CSC-14400: Computer Science I – Lab #5

Dates of Importance:

Tuesday, November 21:	(Optional) Suggested Problems Due for Quiz Extra Credit
Tuesday, November 28:	(Required) In-class Exercise
Thursday, November 30:	(Required) In-class Quiz

Objectives:
This exercise will get you (further) experience with:

- creating your own class.
- iterative execution.
- using arrays.

Preliminaries
Before attempting to complete this assignment, you should be comfortable with the material from Labs 1-4 and Chapters 1-7 of the Horstmann text book.
The following are suggested problems that can be handed in for up to two bonus points in the upcoming quiz:

1. Consider each of the following code sequences *(a)-(i)* and

   - if it would not compile, give "does not compile" as your answer.
   - if it would result in a run-time exception, give "exception" as your answer.
   - if it would result in a (non-null) memory reference, give "memory reference" as your answer.
   - in *any* other case, show the resulting output.

   (a) `int a[]; System.out.print(a);`

   (b) `int b[]; System.out.print(b[0]);`

   (c) `int c[] = new int[20]; System.out.print(c);`

   (d) `int d[] = new int[20]; System.out.print(d[0]);`

   (e) `String e[] = new String[20]; System.out.print(e);`

   (f) `String f[] = new String[20]; System.out.print(f[0]);`

   (g) `String g[] = new String[20]; System.out.print(g[0].length());`

   (h) `String h[] = new String[20]; System.out.print(h.length());`

   (i) `String i[] = new String[20]; System.out.print(i.length);`

2. Suppose you have a class called `Student` which has (among other methods) a method called `getAverage` that takes no inputs and returns a `double`. Give Java code for each of the following:

   (a) Declare a variable that could hold a single object of type `Student` and initialize it using a default constructor.

   (b) Using the variable from the last problem, call the `getAverage` method. Note that this part of the problem and the previous part are essentially review of prior material.

   (c) Declare a variable to hold an array of 10 `Student` objects and initialize the array appropriately (without calling any `Student` constructors yet).

   (d) Write a loop that would initialize each element of the array from the last part to a default constructed `Student`.

   (e) Give code that would print out the average (by calling the `getAverage` method) for the `Student` found at index 7 of your array.

3. *Chapter 7*: p. 362: problems E7.1, E7.6, E7.9, E7.10 a-d,h-j (you do *NOT* need to solve parts e-g)

4. This question does not have to do with arrays, but is a commonly misused concept seen in a significant amount of lab work and quiz answers, so needs a bit more exploration. Consider the following code:

```
public class StaticCling
{
    private int notStatic;
    private static int isStatic;

    public StaticCling(int a, int b)
    {
        notStatic=a;
        isStatic=b;
    }
    // public static void setNotStatic(int newVal) // (part b)
    public void setNotStatic(int newVal)
    {
        notStatic=newVal;
    }
    // public static void setStatic(int newVal) // (part c)
    public void setStatic(int newVal)
    {
        isStatic=newVal;
    }

    public String toString()
    {
        return "nonstatic="+notStatic+", static="+isStatic;
    }
    public static void main(String args[])
    {
        StaticCling sc = new StaticCling(1,2);
        System.out.println(sc);
        StaticCling another = new StaticCling(99,42);
        System.out.println(another);
        System.out.println(sc);
        sc.setNotStatic(-8);
        System.out.println(another);
        System.out.println(sc);
        sc.setStatic(-999);
        System.out.println(another);
        System.out.println(sc);
    }
}
```

(a) What is the output of the above program?

(b) Change the header for the `setStatic` method so that it now reads `public static void setStatic(int newVal)` . Does this still compile? If not, *why* not?

(c) Change the header for the `setNotStatic` method so that it now reads `public static void setNotStatic(int newVal)` . Does this still compile? If not, *why* not?

The Exercise
Write a Java class called `GradeStatistics` that contains (at least) the following _three_ public methods:

1. a constructor that takes a Scanner as its only argument. The (already constructed) Scanner will contain a sequence of integer course assignment scores as follows:
   - the first integer in the Scanner will describe the largest possible score for the associated assignment.
   - all of the remaining integers in the Scanner each represent a score on the assignment. Each score will be between 0 and the largest possible score (noting that both 0 and the largest possible score represent possible values).
2. a method called `showDistribution` that takes no arguments and returns nothing. The method should print out a histogram representing the number of times each data value appeared in the Scanner. For example (assuming a maximum value of 10), if the (additional) data in the Scanner is:

   `8 5 3 4 5 0 1 7 2 8 9 5 3`,

   then the resulting histogram should look something like:

   ```
   [ 10]
   [  9]*
   [  8]**
   [  7]*
   [  6]
   [  5]***
   [  4]*
   [  3]**
   [  2]*
   [  1]*
   [  0]*
   ```

   Where each row is identified by its corresponding score and there is a * for each occurrence of that value found in the Scanner. Note that rows with no stars did not appear in the sequence of integers found in the Scanner.
3. a method called `findMode` that takes no arguments and returns an integer. The integer returned should be the test score that appeared most often. In the case of a tie (i.e. when more than one test score appears most often), this method should return the largest such test score. For the above example, this method would return 5, since it occurred three times in the input, and no other score appeared that often.

****PRE**-lab Work**
When you come into class for the lab exercise, you must have a corresponding java class that:

1. compiles without any errors <u>**and**</u> ...
2. is a sincere attempt at a solution to the problem. Dr. Blythe's judgment on this will be final, no exceptions.

If you do not meet *both* of the above requirements, you will be given a 0 for this lab and will have to leave class immediately - *no exceptions*!!!