PRACTICE ASSESSMENT 3

OVERVIEW

Create a Java console application that utilizes OOP concepts. Your completed application should include classes and methods that perform the required actions and follow proper naming conventions. For this challenge, you will need to submit your code within the Java project named **practice-assessment-3-<username>**. All classes must be created in the package **co.grandcircus**.

BUILD SPECIFICATIONS

The assessment is worth ten points, one for each of the test cases below. **Pay special attention to the spelling and capitalization of the items in bold**.

1. Create a class called **Ingredient**.
    a. Private fields: **String name**, **String unit**, **double quantity**.
        i. Include getters and setters for these fields
    b. Constructors:
        i. no-arg constructor
        ii. overloaded constructor that takes in three arguments to set name, unit, and quantity.
        iii. overloaded constructor that takes in two arguments to set name and quantity. Set unit to empty string (""). Note: empty string is not the same as `null`!
2. Create a class called **Recipe**.
    a. Private fields: **String title**, **List<Ingredient> ingredients**, **String instructions**.
        i. Include getters and setters for these fields
    b. Constructors:
        i. no-arg constructor.
        ii. overloaded constructor with one parameter to set title. It also sets ingredients to an empty ArrayList and instructions to "N/A".
3. Create a subclass of Recipe called **BakingRecipe**.
    a. It has an additional private field: ovenTemperature.
    b. Constructors:
        i. no-arg constructor
        ii. overloaded with two parameters to set title and ovenTemperature. It also sets ingredients to an empty ArrayList and instructions to "N/A". This constructor should call the superclass constructor.

4. Create a **RecipeApp** class with a main method that does the following:
   Create an instance of **Recipe** with the following values:

   a. title: **"S'more"**
   b. ingredients
      i. name: marshmallow, unit: "", quantity: 1.0 (Hint: use the 2-arg constructor)
      ii. name: chocolate, unit: "oz", quantity: 0.5
      iii. name: Graham cracker, unit: "", quantity: 1.0
   c. instructions: "First, you take the graham. You stick the chocolate on the graham. Then, you roast the 'mallow. When the 'mallow's flaming, you stick it on the chocolate. Then, you cover it with the other end. And then you scarf."

   Create an instance of **BakingRecipe** with the following values:
   a. title: **"Baked apple"**
   b. Ingredients
      i. name: apple, unit: "", quantity: 5.0
      ii. name: sugar, unit: "c", quantity: 0.125
      iii. name: cinnamon, unit: "TBSP", quantity: 1.0
   c. bakingTemperature: 350
   d. instructions: "Cut apples in halves or quarters and remove cores. Place in baking dish. Sprinkle with sugar and cinnamon. Bake at 350 for 30 minutes."

5. Within **RecipeApp,** create a method **public static Recipe easiestRecipe()** that will take in a parameter of type **List<Recipe>** and return the **Recipe** with the smallest total quantity of ingredients (ignoring weight).
   a. Call this method from main and pass the **List<Recipe>** containing **Baked apple** and **S'more** as an argument.