Project 3 (in C++): You are to implement the bi-Gaussian automatic threshold selection method of a given histogram.

****

Project points: 12 pts

Language: C++

Due Date: Soft copy (*.zip) and hard copies (*.pdf):

+1 (13/12 pts): early submission, 9/28/2022, Wednesday before midnight
-0 (12/12 pts): on time, 10/2/2022, Sunday before midnight
-1 (11/12 pts): 1 day late, 10/3/2022, Monday before midnight
-2 (10/12 pts): 2 days late, 10/4/2022, Tuesday before midnight
(-12/12 pts): non submission, 10/4/2022, Tuesday after midnight

*** Name your soft copy and hard copy files using the naming convention as given in the project submission requirement.
*** All on-line submission MUST include Soft copy (*.zip) and hard copy (*.pdf) in **the same email attachments** with correct email subject as stated in the email requirement; otherwise, your submission will be rejected.

1. Implement your program as given the specs below.
2. Run your program on data1 and data2.
3. Include in your hard copy *.pdf file as follows:
    - Cover page.
    - source code.
    - Output outFile1 for data1.
    - Output outFile2 for data1.
    - Output outFile1 for data2.
    - Output outFile2 for data2.

****

I. inFile (argv[1]): a text file representing a histogram of a gray-scale image.
The input format as follows: The first text-line is the image header,
follows by a list of pairs <i, j> where i = 0 to max and j is the hist(i)
For example:

```
5  7   0   9 // 5 rows, 6 cols, min is 0 max 9
0  2              // hist[0] is 2
1  8              // hist[1] is 8
2  5                :
3  1                :
:                   :
:                   :
8  5                :
9  2              // hist [9] is 2
```

****

II. a) outFile1(argv [2]): This file includes the followings:

1. Display the input bimodal histogram as a curve. // With a caption.

2. Display of the two best-fitted Gaussians curves. // With a caption.

3. Output the selected threshold. //With a caption

4. Display the histogram curve with a vertical line at the selected threshold on the same page.

5. Display the histogram curve, the two best-fitted Gaussians curves
and the vertical line of the threshold all three on the same page.

b) outFile2 (argv [3]): // To output some intermediate results in methods, for debugging purposes,

****

III. Data structure:

****

- BiMean class
  - (int) numRows
  - (int) numCols
  - (int) minVal
  - (int) maxVal
  - (int) maxHeight // The largest histAry[i] of the given portion of the histogram.
  - (int) maxGVal // The "maximum calculated distribution value",
    // which is the maximum value returned by our gaussian function.
  - (int) offSet // offSet is set to one-tenth of the maxVal- minVal.
    // The assumption: in a bimodal histogram, the first modal occupies at least one-tenth of
    // the histogram population from minVal to maxVal of the histogram
  - (int) dividePt // Initially, dividePt is set to offSet, it increases by 1 at each iteration.
    // The selected threshold value is at the point at dividePt where the "distance"
    // between the two bi-Gaussians curves and the histogram is the minimum.
  - (int *) histAry // a 1D integer array (size of maxVal + 1) to store the histogram.
    // It needs to be dynamically allocated at run time; initialize to zero.
  - (int *) GaussAry // a 1D integer array (size of maxVal + 1) to store the "modified" Gaussian function.
    // It needs to be dynamically allocated at run time.
  - (char**) histGraph // a 2-D integer array (size of maxVal+1 by maxHeight+1), initialize to blank
    // for displaying the histogram curve. It needs to be dynamically allocated at run time.
  - (char**) GaussGraph // a 2-D integer array, size of maxVal+1 by maxHeight+1, initialize to blank,
    // for displaying Gaussian curves in 2D. It needs to be dynamically allocated at run time.

  Methods:
  - constructor (…) // It dynamically allocates all member arrays and initialization.
  - (int) loadHist ( ) // reads and loads the histAry from inFile and **<u>returns</u>** the max among hist[i]. On your own.

  -plotGragh(ary, graph, symbol) // maps 1D array onto 2D array with symbol.
    // i.e., if ary[i] > 0 then graph[i, ary[i]] ← symbol
    // symbol will be '*' for histGraph, and '+' for GaussGraph. On your own.
  - addVertical (…) // add histGraph[thr, j] with '|'where j = 0 to maxHeight. On your own.
  - (double) computeMean (leftIndex, rightIndex.)
    // computes the histogiven portion of from given leftIndex to rightIndex. of the histogram
    // and returns the *weighted* average of the histogram. See algorithm below.
  - (double) computeVar (leftIndex, rightIndex) // computes and returns the *weighted* variance,
    // from the given leftIndex to rightIndex. See algorithm below.
  - modifiedGauss (x, mean, var, maxHeight)
    // The original Gaussian function is
    // g(x) = a* exp (- ((x-b)^2)/(2*c^2)))
    // where a is the height of the Gaussian Bell curve, i.e.,
    // a = 1/(sqrt(c^2 * 2 * pi); b is mean and c^2 is variance
    // Here, the modified method replace 'a' in g(x) with maxHeight
    // G(x) = maxHeight * exp(- ((x-mean)^2 / (2* c^2))
    // The method returns G(x)
    // Alternatively, instead of using maxHeight, one can use
    // G(x) = maxHeight / maxGVal * g(x), where
    // maxGVal is the largest g(x).
    // If you are interest, you may use as such,
    // however, use maxHeight is good enough for this project.

