K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Science and Humanities

**Batch: C1-2** **Roll No.: 16010123036**

**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:**

a) Write a program to find the GCD of two numbers using recursion.
b) 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: | Test case 2: |
|---|---|
| Input:<br>24,28<br>Output:<br>GCD: 4 | Input:<br>24,25<br>Output:<br>GCD: 1 |

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

Example:

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

**Algorithm:**

**Q1)**

```
1. Start
2. Declare two integer variables 'num1' and 'num2' to store two numbers.
3. Prompt the user to enter two numbers and read them into 'num1' and 'num2'.
4. Call the function 'gcd' with 'num1' and 'num2' as arguments and store the result in 'result'.
5. Print the result as "GCD of num1 and num2 is result".
6. End

Function gcd(int a, int b):
1. Start
2. Check if 'b' is equal to 0:
   a. If true, return 'a'.
   b. If false, proceed to the next step.
3. Else, recursively call gcd with 'b' and 'a % b' as arguments.
4. End
```

**Q2)**

1. Start

2. Declare two integer variables 'num1' and 'num2' to store two numbers.

3. Prompt the user to enter two numbers and read them into 'num1' and 'num2'.

4. Call the function 'lcm' with 'num1' and 'num2' as arguments and store the result in 'result'.

5. Print the result as "LCM of num1 and num2 is result".

6. End

Function gcd(int a, int b):

1. Start

2. Check if 'b' is equal to 0:

    a. If true, return 'a'.

    b. If false, proceed to the next step.

3. Else, recursively call gcd with 'b' and 'a % b' as arguments.

4. End

Function lcm(int a, int b):

1. Start

2. Calculate the greatest common divisor (GCD) of 'a' and 'b' using the function 'gcd' and store it in a variable.

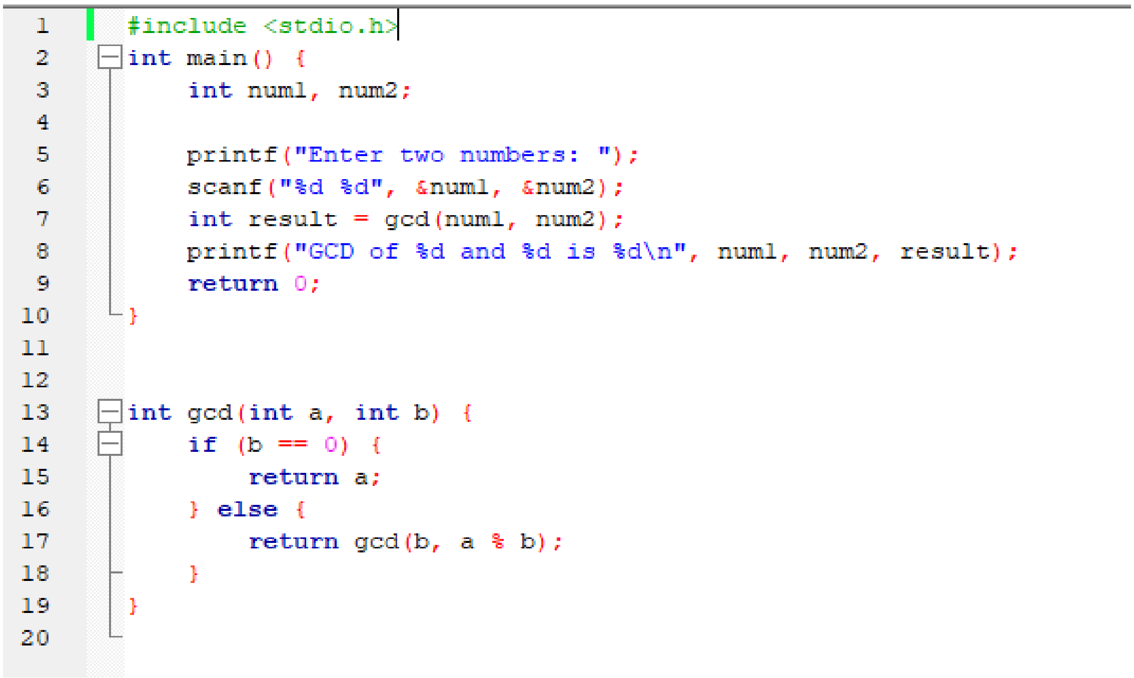
3. Calculate the least common multiple (LCM) of 'a' and 'b' using the formula: (a * b) / gcd(a, b).

