This lab exercise involves writing a program to calculate an *approximation* for the square root of a number entered by the user. In this assignment, you are not allowed to call any library function for square root.

The program must output (in column format) the intermediate values that it uses in its calculations. After the calculations are complete, the program must output the final result. (Refer to the **Sample Output** section of this document for more details.)

Due Date

You must *demonstrate* the solution to this lab exercise to the instructor during class by

Saturday, September 26, 2020,

in order to receive full credit for this work.

REMINDER

The most important detail to remember about any computer work you do is to DOUBLE CHECK that your work is saved on a backup device, such as a flash drive.

In case you do not have a flash drive with you: another technique for saving your work is to compose an EMail message to yourself, and attach the source file as an attachment to that EMail. Do this *before* you shut down or reboot the PC.

Always remember: make small, incremental changes. Test each small change as you go.

Design the Square Root Program

The program must have two nested loops:

- An outer loop that gets input from the user and contains all of the calculations. This loop repeats as long as the user keeps replying "**y**" or "**yes**" to the "**Keep running?**" prompt.
- An inner loop that repeats the approximation calculations until the **difference** value is less than the **tolerance** value.

This program must ask the user to input two values:

- The number for which we need to calculate the square root. (Save this value in the **inputNumber** variable.)
- The "tolerance" (permissible error) for the calculated result.

The algorithm for calculating square root involves starting with an estimate for the square root: the *initial* estimate we will use is **inputNumber / 2.0**. We do not expect the initial estimate to very close to the real square root value, so we perform a calculation to determine **how bad** the initial estimate is. Then we use this information to refine our estimate. We repeat the process until we decide that the estimate is "close enough".

Inputs from the user:

```
double inputNumber; // (we will calculate the square root of this number.)
double tolerance;   // the acceptable error for the approximation.
```

Variables used in the calculations:

```
double estimate;    // a "guess" of the square root result
double quotient;    // inputNumber / estimate
double difference;  // difference between estimate and quotient.
int    n;           // number of "guesses" so far (loop counter).
```

Outputs to the screen:

For each pass through the loop, output the intermediate values of the variables used in the calculations, formatted in columns. (See also the **Sample Output** section of this document.)
After the loop exits, output the final result.

Calculations:

The *initial* value of **`estimate`** is **`inputNumber / 2.0`**.

The program must have a **`do-while`** loop that performs the following steps:

- Calculate: **`quotient = inputNumber / estimate`**.
- Calculate: **`difference = fabs(estimate - quotient).`**
  (The **`fabs`** function is part of the **`cmath`** library.)
- Output: The intermediate values of **`n`**, **`estimate`**, **`quotient`**, and **`difference`**.
- Calculate: a new value for **`estimate`**: the average between the *previous* value of **`estimate`** and the **`quotient`**.

The loop continues until the **`difference`** is less than the **`tolerance`**.

Before the beginning of the inner loop, output column headings for the intermediate values. Use these column headings:

```
 n              estimate              quotient              difference
_               ________              ________              __________
```

REMINDER

The most important detail to remember about any computer work you do is to DOUBLE CHECK that your work is saved on a backup device, such as a flash drive.

In case you do not have a flash drive with you: another technique for saving your work is to compose an EMail message to yourself, and attach the source file as an attachment to that EMail. Do this *before* you shut down or reboot the PC.

Always remember: make small, incremental changes. Test each small change as you go.

Sample Output

In the sample output shown below, text that the user types is shown in **BOLD** font. When the program actually runs, all text is shown in the same font.)

Sample Input / Output

```
Enter a floating point number:  10000
Enter the desired tolerance value:  0.0001
        tolerance = 0.00010000000000000000

Estimating square root of  10000.000000000000000000

  n                 estimate                  quotient                 difference
  _                 ________                  ________                 __________
  1  5000.000000000000000000     2.000000000000000000  4998.000000000000000000
  2  2501.000000000000000000     3.998400639744102314  2497.001599360256022919
  3  1252.499200319871988540     7.984037033673259565  1244.515163286198685455
  4   630.241618676772645813    15.866930560688068397   614.374688116084598732
  5   323.054274618730346447    30.954550939782581764   292.099723678947782446
  6   177.004412779256455224    56.495766647756269663   120.508646131500185561
  7   116.750089713506355338    85.653039107199418822    31.097050606306936515
  8   101.201564410352887080    98.812701742948576111     2.388862667404310969
  9   100.007133076650731596    99.992867432120803528     0.014265644529928068
 10   100.000002543857675620    99.999999745614232438     0.000000508771535124

 The square root of  10000.000000000000000000 is    100.000002543857675620 (+/-
0.000100000000000000)
Keep running?  y
Enter a floating point number:  10000
Enter the desired tolerance value:  0.000000001
        tolerance = 0.000000001000000000

Estimating square root of  10000.000000000000000000

  n                 estimate                  quotient                 difference
  _                 ________                  ________                 __________
  1  5000.000000000000000000     2.000000000000000000  4998.000000000000000000
  2  2501.000000000000000000     3.998400639744102314  2497.001599360256022919
  3  1252.499200319871988540     7.984037033673259565  1244.515163286198685455
  4   630.241618676772645813    15.866930560688068397   614.374688116084598732
  5   323.054274618730346447    30.954550939782581764   292.099723678947782446
  6   177.004412779256455224    56.495766647756269663   120.508646131500185561
  7   116.750089713506355338    85.653039107199418822    31.097050606306936515
  8   101.201564410352887080    98.812701742948576111     2.388862667404310969
  9   100.007133076650731596    99.992867432120803528     0.014265644529928068
 10   100.000002543857675620    99.999999745614232438     0.000000508771535124
 11   100.000000000000000000   100.000000000000000000     0.000000000000000000

 The square root of  10000.000000000000000000 is    100.000000000000000000 (+/-
0.000000001000000000)
Keep running?  y
Enter a floating point number:  25
```

