Name_____

Show the result and turn this sheet with your solutions to the instructor.

(**SimpleMath** Class) Create a class ***SimpleMath*** with attributes *number1* and *number2*, each of which defaults to 1.0. Provide methods that calculate their sum and quotient (*number1* divided by *number2*). The class has *set* and *get* methods for both *number1* and *number2*. It also has **two** constructors: one with no argument and another one that takes two arguments (i.e., *number1* and *number2*). The set methods and constructors should verify that *number1* and *number2* are each floating-point numbers larger than -9999.0 and less than 9999.0 and that *number2* should **not** equal to 0.0 (otherwise, the default value should be used instead). You must also provide a *toString* method that returns a string that displays *number1*, *number2*, sum, and quotient in a meaningful way.

Then write a program to test class *SimpleMath* (this program should be implemented in another class ***SimpleMathTest***) in which you create **two** *SimpleMath* instances with its two different constructors (your program must prompt the user for *number1* and *number2* inputs for one of the constructor). The program will display the information about these two instances (with use of *toString* method). Then choose one of the instances and change its fields/attributes' values (with use of *set* methods) with numbers that are out of the range (-9999.0, 9999.0), display the information about this instance again.

You must follow **Java Code Convention** (meaningful variable/instance names, proper use of indentation and blank lines) in writing your code **and** write a comment (2~4 lines at beginning of one of the classes). Draw a UML class diagram for *SimpleMath* (in the following space).

Over➔

How to Grade the Project (full score: **30** points):

- I'll use the following table to grade different portions of your program.
- If *I cannot compile your project successfully, the Implementation Total scores will be cut by half* (i.e., **multiplied by 0.5**).

UML diagram (**4 points**)	
Java Code Convention (**4 points**)	
Implementation (total **22 points**):	
constructors (**4 points**)	
set methods (**4 points**)	
get methods (**2 points**)	
toString method (**3 points**)	
Methods for calculating sum and quotient (**4 points**)	
Testing class *SimpleMathTest* (**5 points**)	
Implementation total score:	
Compiled successfully?	

(**Bonus 4 points**) Add another constructor that receives a reference to another *SimpleMath* object and test this constructor in the program.