4. Return the calculated LCM.

5. End

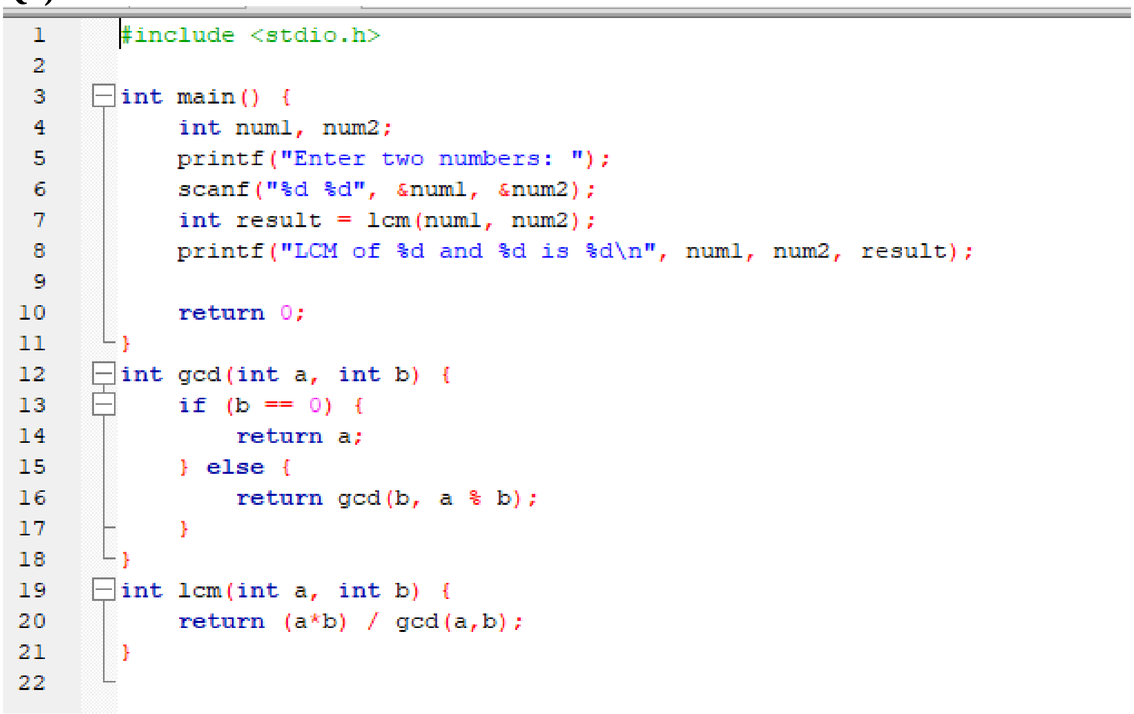
**Implementation details:**

**Q1)**



```c
#include <stdio.h>
int main() {
    int num1, num2;

    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);
    int result = gcd(num1, num2);
    printf("GCD of %d and %d is %d\n", num1, num2, result);
    return 0;
}


int gcd(int a, int b) {
    if (b == 0) {
        return a;
    } else {
        return gcd(b, a % b);
    }
}
```

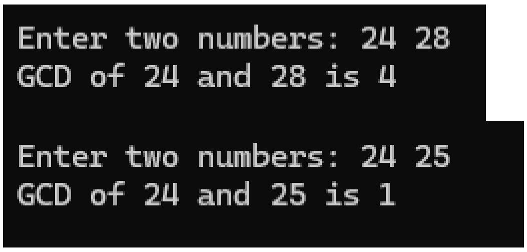
**Q2)**



```c
#include <stdio.h>

int main() {
    int num1, num2;
    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);
    int result = lcm(num1, num2);
    printf("LCM of %d and %d is %d\n", num1, num2, result);

    return 0;
}
int gcd(int a, int b) {
    if (b == 0) {
        return a;
    } else {
        return gcd(b, a % b);
    }
}
int lcm(int a, int b) {
    return (a*b) / gcd(a,b);
}
```

