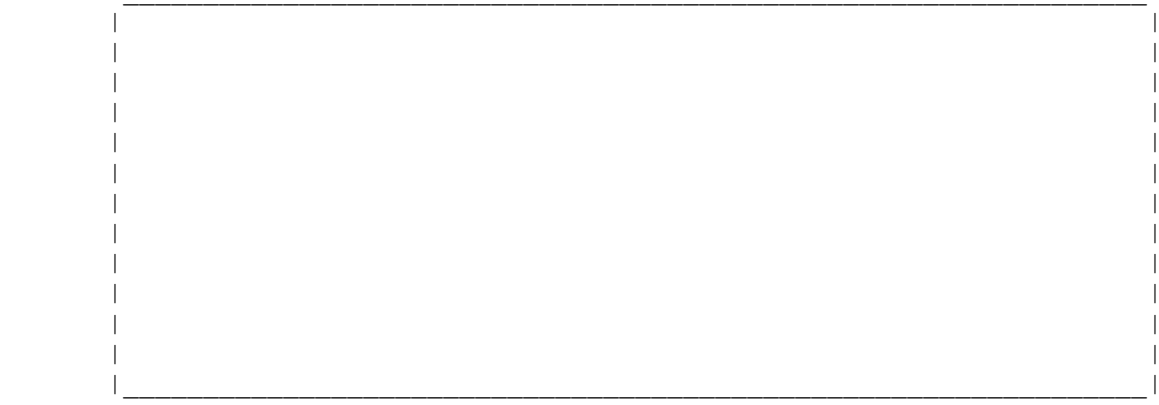
CSE142 Sample Final Exam
Spring 2019

1. Reference Mystery, 5 points. The following program produces 5 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) {
      int x = 2;
      int[] numbers = {2, 4, 6, 0, 1};
      Point origin = new Point(0, 0);

      System.out.println(x + " " + Arrays.toString(numbers) +
                         " (" + origin.x + ", " + origin.y + ")");

      mystery1(x, numbers);
      System.out.println(x + " " + Arrays.toString(numbers));

      mystery2(x, origin);
      System.out.println(x + " (" + origin.x + ", " + origin.y + ")");
   }

   public static void mystery1(int n, int[] a) {
      n++;
      a[n] = -35;
      System.out.println(n + " " + Arrays.toString(a));
   }

   public static void mystery2(int n, Point p) {
      p.x = n;
      n--;
      p.y = n;
      System.out.println(n + " (" + p.x + ", " + p.y + ")");
   }
}
```



2. Array Simulation, 10 points. You are to simulate the execution of a method that manipulates an array of integers. Consider the following method:

```
public static void mystery(int[] arr) {
    for (int i = 1; i < arr.length; i++) {
        if (arr[i] < arr[i - 1]) {
            arr[i] += arr[i - 1];
        } else {
            arr[i] -= arr[i - 1];
        }
    }
}
```

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
[10, 8, 6, 4, 2]	_____
[1, 2, 3, 4, 5]	_____
[5, 10, 8, 1, 3]	_____
[4, 4, 4, 4, 4]	_____
[19]	_____

3. Inheritance Mystery, 6 points. Consider the following classes:

```
public class Vanellope extends Ralph {
    public void method1() {
        System.out.print("Vanellope 1   ");
        method2();
    }

    public void method2() {
        System.out.print("Vanellope 2   ");
    }

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

public class Calhoun extends Felix {
    public void method1() {
        method2();
        System.out.print("Calhoun 1   ");
    }

    public void method2() {
        System.out.print("Calhoun 2   ");
        super.method1();
    }
}

public class Ralph {
    public void method1() {
        System.out.print("Ralph 1    ");
    }

    public void method2() {
        System.out.print("Ralph 2    ");
    }

    public String toString() {
        return "Wreck-It";
    }
}

public class Felix extends Ralph {
    public void method2() {
        System.out.print("Felix 2   ");
        super.method2();
    }

    public String toString() {
        return "Fix-It";
    }
}

// client code
public static void main(String[] args) {
    Ralph[] racers = { new Felix(), new Ralph(), new Vanellope(), new Calhoun() };
    for (int i = 0; i < racers.length; i++) {
        racers[i].method1();
        System.out.println();
        racers[i].method2();
        System.out.println();
        System.out.println(racers[i]);
        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. File Processing, 9 points. Write a static method called processScores that takes as a parameter a Scanner containing a series of lines that represent student records. Each student record takes up two lines of input. The first line has the student's name and the second line has a series of plus and minus characters. Below is a sample input:

```
Kane, Erica
--+-+
Chandler, Adam
++-+
Martin, Jake
+++++++
Dillon, Amanda
++-++-+-
```

The number of plus/minus characters will vary, but you may assume that at least one such character appears and that no other characters appear on the second line of each pair.

