Course Code	**:**	**MCS-011**
Course Title	**:**	**Problem Solving and Programming**
Assignment Number	**:**	**BCA(II)/011/Assignment/2026-27**
Maximum Marks	**:**	**100**
Weightage	**:**	**25%**
Last Dates for Submission	**:**	**31st October, 2026 (For July 2026 Session)**
	:	**30th April, 2027 (For January 2027 Session)**

There are eight questions in this assignment. Each question carries 10 marks. Rest 20 marks are for viva-voce. Answer all the questions. You may use illustrations and diagrams to enhance the explanations. Please go through the guidelines regarding assignments given in the Programme Guide for the format of presentation.

Q1: **(10 Marks)**

Explain the concepts of *recursion* and *iteration* in C with suitable examples. Write two separate C programs to calculate the *Greatest Common Divisor (GCD)* of two positive integers:

- using an iterative approach and
- using a recursive approach

Compare both approaches.

Q2: **(10 Marks)**

Write a C program that accepts *n* integer values from the user and performs the following operations using one-dimensional arrays:

- Find the second largest and second smallest elements.
- Count the number of even and odd numbers.
- Display all duplicate elements, if any.
- Compute the average of all elements.

Q3: **(10 Marks)**

Write a C program to perform operations on a square matrix of order N.

The program should:

- Find the sum of each row and each column.
- Compute the sum of the principal and secondary diagonals.
- Check whether the matrix is symmetric.
- Display the transpose of the matrix.

Q4: **(10 Marks)**

Without using any built-in string handling functions, write a menu-driven C program to perform the following operations on a given string:

1. Reverse the string.
2. Count the number of words.
3. Convert the string to uppercase.
4. Check whether the string is an anagram of another string.
5. Display the frequency of vowels and consonants.

Use the *switch-case* construct for implementing the menu.

Q5: **(10 Marks)**

Explain the concept of *dynamic memory allocation in* C. Write a C program that dynamically allocates memory to store the details of *N products* (Product ID, Product Name, Quantity, Price).

The program should:

- Accept product details.
- Display all products.
- Find the costliest product.
- Calculate the total inventory value.
- Release the allocated memory before termination.

Discuss the advantages of dynamic memory allocation in this context.

Q6: **(10 Marks)**

Explain the role of *structures* and *nested structures* in C programming. Define appropriate structures for a *Library Management System* containing:

- Book ID
- Title
- Author
- Publisher
- Number of Copies

Write functions to:

- Input book details.
- Display all records.
- Search a book using Book ID.
- Display books written by a specified author.

- Find the book having the maximum number of available copies.

Q7: **(10 Marks)**

Write a C program to maintain *electricity consumer records* using file handling.

The program should:

- Read consumer details from a file.
- Calculate the electricity bill based on units consumed.
- Determine whether the consumer is eligible for a subsidy (based on user-defined criteria).
- Store the generated bills in another file.
- Display all processed records.

Explain the file handling operations used in the program.

Q8: **(10 Marks)**

Write a C program that:

- Uses macros to calculate the area and perimeter of a rectangle and a circle.
- Uses conditional compilation (#ifdef, #ifndef, or #if) to selectively compile either the rectangle calculations or the circle calculations.
- Demonstrates the use of a user-defined header file containing the required macros.

Discuss the advantages of using header files and preprocessor directives in modular programming.

Q1: (10 Marks)

Explain the concepts of recursion and iteration in C with suitable examples. Write two separate C programs to calculate the Greatest Common Divisor (GCD) of two positive integers:

- **using an iterative approach and**
- **using a recursive approach**

Compare both approaches.

Answer

Iteration repeats a block of instructions with a loop such as while or for. The variables are updated on each pass, and the loop stops when its condition becomes false. Recursion means that a function calls itself on a smaller version of the same problem. A recursive solution needs a base case to stop further calls; otherwise, the calls continue until the call stack is exhausted.

Both methods can implement the Euclidean algorithm. For two positive integers a and b, GCD(a, b) equals GCD(b, a % b) while b is non-zero. When b becomes zero, the remaining a is the GCD. For example, 72 % 48 = 24 and 48 % 24 = 0, so GCD(72, 48) is 24.

Iterative C program

```
#include <stdio.h>

int main(void) {
    unsigned int a, b, r;
    printf("Enter two positive integers: ");
    if (scanf("%u %u", &a, &b) != 2 || a == 0 || b == 0) {
        puts("Invalid input. Both numbers must be positive.");
        return 1;
    }
    while (b != 0) {
        r = a % b;
        a = b;
        b = r;
    }
    printf("GCD = %u\n", a);
    return 0;
}
```

Recursive C program

```
#include <stdio.h>

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