SC1008 Lab 3 of C++ – Class and Object

1. **(Class Constructors, Destructors and Access Specifier) [10 marks]** Suppose you are asked to implement a class called `Product` to manage the information of different products in an online shopping platform. Below are the detailed members of the class `Product`:
   - Private Members:
     - `string name`: the name of the product,
     - `double price`: the price of the product.
   - Public Members:
     - Constructor that initializes `name` and `price`, and print out necessary information.
     - Destructor that prints a message when an object is deleted.
     - Getters:
       - `getName()` that returns the private attribute name.
       - `getPrice()` that returns the private attribute price.
     - Setter:
       - `setPrice(double newPrice)` that updates price but **only if non-negative**.

   Below is the starting code with missing implementations for you to implement.

```cpp
#include <iostream>
#include <string>
using namespace std;

class Product {
private:
    string name;
    double price;

public:
    // Constructor
    Product(string productName, double productPrice) {
        //TO-DO: Write Your Code Here
        //
        //

    }

    // Destructor
    //TO-DO: Write Your Code Here
    //
    //

    // Getters
    //TO-DO: Write Your Code Here
    //
```

```
    //


    // Setter for price (ensures non-negative value)
    void setPrice(double newPrice) {
        //TO-DO: Write Your Code Here
        //
        //

    }
};


int main() {
    // Creating Product objects
    Product product1("Laptop", 999.99);
    Product product2("Phone", 499.50);

    // Displaying product details
    cout<<endl;
    cout << "The name of Product 1: " << product1.getName() << endl;
    cout<< "The price of Product 1: $" << product1.getPrice() << endl;
    cout<<endl;

    // Modifying product price with setter
    product2.setPrice(550.00);
    cout << "Updated Price of Product 2: $" << product2.getPrice() << endl;

    return 0;
}
```

The corresponding sample output should be:

```
Product created: Laptop ($999.99)
Product created: Phone ($499.5)

The name of Product 1: Laptop
The price of Product 1: $999.99

Updated Price of Product 2: $550
Product deleted: Phone
Product deleted: Laptop
```

2. **(Operator Overloading) [10 marks]** You are asked to implement a class called `Complex` that allows arithmetic operations on complex numbers. In mathematics, a complex number is written as:

$$a + bi$$

where a is the real part and b is the imaginary part. Below are the detailed members of `Complex`:

- Private Members:
  - `double real`: the real part of the complex number.
  - `double imag`: the imaginary part.
- Public Members:
  - Constructor that initializes real and imaginary parts.
  - Operator Overloading:
    - `operator+`, which adds two complex numbers.
    - `operator-`, which subtracts two complex numbers.
    - `operator<<`, which prints the complex number in *a* + *bi* format.

Your tasks are as follows:

- Overload the + operator to allow addition of two complex numbers.
- Overload the - operator to allow subtraction of two complex numbers.
- Overload the << operator to print a complex number in a + bi format.

Below is the starting code with missing implementations for you to implement.

```
#include <iostream>

class Complex {
private:
    double real;
    double imag;

public:
    // Constructor
    Complex(double r, double i) : real(r), imag(i) {}

    // Overloading the + operator
    // TODO: Write Your Code Here
    //
    //

    // Overloading the - operator
    // TODO: Write Your Code Here
    //
    //


    // Overloading the << operator for output
    friend std::ostream& operator<<(std::ostream& out, const Complex& c) {
```

```
        // TODO: Write Your Code Here
        //
        //


    }
};

int main() {
    Complex c1(3.5, 2.0);
    Complex c2(1.5, 4.0);

    Complex sum = c1 + c2;
    Complex diff = c1 - c2;

    std::cout << "First Complex Number: " << c1 << std::endl;
    std::cout << "Second Complex Number: " << c2 << std::endl;
    std::cout << "Sum: " << sum << std::endl;
    std::cout << "Difference: " << diff << std::endl;

    return 0;
}
```

Sample output should be:

```
First Complex Number: 3.5 + 2i
Second Complex Number: 1.5 + 4i
Sum: 5 + 6i
Difference: 2 - 2i
```