Regis University CC&IS
CS210 Introduction to Programming
Java Programming Assignment 4: Objects and Instance Methods

All of your Alice programs have been using objects already. And the predefined function and procedure methods in Alice are equivalent to instance methods for those objects in Java. You also defined your own methods within an Alice class, which had to be applied to a specific object in order to run.

This assignment will apply the same concepts to Java. You will create a new class, and use the new class to create an object in Java. Methods defined within the new class (instance methods, similar to the Alice procedure and function methods) will be called using the dot (.) operator, to send a message to the object.

> WARNING: This assignment will be ***more challenging*** than the previous assignments, so make sure to ***start early*** and ***allocate enough time*** to complete it, including time you might need to seek any necessary help.

Problem Summary

Suppose you wanted to modify your BAC Calculator from Java programming assignment 3.
You could:

- Store all the previous data about one drinker as data fields within an **object**
- Add a drinker identification field
- Revise the calculation methods to be instance methods within the object's class
    - NOTE: This will eliminate the need to pass the drinker's data into the methods via parameters.

Since you will be modifying your BAC Calculator program from Java Assn 3, make sure you use your instructor's feedback to fix any mistakes from Java Assn 3 before you submit Java Assn 4.

Overview of Program

This program will contain two classes, in two separate files within your project:

- A new **Drinker** class to define the data fields and methods for a Drinker object, containing:
    - Five data field (attribute) definitions for a Drinker object
    - A **constructor** method to create a Drinker object
    - Five **setter** methods to set the values for all of the data fields
    - Three **getter** methods to get the values of two of the data fields
    - An instance method to compute the volume distribution for men.
    - An instance method to compute the volume distribution for women.
    - An instance method to compute and display the current BAC.
- The main **BACcalculator** class, modified to define only 3 methods, containing:
    - A **main** method to display the program description, create an object, read the inputs from the user, and call the other methods.
    - A static method to display a description of what the program will do.
    - A static method to display drinker's information.

Since there will be multiple files, the program will be submitted via a **.zip** file (see last page).

Program Requirements

Modify the program you wrote for Java Assn 3, as follows:

1. Define an additional Java **class** (File | New File | Java Class) named:

   `Drinker`

   Within the **Drinker** class, define the following **private** data fields to store properties about the person drinking:

   - drinker identifier (**String**) ← new!
   - weight in kilograms
   - height in meters
   - number of drinks consumed
   - elapsed time since drinking began in hours

   Remember that all private data fields defined at the class-level can be accessed by any method within the ***same*** class (without parameter passing).

   Also within the new **Drinker** class, you will define instance methods that can be used with an object of the Drinker class, as follows:

   - Define a default **constructor** with ***no*** parameters that will create the object with the following initial data values set:

     Drinker identifier set to `""` (an empty String)

     Height and weight, both set to 0.0

     Number of drinks consumed and elapsed time, both set to 0

   - Create **setters** for each data field, as follows:
     - Setter for drinker identifier
       - Input parameters will be two Strings:
         - the drinker's last name
         - the drinker's 10-digit phone number, including dashes (e.g. 333-333-3333)
       - Use the built-in Java **String** methods to ***uppercase the last name,*** and then ***extract*** the first 3 letters of the last name and ***concatenate*** them with the last 4 digits of the phone number to create a unique identifier for the drinker.
       - Set the data field to this value.
     - Setter for weight
       - Input parameter will be weight in pounds
       - Define and use a local ***constant:*** the number of pounds in one kilogram (2.20462)
       - Method will convert the parameter weight to kilograms and then set the weight data field to that value.
     - Setter for height
       - Input parameter will be height in inches
       - Define and use a local ***constant:*** the number of inches in one meter (39.3701)
       - Method will convert the parameter height to meters and then set the height data field to that value.

- Setter for number of drinks consumed
  - No input parameters
  - Use descriptive prompt and read number of drinks consumed
  - Set the data field to the value input by user
- Setter for hours elapsed since drinking started
  - No input parameters
  - Use descriptive prompt and read hours passed while consuming the drinks
  - Set the data field to the value input by user

- Create **getters** for the following data fields in the **Drinker** class.
  - Get the drinker identifier
  - Getter for number of drinks consumed
  - Getter for hours elapsed since drinking started

  Each getter should return the data field's current value.

