CODIGO PARA LEER UNA MATRIZ CUADRADA.

```
#include <iostream>
#include <iomanip> // para setw()
using namespace std;

// Definición de la estructura
typedef struct Matriz {
   int filas;
   int columnas;
   double *entrada;
} Matriz;

// Función para leer una matriz cuadrada
void leerMatriz(Matriz &m) {
   cout << "Ingrese el tamaño de la matriz cuadrada (n x n): ";
   cin >> m.filas;

   // Forzar que sea cuadrada
   m.columnas = m.filas;

   // Validar que el tamaño sea positivo y razonable
   while (m.filas <= 0 || m.filas > 10) {
      cout << " Tamaño inválido. Debe estar entre 1 y 10.\n";
      cout << "Ingrese nuevamente el tamaño: ";
      cin >> m.filas;
      m.columnas = m.filas;
   }

   // Reservar memoria dinámica
```

```
    m.entrada = new double[m.filas * m.columnas];

    // Lectura de elementos
    cout << "\nIngrese los elementos de la matriz (" << m.filas << "x" << m.columnas << "):\n";
    for (int i = 0; i < m.filas; i++) {
        for (int j = 0; j < m.columnas; j++) {
            cout << "Elemento [" << i << "][" << j << "]: ";
            cin >> m.entrada[i * m.columnas + j];
        }
    }
}

// Función para mostrar la matriz con formato
void mostrarMatriz(const Matriz &m) {
    cout << "\n Matriz ingresada (" << m.filas << "x" << m.columnas << "):\n";
    for (int i = 0; i < m.filas; i++) {
        for (int j = 0; j < m.columnas; j++) {
            cout << setw(8) << fixed << setprecision(2)
                 << m.entrada[i * m.columnas + j];
        }
        cout << endl;
    }
}

// Función para liberar memoria
void liberarMatriz(Matriz &m) {
    delete[] m.entrada;
    m.entrada = nullptr;
}

int main() {
```

```
    Matriz A;

    leerMatriz(A);
    mostrarMatriz(A);
    liberarMatriz(A);

    return 0;
}
```