CSE142 Self Assessment 2
Summer 2020

Name _____ Student ID (e.g. 1234567) _____

Section (e.g. AA) _____ Section TA _____

Rules/Guidelines

- This self assessment is closed-book, closed-note. You may only use the cheat sheet provided on Canvas
- If you write your answers on scratch paper, be sure to **clearly label** which question you are answering on the scratch paper.
- Do not focus on style for this self assessment (with an exception for problem 8, Critters). External correctness is all that matters.
- You are limited to Java constructs described in chapters 1 through 10 of the textbook.
- You may **NOT** use break or continue, or return from a void method.
- You are allowed to abbreviate "System.out.print" and "System.out.println" as "S.o.p" and "S.o.pln" respectively. You may also abbreviate "ALWAYS", "NEVER", and "SOMETIMES" to A, S, N respectively. You may **NOT** use any other abbreviations

Advice

- Read all questions carefully. Be sure you understand the questions *before* you begin your answer.
- The questions are not necessarily in order of difficulty. Feel free to skip around. Be sure you are able to at least attempt every question.
- Write clearly and legibly. This will make it easier for us to provide feedback on your self assessment.
- Relax. You are here to learn.

Time

- You have **2 hours** to complete this portion of the self assessment.

Grading

- Log onto Canvas to get the key for this self assessment and grade the mechanical questions once the 2 hours time has elapsed.
- Annotate the programming questions taking note of key differences. You do not need to debug the problem, but have a high level idea of where you may have gone wrong to help prepare for the one on one discussion session with your section TA. The Canvas quiz will help walk you through the types of annotations we are looking for.

Turn In

- Upload a scanned, annotated version of this exam to Canvas and answer the questions on the turn in page in preparation for the one on one.
- Make sure that you have scheduled your one on one session with your section TA

1. Reference Mystery. The following program produces 3 lines of output. Write the output in the box below, exactly as it would appear on the console.

```
import java.util.*;
public class Point {
   int x;
   int y;

   public Point(int initX, int initY) {
      x = initX;
      y = initY;
   }
}

public class ReferenceMystery {
   public static void main(String[] args) {
      Point p = new Point(11, 22);
      int[] a = {33, 44};
      int n = 55;

      mystery1(p, a, n);
      System.out.println(p.x + "," + p.y + " " + Arrays.toString(a) + " " + n);

      a[0] = a[1];
      p.x = p.y;

      n = mystery2(a, n);
      System.out.println(p.x + "," + p.y + " " + n);
   }

   public static int mystery1(Point p, int[] a, int n) {
      n = 0;
      a[0] = a[0] + 11;
      a[1] = 77;
      p.x = p.x + 33;
      System.out.println(p.x + "," + p.y + " " + Arrays.toString(a) + " " + n);
      return n;
   }

   public static int mystery2(int[] a, int n) {
      n = a[0];
      a[0] = a[0] + 11;
      a[1] = n + 11;
      return n;
   }
}
```

2. Array Simulation.

You are to simulate the execution of a method that manipulates an array of integers. Consider the following method:

```
public static void mystery(int[] a) {
   for (int i = 1; i < a.length - 1; i++) {
      a[i] = (a[i - 1] + a[i + 1]) / 2;
   }
}
```

In the left-hand column below are specific arrays of integers. You are to indicate in the right-hand column what values would be stored in the array after method mystery executes if the integer array in the left-hand column is passed as a parameter to mystery.

Original Array	Final Array
[1, 1, 3]	_____
[2, 1, 2, 4]	_____
[6, 13, 0, 3, 7]	_____
[-1, 6, 3, 5, -3]	_____
[7, 2, 3, 1, -3, 12]	_____

3. Inheritance Mystery. Consider the following classes:

```
public class KDot extends Zeltron {
   public String toString() {
      return "KDot " + super.toString();
   }
}

public class Paak extends Zeltron {
   public void method2() {
      System.out.println("Paak 2");
   }

   public String toString()() {
      return "Paak";
   }
}

public class Zeltron {
   public void method1() {
      method2();
      System.out.println("Zeltron 1");
   }
   public void method2() {
      System.out.println("Zeltron 2");
   }

   public String toString() {
      return "Zeltron";
   }
}

public class Gibbs extends Zeltron {
   public void method1() {
      System.out.println("Gibbs 1");
   }

   public void method2() {
      System.out.println("Gibbs 2");
   }
}

// client code
public static void main(String[] args) {
   Zeltron[] zeltrons = { new KDot(), new Gibbs(), new Zeltron(), new Paak() };
   for (int i = 0; i < zeltrons.length; i++) {
      System.out.println(zeltrons[i]);
      zeltrons[i].method1();
      zeltrons[i].method2();
      System.out.println();
   }
}
```

Given the classes to the left, write the output produced by the client code below exactly as it would appear on the console.

