#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

void readMatrix(int array[10][10], int rows, int colums);

void printMatrix(int array[10][10], int rows, int colums);

void matrixAddSub(int arrayone[10][10], int arraytwo[10][10], int rows, int colums, int mul);

void matrixScalarMultiply(int array[10][10], int scalar, int rows, int colums);

void matrixMultiply(int arrayone[10][10], int arraytwo[10][10], int rowsA, int columsA, int columsB);

int main(void){

int i, j, k;

int matrixA[10][10];

int matrixB[10][10];

int rowA, colA;

int rowB, colB;

int operation;

char again = 'Y';

int scalar = 0;

int add = 1;

int sub = -1;

while (again == 'Y'){

printf("\nOperation Menu\n\t1. to Add\n\t2. to Subtract\n\t3.to Multiply two matrices\nEnter yout choice:");

scanf(" %d", &operation);

switch (operation){

case 1:

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

while ((rowA != rowB) && (colA != colB)){

printf("\nMatrices must be the same size\n");

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

}

printf("\n\tEnter elements of Matrix A a %d x %d matrix.\n", rowA, colA);

readMatrix(matrixA, rowA, colA);

printf("\n\t\tMatrix A\n\n");

printMatrix(matrixA, rowA, colA);

printf("\n\tEnter elements of Matrix B a %d x %d matrix.\n", rowB, colB);

readMatrix(matrixB, rowB, colB);

printf("\n\t\tMatrix B\n\n");

printMatrix(matrixB, rowB, colB);

printf("\nThe Sum of matrixA + matrixB is : \n");

matrixAddSub(matrixA, matrixB, rowA, colA, add);

break;

case 2:

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

while ((rowA != rowB) && (colA != colB)){

printf("\nMatrices must be the same size\n");

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

}

printf("\n\tEnter elements of Matrix A a %d x %d matrix.\n", rowA, colA);

readMatrix(matrixA, rowA, colA);

printf("\n\t\tMatrix A\n\n");

printMatrix(matrixA, rowA, colA);

printf("\n\tEnter elements of Matrix B a %d x %d matrix.\n", rowB, colB);

readMatrix(matrixB, rowB, colB);

printf("\n\t\tMatrix B\n\n");

printMatrix(matrixB, rowB, colB);

printf("\nThe difference between matrixA - matrixB is : \n");

matrixAddSub(matrixA, matrixB, rowA, colA, sub);

break;

case 3:

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

while (colA != rowB)

{

printf("\n\nError! column of first matrix not equal to row of second.\n\n");

printf("\nEnter the #rows and #cols for matrix A: ");

scanf("%d%d", &rowA, &colA);

printf("Enter the #rows and #cols for matrix B: ");

scanf("%d%d", &rowB, &colB);

}

printf("\n\tEnter elements of Matrix A a %d x %d matrix.\n", rowA, colA); // with the %d we remember the user the dimentions of the array

readMatrix(matrixA, rowA, colA);

printf("\n\t\tMatrix A\n\n");

printMatrix(matrixA, rowA, colA);

printf("\n\tEnter elements of Matrix B a %d x %d matrix.\n", rowB, colB); // with the %d we remember the user the dimentions of the array

readMatrix(matrixB, rowB, colB);

printf("\n\t\tMatrix A\n\n");

printMatrix(matrixB, rowB, colB);

matrixMultiply(matrixA, matrixB, rowA, colA, colB);

break;

default:

printf("\nIncorrect option! Please choose a number 1-4.");

break;

}

printf("\n\nDo you want to calculate again? Y/N\n");

scanf(" %c", &again);

again = toupper(again);

}

return 0;

}

void readMatrix(int array[10][10], int rows, int colums){

int i, j;

for (i = 0; i < rows; i++){

printf("\t%d entries for row %d: ", colums, i + 1);

for (j = 0; j < colums; j++){

scanf("%d", &array[i][j]);

}

}

return;

}

void printMatrix(int array[10][10], int rows, int colums){

int i, j;

for (i = 0; i < rows; i++) {

for (j = 0; j < colums; j++){

printf("\t%d", array[i][j]);

}

printf("\n");

}

}

void matrixAddSub(int arrayone[10][10], int arraytwo[10][10], int rows, int colums, int mul){

int i, j;

int sumM[10][10];

int scaM[10][10];

for (i = 0; i < rows; i++){

for (j = 0; j < colums; j++){

scaM[i][j] = mul \* arraytwo[i][j];

}

}

for (i = 0; i < rows; i++){

for (j = 0; j < colums; j++){

sumM[i][j] = arrayone[i][j] + scaM[i][j];

printf("\t%d", sumM[i][j]);

}

printf("\n");

}

}

void matrixMultiply(int arrayone[10][10], int arraytwo[10][10], int rowsA, int columsA,int columsB){

int i, j, k;

int mulM[10][10];

for (i = 0; i<rowsA; ++i)

for (j = 0; j<columsB; ++j)

{

mulM[i][j] = 0;

}

for (i = 0; i<rowsA; ++i)

for (j = 0; j<columsB; ++j)

for (k = 0; k<columsA; ++k)

{

mulM[i][j] += arrayone[i][k] \* arraytwo[k][j];

}

printf("\nOutput Matrix:\n");

for (i = 0; i<rowsA; ++i)

for (j = 0; j<columsB; ++j)

{

printf("\t%d ", mulM[i][j]);

if (j == columsB - 1)

printf("\n\n");

}

}