- Move the volume distribution calculation (methods 2 and 3) in Java Assn 3 from the **BACcalculator** class to the new **Drinker** class. Then modify the methods, so they can be used as **instance** methods in the new **Drinker** class, as follows:
  - Eliminate the static keyword (since they will now be used with objects)
  - Eliminate the parameter lists (since they will now use data fields).
  - Modify the formulas, so that they will directly use the object data fields, instead of parameters.
- Move the BAC calculation (method 4) in Java Assn 3 from the **BACcalculator** class to the new **Drinker** class. This method will be modified to ***calculate and display*** the BAC, instead of ***returning*** the BAC. Modify the method, as follows:
  - Change the method header to indicate nothing will be returned.
  - Eliminate all parameters, except the volume distribution and metabolic rate to use.
  - Add a third parameter:
    - ❖ a gender string (which will contain either "male" or "female")
  - Modify the conversion formula to convert the data field weight in kilograms to grams (instead of the previous conversion from pounds to grams) and store the new value. Then use the new value in the BAC formula.
  - After calculating the current BAC, display a description, including the gender, and the BAC value to 3 decimal places. Only one line will be displayed (see next page for examples).
    - ❖ NOTE: No decision statements will be necessary to do this!

    For example, if the gender parameter contains the word "male" and the current BAC is 0.0933333333, the output will be:

```
a male has a current BAC of approximately 0.093
```

    But if the gender parameter contains the word "female" the output will be:

```
a female has a current BAC of approximately 0.093
```

2. Within the original **BACcalculator** class that contains the **main** method:

   - Make sure you deleted the old static calculation methods (2 – 4) from the **BACcalculator** class when you moved them to the **Drinker** class.
   - Modify the assignment 3 static method (5) so that it will display ***only*** the **drinker's information**. The method will:
     - Have three parameters:
       - the Drinker object
       - the previously read height in inches
       - the previously read weight in pounds
     - Use the Drinker object and a **getter** to display the drinker identifier
     - Display the Drinker's height in inches and weight in pounds (originally read from user)
     - Use the Drinker object and the **getters** to display the number of drinks consumed and the elapsed time while drinking.

     Sample of Output from this method

     ```
     Calculations for Drinker XYZ2345,
     who is 66 inches tall and weighs 150 pounds.

     After drinking 6 drinks, in 3 hours:
     ```

   - Modify the **main** method to perform the following tasks (modified from Java Assn 3):
     - Remove the constants for converting the height and weight, since they are now located in the setters (but keep the metabolic rate constants).
     - After displaying program description, create an ***object*** of the new **Drinker** class type.
     - Use descriptive prompts to read:
       - the drinker's last name (you can assume there will be no spaces in the last name and that it will contain at least 3 letters).
       - the drinker's phone number
     - Same as in assn 3, use descriptive prompts to read: height in inches and weight in pounds.

     Sample Input read from main method

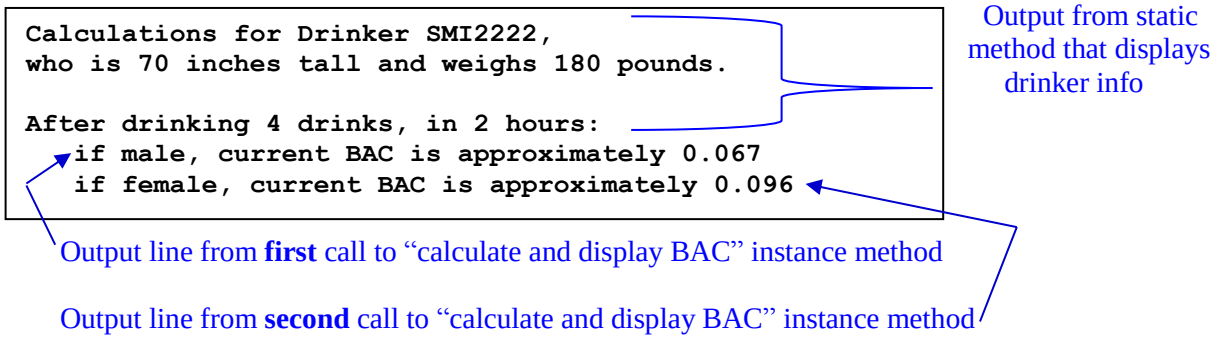
     ```
     Enter the drinker's last name: Smith
     Enter the drinker's phone number: 222-222-2222
     Enter drinker's height (in inches): 70
     Enter the drinker's weight (in pounds): 180
     ```

     - Use the Drinker object and the **setters** to set the values for **all** of the data fields of the object:
       - Be sure to pass any necessary parameters to the **setters**.
       - Note that the **setters** for the number of drinks and time elapsed will prompt the user for input.
     - After setting all the data fields, display a couple of blank lines to separate input and output.

- Using the Drinker object:
  - Call the volume distribution calculation method for **men** (method 2) to compute the volume distribution for men and ***save the result returned***.
  - Call the volume distribution calculation method for **women** (method 3) to compute the volume distribution for women and ***save the result returned***.
- Call the static method to display the drinker's information.
- Use the Drinker object to call the instance method to calculate and display the BAC ***twice***:
  - The first call should pass in "male" as the gender String, the volume distribution computed for men, and the metabolic rate for men.
  - The second call should pass in "female" as the gender String, the volume distribution computed for women, and the metabolic rate for women.

