

## Data Structures

### Lab # 2: Functions, Arrays, Pointers and Dynamic Memory Allocation

---

#### Task-1: Functions and Arrays

(0.4 Points)

Write a C++ Program which creates a dynamic array (starter code lab2\_1\_startcode.cpp is available on Brightspace). The size of the array should be taken as an input from the user through terminal. Once created, the array should be populated with random values in the range of 0-100. The Program should also contain a function called **findMinMax()** which finds the minimum and the maximum values from the array and returns those values to the calling program (e.g. to main function).

Hints: Use “**new**” operator to **dynamically allocate** memory for the array in Heap memory. Use “**pass by reference parameters**” to the function, so that multiple values can be returned. Do not forget to delete the dynamically allocated memory at the end of the program.

The output of the program should look like the following:

```
Enter Size of the Array: 10
array[0]=99
array[1]=8
array[2]=79
array[3]=43
array[4]=65
array[5]=9
array[6]=66
array[7]=39
array[8]=38
array[9]=31
Min : 8
Max : 99
```

Figure 1

#### Task-2, 2D Arrays and Pointers (Matrix Arithmetic)

(0.6 Points)

Please download the starter code called lab2\_2\_startcode.cpp from NYU Class/Brightspace and complete the following functions:

- a) **int\*\* add(int\*\* matrix1, int\*\* matrix2, int rows, int cols)**

A function that adds two matrices (2D arrays) pointed by double pointers matrix1 and matrix2. The method should add both matrices (2D arrays) and return the result to the calling program (i.e. main function in this case). Both matrices have same dimensions.

- b) **int\*\* subtract(int\*\* matrix1, int\*\* matrix2, int rows, int cols)**

A method that subtracts matrix2 (2D arrays) from matrix1 and returns the result to the calling program. Both matrices have same dimensions.

- c) **int\*\* multiply(int\*\* matrix1, int\*\* matrix2, int rows1, int cols1, int rows2, int cols2)**

A function that multiplies two matrices (matrix1 and matrix2) and returns the answer to the calling program. You can assume that the number of columns of first matrix are equal to the number of rows of second matrix (a requirement for matrix multiplication)

- d) **void fill(int\*\* matrix, int rows, int cols)**

A procedure that populates the matrix (2D array) with random integer values in the range of 0 to 20.

e) **void display(int\*\* matrix, int rows, int cols)**

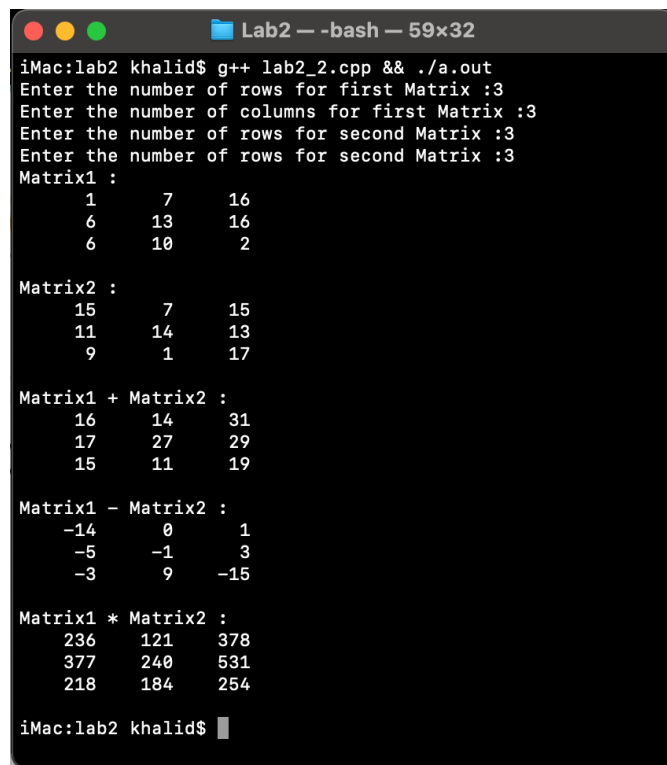
A procedure that prints a matrix (2D array) in two-dimension form. (see *Figure 2*)

f) **void cleanup(int\*\* matrix, int rows)**

A procedure that deletes dynamically created memory pointed by the pointer(s) matrix.

**Note:** Please do not modify the functions headers given in the starter code. When returning a value from a function, also keep in mind that the local variables created in a function will be destroyed upon returning from function.

The output of your program should look like the following:



```
Lab2 — -bash — 59x32
iMac:lab2 khalid$ g++ lab2_2.cpp && ./a.out
Enter the number of rows for first Matrix :3
Enter the number of columns for first Matrix :3
Enter the number of rows for second Matrix :3
Enter the number of rows for second Matrix :3
Matrix1 :
  1   7  16
  6  13  16
  6  10   2

Matrix2 :
 15   7  15
 11  14  13
  9   1  17

Matrix1 + Matrix2 :
 16  14  31
 17  27  29
 15  11  19

Matrix1 - Matrix2 :
-14   0   1
 -5  -1   3
 -3   9 -15

Matrix1 * Matrix2 :
236  121  378
377  240  531
218  184  254

iMac:lab2 khalid$
```

Figure 2

## Code of Conduct

All assignments are graded, meaning we expect you to adhere to the academic integrity standards of NYU Abu Dhabi. To avoid any confusion regarding this, we will briefly state what is and isn't allowed when working on an assignment/lab-task.

Any documents and program code that you submit must be fully written by yourself. You can, of course, discuss your ideas with fellow students, as long as these discussions are restricted to general solution techniques. Put differently, these discussions should not be about concrete code you are writing, nor about specific results you wish to submit. When discussing an assignment with others, this should never lead to you possessing the complete or partial solution of others, regardless of whether the solution is in paper or digital form, and independent of who made the solution, meaning you are also not allowed to possess solutions by someone from a different year or course, by someone from another university, or code from the Internet, etc. This also implies that there is never a valid reason to share your code with fellow students, and that there is no valid reason to publish your code online in any form.

Every student is responsible for the work they submit. If there is any doubt during the grading about whether a student created the assignment themselves (e.g. if the solution matches that of others), we reserve the option to let the student explain why this is the case. In case doubts remain, or we decide to directly escalate the issue, the suspected violations will be reported to the academic administration according to the policies of NYU Abu Dhabi.

(see <https://students.nyuad.nyu.edu/campus-life/community-standards/policies/academic-integrity/> )

---