Homework – Topic 5 – Bowling Scores:

Write a complete Java program to do the following: The main program **reads in and prints** three bowling scores, score1, score2, and score3. It then calls a series of methods to process these scores.

The main program calls a method validGroup() to determine if this set of three values forms a valid group. The method validGroup() receives **four parameters (the 3 scores and a file object).** For the group to be valid, each number must be in the range from 0 to 300 (the possible scores in a bowling game). If the group is valid, **the method prints** a message saying so. If one or more of the numbers is negative or greater than 300, **the method prints** an overall message that the group is invalid. **In addition, for each invalid value, the method prints the score and a message why the score is invalid.** The **method returns a boolean value (true or false) as a signal** indicating the validity of the group. (Hint: use six "if" statements.)

If the group is not valid, the main program skips processing and simply goes on to the next group of three scores.

If the group is valid, the main program calls a method oneGameScore(), sending it **two parameters, the value score1 and a file object.** This score is an integer from 0 to 300 (how can we be sure of this?). The method converts the score into a rating, using the following system: 250 to 300 is a professional game; 200 to 249 is an excellent game; 140 to 199 is a very good game; 100 to 139 is a good game; 50 to 99 is a poor game; below 50 is a horrible game. **The method prints a message with the original score and the bowler's rating.** Then the main program repeats this process for score2 and score3.

Next the main program calls a method avg3Scores(), sending it three parameters: the three scores. The method avg3Scores() finds the average (as an integer) of the three scores and sends it back. **The main program prints the average.** Finally, the main program calls oneGameScore() again, sending it the resulting average from the method avg3Scores().

The main program then prints three blank lines.

Then the main program goes on to the next group of three values. When the main program runs out of groups (**hint: use a sentinel**), **it prints the final values of three counters it has been keeping track of: the total number of groups processed, the number of valid groups, and the number of invalid groups.**

Note:

Data input must come from a file
Output must be file directed.
Do not send prompts to the output file.
Do not use global file objects.
File objects should be declared in main()
and passed to methods as needed.
The program must be properly tested.

Suggested Implementation:

1. Write code to read and print a group of three scores
2. Write code to continue reading and printing groups of three scores until a sentinel is reached.
3. Create an input file from which to read the groups of scores
4. Have the output go to an output file
5. Implement a counter to count the total number of groups processed
6. Implement method validgroup(). The header for validgroup() should be of the form:
   public static boolean validgroup(int score1, int score2, int score3, PrintWriter myout)
7. Implement the counters for the number of valid groups and the number of invalid groups
8. Implement method onegamescore(). The header for method onegamescore() should be of the form:
   public static void onegamescore(int score, PrintWriter myout)
9. Implement method avg3scores(). The header for method avg3scores() should be of the form:
   public static int avg3scores(int score1, int score2 ,int score3)
10. Complete the program and documentation as necessary.

Required Submission:

1. The Java source code file (e.g., HW5.java)
2. The test cases input file (e.g., testCases.txt)
3. The program generated output file (e.g., output.txt)