CECS 277 LAB INHERITANCE 2

OBJECTIVE: Reinforce the lecture material using the automated quiz example by giving you some experience in extending an existing class.

INTRODUCTION: Please remember the coding standards here.

You will start with the Question.java file from the book as your Question supertype. The first subtype has already been done for you in ChoiceQuestion.java. The start of the demonstration code for this is QuestionDemo2.java. You need to copy all three of these into a new Java project called CECS 277 Lab Inheritance 2. Note that this code is straight from the text without any modification. The author violates a number of our coding standards (!gasp!) so you will need to be sure to fix those violations as part of this lab.

PROCEDURE:

1. Create the CECS 277 Lab Inheritance 2 project in your IDE of choice, bring in the above three Java files, and test it to make sure that it works before you start changing it.
2. Create a new subtype to the Question class:
    a. Call this new subtype: `TrueFalseQuestion`.
    b. Create a new constructor for it:
        i. Accepts two arguments: the question statement (a `String`), and a `Boolean`.
            1. The question statement text will be a declaration of some sort. For instance, "It is possible to override every single method that you inherit from a supertype." This is patently false by the way, remember that final methods cannot be overridden.
            2. The `Boolean` argument tells whether the statement is true or false.
                a. A value of true, means that the statement is true, and well, I don't need to tell you the rest.
            3. Use the setText and setAnswer methods of the **supertype** to initialize the TrueFalseQuestion.
                a. Remember, there can be only two answers to a True/False question. So that second parameter determines what the right answer is.
                b. Just trust me on this one, if that second argument is true, then make the correct answer "T", otherwise, make it "F".
    c. Create an override `display` method for the new `TrueFalseQuestion` class.

i. We want the `display` method to cough up the string: "True (T) or False (F): " before the actual statement that the user is evaluating.
ii. Be sure to use the supertype's `display` method as part of your display override.

d. Override the `checkAnswer` method of the supertype so that it takes just the **first character** of the response, **capitalizes it**, and **then** checks the answer. So, the user could get a true false question like the one above and respond with "frankly Scarlett, I don't give a d___" and get it right. The first letter of their response is "f", so you upper case that, and compare it to "F" for the sample question that I just gave you, and voila, it's correct. Be sure to make use of the supertype's `checkAnswer` method in your override.

3. Then, in the demo program, create a new true false question with the arguments "It is possible to override every single method that you inherit from a supertype." and the `Boolean` value of `false`. Test the result and make sure that it works.
4. In your demo program, rather than use a separate variable for each question, create an array of type `Question`, and then use an extended `for` loop to go through the array of `Question` instances to use your `presentQuestion` method. You will have to make a couple of changes to `presentQuestion` along the way.
5. Finally, run the demo, try providing the right answers and the wrong answers to the questions to make sure that the `checkAnswer` routine is working properly in each case.

WHAT TO TURN IN:

- `ChoiceQuestion.java` that adheres to our coding standards.
- `Question.java` that adheres to our coding standards.
- `QuestionDemo2.java`
- `TrueFalseQuestion.java`
- Your sample output as `console.txt`.