This lab exercise has three requirements:

1. Write a program that calculates 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 actual algorithm is described in the **Design the Square Root Program** section of this document.
2. The program must output (in column format) the *intermediate values* that it uses in its calculations. This data must be precisely formatted to look like the **Sample Output** section of this document.
3. 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 *submit* the source code for the solution to this lab exercise to *Moodle* by

Tuesday, June 17, 2025

in order to receive full credit for this work. You must also *demonstrate* the working program to the instructor during class, within two weeks of submitting the source code. (In other words, **submit** the source code by the due date, and do not let yourself get too far behind regarding the demonstration of your working solution during class. This is the same policy that is described in the course *Syllabus*.)

HINT:

To make the formatting of output easier, add these two statements near the beginning of the program:

```
int precision = 18;
int fieldWidth = 25;
```

and then use those variables as arguments for the **`setprecision()`** and **setw()** directives to format the output.

Design the Square Root Program

The program must have two **nested loops**:

- An **outer loop** that gets input values from the user, outputs column headings and displays the final result. 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 and outputs intermediate results until the **difference** value is less than the **tolerance** value.

The outer loop must ask the user to input three 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 initial "**estimate**" for the square root.

The algorithm for calculating square root involves starting with an estimate for the value of the square root. We do not necessarily expect the initial estimate to be 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 latest 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.
double estimate;     //  an initial "guess" of the square root result
```

Variables used in the calculations:

```
double quotient;    // inputNumber / estimate
double difference;  // difference between estimate and quotient.
double result;      // copy of the estimate variable, before the next calculation
int    n;           // number of "guesses" so far (loop counter).
```

Outputs to the screen:

For each pass through the inner 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 inner loop exits, output the final result.

Calculations:

The **inner loop** of the program must be 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`**.
- Copy: Save a copy of the latest **`estimate`** value in **`result`**.
- Calculate: a <u>new</u> 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
_                  ________                  ________                 __________
```

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
Enter the initial estimate:  5
        tolerance = 0.00010000000000000000

Estimating square root of  10000.000000000000000000

  N                 estimate                  quotient               difference
  _                 ________                  ________               __________
  1     5.000000000000000000  2000.000000000000000000  1995.000000000000000000
  2  1002.500000000000000000     9.975062344139651316   992.524937655860298946
  3   506.237531172069850527    19.753572945979396280   486.483958226090464905
  4   262.995552059024646496    38.023456753198921376   224.972095305825718015
  5   150.509504406111773278    66.440986829758827525    84.068517576352945753
  6   108.475245617935300402    92.186931156822382150    16.288314461112918252
  7   100.331088387378841276    99.670004190425501633     0.661084196953339642
  8   100.000546288902171455    99.999453714082122247     0.001092574820049208
  9   100.000000001492139745    99.999999998507860255     0.000000002984279490

 The square root of  10000.000000000000000000
                 is    100.000000001492139745
               (+/-      0.000100000000000000)
Keep running?  y
Enter a floating point number:  10000
Enter the desired tolerance value:  0.0000000000001
Enter the initial estimate:  5
        tolerance = 0.00000000000010000000

Estimating square root of  10000.000000000000000000

  N                 estimate                  quotient               difference
  _                 ________                  ________               __________
  1     5.000000000000000000  2000.000000000000000000  1995.000000000000000000
  2  1002.500000000000000000     9.975062344139651316   992.524937655860298946
  3   506.237531172069850527    19.753572945979396280   486.483958226090464905
  4   262.995552059024646496    38.023456753198921376   224.972095305825718015
  5   150.509504406111773278    66.440986829758827525    84.068517576352945753
  6   108.475245617935300402    92.186931156822382150    16.288314461112918252
  7   100.331088387378841276    99.670004190425501633     0.661084196953339642
  8   100.000546288902171455    99.999453714082122247     0.001092574820049208
  9   100.000000001492139745    99.999999998507860255     0.000000002984279490
 10   100.000000000000000000   100.000000000000000000     0.000000000000000000

 The square root of  10000.000000000000000000
                 is    100.000000000000000000
               (+/-      0.000000000000100000)
```

Sample Input / Output

```
Keep running?  y
Enter a floating point number:  25
Enter the desired tolerance value:  0.01
Enter the initial estimate:  1
        tolerance = 0.01000000000000000021

Estimating square root of     25.000000000000000000

  N                  estimate                   quotient               difference
  _                  ________                   ________               __________
  1     1.000000000000000000    25.000000000000000000    24.000000000000000000
  2    13.000000000000000000     1.923076923076923128    11.076923076923076650
  3     7.461538461538461675     3.350515463917525860     4.111022997620935371
  4     5.406026962727993990     4.624468241161800997     0.781558721566192993
  5     5.015247601944897937     4.984798754563000145     0.030448847381897792
  6     5.000023178253949041     4.999976821853496567     0.000046356400452474

 The square root of     25.000000000000000000
                 is      5.000023178253949041
               (+/-      0.010000000000000000)
Keep running?  y
Enter a floating point number:  25
Enter the desired tolerance value:  0.0000000000001
Enter the initial estimate:  1
        tolerance = 0.00000000000010000000