Sample Input / Output

```
Enter the desired tolerance value:  0.01
        tolerance = 0.01000000000000000021

Estimating square root of     25.000000000000000000

  n                 estimate                 quotient                 difference
  _    ____________________     ____________________    ____________________
  1    12.500000000000000000     2.000000000000000000    10.500000000000000000
  2     7.250000000000000000     3.448275862068965303     3.801724137931034697
  3     5.349137931034482207     4.673650282030621383     0.675487649003860824
  4     5.011394106532551795     4.988631799564816838     0.022762306967734958
  5     5.000012953048684317     4.999987046984871952     0.000025906063812364

 The square root of     25.000000000000000000 is      5.000012953048684317 (+/-
0.010000000000000000)
Keep running?  y
Enter a floating point number:  25
Enter the desired tolerance value:  0.0000001
        tolerance = 0.00000010000000000000

Estimating square root of     25.000000000000000000

  n                 estimate                 quotient                 difference
  _    ____________________     ____________________    ____________________
  1    12.500000000000000000     2.000000000000000000    10.500000000000000000
  2     7.250000000000000000     3.448275862068965303     3.801724137931034697
  3     5.349137931034482207     4.673650282030621383     0.675487649003860824
  4     5.011394106532551795     4.988631799564816838     0.022762306967734958
  5     5.000012953048684317     4.999987046984871952     0.000025906063812364
  6     5.000000000016777690     4.999999999983222310     0.000000000033555381

 The square root of     25.000000000000000000 is      5.000000000016777690 (+/-
0.000000100000000000)
Keep running?  y
Enter a floating point number:  2
Enter the desired tolerance value:  0.00001
        tolerance = 0.00001000000000000000

Estimating square root of      2.000000000000000000

  n                 estimate                 quotient                 difference
  _    ____________________     ____________________    ____________________
  1     1.000000000000000000     2.000000000000000000     1.000000000000000000
  2     1.500000000000000000     1.333333333333333259     0.166666666666666741
  3     1.416666666666666519     1.411764705882353033     0.004901960784313486
  4     1.414215686274509665     1.414211438474870075     0.000004247799639590

 The square root of      2.000000000000000000 is      1.414215686274509665 (+/-
0.000010000000000000)
Keep running?  y
Enter a floating point number:  2
```

Sample Input / Output

```
Enter the desired tolerance value:  0.0000001
        tolerance = 0.00000010000000000000

Estimating square root of      2.000000000000000000

  n                 estimate                 quotient                 difference
  _     1.000000000000000000     2.000000000000000000     1.000000000000000000
  2     1.500000000000000000     1.333333333333333259     0.166666666666666741
  3     1.416666666666666519     1.411764705882353033     0.004901960784313486
  4     1.414215686274509665     1.414211438474870075     0.000042247799639590
  5     1.414213562374689870     1.414213562371500199     0.000000000003189671

 The square root of      2.000000000000000000 is      1.414213562374689870 (+/-
0.000000100000000000)
Keep running?  y
Enter a floating point number:  10
Enter the desired tolerance value:  0.001
        tolerance = 0.00100000000000000002

Estimating square root of     10.000000000000000000

  n                 estimate                 quotient                 difference
  _     5.000000000000000000     2.000000000000000000     3.000000000000000000
  2     3.500000000000000000     2.857142857142857206     0.642857142857142794
  3     3.178571428571428825     3.146067415730336769     0.032504012841092056
  4     3.162319422150882797     3.162235898737389750     0.000083523413493047

 The square root of     10.000000000000000000 is      3.162319422150882797 (+/-
0.001000000000000000)
Keep running?  y
Enter a floating point number:  10
Enter the desired tolerance value:  0.0000001
        tolerance = 0.00000010000000000000

Estimating square root of     10.000000000000000000

  n                 estimate                 quotient                 difference
  _     5.000000000000000000     2.000000000000000000     3.000000000000000000
  2     3.500000000000000000     2.857142857142857206     0.642857142857142794
  3     3.178571428571428825     3.146067415730336769     0.032504012841092056
  4     3.162319422150882797     3.162235898737389750     0.000083523413493047
  5     3.162277660444136274     3.162277659892622328     0.000000000551513946

 The square root of     10.000000000000000000 is      3.162277660444136274 (+/-
0.000000100000000000)
Keep running?  n

C:\CSC237\Lab\Lab05a_SquareRoot\SquareRoot\Debug\SquareRoot.exe (process 13908)
exited with code 0.
```

Demonstrate the Working Program to the Instructor

Demonstrate the working program to the instructor.

Be sure to save a copy of the source file in a safe place for future reference.