Write a complete C++ program, including a good comment in each function and in the main function, to do the following:

**Outline:** The main function will read in three integers representing a student's scores on three tests. The main function will call a function to determine if these three scores are valid. If they are, the main function will call other functions to classify the scores in various ways. The main function will repeat this process for the entire set of data, then print some counters.

**Details** (a description of each function and then the main function):

0. Write a function `introduction` which will not receive any parameters (and will not return an answer). This function will print a description of the program.

1. Write a function `validate` which will receive three integers representing three test scores. The function will determine whether or not the three scores are valid—each in the range from 0 - 100.

   For each score, if the score is invalid, the function will print it, with a message explaining why it is invalid--too big or too small. If all three scores are valid, the function will return a signal to the main function saying this is a valid set of scores; if any score is invalid (or two or more are invalid), the function will return a signal saying this is an invalid set of scores. (A Boolean variable—a variable with data type bool—is perfect for the answer to be returned by the function.)

2. Write a function `compare` which will receive three integers, each a value from 0 to 100. The function will compare these numbers and print one of the following four messages: (1) all three numbers are the same; (2) two numbers are the same and the third is larger (there are three ways this can happen); (3) two numbers are the same and the third is smaller (also three ways for this to occur); (4) all three numbers are different. The function will print, but it will not return a value.

3. Write a function `roundscore` which will receive one integer, a number from 0 to 100. This function will round the integer to the nearest 10; (rounding up for 5 to 9, and rounding down for 0 to 4). The function will return the rounded value.
   For example, if you send the function 54, it will return 50; if you send 70, it will return to 70; if you send 88, it will return 90.
**Main Function:**

The main function will call the function *introduction* one time, to start things off. After this the main function will read in three numbers. The main function will print the numbers as soon as they are read in.

The main function will call the function *validate* to determine if this group is valid. The main function will print the result of the function call (and add to counters).

If the function validate says it is a valid group, the main function will then process this set of data. However, if the function validate says the group is not valid, the main function will simply go back to read the next group of data values. Be sure to skip a few lines on the output display.

If the group is valid, first the main function will send the original three numbers to a function called *compare*. Then the main function will send the numbers one at a time to another function called *roundscore*, producing three rounded scores. (There will be three calls to roundscore.) The main function will print each rounded score. Finally, the main function will send the three *rounded scores* to the function *compare* again. How is this call different from the first call to this function?

The main function will continue this process for the entire set of test scores. You decide how to determine the end of the entire data set. Be sure to explain your choice to the user and in a comment.

When the main function runs out of groups (you must decide when the main function has run out of data), the main function will print the final values of three counters it has been keeping track of: the total number of groups processed, how many groups were valid, and how many groups were invalid.

Use three separate counters. Do NOT compute one from the others.

**DATA:** You will be judged on the quality of your data. Include at least 20 groups of various types. Be sure to include each possible comparison message at least 3-4 times. Include at least 5 invalid groups (you must spread them through the entire set of data), some with the first grade invalid, some with the second or third invalid, and some with more than one invalid grade. Have several cases where the original grades are different but the rounded grades are equal.

You are expected to use a data file for the input to the program; you also should use a file for output. Be sure to submit a copy of both the input and output files.

**STYLE:** Each function should have a good comment explaining its role in the program and what parameter(s) it will receive.

**OUTPUT:** Make sure your output is clean, neat and easy to read.