Assignment # 2

C++ Functions and Filing

Submission Dead Line: Friday 19/10/2018

Question 1: It is known from Babylonian times that the square root of a number can be approximated by a method now known as 'divide and average' method. Since it was used by the Babylonians, it is also called the Babylonian algorithm.
METHOD: Suppose we want to compute the square root of a number N.
Let A>0 be a guess for square root (N), then a better approximation is given by: B=(A+N/A)/2.
We can then improve the approximation B using C=(B+N/B)/2.

EXAMPLE: For example, lets compute: square root of N=2.
Let our initial guess be 1.
The next better approximation is (1+2/1)/2=1.5
The next better approximation is (1.5+2/1.5)/2=1.416667
The next better approximation is (1.416667+2/1.416667)/2=1.414216
And so on. The more you repeat this, the closer you will get to the actual answer. Just keep in mind that the square root of 2 is an irrational number, which means that you can keep on improving your approximation forever.

1. Write a C++ Function square_root (int n) to implement above mentioned method of calculating square root and will return the calculated square root. You have to run 10 iterations for approximation.
2. Now write a C++ Function compare_Sqrt() which will read all numbers listed in a file **input.txt** and then compare the computed square root by your method square_root (int n), with the square root function sqrt() of C++. After that it will print in each line the number N, both square roots of N and the difference in a file **output.txt**, you can use space as a delimiter sign. Note that the output for all numbers should be printed in separate line.

Question 2: The game of "**23**" is a two-player game that begins with a pile of 23 toothpicks. Players take turns, withdrawing either 1, 2, or 3 toothpicks at a time. The player to withdraw the last toothpick loses the game. Write a C++ Function for human vs. computer program that plays "23".

The human should always move first. When it is the computer's turn, it should play according to the following rules:

1. If there are more than 4 toothpicks left, then the computer should withdraw 4 – X toothpicks, where X is the number of toothpicks the human withdrew on the previous turn.
2. If there are 2 to 4 toothpicks left, then the computer should withdraw enough toothpicks to leave 1.
3. If there is 1 toothpick left, then the computer has to take it and loses.

When the human player enters the number of toothpicks to withdraw, the program should perform input validation. Make sure that the entered number is between 1 and 3 and that the player is not trying to withdraw more toothpicks than exist in the pile.