```cpp
// complex_ops.cpp
#include <iostream>
using namespace std;

class Complex {
private:
    double real, imag;
public:
    // Default constructor
    Complex(): real(0), imag(0) {}

    // Parameterized constructor
    Complex(double r, double i): real(r), imag(i) {}

    // Copy constructor (default would also work)
    Complex(const Complex &c): real(c.real), imag(c.imag) {}

    // Operator + (add)
    Complex operator+(const Complex &c) const {
        return Complex(real + c.real, imag + c.imag);
    }

    // Operator * (multiply)
    Complex operator*(const Complex &c) const {
        double r = real * c.real - imag * c.imag;
        double i = real * c.imag + imag * c.real;
        return Complex(r, i);
    }

    // Friend overloads for input and output
    friend ostream& operator<<(ostream &out, const Complex &c) {
        out << c.real;
        if (c.imag >= 0) out << " + " << c.imag << "i";
        else out << " - " << -c.imag << "i";
        return out;
    }

    friend istream& operator>>(istream &in, Complex &c) {
        // Expect two doubles: real imag
        in >> c.real >> c.imag;
        return in;
    }
};

int main() {
    cout << "Complex class demo\n";
    Complex a(2.5, 3.0);
    Complex b(1.0, -4.0);
    Complex c = a + b;
    Complex d = a * b;
    cout << "a = " << a << "\n";
    cout << "b = " << b << "\n";
    cout << "a + b = " << c << "\n";
    cout << "a * b = " << d << "\n";

    cout << "\nEnter complex (real imag): ";
    Complex x;
    cin >> x;
    cout << "You entered: " << x << "\n";
    return 0;
}
```