60-141 – Introduction to Programming II   Winter, 2017

Lab #3: Arrays

(Due at the beginning of the next lab period)

Objectives: Practice dealing with 2D arrays.

Pre-requisite(s):

- Read and review chapter 6.

Manipulating a 2D array:

1. Create a two dimensional array (e.g. **int A2D [M][N] ;**) of size **M x N** to store integer values. Use **`#define M 6`** and **`N 5`** to start. (Using symbolic constants instead of hard coding the array sizes improves scalability).
2. Populate the array pseudo-randomly, with integer numbers between 1 and **M x N**, inclusive, so that every array element is unique (no duplicates).
3. Print the array in a table format (use formatting codes to achieve this).
4. Use Linear Search to find if a number ***n*** is found in the array, where ***n*** is an integer between 1 and **M x N** (inclusive) entered by the user.
5. Apply a single LEFT shift operation to the array. LEFT shift means move every element one position to the LEFT; the first element becomes the last one, and the first element in each row moves up to become the last element in the previous row.

   Example: Left shift of a 2 x 4 array: becomes:

   ```
   4 8 3 2          8 3 2 5
   5 6 1 7          6 1 7 4
   ```
6. Print the shifted array.

Summary of the lab requirements: You must create an interactive menu within **main()** for this program (call it **Lab3.c**) with choices to fill the array with random numbers, search the array, left shift the array, print the array and either repeat a menu item or quit.

Design and document the following functions (REQUIRED):

- **PrintArray2D()** **--** to print the array.
- **PopulateRandom2D()** **--** to populate the array with pseudo-random numbers.
- **LinearSearch2D()** **--** to search the array for a value, return true if found.
- **LeftShift2D()** **--** to left shift the array.

Each one of the functions above accepts a 2D array as a parameter, along with any additional parameters that you may find necessary. The return types of the functions are also your choice. Do NOT use global (i.e. file scope) variables in this program.

Important Note: When passing 2D arrays as parameters to functions, it is necessary to specify the number of columns explicitly. Hence: **int foo(int A[][N]);** is valid and permits references to all elements within **A**, such as **A[i][j] for 0<=i and 0<=j<N**.

EVALUATION OF WORK AND ATTENDANCE: Total 5 marks.
You need to show to your lab instructor the work you have completed for this lab assignment, generally in the form of a working program. The marks you will receive for the lab are made of two parts, the programming part and lab attendance. Do not email your work to the Instructor or to any teaching assistant.

Lab Work Mark: You will be evaluated based on your solution for the problem assigned, using the following scheme: You must attend the lab and show your work in order to earn any of the following marks.

0 mark = No appreciable and relevant work done.
1 marks = Incomplete code / does not compile, with no, little or invalid documentation.
2 marks = Complete running program with no, little or invalid documentation.
3 marks = Incomplete code / does not compile, with suitable documentation.
4 marks = Complete running program with suitable documentation (minor errors may be accepted).

Attendance Mark: You will receive 1 mark for attendance during the lab period.

IMPORTANT:
ASK QUESTIONS IF YOU GET STUCK, BUT DO YOUR OWN CODING. SUBMITTING WORK COPIED FROM OTHER STUDENTS WILL RESULT IN A MARK OF ZERO (0) FOR THE LAB WORK MARK.