For each student you should produce a line of output with the student's name followed by a colon followed by the percent of plus characters. For example, suppose the input above is stored in a Scanner called input and we make the following call:

```
processScores(input);
```

The following output should then be produced:

```
Kane, Erica: 40.0% plus
Chandler, Adam: 75.0% plus
Martin, Jake: 100.0% plus
Dillon, Amanda: 62.5% plus
```

You do not need to round the percentages; you should output the exact value. You must exactly reproduce the format of this log.

5. File Processing, 10 points. Write a static method called evaluate that takes as a parameter a Scanner containing a series of tokens that represent a numeric expression involving addition and subtraction and that returns the value of the expression. For example, suppose a Scanner called data contains the following tokens:

```
4.2 + 3.4 - 4.1
```

Then suppose we make the following call:

```
evaluate(data)
```

In this case, the call should return 3.5 because 4.2 + 3.4 - 4.1 evaluates to 3.5. Every expression will begin with a real number and then will have a series of operator/number pairs that follow. The operators will be either "+" (addition) or "-" (subtraction). As in the example above, there will be spaces separating numbers and operators. Assume the expression is legal.

Your program should evaluate operators sequentially from left to right. For example, suppose a Scanner called data2 contains the following tokens:

```
7.3 - 4.1 - 2.0
```

Then suppose we make the following call:

```
evaluate(data2)
```

In this case, the method should return 1.2 by evaluating the operators as follows:

```
7.3 - 4.1 - 2.0 = (7.3 - 4.1) - 2.0 = 3.2 - 2.0 = 1.2
```

The Scanner might contain just a number, in which case your method should return that number as its result. For example, suppose a Scanner called data3 contains the following tokens:

```
9.8
```

Then suppose we make the following call:

```
evaluate(data3)
```

In this case, the method would return 9.8.

You may assume that the Scanner will only contain numbers, plus signs, and minus signs and that the tokens in the Scanner will alternate between numbers and operators. You may also assume that both the first and last tokens in the Scanner will be numbers and that the Scanner will contain at least one number. You may not construct any extra data structures to solve this problem.

6. Arrays, 10 points. Write a static method named repeatedSequence that accepts two arrays of integers a1 and a2 as parameters and returns true if a2 is composed entirely of repetitions of a1 and false otherwise. For example, if a1 stores the elements {2, 1, 3} and a2 stores the elements {2, 1, 3, 2, 1, 3, 2, 1, 3}, the method would return true.

   If the length of a2 is not a multiple of the length of a1, your method should return false. The following table shows some calls to repeatedSequence and their expected results:

Arrays	Method Call	Return Value
`int[] a1 = {2, 1};` `int[] a2 = {2, 1, 2, 1, 2, 1};`	`repeatedSequence(a1, a2)`	`true`
`int[] a3 = {2, 1, 3};` `int[] a4 = {2, 1, 3, 2, 1, 3, 2};`	`repeatedSequence(a3, a4)`	`false`
`int[] a5 = {23};` `int[] a6 = {23, 23, 23, 23};`	`repeatedSequence(a5, a6)`	`true`
`int[] a7 = {5, 6, 7, 8};` `int[] a8 = {5, 6, 7, 8};`	`repeatedSequence(a7, a8)`	`true`
`int[] a9 = {5, 6};` `int[] a10 = {5, 6, 7, 5, 6, 5};`	`repeatedSequence(a9, a10)`	`false`

   You may assume that both arrays passed to your method are not null and have a length of at least 1. Your method must not modify the contents of the arrays passed as parameters.

7. Classes, 10 points. Suppose that you are provided with a pre-written class Date as described below. Assume that the fields, constructor, and methods shown are implemented. You may refer to them or use them in solving this problem.

```
// A Date object represents a date during the year,
// such as January 1, July 12, or December 25.
public class Date {
    private int month;
    private int day;

    // Constructs a new date for the given month and day
    public Date(int m, int d) { /* implementation omitted */ }

    // returns the field values
    public int getMonth() { /* implementation omitted */ }
    public int getDay() { /* implementation omitted */ }

    // returns String for date; for example: "1/1" or "12/25"
    public String toString() { /* implementation omitted */ }

    // your method will go here
}
```

Write an instance method named isSummer that will be placed inside the Date class. The isSummer method takes no parameters and returns true if this Date object represents a date during the summer and false otherwise. For the purposes of this problem assume the first day of summer is June 21 and the last day of summer is September 20.

