COMP1006/1406 – Winter 2023

Submit your `.java` files to Gradescope.

The assignment is out of 10 and is worth 10% of your final grade.

This is Part One of the assignment. Part Two will posted soon.

The assignment has two parts. While the due date for the entire assignment is Friday, February 17th, the intention is that part one should be completed by Friday, February 10th, and that part two should be completed by Friday, February 17th.

Start early!

This assignment will completely be graded for correctness.
Be sure that your code compiles and runs. Test your code!

There is a 48-hour grace period in which submissions will be accepted without penalty. If need be, you can submit this assignment up to Sunday, February 19th, at 11:59pm without any penalty. However, there will not be any office hours and no guarantees that questions will be answered on discord over the weekend. Be sure to start early and submit often.

Do NOT change the input types, output types or modifiers of the methods you are completing. If you change any of these, it will not compile when submitted.

Part One

Temperature [4 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**).

A *temperature* consists of a value (magnitude) and a scale. For example, if the temperature is $-17.4^\circ C$, then its *value* is -17.4 and its *scale* is Celsius. The valid scales that we will consider for our Temperature objects will be Celsius, Fahrenheit or Kelvin. Once a scale has been set, a Temperature object will always display its temperature in that scale until the scale is changed. The default scale is Celsius if not specified.

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

The three scales are represented by Strings in the provided Scale class (class attributes). For this assignment, the purpose of the Scale class is to provide a consistent naming scheme for the different scales. Essentially, we assign fixed names for the three scales and use these everywhere in the code (so that a simple spelling mistake in your code does not result in failing every text case).

Some examples of using a Temperature object:

```
Temperature t = new Temperature(10.1);
System.out.println(t.getScale());     // displays "CELSIUS"
System.out.println(t.toString());     // displays "10.1C"
t.setScale(Scale.FAHRENHEIT);         // change scale
System.out.println(t.toString());     // displays "50.18F"  (notice it converted the value!)
System.out.println(t.getScale());     // displays "FAHRENHEIT"
t = new Temperature(12.25, "KELVIN"); // scale input is not from Scale!
System.out.println(t.getScale());     // displays "NONE"
System.out.println(t.toString());     // displays "0.0N"
```

Note: You must provide the **state** (instance attributes) for the Temperature class. You must decide what state to store for this problem. But, you must only use instance attributes. You should have no static attributes in your class.

Note: You must use **information hiding** in your class.

Note: Temperature values are floating point numbers. If an object's value is expected to be 0.1 and your output (from getValue() is 0.09999999999998), that is OK. You are **not** asked to perform any rounding in the getValue() method.

Note: The provided toString() method will always display your Temperature objects using three (3) decimal places. Rounding *will* happen automatically here in the String output.

Note: A program called SimpleTemperatureProgram is provided with the code shown above that you can use as a starting point for your own testing if you wish.

Include your Temperature.java file in your .zip file when you submit.

Part One Reminders

1) You should have no static attributes in your class.
2) Read the specifications in the skeleton file carefully!
3) Do **not** use Strings that *look* like the attributes from `Scale`
4) Be sure to use **information hiding**!