McGill University
School of Computer Science

COMP-206

Mini Assignment #4

Due: February 27, 2019 on myCourses at 23:30

A matrix in algebra is an array of numbers arranged in rows and columns. In C, it is common to implement Matrices using two dimensional arrays. We will be focusing on manipulating a fixed size 5x5 square matrix (5 rows and 5 columns).
Create a C program called **matrix.c** that implements multiple functions doing each a specific task. The functions that you have to implement are the following:

1. Write a function that will fill a 5x5 matrix by randomly generated numbers. You can use rand() and srand() functions provided by <stdlib.h> library to generate random numbers between 1 and 100. The signature of the function is:
   void fillMatrix(int matrix[rows][cols]);
   rows and cols are global constants defined as follow at the beginning of your c file:
   #define rows 5
   #define cols 5

2. Write a function that given a 5x5 matrix as parameter, will print each row of the matrix to the screen line by line. The signature of the function is:
   void printMatrix(int matrix[rows][cols]);
   Use array notation in your implementation.

3. In algebra, a transpose of a matrix is a matrix whose rows are the columns of the original. Write a function that will take a 5x5 matrix as parameter and then transpose it in-place which means without the use of an extra or a temporary matrix while transposing. The signature of the function is:
   void transposeMatrix(int matrix[rows][cols]);
   Use pointer notation in your implementation.

4. Write a function that given three matrices as parameters (two as input matrices and one as output), will calculate the product of the first 2 matrices and place the result in the third one.
   The signature of the function is:
   void multiplyMatrix(int m1[2][cols], int m2[rows][cols], int resultMatrix[rows][cols])
   Note that the first matrix isn't 5x5. Extra space in resultMatrix should be filled with zero. resultMatrix will hold the result of the multiplication of m1 X m2.
   Use pointer notation in your implementation.

The main() function that will be used to test your functions is provided as follow:

```
int main()
{
    int matrix[rows][cols];
    fillMatrix(matrix);    // Q1
    printMatrix(matrix);   // Q2
    transposeMatrix(matrix);  // Q3
    printMatrix(matrix);
    int matrix2[2][cols]={0,1,2,3,4,5,6,7,8,9};
    int matrixResult[rows][cols];
    multiplyMatrix(matrix2, matrix, resultMatrix); // Q4
    printMatrix(matrixResult);
    return 0;
}
```

WHAT TO HAND IN

Everything must be submitted to My Courses before the due date. Remember that you can hand in your assignment up to two days late but there will be a penalty of 5% each day. After that, your assignment will not be accepted. Please hand in the following:

- A single C code file called **matrix.c**

HOW IT WILL BE GRADED

The assignment is worth a total of 20 points.

- 20 points - matrix C program implementation as follow:
  - 6 points for fillMatrix function
    - 3 points for properly assign random values to the matrix
    - 2 points for generating non repeated random numbers with the use of srand
    - 1 points for respecting the range of numbers generated between 1 and 100
  - 2 points for printMatrix function
    - 1.5 points for pretty printing properly as instructed
    - 0.5 point for using array notation
  - 6 points for transposeMatrix function
    - 3 points for transposing properly the matrix

    - 2 points for using pointer notation
    - 1 points for inplace transposing without the use of temp array
- 6 points for multiplyMatrix function
    - 3 points for proper multiplication of the matrices
    - 2 points for using pointer notation
    - 1 points for filling unused space with 0

GRADING RULES

The following rules are followed by the TA when grading assignments:

- A program must run in order to get a grade (even if it does not run well). If it does not run (does not compile) it will receive a zero. (Make sure to run your programs from Trottier – they sometimes do not run the same from home when logging in using putty.)
- The TA will grade using the mimi.cs.mcgill.ca server.
- All questions are graded proportionally (assuming the program runs at all). This means that if 40% of the question is correct, you will receive 40% of the grade.
- The TA will compile the c code then runs it and observe the execution.
- The TA will check whether the printMatrix function is printing to the screen the rows one by one. Numbers should be separated by a tabulation. Each row must be printed on a separate line. The TA will also check if the implementation is using the array notation as specified in the instructions.
- The TA will check if the randomly generated numbers are comprised within the specified range [1 - 100]
- The TA will check whether the transposeMatrix function is displaying the new matrix and that it corresponds to the transpose of the previous matrix. The TA will also check if the implementation is using the pointer notation as specified in the instructions.
- The TA will check whether the multiplyMatrix function is displaying the new matrix and that it corresponds to the multiplication of the previous matrix by the newly provided one. The TA will also check if the implementation is using the pointer notation as specified in the instructions.