EGR 125 - Introduction to Engineering Methods (C++)   Due date: ___________________
File: N125-Ch8LB_and_Ch12LB

Chapter 8 and Chapter 12 Homework: Multidimensional Arrays and the Vector Class

Reading Assignment:

Read Chapters 8 and 12 in Introduction to Programming with C++, 3rd Edition by Liang

Problem Assignment:

1. (10 pts) Work the following ***Quick Check Exercises*** in the textbook. As usual, include the instructions and the given code with the answers.
   - ***Quick Check Exercise*** 8.4 on p. 313
   - ***Quick Check Exercise*** 8.5 on p. 314
2. (22 pts) ***Sum the major diagonal in a matrix***: Programming Exercise 8.2 on page 327. Additional details:
   - The size of an array cannot be changed based on user input (dynamic memory allocation), so the matrix should be dimensionsed to the max size ever expected (10 x 10 perhaps).
   - Prompt user to enter N (the size of the N x N matrix). The program should work for any N >= 2. N should not be used anywhere for matrix dimensions. It is only used in for loops to control user input and printing. Note that a 3x3 matrix is really just using the upper corner of the 10x10 matrix.
   - Prompt the user to enter the elements in the matrix row-by-row.
   - Display the NxN matrix.
   - Display the sum of the elements in the major diagonal. The sum should be displayed from the main function, not from the function sumMajorDiagonal.
   - Include a printout of the main program and the function.
   - Include printouts for the test case in the textbook as well as for a 2x2 matrix and a 3x3 matrix.

(continued on the following page)

3. (22 pts) ***Matrix of velocities***: Write a C++ program to convert velocities to alternate units as described below:
   - Use a single 2D array with 3 columns and 21 rows for the velocity values as follows:
     - column 0 – velocity in mi/h (mph)
     - column 1 – velocity in km/h
     - column 2 – velocity in ft/s
   - The 3D array is illustrated below where velocity varies from 0 – 100 mi/h:

0		
5		
10		
15		
20		
25		
30		
35		
40		
45		
50		
55		
60		
65		
70		
75		
80		
85		
90		
95		
100		

   - Use a loop or a list to assign the velocity values to column 0.
   - Use a loop to calculate and assign the velocities in columns 1 and 2.
   - Neatly display the velocity in three columns with a table heading. Select a reasonable number of digits for each column.

4. (22 pts) ***Vector Class***: Write a C++ program that reads an unspecified number of temperatures in Fahrenheit from the keyboard (prompt the user to enter them one at a time). The values should be added to a vector using the ***vector class*** with the ***push_back*** function. Instruct the user to enter an invalid temperature (<-459.4) to indicate that they are done entering values. Use functions and algorithms in the **<*algorithm*>** and **<*numeric*>** libraries to find the max, min, sum, and count (number of values) and display the results. Also sort the temperatures in increasing order and display the values. Include units with all outputs. Print the program along with an example that includes at least 10 valid temperatures.

5. (24 points) Determine the output for each part below by hand (do not compile the programs).

Part A	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = 0; i < R; i++)`
	`{   for (int j = 0; j < C; j++)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part B	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = 1; i < R; i++)`
	`{   for (int j = 1; j < C; j++)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part C	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int j = 0; j < C; j++)`
	`{   for (int i  = 0; i < R; i++)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part D	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = R-1; i >= 0; i--)`
	`{   for (int j = C-1; j >= 0; j--)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part E	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = 0; i < R; i++)`
	`{   for (int j = 0; j < i+1; j++)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part F	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = 0; i < R; i++)`
	`{   for (int j = 0; j < C; j++)`
	`        if(i>=j)cout<<setw(3)<<A[i][j];`
	`    cout << endl;  }`	

Part
G
	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = R-1; i >= 0; i--)`
	`{   for (int j = 0; j < C; j++)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	

Part
H
	`const int R = 4, C = 3;`
	`double A[R][C]={1,2,3,4,5,6,7,8,9,10,11,12};`
	`for (int i = 0; i < R; i++)`
	`{   for (int j = C-1; j >=0; j--)`
	`        cout << setw(3) << A[i][j];`
	`    cout << endl;  }`	