CSCI 201 – Computer Science 1

Homework 5: Comparing algorithms for computing the square root.
Due on Monday February 21
Objectives. The objectives of this assignment are:
1. Learn how to track the number of operations evaluated when a program runs.
2. Learn to design a converging loop to compute better approximations.
3. Apply a method to estimate the efficiency of two different converging processes.

IMPORTANT
The sample code for this assignment is available in `CourseInfo/TriangleANDSqrt/sqrtTest.cpp` in `CourseFiles`.

In class we studied two approaches that can be used to estimate the square root of a number. The two approaches for the square root are the *Divide-and-conquer* and *Newton's method.* In both cases, we have a pre-defined constant `eps` which is used to check the goodness of the guess. The loop exits when the absolute value of the difference between the input number and the square of the guess is less than `eps`.

Question 1. Newton's method works by starting with an initial guess; we'll use $(num + 1)/2$ as the initial guess. For each iteration, the `new guess` is computed as $(oldGuess - \frac{oldGuess^2 - num}{2 * oldGuess})$. List the sequence of operations that will be performed when we use this approach, and identify the repetition and the exit condition.

Question 2. Construct a flowchart for the Newton's method using Raptor.

Question 3. Write a `C++` program that implements your flowchart. The program must count the number of divisions and multiplications that the loop performed. Compile and test in a script session.

Comparing the two methods. When comparing two different approaches, it is important that we have a common framework. In the realm of computing this is done by creating a "test-suite" on which both the approaches are applied. The test-suite defines the data set and all the parameters needed for the problem. In this case our data set is the numbers for which we compute the square root; the additional parameter is the amount of accuracy needed, which is specified by the value of `eps`. Our process is as follows: *For each value of eps run the program on all the numbers in the data set and note the sum of the number of multiplications and divisions in each case.* This process is carried out for both programs and the results are noted in 2 tables.

Step 1. *Defining the test suite.* We will use the ten input values starting from 0.2, increasing by 10% each time. The `eps` values are $0.1, 0.05, 0.02, 0.01, 0.005, 0.002, 0.001, 0.0005, 0.0002$.

Step 2. *Collecting results.* Construct a table with one row for each data value and one column for each `eps` value. We will have two such tables, one for divide-and-conquer and one for Newton's method. The table will be filled by running the program and recording the number of multiplications and divisions of the loop. For instance, the entry in the Newton table corresponding to data value 0.2 and `eps` value 0.002 will be the number of multiplications and divisions performed by Newton's

method to find the square root of 0.2 with `eps` set to 0.002. **For each column, compute the average.**

Step 3. *Comparing the results.* Compare the data from the two tables from two viewpoints:
The total number of operations. Which method do you think is performing fewer operations? Explain.
The scalability of the algorithm. **Scalability** refers to how well the algorithm *scales up*, i.e., how does the amount of resources used (in this case, the number of operations) increase with increased requirement (in this case, greater accuracy)?. Scalability is usually represented by a graph, that plots the resources used against the requirements. Can you think of a way of doing that?

What to upload on Github. Create a folder named Hwork5 within your Github assignment and upload/create the following:

1. Answer to Question 1.
2. Flowchart from Question 2.
3. Script file and `.cpp` file from Question 3.
4. The table of results from Step 2.
5. Answers to questions in Step 3.