**CSC 1101 – Problem Solving and Programming Laboratory**

**Lab 21 – Omar Faruk**

**25 points – Due Dec 1, end of class**

**a)** Save this document with your name and the homework number somewhere in the file name.

**b)** Type/paste your answers into the document.

**c)** Submit this document and your .cpp file(s) to the Canvas item where you downloaded this document. Do not submit a zip file but individually attach your files.

***Title: Product of Matrices***

Write a C++ program to find the product of two matrices. Each matrix contains integer values. We can multiply two matrices if the number of columns in first matrix is equal to the number of rows in second matrix. The size of multiplication matrix is **row of first matrix X column of second matrix**. You can follow the algorithm to write the code:

matrixMultiply(A, B):

Assume dimension of A is (m x n), dimension of B is (p x q)

Begin

if n is not same as p, then exit

otherwise define C matrix as (m x q)

for i in range 0 to m - 1, do

for j in range 0 to q – 1, do

for k in range 0 to p, do

C[i, j] = C[i, j] + (A[i, k] \* B[k, j])

done

done

done

End

You can also follow the sources:

1. <https://en.wikipedia.org/wiki/Matrix_multiplication>
2. sample app "Arrays-Multi-Dimensional.cpp" on Canvas.

Define matrices:

You can define fixed value of first 2D array with (row=2, column=3) and second 2D array (row=3, column=2). Product matrix contains 2X2 dimension.

Follow the sample I/O and make proper alignment.

**Sample Input/Output:**

Text

Description automatically generated

*[your program code here]\**

//==========================================================

//

// Title: Matrix to Matrices!

// Course: CSC 1101

// Lab Number: 21

// Author: Omar Faruk

// Date: 12/1/2020

// Description:

// Creating an application taking two fixed Matrices and

// creating a 3rd matrix if columns in matrix 1 is identical to

// columns in matrix 2 by multiplying the two matrices.

//

//==========================================================

#include <cstdlib> // For several general-purpose functions

#include <fstream> // For file handling

#include <iomanip> // For formatted output

#include <iostream> // For cin, cout, and system

#include <string> // For string data type

using namespace std; // So "std::cout" may be abbreviated to "cout"

// Declare Constants

const int COLFMT1 = 6;

const int ROW\_SIZE = 3, ROW\_SIZE2 = 2;

const int COL\_SIZE = 2, COL\_SIZE2 = 3;

// Print 2D Array Functions for different sizes

void print2DArray1(string heading,

int arr[ROW\_SIZE][COL\_SIZE])

{

// Loop to print array numbers

cout << "\n" + heading << endl;

cout << endl;

for (int i = 0; i < ROW\_SIZE; i++)

{

for (int j = 0; j < COL\_SIZE; j++)

cout << setw(COLFMT1) << right << arr[i][j];

cout << endl;

}

}

void print2DArray2(string heading,

int arr[ROW\_SIZE2][COL\_SIZE2])

{

// Loop to print array numbers

cout << "\n" + heading << endl;

cout << endl;

for (int i = 0; i < ROW\_SIZE2; i++)

{

for (int j = 0; j < COL\_SIZE2; j++)

cout << setw(COLFMT1) << right << arr[i][j];

cout << endl;

}

}

void print2DArray3(string heading,

int arr[ROW\_SIZE2][COL\_SIZE])

{

// Loop to print array numbers

cout << "\n" + heading << endl;

cout << endl;

for (int i = 0; i < ROW\_SIZE2; i++)

{

for (int j = 0; j < COL\_SIZE; j++)

cout << setw(COLFMT1) << right << arr[i][j];

cout << endl;

}

}

int main()

{

// Declare constant

const int M = 2, N = 3, P = 3, Q = 2;

// Show application header

cout << "Welcome to Matrix to Matricies!" << endl;

cout << "--------------------------" << endl << endl;

// Define Two Matrices

int arrMatrix1[M][N] =

{

{3, 4, 5},

{7, 8, 10}

};

int arrMatrix2[P][Q] =

{

{7, 8},

{2, 3},

{11, 12}

};

int arrMatrix3[2][2] =

{

{0, 0},

{0, 0},

};

// Multiply Matrices if matrix 1 and 2 value are same

if ( N != P)

{

cout << "\nNo columns and rows in corresponding matrix are the same" << endl;

}

else

{

for (int i = 0; i < M; i++)

{

for (int j = 0; j < Q; j++)

{

for (int k = 0; k < P; k++)

{

arrMatrix3[i][j] = arrMatrix3[i][j] + (arrMatrix1[i][k] \* arrMatrix2[k][j]);

}

}

}

}

// Print Matrix Arrays

print2DArray2("The First Matrix (size: 2X3):", arrMatrix1);

print2DArray1("The Second Matrix (size: 3X2):", arrMatrix2);

print2DArray3("The Product of two matrices is :", arrMatrix3);

// Show application close

cout << "\nEnd of Matrix to Matricies!" << endl;

}

*[your program output here]\*\**

