Quiz2

Week 1

Monday

Name: _____

Instructions:

1. **Wait to start until the TA tells you to do so. Do not turn the page yet.**
2. You will have forty minutes to complete up to two of the two questions. We will grade both questions and you will keep the higher score of the two.
3. Please turn in this quiz sheet when finished. You may either sit quietly at your desk (no laptops), or, preferably, leave the room and return when the quiz is over for the class.
4. The maximum possible score for Problem 1 (the problem you have seen) is 80 points; the maximum score of Problem 2 (the unseen problem) is 100 points.

Problem 1: Sum Neighbors Plus (max score of 80 points)

Imagine that the user specifies a width and height of a grid, and provides a tile in that grid. You will write code to return the sum of the adjacent neighbors of that tile that are directly above, or to the left and right. For example, in the grid below,

tiles 23 will would return the list [16, 30, 22, 24] which sums to 92 and tile 70 would return the list [63, 0, 69, 0], which sums to 132. If a neighbor doesn't exist, you should return a 0 for that item in your result array before summing.
You may use the following formulas in your solution as needed:

```
row = (tile - 1) / width
```

Please do not use any Java code or libraries that we have not covered in class; this defeats the purpose of the assessment.

See the test cases below, and then complete the template.

```
/*
            SAMPLE ASSESSMENT2_8
*/
import java.util.Arrays;

public class Assess2_8_Sample{

  public static void main(String[] args){
    System.out.println("test1: " + (plusSign(1,1,1) == 0));
    System.out.println("test2: " + (plusSign(2,1,1) == 2));
    System.out.println("test3: " + (plusSign(2,1,2) == 1));
    System.out.println("test4: " + (plusSign(1,2,1) == 2));
    System.out.println("test5: " + (plusSign(1,2,2) == 1));
    System.out.println("test6: " + (plusSign(1,3,1) == 2));
    System.out.println("test7: " + (plusSign(1,3,2) == 4));
    System.out.println("test8: " + (plusSign(1,3,3) == 2));
    System.out.println("test9: " + (plusSign(3,1,1) == 2));
    System.out.println("test10: " + (plusSign(3,1,2) == 4));
    System.out.println("test11: " + (plusSign(3,1,3) == 2));
    System.out.println("test12: " + (plusSign(2,3,1) == 5));
    System.out.println("test13: " + (plusSign(2,3,2) == 5));
    System.out.println("test14: " + (plusSign(2,3,3) == 10));
    System.out.println("test15: " + (plusSign(2,3,4) == 11));
    System.out.println("test16: " + (plusSign(2,3,5) == 9));
    System.out.println("test17: " + (plusSign(3,2,1) == 6));
    System.out.println("test18: " + (plusSign(3,2,2) == 9));
    System.out.println("test19: " + (plusSign(3,2,3) == 8));
    System.out.println("test20: " + (plusSign(3,2,4) == 6));
    System.out.println("test21: " + (plusSign(3,2,5) == 12));
    System.out.println("test22: " + (plusSign(3,2,6) == 8));
    System.out.println("test23: " + (plusSign(3,3,1) == 6));
    System.out.println("test24: " + (plusSign(3,3,2) == 9));
    System.out.println("test25: " + (plusSign(3,3,3) == 8));
    System.out.println("test26: " + (plusSign(3,3,4) == 13));
    System.out.println("test27: " + (plusSign(3,3,5) == 20));
    System.out.println("test28: " + (plusSign(3,3,6) == 17));
    System.out.println("test29: " + (plusSign(3,3,7) == 12));
    System.out.println("test30: " + (plusSign(3,3,8) == 21));
    System.out.println("test31: " + (plusSign(3,3,9) == 14));
    System.out.println("test32: " + (plusSign(5,5,2) == 11));
    System.out.println("test33: " + (plusSign(5,5,19) == 76));
    System.out.println("test34: " + (plusSign(5,5,13) == 52));
    System.out.println("test35: " + (plusSign(5,5,25) == 44));

  }
```

```
  public static int plusSign(int width, int height, int tile){
    int result = 0;









    return result;
  }

}
```

```
//extra sheet if needed

  }
}
```

Problem 2: Is Square (max score of 100 points)

Write code that will return whether or not four tiles all share the same point, that is, they are adjacent to one another and make a 2x2 square. The four tiles will be passed into your method in ***sorted numeric order*** (this should make your code easier). If the four tiles make such a square return "all touch" otherwise return "no". The width and height will both be at least 3 or larger, and the **first tile will never be in the last column** (so you don't have to check for this edge case)

Here are some example test cases for a function defined as:

```
isSquare(int width, int height, int t1, int t2, int t3, int t4)
```

```
isSquare (3, 4, 2, 3, 5, 6) will return "all touch"
isSquare (3, 4, 4, 5, 7, 8) will return "all touch"
isSquare (3, 4, 2, 3, 5, 9) will return "no"
isSquare (3, 4, 1, 4, 5, 7) will return "no"
```

Note that the **bolded tiles above** will always be in sorted order (so the first tile would always be in the top left of the square we expect to check for).

Complete the template below. Please do not use any Java code or libraries that we have not covered in class; this defeats the purpose of the assessment.

```
public static String isSquare(int width, int height, int t1, int t2, int t3, int t4) {

        // YOUR ANSWER HERE
```

// extra page if needed

}

SCRATCH PAPER

SCRATCH PAPER