Abstract/Inner Class Exercise

Exercises

1. Define a class named Factorial that has the following methods:
   a. Constructor - Factorial(lowerLimit, upperLimit)
   b. Implement an iterator which will return the factorials of numbers starting at lowerLimit and ending at upperLimit. Make sure the class implements the Iterable interface.
   c. Define a main method that uses the iterator you defined above.

2. A Car class is defined as follows:

   public abstract class Car {
      private String make;
      public abstract void start();
      public void setMake(String make) { this.make = make; }
      public String getMake() { return make; }
   }

   Complete the assignment statement below so we can define a hybrid car object that has an instance variable representing battery power. In addition, the object will be associated with a start() method that decreases the battery power by 100 units. The initial battery power is 3000 units. **You must use an anonymous inner class.**

   public static void main(String[] args) {
      Car Hybrid = // COMPLETE THIS ASSIGNMENT
   }

3. Define an Abstract class named Drink which defines the following methods:
   a. Constructor with two parameters: serialNo (drink’s serial number) and description (brief description about the drink). A drink has a sugarContent which is by default is 0.
   b. toString method that prints the serial number followed by the description and the sugar content of the drink.
   c. An increaseSugarContent method that allow us to increase the sugar content by a particular integer amount. For example, if p is a concrete class that extends Drink, then we should be able to call increaseSugarContent as follows:
      p.increaseSugarContent(10).increaseSugarContent(20);
   d. An abstract method named getIngredients which returns a string with the drink’s ingredients.

4. Define a Concrete class named Popsi (yes, o not e 😊) which extends the Drink class above. A Popsi has as ingredients sugar and caffeine. The class constructor will initialize the Drink with a serial number equal to 10 and “Great carbonated soda” as a description.

5. Define an finalize() method for the Popsi class. In order to test the finalize() method use the System.gc() method which suggests that the Java Virtual Machine expend effort toward recycling unused objects in order to make the memory they currently occupy available for quick reuse. Hint: set references to null.