4. ArrayList Debugging.

Consider a static method called cloneOddsRemoveEvens that takes an ArrayList of integer values as a parameter and that replaces each odd number with two of that number and that removes all even values. For example, suppose that a variable called list stores the following sequence of values:

```
[3, 8, 19, 42, 7, 26, 27, -8, 193, 204, 6, -4]
```

and we make the following call:

```
cloneOddsRemoveEvens(list);
```

Afterwards the list should store the following sequence of values:

```
[3, 3, 19, 19, 7, 7, 27, 27, 193, 193]
```

Notice that each odd number has been duplicated (3, 19, 7, 27, 193) and that the even values have been removed (8, 42, 26, -8, 204, 6, -4). Below are several examples of how an ArrayList should be changed by the method.

ArrayList contents before call	ArrayList contents after call
[]	[]
[82]	[]
[3, 8, 15]	[3, 3, 15, 15]
[2, 4, 6, 8]	[]
[17, 5, 3, 9]	[17, 17, 5, 5, 3, 3, 9, 9]
[1, 2, 3, 4, 5, 6]	[1, 1, 3, 3, 5, 5]

The following is a proposed implementation of cloneOddsRemoveEvens:

```
public static void cloneOddsRemoveEvens(ArrayList<Integer> list) {
   for (int i = 0; i < list.size(); i++) {
      int val = list.get(i);
      if (val % 2 == 0) {
         list.remove(i);
      } else {
         list.add(i + 1, val);
      }
   }
}
```

This implementation has one or more bugs. Modify the implementation so that it behaves as described above. Your modified method should retain the same basic approach to the problem as the buggy implementation; you should not write an entirely new implementation.

You may not construct any extra data structures to solve this problem. You must solve it by manipulating the ArrayList you are passed as a parameter. See the cheat sheet for a list of available ArrayList methods.

5. File Processing.

Write a static method called underline that accepts as a parameter a Scanner containing an input file and that prints to System.out the same text with certain lines underlined. The lines to be underlined all begin with a period. The period should not be printed. You should print the text that follows the period on a line by itself followed by a line of dashes equal in length to the text that follows the period. For example, consider the following input:

```
.Statement of Purpose
I didn't expect to major in computer science until I took cse142.
I liked it more than I expected and that got me hooked on cs.

.High School Performance
I got very good grades in high school, graduating in the top 10% of
my class.

.College Performance
I have done well in my college classes, with an overall gpa of 3.5.
```

If the text above is stored in a Scanner called input and we make this call:

```
underline(input);
```

the method should print the following output to System.out:

```
Statement of Purpose
--------------------
I didn't expect to major in computer science until I took cse142.
I liked it more than I expected and that got me hooked on cs.

High School Performance
-----------------------
I got very good grades in high school, graduating in the top 10% of
my class.

College Performance
-------------------
I have done well in my college classes, with an overall gpa of 3.5.
```

Notice that some of the input lines can be blank lines. You may not construct any extra data structures to solve this problem.

6. File Processing.

Write a static method called redact that takes as a parameter a Scanner containing a text with special markers indicating sensitive words that are to be replaced. It prints the resulting text with each word on a separate line. The idea is that the text contains potentially classified information and extra "content markers" have been included throughout to indicate classified material. The special string "CLASSIFIED" appears in various parts of the text to indicate sensitive material (with exactly this casing). Each occurrence of the string will be followed by an integer indicating how many words are to be redacted. For each of those words, you should print the text "[redacted]" instead of printing the word. For example, if a Scanner called text contains the following:

```
four score CLASSIFIED 3 and seven years ago our CLASSIFIED 1 fathers
brought forth CLASSIFIED 2 on this continent
```

There are three indications of classified material. If you were to call the method as follows:

```
redact(text);
```

the following output should be produced:

```
four
score
[redacted]
[redacted]
[redacted]
ago
our
[redacted]
brought
forth
[redacted]
[redacted]
continent
```

You are to exactly reproduce the format of this output. Notice that line breaks in the input are not meaningful. You may not construct any extra data structures to solve this problem.