The **main** method code should ***not*** directly calculate any results, nor display any drinker information or BAC information. All this should be done from within methods that are called by the main method.

Complete Sample Output



NOTE: For an example of a Java program using objects, see Online Content section 10.13.

WARNING: The methods must be implemented exactly as specified in these requirements. If your program produces correct output, but you did not implement the methods as specified (with correct parameter passing and return values), you will lose a significant number of points.

See next page for an outline of what your code should look like.

Coding Standards

The program must also follow the **CS210 Coding Standards** from Content section 6.10.

You must ***include*** the following comments:

- Comments at the ***top of the file, above*** the **main** class, describing what the class does
  - Include **tags** with the author's name (i.e. your full name) and the version of the code (e.g. @version 2.0, Java Assn 4 modified from Java Assn 3)
- Comments at the ***top of each method***, ***above*** the method header, describing what the method does (***only*** this method – do not refer to actions of any other methods)
  - Include **tags** with names and descriptions of ***each*** parameter and return value.

Delete any default comments supplied by the IDE that you did not use.

Debugging and Testing

Run, test, and debug your Java program, until it works.

Then test your program with different inputs to make sure it provides correct results.

Outline of what your **`BACcalculator.java`** code file should look like:

```
/*
 * The blood alcohol content calculator program will do the following:
 *   Put program description here (expand to as many lines as needed)
 */
import java.util.Scanner;

/**
 * @author Mary Jones          // your full name
 * @version 2.0, Java Assn 4 modified from Java Assn 3
 */
public class BACcalculator {

    public static void main(String[] args) {
        // Call to method 1 goes here - display program description

        // Statement to create object goes here

        // Statements to read user inputs go here

        // Statements to call setters go here

        // Statements to call methods to compute volume distribution to here

        // Statement to call method to display drinker info goes here

        // Statements to call "calculate and display BAC" method twice go here
    }

    // Existing method to display program description goes here

    /**
     *  Description of method to display drinker information
     *
     * @param name - description of parameter
     *               (one @param comment line for each parameter)
     */
    public static void methodName (datatype parameter, ...) {
        // Method body statements go here
    }
}
```

Outline of what your **`Drinker.java`** code file should look like:

```
public class Drinker {
    // Data field definitions go here
    //      (i.e. list of private dataTypes & names)

    /**
     * Constructor
     *
     * @param name - description of parameter
     *               (one @param comment line for each parameter)
     */
    public Drinker () {
        // Statements to initialize data field values go here
    }

    /**                                  ← All setter methods go here (one example shown)
     * Setter setterName
     *
     * @param name - description of parameter (if necessary)
     *               (one @param comment line for each parameter)
     */
    public void setDatafieldname () {
        // Statements to calculate and set field value go here
    }

    /**                                  ← All getter methods go here (one example shown)
     * Getter getterName
     *
     * @return datafieldname - description (if necessary)
     */
    public dataTypeReturned getDatafieldname () {
        // Statement to return data field value goes here
    }

    /**            ← Both volume distribution calculation methods go here (one example shown)
     * Description of method to calculate volume distribution for men
     *
     * @return name - description (if necessary)
     */
    public dataTypeReturned methodName () {
        // Method body statements go here
    }

    /**
     * Description of method to calculate and display BAC
     *
     * @param name - description of parameter (if necessary)
     *               (one @param comment line for each parameter)
     */
    public void methodName (datatype parameter, ...) {
        // Method body statements go here
    }

}
```

Submission

This programming assignment is due by midnight of the date listed in the **Course Assignments by Week**.

Now that you have multiple class files within your project, you will submit a single **zip** file containing all of the files in your project.

- First export your project from NetBeans:
  - Highlight the project name.
  - Click on **File** from the top menu, and select **Export Project**.
  - Select **To ZIP**
  - Name your export file in the following format:
    `<lastname>Assn<x>.zip`

    For example:
    `SmithAssn4.zip`

    NOTE: Save this zip file to some other directory, not your project directory.

- Then submit your **.zip** file to the **Java Prog Assn 4** assignment submission folder (located under the **Assignments/Dropbox** tab in the online course).

  - Warning: Only NetBeans export files will be accepted.
    Do not use any other kind of archive or zip utility.

Grading

Your program will be graded using the **rubric** that is linked on the same assignment page from which this program requirements file was downloaded.

WARNING:
*Programs submitted more than 5 days past the due date will **not** be accepted,*
and will receive a grade of 0.