COMP1406 – Winter 2019

Submit a single file called `assignment2.zip` to the submission server
http://134.117.31.149:9091/

Your zip file must contain a directory called `comp1406a2` and all of your `.java` files must be in this directory. Do not include your `.class` files.

Do NOT import or use any classes other than ones provided in the java.lang package or specified in the assignment. You can use String, Object, any primitive Wrapper classes (Integer, Double, etc.).

This assignment has 50 marks.
5 marks for style (hand marked by a TA)
45 marks for correctness marked by the server.

0 Style [5 marks]

Part of your grade for this assignment will be for style. Style for this assignment will consist of

(a) Good use of inline comments.

(b) Good attribute, method and variable names (when you create them).

(c) Good use of whitespace (indenting, empty lines).

(d) Good attribute choices for storing the state of your objects.

1 Temperature [15 marks]

Complete the provided `Temperature` class. Add any attributes and helper methods as needed but keep in mind that testing will involve only the methods you are asked to write/complete. You must complete the constructors and methods in the provided class (without changing any signatures, return types, or modifiers).

In this problem you will need to be able to convert temperatures between Celsius, Fahrenheit and Kelvin. For help, see `https://en.wikipedia.org/wiki/Conversion_of_units_of_temperature`

A temperature object holds a single temperature and displays it in one of the three scales. Once a scale has been set, it will display the temperature in that scale until changed. The default scale is Celsius if not specified.

For one of the methods, `setTemp(double temp, String scale)`, the specification is a bit open ended. In this method, you will try to allow any scale input string that *resembles* the intended input. At the very least, it must work correctly for the three input strings `"CELSIUS"`, `"FAHRENHEIT"` and `"KELVIN"`. In order to receive full marks for this method it must be able to process more strings correctly. For example, see *** in the examples below. (Don't worry about processing typos of the names. Case and shortened versions should be handled.)

Some examples of using a `Temperature` object:

```
Temperature t = new Temperature(10.1);
System.out.println(t.getScale());    // outputs the char 'C'
System.out.println(t);               // outputs 10.1C
```

```
t.setScale(Scale.FAHRENHEIT);
System.out.println(t);                // outputs 50.18F
System.out.println(t.getScale());     // outputs the char 'F'
t.setTemp(12.25, "Kel");              // ***
System.out.println(t);                // outputs 12.25K
```

When you set a temperature (without explicitly stating the scale), it is assumed that the input temperature is in whatever scale the object is currently using.

Note: You should have **no** static attributes or methods in your class (unless they were supplied in the starter code).

2 Weather [30 marks]

This will be your first problem where you need to deal with several classes. This is your first **larger** problem to solve and it can be overwhelming if you try to sit down and solve the it in one go. Before starting to code, be sure to read through all the provided classes that you will use (and not make). Read through the `BuildDatabase` program to understand how to use these other provided classes. Do ONE thing at a time.

You will implement a simple weather database. The database will store several weather stations and allow for basic queries about the data. Each weather station keeps some records of the weather (at the location of the station). A weather record consists of the temperature at a given time.

You are provided with some classes that you must NOT change: `Compare`, `Scale`, `TimeStamp` and `WeatherReport`.

You will need to use your `Temererature` class from the previous problem in this problem. You must complete the methods in the `WeatherDatabase` and `WeatherStation` classes. The API is specified as javadoc comments in the `.java` files. You must decide how to store the state needed for the objects of these classes. Add any attributes and helper methods that you need. Keep in mind that we will only test your code by calling the methods that you are asked to complete. You are NOT allowed to use any external classes unless they are in `java.lang` or `comp1406a2`.

There is a `BuildDatabase` program also included in the `comp1406a2` package. This program will build a small database consisting of three weather stations that each hold some number of weather reports. Read through this code to see the classes/objects in use. You can use this class as a starting point for testing your classes.

Submission Recap

A complete assignment will consist of a single file (`assignment2.zip`) that has a single folder/directory called `comp1406a2`. The `comp1406a2` folder will have the following three files included:

- `Temperature.java`
- `WeatherStation.java`
- `WeatherDatabase.java`

All files must have the `package comp1406a2;` directive as the first line. Your code will NOT compile if it does not have this and you will receive zero correctness marks if your code does not compile.

Do NOT modify `Scale.java`, `Compare.java`, `TimeStamp.java`, or `WeatherReport.java`. We will use our own copy of these files and any changes you make will be lost.