Assignment: Operator Overloading for a 3D Vectors Class in C++

This assignment is concerned with operator overloading learning outcome. In this task, you will learn how to overload specific operators for specific datatypes.

Objective:

In this assignment, you will implement a class named `Vector3D` in C++ to represent a vector in a 3D coordinate system. The class should have two points in 3D space: a source point called 'p' and a destination point called 'q'. You will also overload several operators to perform common vector operations, such as addition, subtraction, dot product, and cross product.

Requirements:

1. Define a class `Vector3D` with the following attributes:

- Two private member variables to represent the source point (`p`) and destination point (`q`) in 3D space. Each point should be represented as an `array<double, 3>`, where the first element represents the x-coordinate, the second element represents the y-coordinate, and the third element represents the z-coordinate.

2. Implement the following constructors and destructor:

- Default constructor: Initializes both points to the origin (0, 0, 0).

- Parameterized constructor: Accepts two points as arguments and initializes the source and destination points accordingly.

- Copy constructor: Initializes a new object with the same source and destination points as the object passed as an argument.

- Destructor: Releases any allocated resources.

3. Overload the following operators for the `Vector3D` class:

- `operator+`: Adds two vectors using the head-to-tail rule. The result should be a new `Vector3D` object representing the sum of the two vectors.

- **`operator-`:** Subtracts two vectors. The result should be a new `Vector3D` object representing the difference between the two vectors.

- `operator*`: Calculates the dot product of two vectors. The result should be a `double` representing the scalar dot product.

- **`operator^`:** Calculates the cross product of two vectors. The result should be a new `Vector3D` object representing the vector cross product.

- **`operator++`:** Increments the magnitude of a vector in all three directions (x, y, and z) by 1. This should modify the `Vector3D` object in-place (use both prefix and postfix overloads).

- **`operator--`:** Decrements the magnitude of a vector in all three directions (x, y, and z) by 1. This should modify the `Vector3D` object in-place (use both prefix and postfix overloads).

4. Implement appropriate getters and setters for the source and destination points.

5. Write a test program to demonstrate the functionality of the `Vector3D` class and its overloaded operators.

Notes:

- Make sure to follow best practices for C++ programming and object-oriented design, such as proper encapsulation, error handling, and code readability.

- Include necessary header files, and use appropriate namespaces as needed.

- Comment your code to explain the functionality of each method, constructor, and operator.

Submission:

Submit your completed C++ source code file(s) for the `Vector3D` class implementation, along with any required header files and the test program. Ensure your code is well-commented and adheres to the assignment requirements.