COSC 1437 GE1

Problem Statement:

Write a C++ program that generates 6 random lottery numbers between 0 and 9 inclusive. There should be no duplicate numbers. The program should then repeatedly generate sets of 6 non-duplicate random numbers in the range 0 - 9 to see how many sets of 6 random numbers match the originally generated lottery numbers.

Matches are of two types. Type 1 matches require numbers to match exactly by position. For example, suppose the original lottery numbers were 7, 2, 1, 9, 4, and 5 (in that specific order). Then a Type 1 match would be the numbers 7, 2, 1, 9, 4, and 5 (in that exact same specific order). No other sets of numbers would form a Type 1 match.

A Type 2 match will occur no matter what order the numbers occur. For example, suppose the original lottery numbers were 7, 2, 1, 9, 4, and 5 (in that specific order), then a Type 2 match would be the numbers 7, 2, 1, 9, 4, and 5 in ANY order, for example both the set 1, 9, 2, 7, 4, and 5 and the set 4, 9, 1, 5, 7 and 2 plus many others would form Type 2 matches.

As previously mentioned, for the simulation, 6 non-duplicate random lottery numbers will be generated. After that, the simulation loop will run, generating 10 Million sets of 6 non-duplicate numbers in the range 0 to 9. During the simulation, a counter will be maintained for Type 1 matches and another counter will keep track of Type 2 matches. The program will then display the percentage of guesses that result in Type 1 matches and the percentage of guesses that result in Type 2 matches.

Seed the random number generating function using the system time so that each time the program is run a different sequence of random numbers will be generated. Each run of the program should produce similar but probably not the exact same results.

Analysis: The expected results produced by this simulation can be calculated using some simple math. If duplicate numbers were allowed, then there are 10 * 10 * 10 * 10 * 10 * 10 = 1,000,000 possible different sets of lottery numbers. For this situation, on average we would expect to guess the correct lottery number once if we bought 1,000,000 different lottery tickets. If we made 10 million guesses, then we would expect to guess the correct lottery numbers 10 times assuming Type 1 matches.

However, in our problem statement, duplicates are NOT allowed. So there are only 10 * 9 * 8 * 7 * 6 * 5 = 151200 possible lottery numbers. Assuming Type 1 matches, we would expect to be correct on average once every 151200 guesses. If we made 1 million guesses, we would expect to be correct about 6 times (1,000,000 / 151200)

Now if order doesn't matter, we should win more often. If I have 6 different numbers, I can arrange these numbers 720 different ways (6 * 5 * 4 * 3 * 2). So for Type 2 matches I would expect to be 720 times more likely to win than with non-duplicate Type 1 matches.

Use modular programming techniques that demonstrate to me your mastery of using functions and using parameters to pass data between functions. One possible functional breakdown for this program would be as follows:

A function to test for Type 1 matches

A function to test for Type 2 matches

A function to generate 6 non-duplicate random numbers

A function to display results

A function used during program development to display a set of 6 non-duplicate lottery numbers

A main function