7. Arrays.

Write a static method called minGap that takes an array of integers as a parameter and returns the minimum gap between adjacent values in the array. The gap between two adjacent values in a list is defined as the second value minus the first value. Consider a variable called list defined as follows:

```
int[] list = {1, 3, 6, 7, 12};
```

The first gap is 2 (3 - 1), the second gap is 3 (6 - 3), the third gap is 1 (7 - 6) and the fourth gap is 5 (12 - 7). Thus, the call:

```
minGap(list)
```

should return 1 because that is the smallest gap in the list. Notice that the minimum gap could be a negative number. If the list has fewer than 2 elements, your method should return 0. Below are several examples of what value would be returned for a given array.

Array passed as parameter	Value Returned
{}	0
{8}	0
{4, 17}	13
{3, 5, 11, 4, 8}	-7
{2, 8, 8, 10, 15}	0

You may not construct any extra data structures to solve this problem and your method should not alter the array passed as a parameter.

8. Critters.

Write a class called Ferret that extends the Critter class. The instances of the Ferret class always infect when an enemy is in front of them, otherwise hop if possible, and otherwise randomly choose between turning left and turning right (each choice equally likely). Their appearance changes based on whether they recently attempted to infect. They initially display as "I=0" indicating that they have not attempted to infect recently. After an infect move, they should display as "I=5". As the ferret makes other moves that are not infecting, this display should change to "I=4", "I=3", "I=2", "I=1", and "I=0". It should then stay at "I=0". Notice, however, that it can go back to "I=5" in the middle of this process because it might infect again before reaching "I=0". The ferrets should be blue when they are displaying as "I=0" and red otherwise.

Use a Random object to make random choices but each Ferret should construct only one such object. As in assignment 8, fields must be declared private, fields initialized to a non-default value must be set in a constructor, and all updates to fields must occur in the getMove method.

9. Arrays.

Write a static method called splice that takes as parameters an array of integers and a "from index" (inclusive) and "to index" (exclusive) and that returns a new array containing the result of splicing together three segments of the array. The from and to indexes specify a sublist. For example, if list stores the following:

```
[7, 5, 8, 5, 9, 7, 2, 3]
```

then the call splice(list, 2, 4) indicates that the array should be rearranged using the sublist from 2 to 4:

```
    [7, 5]       [8, 5]    [9, 7, 2, 3]
before sublist   sublist   after sublist
```

The new array should contain the values after the sublist followed by the values in the sublist followed by the values before the sublist. For this example, it would return

```
[9, 7, 2, 3, 8, 5, 7, 5]
```

You may assume that the indexes passed as parameters specify a legal sublist of the list, although it might be empty. The method should not construct any extra data structures other than the array to be returned and it should not alter its array parameter. You are not allowed to call methods of the Arrays class to solve this problem

10. Programming.

Write a static method called findIndexes that takes an integer n and an array of integer values as parameters and finds the indexes of the first occurrence of the values 0 through n-1 inclusive, returning those indexes in an array. If some value is not found, then the value at the corresponding index should be -1. For example, suppose that a variable called list has been defined as follows:

```
int[] list = {3, 8, 5, 7, 12, -8, 3, 1, 4, 0, 6};
```

Consider the call of findIndexes(10, list). Because the first parameter of the call is 10, the method will be searching for the index of the first occurrence of each of the values between 0 and 9 inclusive. The resulting array that is constructed and returned will be of length 10. In the input array, the value 0 first appears at index 9. The value 1 first appears at index 7. The value 2 does not appear at all. The value 3 first appears at index 0. And so on. Therefore, the array that is constructed and returned will be:

```
{9, 7, -1, 0, 8, 2, 10, 3, 1, -1}
```

Note that because the value 2 did not occur in list, the value at index 2 in the resulting array is -1.

If the variable list had instead been defined as follows:

```
int[] list = {2, 7, 9, 3, 4, 5, 2, 0, 6, 42};
```

then a call of findIndexes(6, list) would return this array:

```
{7, -1, 0, 3, 4, 5}
```

As in the examples above, you cannot make any assumptions about the range of values stored in the array to be examined, but you may assume that the parameter n is greater than 0. You may not construct any String objects or extra data structures other than the array that is returned to solve this problem.

11 Bonus.

A new challenger has appeared! Your section TA is joining the Critter Tournament. Draw your section TA in Critter form.