Name: _____ class & section _____

You must:

- **follow all style guidelines in Appendix I** and **F:** provide full javadoc style comments for each method and class (search the internet for more details). Do **NOT** create a java package or .jar
- spell names **exactly** as they are in these instructions so my tester will work with your class
- provide **both electronic and physical deliverables** (See below).

Bulgarian Solitaire explained:

In this assignment, you will model the game of Bulgarian Solitaire. The game starts with 45 cards. (They need not be playing cards. Unmarked index cards work just as well.) Randomly divide them into some number of piles of random size. For example, you might start with piles of size 20, 5, 1, 9, and 10. In each round, you take one card from each pile, forming a new pile with these cards. For example, the sample starting configuration would be transformed into piles of size 19, 4, 8, 9, and 5. The solitaire is over when the piles have size 1, 2, 3, 4, 5, 6, 7, 8, and 9, in some order. (It can be shown that you always end up with such a configuration.) In your program, produce a random starting configuration and print it. Then keep applying the solitaire step and print the result. Stop when the solitaire final configuration is reached.

This game can be played with any number that is the sum of the integers from 1 to **n** to produce **n** piles: six cards and 3 piles, ten cards and 4 piles, 15 cards and 5 piles, etc. Here is a sample possible game play output:

The 10 card initial piles are: 4 4 2
10 card step: 1, current piles are: 3 3 1 3
10 card step: 2, current piles are: 2 2 2 4
10 card step: 3, current piles are: 1 1 1 3 4
10 card step: 4, current piles are: 2 3 5
10 card step: 5, current piles are: 1 2 4 3

BulgarianTester (you must show expected result for every case/input):

This class' main method must instantiate an object of the Bulgarian class and an object of the Scanner class. The tester you submit must produce output and expected results for at least the three following (good, extreme & bad input):

1. Since input data might cause an exception, (non-numeric input), it must be encased in a try block, and have an accompanying catch block for the possible IllegalArgumentException (you can use an if statement). **See section 11.4**. Print out a message noting that the data is out of range / being ignored, and have the program continue if possible.
2. You must use only one random number generator for each game played and the **numberOfPiles** must be used as the seed for the random number generator.
3. For the testing, ou must use the number of piles (n) to determine the number of cards(sum of the integers from 1 to n, inclusive) to place randomly in the initial pile creation phase.
4. Submit a tester that demonstrates at least the following :
    a. the results of trying to play a 0 pile game
    b. the results of trying to play a -2 pile game
    c. a 4 pile game
    d. a 6 pile game
    e. a 9 pile game
    f. at least one other test case

Bulgarian class: must have at least the following instance variables with these names:

1. int ***soughtPiles*** - the number of uniquely sized piles needed to win
2. int ***deckSize*** - integer value set to the sum of the integers from 1 to ***piles***, inclusive
3. ***int steps*** - to **count** the number of steps taken to produce the required piles
4. ***piles*** - an integer array or ArrayList to store the number of cards in each pile
5. ***static Random generator*** - the one and only to be used in a given game

The Bulgarian class must have at least the following methods with these names:

1. a constructor **Bulgarian(int numberOfPiles)** that **initializes all instance variables properly**
2. You may use an ArrayList<Integer> for the parameter or return type instead of an integer array in these methods
3. public int[] ***createPiles***() method to place all cards from the deck into a random number of random sized piles.
4. public int[] ***playARound***(int[] piles) to complete the removal of cards and creation of a new pile step.
5. private int[ ] ***removeZeros*** (int[ ] piles) to eliminate piles with zero cards from the current set.
6. public boolean ***winningConfig***(int[ ] piles)
7. public String ***toString*** must return a String containing **all** instance variables with clear, brief headings

You must submit all physical and electronic Deliverables. See Canvas for due date.

Physical: All Pages must be stapled in order (print on both sides if possible to save paper).

1. All pages of these instructions (printed from the web), with your **name clearly printed** on it, as a cover sheet.
2. Printed Source Code with Comments (including all javadoc style comments, **no line wrapping**)
3. Sample Input and Output screen captures or console copies for each of the required tests (printed)
4. a simple **test plan** including explanations of any discrepancies and reasons for each test.
   4.1. Show actual input and include all of the outputs (expected & actual) in the test plan
   4.2. **ALL** values output as well as **ALL** expected output.
   4.3. Remember to test an <Enter> for expected inputs, a non-numeric character and just **one** numeric input.
   4.4. test multiple **sets** of data in one run.

Electronic:

5. Copies **of #3 and #4-4.4 above** as an rtf, All **.html (javadocs)**, and **.java** files **all** together as **one .zip** file. Do not use **rar** or any other archive format
6. Name the file: "<YourName>_p<#>.zip".
7. Submit this single **zip** file on **Canvas**
