COSC 1410, Spring 2016
Assignment 5

[1] Objective: This assignment is to help you have a better understanding of functions with call-by-value and call-by-reference parameters. You are also expected to use several library functions from <cmath> and <cstdlib> library. You are going to do some interesting calculations with limited syntax that we have learned so far. Do not use arrays in this assignment.

[2] Description: After getting all the test papers graded with numerical scores, Dr. Huang would like to see the following statistics of student performance.

```
Number of Scores
Maximum Score
Average Score
Minimum Score
Standard Deviation
```

Requirements:

1. You should not use array in this assignment.
2. You should not use global variables except for constants.
3. You must write at least three functions (see below). These functions must use some value-parameters and some call-by-reference parameters.

The traditional way to compute the standard deviation is given at left below. It requires the computation of the average first and use the average to compute the standard deviation. To do would require you to use the x_i twice which is not possible without using array. Fortunately, there is another way to calculate the standard deviation (on the right). You can use one loop to compute $\sum_{i=1}^{N} x_i$ and $\sum_{i=1}^{N} x_i^2$. You can then compute the standard deviation easily.

$$Std = \sqrt{\sum_{i=1}^{N} (x_i - \mu)^2} = \sqrt{\frac{\left(\sum_{i=1}^{N} x_i^2\right)}{N} - \left(\frac{\sum_{i=1}^{N} x_i}{N}\right)^2}.$$

Here are the three functions that you should implement. I omitted the parameter list in the list below. You have to decide what are the parameters and how to pass them. You are allowed to write more functions.

- int getScore(); // returns a random number generated within a range.
- void getStats() // do all the input while computing average, std, max and min. Return average, std, max and min.
- void printStats(); // print N, max, average, min, std.

[3] Input: Since we don't have a convenient way to get a large number of scores into the program, we have decided to generate the scores by using the random number generator. For the scores to look somewhat real, this is what you need to do. (1) Generate 20% integer numbers between 0 and 100, and (2) Generate 80% numbers between 40 and 70 for a total of 100 numbers, (3) Set N = 100 (You may want to test your program with a smaller N first). Your program does not have to remember all these numbers generated. You must initialize random number generator by calling the below function in your main program to generate different random numbers in each run.

```
srand (time(NULL));  // remember to add #include<time.h> at the beginning
```

Note: After learning how to use array later, you may come back and redo this assignment and store the values into an array so that you can assign a letter grade to each student. There is no way we can do that at this point. Also, you will learn how to read a list of scores from a file later this semester.

[4] Output: A sample output looks like:

```
Scores are as follows:
1: 40
2: 9
3: 44
...
99: 42
100:40

Number of Scores:    100
Maximum Score:       97
Average Score:       54.03
Minimum Score:       3
Std deviation:       14.58
```

You should also print the all the numbers generated. I am saving space here.

[5] Due date: Wednesday, March 21, 2016.