- setZero(Ary) // Set 1D Ary to zero; on your own.
- (int) biMeanGauss (thrVal) // this is the principle method that
  // determines the best threshold selection (via fitGauss method)
  // where the two Gaussian curves fit the histogram the best; see algorithm below
- fitGauss(...) // computes the Gaussian curve fitting to the histogram; see algorithm below
- bestFitGauss(...)// put the 2 best fit Gaussians curves onto GaussAry; see algorithm below
- plotAll(…) // overlay histGraph with add vertical line and GaussGraph to File1.
  // on your own. You should know how to write this method.

// You may add methods as deem fit.

****
IV. Main (...)
****

```
step   0: inFile, outFile1, outFile2 ← open via argv[]
step   1: numRows, numCols, minVal, maxVal ← read from inFile
step   2: maxHeight ← loadHist (histAry, inFile)
step   3: use constructor to dynamically allocates all member arrays and initialization.
step   4: plotGraph (histAry, histGraph, ‘*’)
           outFile1 ← histGraph // output histGraph with caption
Step   5: offSet ← (int) (maxVal - minVal)  / 10
           dividePt ←  offSet
step   6: bestThrVal ← biMeanGauss (dividePt, outFile2)
step   7: bestFitGauss (bestThrVal, GaussAry)
step   8: plotGraph (GaussAry, GaussGraph, ‘+’)
          outFile1 ← GaussGraph // output GaussGraph with caption
step   9: outFile1 ← output bestThrVal // with caption
step   10: addVertical(histGraph, ‘|’)
           outFile1 ← histGraph // output histGraph with caption
step   11: outFile1 ←plotAll(…)  // with caption
step   12: close all files
```

****
V. (int) biMeanGauss (dividePt, outFile2)
****

```
Step 0: (double) sum1
        (double) sum2
        (double) total
        (double) minSumDiff
        bestThr ← dividePt
         minSumDiff ← 999999.0 // a large value

Step 1: setZero (GaussAry) // reset for each iteratio
step 2: sum1 ← fitGauss (0, dividePt, GaussAry)
                // fitting the first Gaussian curve
Step 3: sum2 ← fitGauss (dividePt, maxVal, GaussAry)
                // fit the second Gaussian curve
Step 4: total ← sum1 + sum2
Step 5: if total < minSumDiff
                 minSumDiff ← total
                 bestThr ← dividePt
Step 6: outFile2 ← print dividePt, sum1, sum2, total, minSumDiff and bestThr
Step 7: dividePt ++
step  8: repeat step 1 to step 9 while dividePt < (maxVal – offSet)
step 9: return bestThr
```

```
*************************************
VI. double fitGauss(leftIndex, rightIndex, GaussAry)
*************************************
Step 0: (double) mean
        (double) var
        (double) sum
        (double) Gval
        (double) maxGval
        sum ← 0.0
step 1: mean ←  computeMean (leftIndex, rightIndex, maxHeight)
        var ← computeVar (leftIndex, rightIndex, mean)
Step 2: index ← leftIndex
Step 3: Gval ← modifiedGauss (index, mean, var, maxHeight)
Step 4: sum += abs (Gval – (double)histAry[index])
Step 5: GaussAry[index] ← (int) Gval
Step 6: index ++
Step 7: repeat step 3 – step 6 while index <= rightIndex
Step 8: return sum
*************************************
VII. (double) computeMean (leftIndex, rightIndex, maxHeight)
*************************************
Step 0: maxHeight ← 0 // maxHight came via parameter, it is NOT local variable!
        sum ← 0
        numPixels ← 0
Step 1: index ← leftIndex
Step 2: sum += (hist[index] * index)
       numPixels += hist[index]
Step 3: if hist[index] > maxHeight
                maxHeight ← hist[index]
Step 4: index++
Step 5: repeat Step 2 to step 4 while index < rightIndex

Step 6: return (double)sum / (double) numPixels
*************************************
VIII. (double) computeVar (leftIndex, rightIndex, mean)
*************************************
Step 0: sum ← 0.0
        numPixels ← 0
Step 1: index ← leftIndex
Step 2: sum += (double) hist [index] * ((double) index – mean)^2)
       numPixels += hist[index]
Step 3: index++
Step 4: repeat Step 2 to step 3 while index < rightIndex
Step 5: return (double) sum / (double) numPixels
*************************************
IX.(double) modifiedGauss (x, mean, var, maxHeight)
*************************************
return (double) (maxHeight * exp ( - ( (x-mean)^2 / (2*var) )
                // double check the equation!!
*************************************
X. bestFitGauss (bestThrVal, GaussAry) // this method call fitGauss to get the two best-fitted Gaussian
*************************************
step 0: sum1 (double), sum2 (double)
Step 1:  set1DZero(GaussAry)
step 2: sum1 ← fitGauss(0, bestThrVal, GaussAry)
Step 3: Sum2 ← fitGauss(bestThrVal, maxVal)
// Note, we ignore sum1 and sum2 here.
```