For example, suppose the following objects are declared in client code:

```
Date d1  = new Date(1, 1);
Date d2  = new Date(5, 18);
Date d3  = new Date(6, 20);
Date d4  = new Date(6, 21);
Date d5  = new Date(7, 4);
Date d6  = new Date(9, 20);
Date d7  = new Date(9, 21);
Date d8  = new Date(10, 3);
Date d9  = new Date(12, 25);
Date d10 = new Date(12, 31);
```

The table below shows some calls to isSummer and their expected results:

Call	Return Value
d1.isSummer()	false
d2.isSummer()	false
d3.isSummer()	false
d4.isSummer()	true
d5.isSummer()	true
d6.isSummer()	true
d7.isSummer()	false
d8.isSummer()	false
d9.isSummer()	false
d10.isSummer()	false

Your method should not modify the state of the Date object. You may assume that the state of the Date object is valid at the start of the call, that the month is always between 1 and 12 (inclusive) and that the day is always between 1 and 31 (inclusive) and represents an actual date. You may also assume the date is not in a leap year.

8. Critters, 15 points. Write a class named Rhino that extends the Critter class from homework 8. Write the complete class with any fields, constructors, etc. that are necessary. All unspecified aspects of Rhinos use the default behavior.

   Rhinos are constructed with an initial speed (an integer) and direction (a Direction value). The Rhino should move a number of steps equal to its speed in the given direction, then choose a new direction randomly. However, it is difficult for Rhinos to change direction. Therefore, when it is time for a Rhino to choose a new direction, it should have a 40% chance to continue in the same direction and a 20% chance to change to each other direction. (Rhinos never stay in the same place.)

   Rhinos always eat when encountering food and always pounce when fighting (to take advantage of their movement). In addition, eating and fighting affect the Rhino's speed. A Rhino will increase its speed by one after eating (since it now has more energy) and will decrease its speed by one after fighting (since pouncing on its opponent slows it down). Neither eating nor fighting interrupts the Rhino's movement; the Rhino should continue in the same direction until a number of steps equal to or greater than its new speed have been taken. (If fighting causes a Rhino's speed to drop below the number of steps it has taken in the current direction, it should consider its cycle ended on its next move.) A Rhino's speed should never drop below one. If it would, it should be set to one instead.

   Below is an example of the movement of a Rhino constructed with initial speed 3 and initial direction EAST:

   ```
   EEEWW (eat) WWEEEENN (fight) NWWW (randomly continue in same direction)
   WWWSSS (eat) SEEE (fight) NNN (fight) SSEEWW ...
   ```

   Rhinos are gray, unless they are moving very fast (with speed greater than 6) in which case they are white.

9. Arrays, 15 points. Write a static method named banish that accepts two arrays of integers a1 and a2 as parameters and removes all occurrences of a2's values from a1. An element is "removed" by shifting all subsequent elements one index to the left to cover it up, placing a 0 into the last index. The original relative ordering of a1's elements should be retained.

   For example, suppose the following two arrays are declared and the following call is made:

```
int[] a1 = {42, 3, 9, 42, 42, 0, 42, 9, 42, 42, 17, 8, 2222, 4, 9, 0, 1};
int[] a2 = {42, 2222, 9};
banish(a1, a2);
```

   After the call has finished, the contents of a1 should be:

```
{3, 0, 17, 8, 4, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

   Notice that all occurrences of the values 42, 2222, and 9 have been removed and replaced by 0s at the end of the array, and the remaining values have shifted left to compensate.

   Do not make any assumptions about the length of the arrays or the ranges of values each might contain. For example, each array might contain no elements or just one element, or very many elements. (If a2 is an empty array that contains no elements, a1 should not be modified by the call to your method.) You may assume that the arrays passed are not null. You may assume that the values stored in a2 are unique and that a2 does not contain the value 0.

   You may not use any temporary arrays to help you solve this problem. You also may not use any other data structures or complex types such as Strings or ArrayLists. But you may declare as many primitive variables (such as ints) as you like.

10. Programming, 10 points. Write a static method named collapseDigits that accepts an integer parameter and that returns a new number obtained by replacing every pair of repeated adjacent digits with one of that digit. For example, the integer 558834226 has three repeated adjacent digits: 55, 88 and 22. This means that collapseDigits(558834226) should return the integer 583426.

    The table below shows some calls to collapseDigits and their expected results:

Call	Return Value
collapseDigits(44223553)	42353
collapseDigits(346623)	34623
collapseDigits(14436344)	143634
collapseDigits(121212)	121212
collapseDigits(0)	0
collapseDigits(8)	8
collapseDigits(44)	4
collapseDigits(22005)	205

    You may assume that the integer passed as a parameter to your method is greater than or equal to 0. You may also assume that no digit in the parameter is repeated more than twice in a row, though a digit may appear more than twice total (as in the third sample call above). You may not construct any complex data structures (e.g. Strings, arrays, ArrayLists) to solve this problem, though you may create as many primitive variables (e.g. int, double, char) as you want.