// Librerias

#include <iostream>

#include <random>

#include <chrono>

using namespace std;

// Prototipos de las funciones

size\_t menu();

size\_t leer\_numero(string);

int\*\* genere\_arreglo(int, int);

void imprime\_arreglo(int\*\*, int, int);

int\*\* suma(int\*\*, int\*\*, int, int);

int\*\* resta(int\*\*, int\*\*, int, int);

int\*\* multiplicacion(int\*\*, int\*\*, int, int);

int\*\* producto\_escalar(int, int\*\*, int, int);

int\*\* opuesta(int\*\*, int, int);

int\*\* transpuesta(int\*\*, int, int);

// Funcion principal

int main() {

    size\_t opcion;

    while(opcion != 7) {

        opcion = menu();

        if (opcion > 0 && opcion < 7) {

            size\_t filas, columnas;

            filas = leer\_numero("Ingrese el numero de filas de la matriz ]0, 10]:");

            columnas = leer\_numero("Ingrese el numero de columnas de la matriz ]0, 10]:");

            if  (filas <= 0 || filas > 10 || columnas <= 0 || columnas > 10 ) {

                cout << "Por favor ingrese el numero de filas y columnas en el intervalo establecido" << endl << endl;

                continue;

            }

            cout << "================================================" << endl;

            int\*\* matriz\_a = genere\_arreglo(filas, columnas);

            cout << "Matriz A:" << endl;

            imprime\_arreglo(matriz\_a, filas, columnas);

            if (opcion == 1) {

                int\*\* matriz\_b = genere\_arreglo(filas, columnas);

                cout << "Matriz B:" << endl;

                imprime\_arreglo(matriz\_b, filas, columnas);

                int\*\* resultado = suma(matriz\_a, matriz\_b, filas, columnas);

                cout << "Suma:" << endl;

                imprime\_arreglo(resultado, filas, columnas);

            }

            if (opcion == 2) {

                int\*\* matriz\_b = genere\_arreglo(filas, columnas);

                cout << "Matriz B:" << endl;

                imprime\_arreglo(matriz\_b, filas, columnas);

                int\*\* a\_menos\_b = resta(matriz\_a, matriz\_b, filas, columnas);

                cout << "Resta A-B:" << endl;

                imprime\_arreglo(a\_menos\_b, filas, columnas);

                int\*\* b\_menos\_a = resta(matriz\_b, matriz\_a, filas, columnas);

                cout << "Resta B-A:" << endl;

                imprime\_arreglo(b\_menos\_a, filas, columnas);

            }

            if (opcion == 3) {

                int filas\_b = columnas;

                int columnas\_b = filas;

                int\*\* matriz\_b = genere\_arreglo(filas\_b, columnas\_b);

                cout << "Matriz B:" << endl;

                imprime\_arreglo(matriz\_b, filas\_b, columnas\_b);

                int\*\* a\_por\_b = multiplicacion(matriz\_a, matriz\_b, filas, columnas);

                cout << "Multiplicacion A\*B:" << endl;

                imprime\_arreglo(a\_por\_b, filas, filas);

                int\*\* b\_por\_a = multiplicacion(matriz\_b, matriz\_a, filas\_b, columnas\_b);

                cout << "Multiplicacion B\*A:" << endl;

                imprime\_arreglo(b\_por\_a, filas\_b, filas\_b);

            }

            if (opcion == 4) {

                size\_t escalar;

                escalar = leer\_numero("Ingrese escalar para multiplicar la matriz:");

                int\*\* resultado = producto\_escalar(escalar, matriz\_a, filas, columnas);

                cout << "Producto escalar:" << endl;

                imprime\_arreglo(resultado, filas, columnas);

            }

            if (opcion == 5) {

                int\*\* resultado = opuesta(matriz\_a, filas, columnas);

                cout << "Opuesta:" << endl;

                imprime\_arreglo(resultado, filas, columnas);

