Programming Project 1

The first programming project involves writing a program that computes the minimum, the maximum and the average weight of a collection of weights represented in pounds and ounces that are read from an input file. This program consists of two classes. The first class is the `Weight` class, which is specified in integer pounds and ounces stored as a double precision floating point number. It should have five public methods and two private methods:

1. A public constructor that allows the pounds and ounces to be initialized to the values supplied as parameters.
2. A public instance method named `lessThan` that accepts one weight as a parameter and returns whether the weight object on which it is invoked is less than the weight supplied as a parameter.
3. A public instance method named `addTo` that accepts one weight as a parameter and adds the weight supplied as a parameter to the weight object on which it is invoked. It should normalize the result.
4. A public instance method named `divide` that accepts an integer divisor as a parameter. It should divide the weight object on which the method is invoked by the supplied divisor and normalize the result.
5. A public instance `toString` method that returns a string that looks as follows: *x* lbs *y* oz, where *x* is the number of pounds and *y* the number of ounces. The number of ounces should be displayed with three places to the right of the decimal.
6. A private instance method `toOunces` that returns the total number of ounces in the weight object on which is was invoked.
7. A private instance method `normalize` that normalizes the weight on which it was invoked by ensuring that the number of ounces is less than the number of ounces in a pound.

Both instance variable must be private. In addition the class should contain a private named constant that defines the number of ounces in a pound, which is 16. The must not contain any other public methods.

The second class should be named Project1. It should consist of the following four class (static) methods.

1. The main method that reads in the file of weights and stores them in an array of type `Weight`. It should then display the smallest, largest and average weight by calling the remaining three methods. The user should be able to select the input file from the default directory by using the `JFileChooser` class. The input file should contain one weight per line. If the number of weights in the file exceeds 25, an error message should be displayed and the program should terminate.
2. A private class method named `findMinimum` that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the smallest weight in that array.

3. A private class method named `findMaximum` that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the largest weight in that array.
4. A private class method named `findAverage` that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the average of all the weights in that array.

Be sure to follow good programming style, which means making all instance variables private, naming all constants and avoiding the duplication of code. Furthermore you must select enough different input files to completely test the program.

Grading Rubric:

The following grading rubric will be used to determine your grade:

Attribute	**Meets**
Weight class	40 points It should have five public methods and two private methods: 1. A public constructor that allows the pounds and ounces to be initialized to the values supplied as parameters. 2. A public instance method named lessThan that accepts one weight as a parameter and returns whether the weight object on which it is invoked is less than the weight supplied as a parameter. 3. A public instance method named addTo that accepts one weight as a parameter and adds the weight supplied as a parameter to the weight object on which it is invoked. It should normalize the result. 4. A public instance method named divide that accepts an integer divisor as a parameter. It should divide the weight object on which the method is invoked by the supplied divisor and normalize the result. 5. A public instance toString method that returns a string that looks as follows: x lbs y oz, where x is the number of pounds and y the number of ounces. The number of ounces should be displayed with three places to the right of the decimal.

	6. A private instance method toOunces that returns the total number of ounces in the weight object on which is was invoked. 7. A private instance method normalize that normalizes the weight on which it was invoked by ensuring that the number of ounces is less than the number of ounces in a pound. Both instance variable must be private. In addition the class should contain a private named constant that defines the number of ounces in a pound, which is 16. The must not contain any other public methods.
Project1 class	40 points It should consist of the following four class (static) methods. 1. The main method that reads in the file of weights and stores them in an array of type Weight. It should then display the smallest, largest and average weight by calling the remaining three methods. The user should be able to select the input file from the default directory by using the JFileChooser class. The input file should contain one weight per line. If the number of weights in the file exceeds 25, an error message should be displayed and the program should terminate. 2. A private class method named findMinimum that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the smallest weight in that array. 3. A private class method named findMaximum that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the largest weight in that array. 4. A private class method named findAverage that accepts the array of weights as a parameter together with the number of valid weights it contains. It should return the average of all the weights in that array.
Test Cases	10 points

	Test cases are supplied in the form of table with columns indicating the input values, expected output, actual output and if the test case passed or failed. Select at least 3 different input files to completely test the program. Test cases were included in the supporting word or PDF documentation.
Documentation and Style guide	10 points Screen captures were provided and labeled for compiling your code, and running each of your test cases. Header comments include filename, author, date and brief purpose of the program. In-line comments used to describe major functionality of the code. Meaningful variable names and prompts applied. Class names are written in UpperCamelCase. Variable names are written in lowerCamelCase. Constant names are in written in All Capitals. Braces use K&R style**.** Declare all instance variables private. Avoids the duplication of code.