**Friend Function:**

1. Create a C++ class representing a Circle with private data members for radius. Implement a friend function calculateArea that calculates and returns the area of the circle.
2. Design a C++ program with two classes, Student and Marksheet. Use a friend function to access and display the student's details and marks from both classes.
3. Create a C++ class BankAccount with private data members for account balance. Implement a friend function withdraw that allows you to withdraw money from the account while maintaining proper validation.
4. Develop a C++ program to show the usage of friend classes, where one class can access the private members of another class.

**Inline Function:**

1. Write a C++ program to calculate the factorial of a number using an inline function.
2. Create an inline function in C++ to find the square of a number and display the result.
3. Design a program with an inline function to check if a given number is even or odd.
4. Implement an inline function in C++ to compute the area of a rectangle with user-defined input.
5. Write a C++ program with an inline function to calculate the volume of a sphere given its radius.

**Static Data Member:**

1. Create a C++ class for a Book with a static data member to keep track of the total number of books available. Implement appropriate member functions to add and remove books from the inventory.
2. Develop a C++ program to demonstrate the usage of a static data member to maintain a count of objects created for a particular class.
3. Write a C++ program to represent a BankAccount class with a static data member for the interest rate. Implement a static member function to update the interest rate for all accounts.
4. Design a C++ class for a Car with a static data member to store the total number of cars manufactured by a company. Implement a constructor to update this count when a new car is created.
5. Create a C++ class for a University with static data members to keep track of the total number of students and professors. Implement functions to add new students and professors and display the counts.

**Static Member Function:**

1. Design a C++ class for a Calculator with static member functions for basic arithmetic operations like addition, subtraction, multiplication, and division.
2. Create a C++ program to demonstrate the usage of a static member function to calculate the factorial of a number.
3. Implement a C++ class Date with a static member function to validate whether a given date is valid or not.
4. Develop a C++ program to simulate a Bank system with static member functions to calculate interest and display bank statistics.
5. Write a C++ class MathUtilities with a static member function to find the maximum and minimum of a set of numbers.

**Default Arguments:**

1. Create a C++ function that calculates the volume of a cube with default arguments for length, width, and height.
2. Implement a C++ function for calculating the area of a rectangle with default arguments for length and width.
3. Design a C++ program with a function that calculates the total cost of a product with default arguments for quantity and unit price.
4. Write a C++ program with a function that calculates the area of a circle with a default argument for radius.
5. Develop a C++ program with a function that prints a message with a default argument for the number of times it should be repeated.

**Type Conversion:**

1. Create a C++ program that demonstrates implicit type conversion by adding an integer and a floating-point number.
2. Implement a C++ program that uses explicit type conversion to convert a floating-point number to an integer.
3. Develop a C++ program that showcases the risks of data loss when converting a larger data type to a smaller one and how to handle it.

**New Operators and Delete Operators:**

1. Write a C++ program that dynamically allocates memory for an integer using the new operator and then frees it using the delete operator.
2. Create a C++ program that dynamically allocates an array of integers using new[] and releases the memory using delete[].
3. Implement a C++ program that allocates memory for a string on the heap using new and deallocates it using delete.
4. Develop a C++ program that uses the new operator to create an object of a user-defined class and then deletes it using delete.

**Arrays of Objects:**

1. Create a C++ class Student with data members for name, roll number, and marks. Create an array of Student objects and display their details.
2. Implement a C++ program to sort an array of objects of a user-defined class based on a specific criteria (e.g., age, name).
3. Write a C++ program to find the average marks of students stored in an array of Student objects.
4. Develop a C++ program to demonstrate the usage of arrays of objects in a library system, where each object represents a book.
5. Design a C++ program that creates an array of objects representing geometric shapes (circle, rectangle, triangle) and calculates the total area of all shapes in the array.