Name: _____ Andrew Id: _____

15-121 Fall 2023 Assessment 1

Up to 50 minutes. No calculators, no notes, no books, no computers. Show your work!

1. **Short Answer**.

   Answer (a) and (b) in 1 sentence, longer answers will be marked incorrect.

   (a) (2 points) What is the difference between `public` and `private` when defining instance variables?

   (b) (2 points) Describe the relationship between a class and an object.

   (c) (1 point) Write one or two lines of code to generate a random integer between 0 and 15, inclusive. (Meaning any of the integers from 0 to 15, including 0 and 15, could be the random number chosen.)

2. (4 points) **Code Tracing:** Indicate what the following program prints. Place your answer (and nothing else) in the box under the code.

```
public class IncrementorExercise {
    public static void main(String[] args) {
        int a = 5;
        int b = 6;

        System.out.println(++b - --a + ++a - b++ - ++b + b-- + --b);
        System.out.println(a);
        System.out.println(b);
    }
}
```

3. (5 points) **Code Tracing:** Indicate what the following program prints. Place your answer (and nothing else) in the box under the code. Hints: There are 10 lines of output. Watch out for static.

```
public class A1CT {
    private int a = 0;
    private int b = 0;
    public static int c = 0;

    public A1CT(int b, int a) {
        this.a = a;
        c = b + a;
    }

    public int update(int a) {
        this.b = a;
        return this.a + this.b;
    }

    public String toString() {
        return "a: " + this.a + ", b:" + this.b;
    }

    public static void main(String[] args) {
        A1CT t1 = new A1CT(10, 15);
        System.out.println(t1);

        A1CT t2 = new A1CT(8, 20);
        System.out.println(t2);

        System.out.println("c: " + t1.c);
        System.out.println("c: " + t2.c);

        System.out.println(t1.update(7));
        System.out.println(t2.update(9));

        System.out.println(t1);
        System.out.println(t2);
        System.out.println("c: " + t1.c);
        System.out.println("c: " + t2.c);
    }
}
```

4. (10 points) **Free Response:**

   Write a `Rectangle` class with the following properties:

   - A rectangle can be constructed in one of two ways:
     1. By specifying the length of of the two sides, each as a double. For example, calling...
        `Rectangle r1 = new Rectangle(3,5);`
        would create a rectangle with two sides of length 3 and two sides of length 5.
     2. By specifying the coordinates of two diagonal corners. For example, calling...
        `Rectangle r2 = new Rectangle(1,8,5,3);`
        would create a rectangle where one corner is at (1,8) and the opposite diagonal corner is at (5,3). In this case, that means it has two sides of length 4 and two sides of length 5. (Note: You do not need to store the coordinates. We are only concerned with the dimensions of the rectangle.)
   - Rectangles are immutable: Once they are created, they never change. (So all instance variables should be private and you do not need to create setters for them.)
   - A rectangle has a method that can be used to calculate its area.
   - When printed, a rectangle should print out its dimensions. For example, `System.out.println(r1)` should display: `3.0x5.0 Rectangle`. `System.out.println(r2)` should display: `4.0x5.0 Rectangle`.

5. **Free Response**

In this problem you will write a variety of methods in an `IntegerList` class. An `IntegerList` is used to store and operate on a list of integers. The size of an `IntegerList` is fixed: Once it is created, no new integers are added or removed.

Consider the following code for the `IntegerList` class:

```
public class IntegerList {
    // This class only has one instance variable.  You may not add any more.
    private int[] arr;

    public IntegerList(int[] arg) {
        // You will write this code
    }

    private boolean swap(int i1, int i2) {
        // You will write this code
    }

    public void reverse() {
        // You will write this code
    }

    public void permute() {
        // You will write this code
    }

    public String toString() {
        return Arrays.toString(this.arr);
    }

    public static void main(String[] args) {
        int[] src = { 5, 10, 15, 20 };
        IntegerList l = new IntegerList(src);
        System.out.println(l);
        l.swap(0, 2);
        System.out.println(l);
        l.reverse();
        System.out.println(l);
        l.permute();
        System.out.println(l);
    }
}
```

When executed, the main method above printed out the following:

```
[5, 10, 15, 20]
[15, 10, 5, 20]
[20, 5, 10, 15]
[15, 20, 10, 5]
```

Note that the last line of output is randomized and would be different if main was run again.

(a) (3 points) Write the constructor, as specified below.

```
/**
 * Create a new IntegerList containing the values found in the array arg. Later
 * changes to the new IntegerList should not modify arg, so you need to copy the
 * values from arg, not simply assign arg to arr.
 *
 * @param arg An array containing the integers to put into the IntegerList.
 */
public IntegerList(int[] arg) {
```

(b) (3 points) Write the method `swap`, as specified below.

```
/**
 * Swap two elements in the list.
 *
 * This is meant to be a helper method for some of the other methods below.
 *
 * For example, if the integer list currently contains [5, 10, 15] and swap(0,2)
 * is called, then afterwards the triple will contain [15, 10, 5]
 *
 * You need to verify that the arguments passed in are valid. In the previous
 * example, calling swap(0,3) is not valid because index 3 is not valid in that
 * IntegerList.
 *
 * @param i1 The index of the first element
 * @param i2 The index of the element to swap with the first element.
 * @return True if the swap is successful, and false otherwise
 */
private boolean swap(int i1, int i2) {
```

(c) (5 points) Write the method `reverse`, as specified below.

```
/**
 * Reverse the order of the items in the list without creating a new array. (If
 * you can't figure out how to do it without a new array, then partial credit
 * can be awarded for solutions that use a new array.)
 */
public void reverse() {
```

(d) (5 points) Write the method `permute`, as specified below.

```
/**
 * Randomize the order of the items in the list. All possible outcomes should be
 * equally likely.
 *
 * A recommended algorithm is as follows:
 *
 * 1. Choose one of the elements at random, and swap it to be the first element
 * in the list.
 *
 * 2. From the remaining elements (not including the first element), choose one
 * at random and swap it to be the second element in the list.
 *
 * 3. From the remaining elements (not including the first or second element),
 * choose one at random and swap it to be the third element in the list.
 *
 * 4. etc.
 */
public void permute() {
```