Complete the following program with the given declarations. All user input and output **must** occur in your `main` function. You are permitted to create other functions if you feel they would help you complete the task. Call your C++ file ComplexNumbers. This is to be submitted to Classroom by the date on the assignment. You are only completing one C++ file for this mini-project.

In your `main` function, display a menu to ask the user for their desired task. They are to first choose their operation and then enter the values required. Their options are listed below. When the process is complete, ask the user if they want to do it again.

In mathematics, complex numbers are used as the basis of a vector system on the coordinate plane. When graphing, we think of the real part (a) as the *x*-value and the imaginary part (b) as the *y*-value. Create a class called `Complex` inside of your program. Your class must contain private attributes for the a and b values for a given `Complex` object. You must also create a constructor to set the values for your object. Your class must also contain the following methods.

`Complex conjugate()`
This method returns the complex conjugate for the `Complex`.

`double norm()`
This method returns the norm of the `Complex`.

`double real()`
This method returns the real part of the `Complex`.

`double imag()`
This method returns the imaginary part of the `Complex`.

`double dotProduct(Complex other)`
This method returns the dot product of the `Complex` and the parameter.

`Complex scalarProduct(double value)`
This method returns the scalar product of the `Complex` and the parameter.

`double angleBetween(Complex other)`
This method returns the angle between the `Complex` and the parameter (in degrees).

Your class must also contain overloaded operators for addition, subtraction, multiplication, division, equality, and conversion (so that your `Complex` prints out in proper a + bi form). In your `main` function, you must allow the user to pick their choice of operations and enter the values for their `Complex` object(s) and required other values. After the user has entered the information, display the requested result.

This project will be graded on:

1. Documentation (4 points): Your code must be fully commented and employ standard C++-style conventions.
2. User-Friendliness (4 points): All interactions with the user must be clear and unambiguous.
3. Methods (21 points): Each function has the proper declaration and works as expected.
4. Overloaded operators (12 points): Each operator works as expected.

TOTAL: 41 points