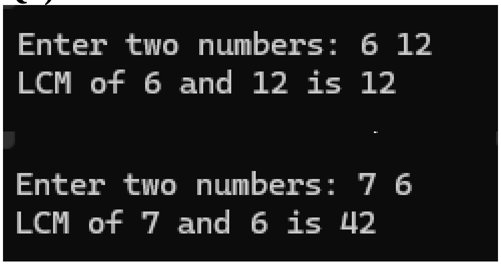
**Output(s):**

**Q1)**



```
Enter two numbers: 24 28
GCD of 24 and 28 is 4

Enter two numbers: 24 25
GCD of 24 and 25 is 1
```

**Q2)**



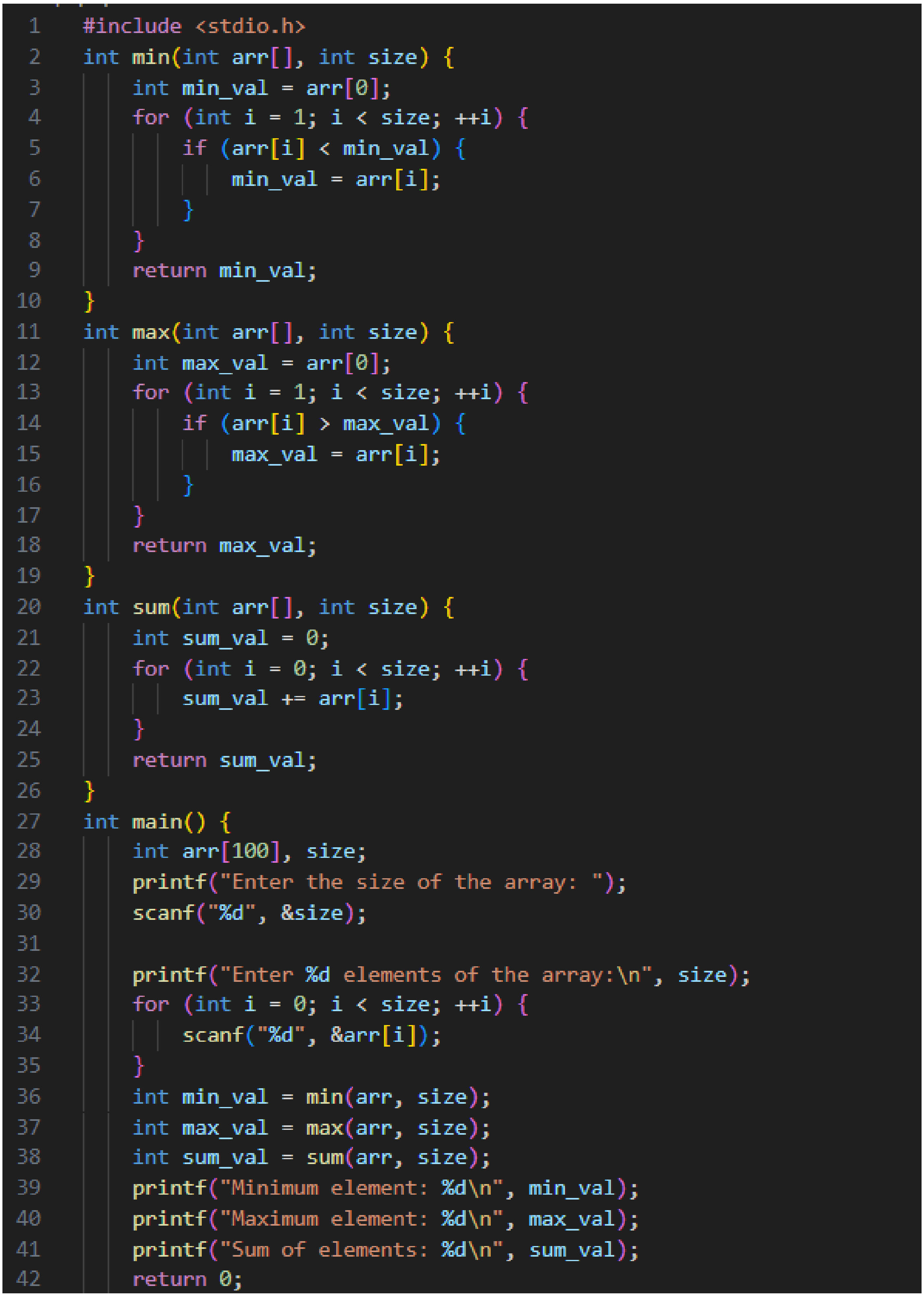
```
Enter two numbers: 6 12
LCM of 6 and 12 is 12

Enter two numbers: 7 6
LCM of 7 and 6 is 42
```