Estimating square root of     25.000000000000000000

  N                  estimate                   quotient               difference
  _                  ________                   ________               __________
  1     1.000000000000000000    25.000000000000000000    24.000000000000000000
  2    13.000000000000000000     1.923076923076923128    11.076923076923076650
  3     7.461538461538461675     3.350515463917525860     4.111022997620935371
  4     5.406026962727993990     4.624468241161800997     0.781558721566192993
  5     5.015247601944897937     4.984798754563000145     0.030448847381897792
  6     5.000023178253949041     4.999976821853496567     0.000046356400452474
  7     5.000000000053722360     4.999999999946277640     0.000000000107444720
  8     5.000000000000000000     5.000000000000000000     0.000000000000000000

 The square root of     25.000000000000000000
                 is      5.000000000000000000
               (+/-      0.000000000000100000)
Keep running?  y
Enter a floating point number:  2
Enter the desired tolerance value:  0.00001
Enter the initial estimate:  1
        tolerance = 0.00001000000000000000

Estimating square root of      2.000000000000000000

  N                  estimate                   quotient               difference
```

Sample Input / Output

```
  _                ________          ________            _________
  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
Enter the desired tolerance value:  0.0000000000001
Enter the initial estimate:  1
        tolerance = 0.000000000000100000

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
  5     1.414213562374689870     1.414213562371500199     0.000000000003189671
  6     1.414213562373094923     1.414213562373095145     0.000000000000000222

 The square root of      2.000000000000000000
                 is      1.414213562373094923
               (+/-      0.000000000000100000)
Keep running?  y
Enter a floating point number:  10
Enter the desired tolerance value:  0.001
Enter the initial estimate:  5
        tolerance = 0.001000000000000000002

Estimating square root of     10.000000000000000000

  N                 estimate                 quotient               difference
  _                ________                 ________               _________
  1     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:  0.0000000000001
Enter the initial estimate:  5
```

Sample Input / Output

```
        tolerance = 0.00000000000010000000

Estimating square root of     10.000000000000000000

  N                  estimate                  quotient               difference
  _                  ________                  ________               __________
  1     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
  6     3.162277660168379079     3.162277660168379523     0.000000000000000444

 The square root of     10.000000000000000000
                is      3.162277660168379079
              (+/-      0.000000000000100000)
Keep running?  n
```

This algorithm also corrects itself if the initial estimate of the square root is obviously wrong:

Sample Result for Obviously Incorrect Initial Estimate

```
Enter a floating point number:  2
Enter the desired tolerance value:  0.0000000000001
Enter the initial estimate:  25
        tolerance = 0.00000000000010000000

Estimating square root of      2.000000000000000000

  N                  estimate                  quotient               difference
  _                  ________                  ________               __________
  1    25.000000000000000000     0.080000000000000002    24.920000000000001705
  2    12.539999999999999147     0.159489633173843709    12.380510366826156243
  3     6.349744816586921026     0.314973287552526959     6.034771529034394177
  4     3.332359052069723937     0.600175421900530970     2.732183630169192856
  5     1.966267236985127509     1.017155736707791025     0.949111500277336484
  6     1.491711486846459156     1.340741837570805339     0.150969649275653817
  7     1.416226662208632359     1.412203324064639443     0.004023338143992916
  8     1.414214993136635901     1.414212131611001677     0.000002861525634223
  9     1.414213562373818789     1.414213562372371280     0.000000000001447509
 10     1.414213562373094923     1.414213562373095145     0.000000000000000222

 The square root of      2.000000000000000000
                is      1.414213562373094923
              (+/-      0.000000000000100000)
Keep running?  y
```

Sample Result for Obviously Incorrect Initial Estimate

```
Enter a floating point number:  100
Enter the desired tolerance value:  0.0000000000001
Enter the initial estimate:  5000
        tolerance = 0.00000000000010000000

Estimating square root of    100.000000000000000000

  N                 estimate                  quotient               difference
  _                 ________                  ________               __________
  1  5000.000000000000000000     0.020000000000000000  4999.979999999999563443
  2  2500.010000000000218279     0.039999840000639997  2499.970000159999472089
  3  1250.024999920000482234     0.079998400037119125  1249.945001519963398096
  4   625.052499160018783186     0.159986561343864242   624.892512598674898072
  5   312.606242860681334150     0.319891244285120779   312.286351616396189002
  6   156.463067052483239650     0.639128465802453305   155.823938586680782237
  7    78.551097759142848531     1.273056683518603460    77.278041075624244627
  8    39.912077221330726218     2.505507279048751368    37.406569942281976182
  9    21.208792250189738127     4.715025675217567880    16.493766574972170247
 10    12.961908962703653003     7.714913002995012370     5.246995959708640633
 11    10.338410982849332242     9.672666347458298119     0.665744635391034123
 12    10.005538665153814293     9.994464400829208728     0.011074264324605565
 13    10.000001532991511510     9.999998467008722969     0.000003065982788542
 14    10.000000000000117240     9.999999999999882760     0.000000000000234479
 15    10.000000000000000000    10.000000000000000000     0.000000000000000000

 The square root of    100.000000000000000000
                is     10.000000000000000000
              (+/-      0.000000000000100000)
Keep running?  n
```

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.