Lab 11

Task 1: Function Overloading in C++ Classes

Description:
You are tasked with implementing a simple class in C++ called MathOperations, which will demonstrate the concept of function overloading. The class will contain multiple member functions with the same name but different parameter lists, showcasing the versatility of function overloading in C++.

Instructions:

- Create a C++ class named MathOperations.
- Implement the following member functions within the class:
    - add: This function should take two integer parameters and return the sum of the two integers.
    - add: Overload the add function to take three double parameters and return the sum of the three doubles.
- Create a main function to test the functionality of your MathOperations class.
- Inside the main function, create an instance of the MathOperations class.
- Test each overloaded function by calling them with different sets of arguments.
- Print the results of each function call to the console.

Task 2: Unary Operator Overloading Example in C++

Description
You are tasked with implementing a class called Vector2D to represent 2D vectors. Implement unary operator overloading for negation to allow negating the vector's components.

Task:

- Define the Counter class with an integer member variable count initialized to 0.
- Implement the unary operator overloading for both prefix and postfix increment operations:
    - For prefix increment (++count), the count should be incremented by 1.
    - For postfix increment (count++), the count should be incremented by 1 after the current value is used.
- Write a main() function to demonstrate the usage of both prefix and postfix increment operators on a Counter object.

Task 3: Binary Operator Overloading Example in C++

Description
You are assigned the task of implementing a Matrix class in C++ to represent matrices. Implement binary operator overloading for multiplication (*) to allow multiplying two matrices.

- Define the Matrix class with a private member variable data to store the matrix elements as a 2D int matrix.
- Make required constructors.
- Implement binary operator overloading for multiplication (*) to allow multiplying two matrices:
- For two matrices A and B, the multiplication operation A * B should return a new matrix that represents the result of multiplying matrices A and B.
- Write a main() function to demonstrate the usage of binary operator overloading for matrix multiplication.