            }

            if (opcion == 6) {

                int\*\* resultado = transpuesta(matriz\_a, filas, columnas);

                cout << "Transpuesta:" << endl;

                imprime\_arreglo(resultado, columnas, filas);

            }

            cout << "================================================" << endl;

        }

    }

    return 0;

}

// Definiciones de las funciones

size\_t menu() {

    size\_t opcion;

    cout << "Elija una opcion:" << endl;

    cout << "1. Suma" << endl;

    cout << "2. Resta" << endl;

    cout << "3. Multiplicacion" << endl;

    cout << "4. Producto Escalar" << endl;

    cout << "5. Opuesta" << endl;

    cout << "6. Transpuesta" << endl;

    cout << "7. Salir" << endl;

    cin >> opcion;

    return opcion;

}

size\_t leer\_numero(string mensaje) {

    size\_t numero;

    cout << mensaje << endl;

    cin >> numero;

    cout << endl;

    return numero;

}

int\*\* genere\_arreglo(int filas, int columnas) {

    default\_random\_engine generador;

    generador.seed(chrono::system\_clock::now().time\_since\_epoch().count());

    uniform\_int\_distribution<int> distr(1,10);

    int\*\* matriz;

    matriz = new int\*[filas];

    for (int x=0; x<filas; x++) {

        matriz[x] = new int[columnas];

        for (int y=0; y<columnas; y++) {

            matriz[x][y] = distr(generador);

        }

    }

    return matriz;

}

void imprime\_arreglo(int\*\* matriz, int filas, int columnas) {

    for (int x=0; x<filas; x++) {

        for (int y=0; y<columnas; y++) {

            cout << matriz[x][y] << "\t";

        }

        cout << "\t" << endl;

    }

    cout << "\t" << endl;

}

int\*\* suma(int\*\* matriz\_a, int\*\* matriz\_b, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[filas];

    for (int x=0; x<filas; x++) {

        resultado[x] = new int[columnas];

        for (int y=0; y<columnas; y++) {

            resultado[x][y] = matriz\_a[x][y] + matriz\_b[x][y];

        }

    }

    return resultado;

}

int\*\* resta(int\*\* matriz\_a, int\*\* matriz\_b, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[filas];

    for (int x=0; x<filas; x++) {

        resultado[x] = new int[columnas];

        for (int y=0; y<columnas; y++) {

            resultado[x][y] = matriz\_a[x][y] - matriz\_b[x][y];

        }

    }

    return resultado;

}

int\*\* multiplicacion(int\*\* matriz\_a, int\*\* matriz\_b, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[filas];

    for (int x=0; x<filas; x++) {

        resultado[x] = new int[filas];

        for (int y=0; y<filas; y++) {

            int numero = 0;

            for (int z=0; z<columnas; z++) {

                numero += matriz\_a[x][z] \* matriz\_b[z][y];

            }

            resultado[x][y] = numero;

        }

    }

    return resultado;

}

int\*\* producto\_escalar(int escalar, int\*\* matriz, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[filas];

    for (int x=0; x<filas; x++) {

        resultado[x] = new int[columnas];

        for (int y=0; y<columnas; y++) {

            resultado[x][y] = matriz[x][y] \* escalar;

        }

    }

    return resultado;

}

int\*\* opuesta(int\*\* matriz, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[filas];

    for (int x=0; x<filas; x++) {

        resultado[x] = new int[columnas];

        for (int y=0; y<columnas; y++) {

            resultado[x][y] = matriz[x][y] \* -1;

        }

    }

    return resultado;

}

int\*\* transpuesta(int\*\* matriz, int filas, int columnas) {

    int\*\* resultado;

    resultado = new int\*[columnas];

    for (int x=0; x<columnas; x++) {

        resultado[x] = new int[filas];

        for (int y=0; y<filas; y++) {

            resultado[x][y] = matriz[y][x];

        }

    }

    return resultado;

}