**CPP Problem Design**

|  |
| --- |
| **Subject:** **Complex** |
| **Contributor: 林承達, 陳俊儒, 廖宣瑋** |
| **Main testing concept:** Operator overloading   |  |  | | --- | --- | | **Basics** | **Functions** | | ■ C++ BASICS  □ FLOW OF CONTROL  ■ FUNCTION BASICS  □ PARAMETERS AND OVERLOADING  □ ARRAYS  ■ STRUCTURES AND CLASSES  ■ CONSTRUCTORS AND OTHER TOOLS  ■ OPERATOR OVERLOADING, FRIENDS,AND REFERENCES  □ STRINGS  □ POINTERS AND DYNAMIC ARRAYS | □ SEPARATE COMPILATION AND NAMESPACES  □ STREAMS AND FILE I/O  □ RECURSION  □ INHERITANCE  □ POLYMORPHISM AND VIRTUAL FUNCTIONS  □ TEMPLATES  □ LINKED DATA STRUCTURES  □ EXCEPTION HANDLING  □ STANDARD TEMPLATE LIBRARY  □ PATTERNS AND UML | |
| **Description:**  Define a class for complex numbers named **Complex**. A complex number is a number formed as **a + b \* i**, where **a** and **b** are numbers of type double, and **i** is a number that represents the quantity .   * The class Complex represent a complex number with two values:   **realValue (double)**, **imaginaryValue(double)**   * The class Complex has three constructors: * **Complex()**: construct a complex number where both realValue and imaginaryValue are 0. * **Complex(double r):** construct a complex number where the realValue is r and the imaginaryValue is 0. * **Complex(double r, double i)**: construct a complex number where the realValue is r and the imaginaryValue is i. * You should implement the following functions: * **double real()**: return the realValue. * **double imag()**: return the imaginaryValue. * **double norm()**: return the value of . * **double real(Complex c)**: return the realValue of a complex number c. * **double imag(Complex c)**: return the imaginaryValue of a complex number c * **double norm(Complex c)**: return the value of of a complex number c. * And you are required to Overload all the following operators: * **==**: Judge if the real and imaginary parts of two complex numbers are equal. * **+**: Add another complex number or a double type number. * **-**: Minus another complex number or a double type number. * **\***: Multiply with another complex number or a double type number. * **/**: Divide with another complex number or a double type number. * **>>**: Get the value of a complex number from the input stream with the format “x = realValue + imaginaryValue\*i”. * **<<**: Send complex numbers to the output stream with the format “realValue + imaginaryValue\*i”.   **Input:**  The input is a complex number in the format “x = realValue + imaginaryValue\*i”, where x is the variable name of the complex number.  \*\*The main() function in your submission will be replaced when judging.  \*\*You can use the main() function in “Other Notes” to test your program.  **Output:**  The result of executing your program with the given main function.  **Sample Input / Output：**   |  |  | | --- | --- | | Sample Input | Sample Output | | x = 3 + 4\*i  y = 5 + 6\*i | x = 0 + 0\*i  y = 3 + 0\*i  z = -3.2 + 2.1\*i  testing members and support functions as well as output operator:  complex number x = 3 + -4\*i  real part: 3  real part from friend real(x): 3  imaginary part: -4  imaginary part from friend imag(x) : -4  norm: 5  test operator ==:  x!=y  test complex arithmetic and output routines:  x = 3 + -4\*i  y = 1 + -1\*i  z = -3.2 + 2.1\*i  z = x + y = 4 + -5\*i  z = x \* y = -1 + -7\*i  z = x - y = 2 + -3\*i  z = x / y = 3.5 + -0.5\*i  d: 2 x: 3 + -4\*i  x+d: 5 + -4\*i  x-d: 1 + -4\*i  x\*d: 6 + -8\*i  x/d: 1.5 + -2\*i  d+x: 5 + -4\*i  d-x: -1 + 4\*i  d\*x: 6 + -8\*i  d/x: 0.24 + 0.32\*i  two/x: 0.24 + 0.32\*i  Getting data from standard input:  data read is:  x = 3 + 4\*i  y = 5 + 6\*i | |
| **□ Eazy,Only basic programming syntax and structure are required.**  **■ Medium,Multiple programming grammars and structures are required.**  **□ Hard,Need to use multiple program structures or more complex data types.** |
| **Expected solving time:**  30 minutes |
| **Other notes:**  #include"complex.h"  int main()  {  // test constructors  Complex x, y(3), z(-3.2, 2.1);  cout << "x = " << x << "y = " << y << "z = " << z << endl << endl;  x = Complex(3, -4);  cout << "testing members and support functions as well as"  << " output operator:\n"  << "complex number x = " << x << endl  << "real part: " << x.real() << endl  << "real part from friend real(x): " << real(x) << endl  << "imaginary part: " << x.imag() << endl  << "imaginary part from friend imag(x) : " << imag(x) << endl  << "norm: " << norm(x) << endl << endl;  cout << "test operator ==:" << endl << endl;  if (x == y)  cout << "x = y" << endl << endl;  else  cout << "x!=y" << endl << endl;  cout << "test complex arithmetic and output routines: \n\n";  y = Complex(1, -1);  cout << "x = " << x << "y = " << y << "z = " << z << endl << endl;  z = x + y;  cout << "z = x + y = " << z << endl;  z = x \* y;  cout << "z = x \* y = " << z << endl;  z = x - y;  cout << "z = x - y = " << z << endl;  z = x / y;  cout << "z = x / y = " << z << endl << endl;  //test of automatic conversion double -> complex by the constructor.  double d(2.0);  cout << "d: " << d << " x: " << x << endl;  cout << "x+d: ";  z = x + d;  cout << z << endl;  z = x - d;  cout << "x-d: ";  cout << z << endl;  z = x \* d;  cout << "x\*d: ";  cout << z << endl;  z = x / d;  cout << "x/d: ";  cout << z << endl;  z = d + x;  cout << "d+x: ";  cout << z << endl;  z = d - x;  cout << "d-x: ";  cout << z << endl;  z = d \* x;  cout << "d\*x: ";;  cout << z << endl;  z = d / x;  cout << "d/x: ";;  cout << z << endl;  //test whether double/complex and complex/complex give same result:  Complex two(2, 0);  cout << "two/x: ";  cout << two / x << endl;  cout << "\nGetting data from standard input: \n";  cin >> x >> y;  cout << "data read is: x = " << x << " y = " << y << endl << endl;  return 0;  } |