Project 6 (C++): Thinning is the 2nd methods to obtain the skeletons of objects in a given binary image. The thinning of an object is like peeling off one layer of object from 4 sides (north, south, west, and east) in iterations, until the object becomes a skeleton. Implement the thinning algorithm given in the lecture notes.

*** This is an easy project, for your own good, do the project on your own!

What you need to do:
1) You will have two (2) date files: image1 and image2 to test your program.
2) Run your program two times using each of test data
3) Include in your hard copies:
- Project cover page
- Project source code
- reformatPrettyPrint outFile1 for image1
- reformatPrettyPrint outFile2 for image1
- reformatPrettyPrint outFile1 for image2
- reformatPrettyPrint outFile2 for image2

****
Language: C++
Project points: 10pts
Due Date: Soft copy (*.zip) and hard copies (*.pdf):

10/10 on time: 4/9/2021 Friday before midnight.
+1 early submission: 4/5/2021 Monday before midnight.
-1 for 1 day late: 4/10/2021 Saturday before midnight.
-2 for 2 days late: 4/11/2021 Sunday before midnight.
-10/10: after 4/11/2021 Sunday after midnight.
-5/10: does not pass compilation
0/10: program produces no output
0/10: did not submit hard copy.

*** Follow "Project Submission Requirement" to submit your project.

****
I. Input: inFile (argv [1]): a binary image
****
II. Outputs: There are two outfiles:
a) outFile1 (argv [2]):to store the final thinning result with the image header.
b) outFile2 (argv [3]):
- reformatPrettyPrint input image with proper caption.
- reformatPrettyPrint after completing each cycle (after thinning all sides) with proper caption, i.e.,
("result of thinning : cycle – 1")
("result of thinning : cycle – 2")

****
III. Data structure:
****
- A Thinning class
- (int) numRows
- (int) numCols
- (int) minVal
- (int) maxVal
- (int) changeflag
- (int) cycleCount
- (int **) aryOne // a 2D array, need to dynamically allocate at run time of size numRows + 2 by numCols + 2.
- (int **) aryTwo // the same as firstAry

- methods:
- constructor(...)
// need to dynamically allocate aryOne and aryTwo
// assign values to numRows,..., etc.

- zeroFrame (Ary)// framing the extra 2 rows and extra 2 columns with zeros.
- loadImage (inFile, aryOne) // Read from the input file onto inside frame of firstAry
- copyArys () // always copy from aryTwo to aryOne

*** The following four thinning operations are given in the lecture notes; check aryOne and write the result to aryTwo; make sure only operate on pixels inside the frame of arrays

- NorthThinning (aryOne, aryTwo) // See the lecture note;
- SouthThinning (aryOne, aryTwo) //
- WestThinning (aryOne, aryTwo) // :
- EastThinning (aryOne, aryTwo) // :

- reformatPrettyPrint (aryTwo, file) // reuse code from your previous project

****
III. main (...)
****

step 0: inFile ← open input file from argv[1]
numRows, numCols, minVal, maxVal ← read from inFile
outFile1 ← open from argv [2]
outFile2 ← open from argv [3]
outFile1 ← write numRows, numCols, minVal, maxVal
dynamically allocate firstAry of size numRows + 2 by numCols + 2.
dynamically allocate secondAry of size numRows + 2 by numCols + 2.

step 1: zeroFrame(firstAry)
zeroFrame(secondAry)

step 2: loadImage (inFile, firstAry)
step 3: cycleCount ← 0
step 4: reformatPrettyPrint (firstAry, outFile2) // This print is before thinning

step 5: changeFlag ← 0

step 6: NorthThinning (firstAry, secondAry)
copyArys (…)

step 7: SouthThinning (firstAry, secondAry)
copyArys(...)

step 8: WestThinning (firstAry, secondAry)
copyArys(...)

step 9: EastThinning (firstAry, secondAry)
copyArys(...)

step 10: cycleCount ++

Step 11: reformatPrettyPrint (firstAry, outFile2)

Step 12: repeat step 5 to step 11 while changeFlag > 0

step 13: outFile1 ← output inside frame of firstAry from [1][1] *without* extra rows and cols

step 14: close all files