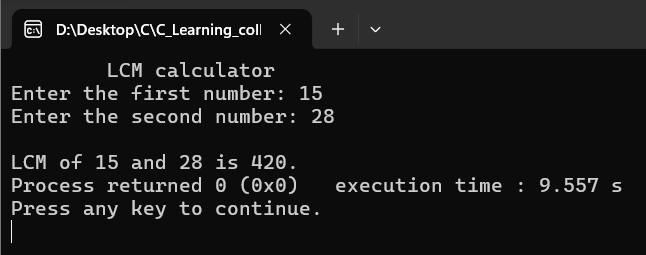
* Test case 2: Negative numbers



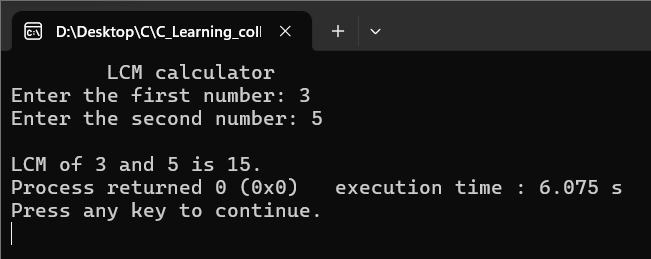
* Test case 3: Equal numbers



* Test case 4: Coprime numbers



* Test case 5: Prime numbers



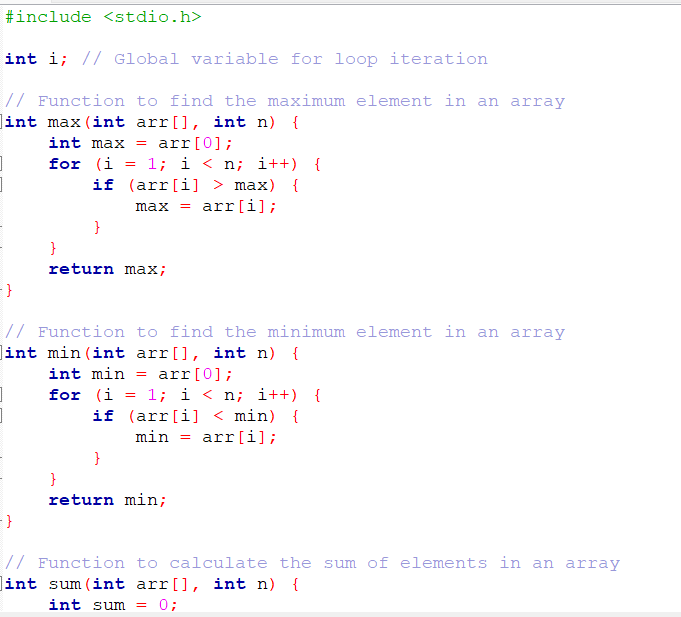
**Conclusion:**

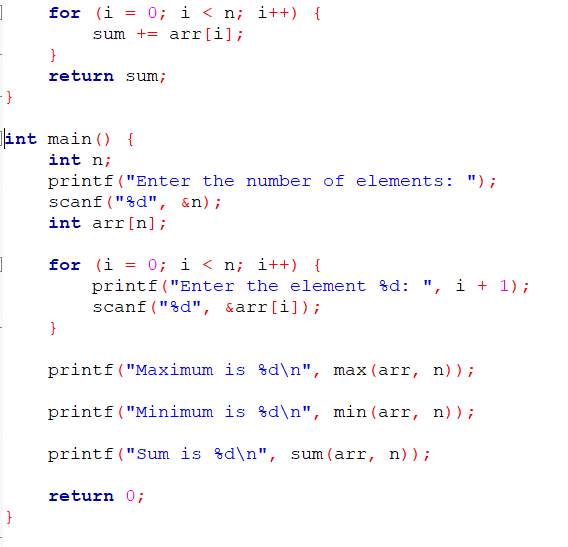
In this experiment, I learned about functions and the method to call them within the same function (recursion). By using functions reusability of code is made possible. I've learned how Euclid's algorithm can be used to efficiently calculate the GCD of two numbers recursively. I've realized the importance of testing code with different inputs to ensure it works correctly in various scenarios. This includes testing with positive numbers, negative numbers, equal numbers, coprime numbers, etc.

**Post Lab Questions**

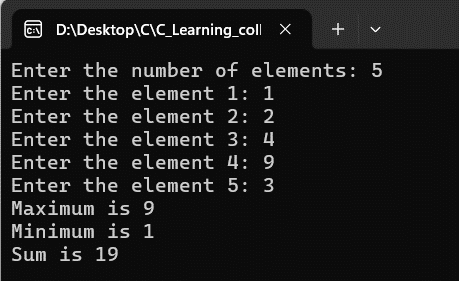
1. Write a C program to find the minimum, maximum and sum of elements in an array using functions.

Code:





Output:



1. Virtual Lab for functions.

<https://cse02-iiith.vlabs.ac.in/exp/cp-recursion/simulation.html>

<https://cse02-iiith.vlabs.ac.in/exp/functions/simulation.html>

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_ Signature of faculty in-charge**