### Conclusion:

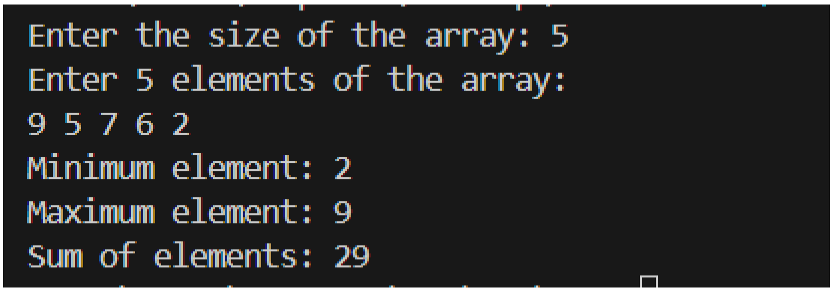
In this experiment, we effectively utilized user-defined functions in C to find the Greatest Common Divisor (GCD) of two numbers using recursion and then used this function to compute the Least Common Multiple (LCM). This modular approach enhances code organization and readability, promoting better software engineering practices. By employing structured programming techniques, we demonstrated how to design efficient and maintainable programs for mathematical computations.

### Post Lab Questions

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



```
#include <stdio.h>
int min(int arr[], int size) {
    int min_val = arr[0];
    for (int i = 1; i < size; ++i) {
        if (arr[i] < min_val) {
            min_val = arr[i];
        }
    }
    return min_val;
}
int max(int arr[], int size) {
    int max_val = arr[0];
    for (int i = 1; i < size; ++i) {
        if (arr[i] > max_val) {
            max_val = arr[i];
        }
    }
    return max_val;
}
int sum(int arr[], int size) {
    int sum_val = 0;
    for (int i = 0; i < size; ++i) {
        sum_val += arr[i];
    }
    return sum_val;
}
int main() {
    int arr[100], size;
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    printf("Enter %d elements of the array:\n", size);
    for (int i = 0; i < size; ++i) {
        scanf("%d", &arr[i]);
    }
    int min_val = min(arr, size);
    int max_val = max(arr, size);
    int sum_val = sum(arr, size);
    printf("Minimum element: %d\n", min_val);
    printf("Maximum element: %d\n", max_val);
    printf("Sum of elements: %d\n", sum_val);
    return 0;
```

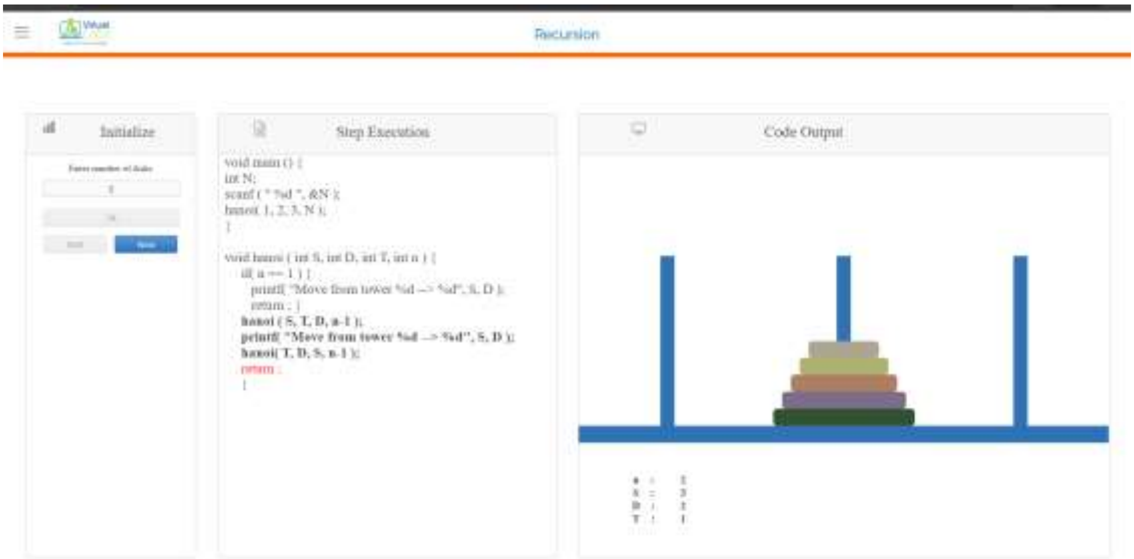


```
Enter the size of the array: 5
Enter 5 elements of the array:
9 5 7 6 2
Minimum element: 2
Maximum element: 9
Sum of elements: 29
```

