Exercise 3

Arrays: Matrix Multiplication

Introduction:

In C++ arrays of arrays are called multi-dimensional arrays. By including two pair of brackets, a two-dimensional array would be created. By iterating the same idea, arrays of higher dimension (i.e. k-dimensional array) is possible. However, only two-dimensional arrays would be covered on this exercise because it is difficult to visualize arrays beyond three-dimensions [1].

```
	datatype arrayName [rowSize][columnSize];
	int twoDimArray[3][4];//two dimensional array with 3 rows and 4 columns
```

When a two-dimensional array is stored in the computer's memory, it is done through row-major ordering. That is, the first row is stored followed by the next row and so on. So in order for the compiler to know where a row would end, the number of columns must be included when passing a two-dimensional array to a function. Refer to the example below

```
	int f1(int twoDimArr[][4]);
```

Initializing a matrix is as follows:

```
	int x[2][2] = { {1,2}, {3,4}};// 1,2 for first row, 3,4 for second row
```

A matrix is defined as a set of numbers in a two-dimensional array, which has entries called elements. Matrix multiplication is defined by as,

If *A* is a *m x n* matrix and *B* is a *n x q* matrix, then the product *C = AB* is a *m x q* matrix. The entry of element C_{ij} is given with the equation [2]:

$$C_{ij} = \sum_{k=0}^{n-1} A_{ik} B_{kj}$$

Where:

i is the number of rows

j is the number of columns

n is the common number or columns for A and rows for B.

For example, compute AB if

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 1 \end{bmatrix}, B = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 3 \end{bmatrix} \; AB = \begin{bmatrix} 1(1)+3(2) & 1(2)+3(1) & 1(1)+3(3) \\ 2(1)+1(2) & 2(2)+1(1) & 2(1)+1(3) \end{bmatrix}$$

`void randInitMatrix (int A[][4], int B[][4])` randomly initialize matrix

`void matrixMultiply(int A[][4], int B[][4], int C[][4])` perform the multiplication process. Matrices A and B are the source matrix, while matrix C is the destination matrix. Note: use iteration operation to accomplish this task.

`void dispMatrices (int A[][4], int B[][4], int C[][4])` would be implemented to display properly the content of matrices A,B, and C.