H7. (复数类)

Create a class called Complex for performing arithmetic with complex numbers. Write a program to test your class.

Complex numbers have the form:
realPart + imaginaryPart * i

where i is

Use double variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should contain default values in case no initializers are provided. Provide public member functions that perform the following tasks:

(a) Adding two Complex numbers: The real parts (实部) are added together and the imaginary parts (虚部) are added together.
(b) Subtracting two Complex numbers: The real part of the right operand is subtracted from the real part of the left operand, and the imaginary part of the right operand is subtracted from the imaginary part of the left operand.
(c) Printing Complex numbers in the form (a, b), where a is the real part and b is the imaginary part.

Tips:
1. add函数的使用方式：Complex c = a.add(b);
2. substract函数的使用方式：Complex d = a.substract(b);

H8. (大整数类)

Create a class HugeInteger that uses a 40-element array of digits to store integers as large as 40 digits each. Provide member functions:

(a) Constructor (need overloaded): input a long int variable, set default value 0; input a string variable;
(b) add (need overloaded): input an int variable; input a int variable; input a string variable;
(c) substract (need overloaded): input an int variable; input a int variable; input a string variable;
(d) input: convert a string variable to HugeInteger object
(e) output: output HugeInteger object as string
(f) For comparing HugeInteger objects, provide functions isEqualTo, isNotEqualTo, isGreaterThan, isLessThan, isGreaterThanOrEqualTo and
isLessThanOrEqualToeach -- each of these is a "predicate" function that simply returns true if the relationship holds between the two HugeIntegers and returns false if the relationship does not hold. Also, provide a predicate function isZero.

If you feel ambitious, provide member functions multiply, divide and modulus.

Tips:
1. 为了节省存储空间，可用 short 类型的变量存储每位的值
2. 注意 HugeInteger 对象的有效值范围
3. 在(a)中，构造函数 input a long int variable 时，注意变量类型之间的转换（从 long 到 short 的转换）
4. int 类型的变量可以被 C++ 编译器直接转化为 long int 类型的变量，所以在实现 add/subtract 函数时，若参数为 int 类型，则可以将它通过参数为 long int 类型的构造函数转化为一个 HugeInteger 对象，然后利用参数为 HugeInteger 类型的 add/subtract 函数完成。
5. 在 subtract 函数中注意考虑出现负数怎么办（注意题目中 HugeInteger 变量没有符号位）