COMP1006/1406 – Fall 2017

Submit a single file called `assignment2.zip` to cuLearn.
The assignment is out of 100 marks.

1: Temperature [30 marks]

Complete the provided `Temperature` class. Add any attributes and helper methods as needed. 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.

Examples:

```
Temperature t = new Temperature(10.1);
System.out.println(t.getScale());    // outputs the char 'C'
System.out.println(t);               // outputs 10.1C
t.setScale("F");
System.out.println(t);               // outputs 50.18F
System.out.println(t.getScale());    // outputs the char 'F'
```

Note: Repeatedly changing the scale should not "change" the value of the temperature. For example,

```
Temperature t = new Temperature(10.1);
System.out.println(t);
for(int i=0; i<10000; i+=1){
   t.setScale("F");
   t.setScale("C");
}
System.out.println(t);
```

Should print out identical strings.

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

Note: Your code must use encapsulation. Your grade will be reduced by 5 marks if you do not use encapsulation.

Mark breakdown: 5 marks for Style, 25 marks for Correctness

Put your `Temperature.java` file in your `assignment2.zip` file.

2: Extreme Temperatures [30 marks]

Complete the `MaxTemp` class. The class consists of a single constructor and a single getter method.

- The constructor takes an array of `Temperature` objects as input.
- The getter method either returns an array of doubles with exactly two doubles in it or returns `null`. If the object was created with one more more `Temperature` objects in the constructor's input array then the output consists of the maximum temperature of all `Temperature` objects passed to the constructor and a count of how many times that maximum was present in the array passed to the constructor (in that order). If zero `Temperature` objects were passed to the constructor (in the array) then the getter returns `null`.

  Note: The max temperature returned must be displayed in the **Kelvin** scale.

  Note: Different `Temperature` objects in the array passed to the constructor may have different temperature scales set for themselves.

Since the `Temperature` objects will store a floating point number for the temperature, you will use the provided `EPSILON` constant in the `MaxTemp` class and consider two temperatures as equal if their absolute difference is smaller than `EPSILON`. Therefore, if `Math.abs(temp1 - temp2) < ESILON` then temp1 and temp2 are considered equal.

For example, if the array

```
new Temperature[]{new Temperature(1001.12, "K"),
                  new Temperature(-200.0, "F"),
                  new Temperature(1001.11, "K")}
```

is passed to the constructor, and `EPSILON = 0.1`, then the `getMax` getter will return `[1001.12, 2.0]`.

The return value should still display the maximum temperature (in K). In the example above, even though we consider 1001.11 and 1001.12 the "same", 1001.12 is still the max to be returned.

Note: Your code must use encapsulation. Your grade will be reduced by 5 marks if you do not use encapsulation.

Mark breakdown: 5 marks for Style, 25 marks for Correctness

Put your `MaxTemp.java` file in your `assignment2.zip` file.

3: Tic-Tac-Toe [40 marks]

Implement the game of Tic-Tac-Toe. Your game must consist of at least three (3) classes: `Board`, `Player`, and `TicTacToeApp`. You can add more classes if you wish.

- The `TicTacToeApp` program will run the game, using a `Board` object and two `Player` objects.
- The `Player` class will implement a player in the game. A player could be a human user (and so user input will be needed) or a computer.
- The `Board` class will be the board in the game. It can either be a 3x3 grid or a 5x5 grid. The constructor of the class will create an appropriate grid.
- User input for specifying which grid element in the board will be a comma separated list of two numbers, where the first number is row (starting at 0) and the second number is the column (starting at 0). The amount of whitespace around the numbers (and comma) can be arbitrary.
- A single command line argument (for the `TicTacToeApp` program) will specify if the game is a 3x3 or 5x5 grid. Acceptable command line arguments will be `"3"`, `"5"`, `"three"` and `"five"`. For any other command line argument (or lack) your program will terminate with a usage message (of how to run the program properly).
- The game board must be displayed before any player makes a move.
- The game should be easy to play.

Note: Style for this problem includes class design. Encapsulation is required.

Mark breakdown: 15 marks for Style, 25 marks for Correctness

Put your `Board.java`, `TicTacToeApp.java`, `Player.java` and any other `.java` files you created in your `assignment2.zip` file.

Submission Recap

A complete assignment will consist of a single file (`assignment2.zip`) with at least following five files included: `Temperature.java`, `MaxTemp.java`, `TicTacToeApp.java`, `Board.java` and `Player.java`.