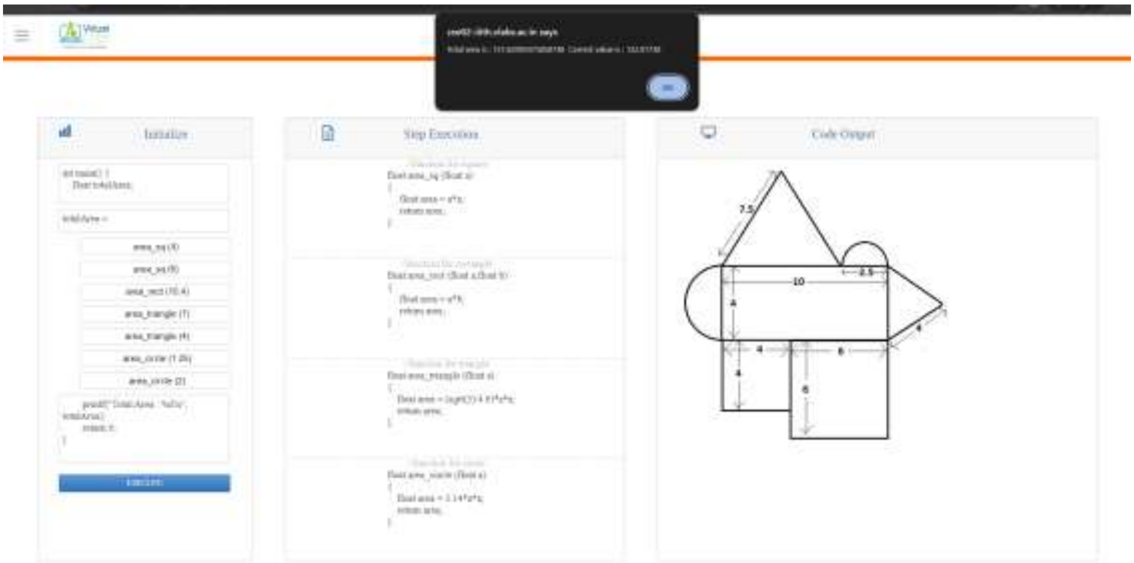
2. Virtual Lab for functions.
   https://cse02-iiith.vlabs.ac.in/exp/cp-recursion/simulation.html

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

Sol:



```
void main () {
int N;
scanf ( " %d ", &N );
hanoi( 1, 2, 3, N );
}

void hanoi ( int S, int D, int T, int n ) {
  if( n == 1 ) {
    printf( "Move from tower %d --> %d", S, D );
    return ; }
  hanoi ( S, T, D, n-1 );
  printf( "Move from tower %d --> %d", S, D );
  hanoi( T, D, S, n-1 );
  return ;
  }
```



[illegible] says

[illegible]

Initialize

```
int main() {
  float totalArea;

totalArea =
        area_sq (4)
        area_sq (8)
        area_rect (10,4)
        area_triangle (7)
        area_triangle (4)
        area_circle (1.25)
        area_circle (2)

        printf("Total Area : %f\n",
totalArea);
        return 0;
}
```

Step Execution

```
//function for square
float area_sq (float a)
{
  float area = a*a;
  return area;
}

//function for rectangle
float area_rect (float a,float b)
{
  float area = a*b;
  return area;
}

//function for triangle
float area_triangle (float a)
{
  float area = (sqrt(3)/4)*a*a;
  return area;
}

//function for circle
float area_circle (float a)
{
  float area = 3.14*a*a;
  return area;
}
```

Code Output

**Date: _____________**                                        **Signature of faculty in-charge**