Universidad Nacional de Moquegua

Universidad del Perú

Facultad de Ingeniería

Escuela Profesional de Ingeniería de Sistemas e Informática.

Docente:

APAZA ALANOCA, HONORIO

Alumno(a):

PANCCA CCALLA, MAYERLING DAYANA

Curso:

ANALISIS Y DISEÑO DE ALGORITMOS

Ciclo:

III

Resolucion de Ejercicios

El objetivo de este trabajo es proporcionar una base sólida para entender cómo manejar y resolver problemas relacionados con matrices, promoviendo buenas prácticas de programación y el desarrollo de habilidades lógicas.

1 Suma de la diagonal principal y secundaria

Dada una matriz cuadrada mmm de números enteros de dimensión ddd, escribe un algoritmo que calcule la suma de:

Los elementos de la diagonal principal (de la esquina superior izquierda a la esquina inferior derecha). Los elementos de la diagonal secundaria (de la esquina superior derecha a la esquina inferior izquierda).

Complejidad Temporal

- **Lectura de la matriz:** Recorre todos los elementos con un ciclo anidado, por lo que la complejidad es $O(n^2)$.
- **Cálculo de sumas de las diagonales:** Solo se recorre la matriz una vez, con complejidad $O(n)$.
- **Mostrar la matriz:** Se recorre nuevamente todos los elementos con complejidad $O(n^2)$.

Complejidad total: $O(n^2)$ (dominado por la lectura y visualización de la matriz).

Complejidad Espacial

- **Matriz:** Se necesita espacio para una matriz de $n \times n$, lo que requiere $O(n^2)$ de espacio.
- **Variables auxiliares:** Solo se usan algunas variables como sp, ss, i, j, lo cual es $O(1)$.

Complejidad total: $O(n^2)$ (dominado por la matriz).

Conclusión

- **Complejidad Temporal:** $O(n^2)$
- **Complejidad Espacial:** $O(n^2)$

2 Rotación de una matriz 90°

Dada una matriz cuadrada mmm de dimensión ddd, escribe un algoritmo que permita rotar la matriz 90° en el sentido de las agujas del reloj.

Complejidad Total

Cada una de estas operaciones (entrada, impresión, rotación e impresión) tiene una complejidad de $O(n^2)$.

Dado que no hay operaciones anidadas dentro de cada bloque (los bucles están secuenciales), la complejidad total sigue siendo $O(n^2)$, ya que cada operación individual tiene la misma complejidad.

Por lo tanto, la complejidad total del programa es $O(n^2)$ debido a los bucles anidados que recorren la matriz de $n \times n$ en varias etapas del proceso.

3 Perímetro de la matriz

Dada una matriz mmm de números enteros y dimensión ddd, escribe un algoritmo que calcule la suma de los elementos en el "perímetro" de la matriz (es decir, los elementos que están en los bordes)

Complejidad Total

- **Entrada de datos:** Tiene una complejidad de $O(d^2)$.
- **Cálculo de la suma del perímetro:** Tiene una complejidad de $O(d)$ debido a las operaciones sobre las filas y columnas del borde de la matriz.

Por lo tanto, la complejidad total del programa es $O(d^2)$, ya que el paso más costoso (rellenar la matriz) tiene una complejidad de $O(d^2)$.

En conclusión, la complejidad total de este programa es $O(d^2)$ debido al proceso de llenar la matriz, que requiere recorrer todos los elementos de la matriz $d \times d$.

4 Transpuesta de una matriz

Dada una matriz m x m de números enteros y dimensión d x d, escribe un algoritmo que calcule la suma de los elementos en el "perímetro" de la matriz (es decir, los elementos que están en los bordes).

Complejidad Total

- **Entrada de datos:** Tiene una complejidad de $O(n \cdot m)$, ya que se recorren todos los elementos de la matriz para leerlos.
- **Transposición de la matriz:** Tiene una complejidad de $O(n \cdot m)$ debido a las operaciones sobre cada elemento de la matriz original.
- **Impresión de la matriz original y de la transpuesta:** También tiene una complejidad de $O(n \cdot m)$ porque se recorren todos los elementos de ambas matrices para imprimirlos.

Por lo tanto, la complejidad total del programa es $O(n \cdot m)$, ya que las operaciones dominantes (lectura, transposición e impresión) tienen una complejidad de $O(n \cdot m)$.

En conclusión, la complejidad total de este programa es $O(n \cdot m)$ debido al proceso de recorrer todas las posiciones de la matriz para realizar las operaciones.

5 Verificar simetría de una matriz

Escribe un algoritmo que determine si una matriz cuadrada m y m es simétrica (es decir, si m[i][j]=m[j][i] para todos i,j).

Complejidad Total

- **Entrada de datos:** Tiene una complejidad de $O(m^2)$, ya que se recorren todos los elementos de la matriz para leerlos.
- **Verificación de simetría:** El algoritmo recorre solo la mitad superior de la matriz, comparando los elementos $mat[i][j]$ con $mat[j][i]$. Esto tiene una complejidad de $O\left(\frac{m^2}{2}\right)$, lo cual es equivalente a $O(m^2)$.

Por lo tanto, la complejidad total del programa es $O(m^2)$, ya que tanto la lectura de la matriz como la verificación de simetría requieren recorrer todos los elementos de la matriz en el peor de los casos.

En conclusión, la complejidad total de este programa es $O(m^2)$ debido a que se recorre toda la matriz para verificar la simetría y para llenar los datos.

6 Recorrido Espiral

Escribe un algoritmo que recorra una matriz cuadrada o rectangular M en forma de espiral, comenzando desde la esquina superior izquierda y moviéndose en sentido horario. El algoritmo debe devolver los elementos en el orden en que son visitados.

Complejidad Total

- **Entrada de datos:** Tiene una complejidad de $O(n \cdot m)$, ya que se recorren todos los elementos de la matriz para leerlos.
- **Impresión en espiral:** El algoritmo recorre la matriz de manera ordenada en espiral, visitando cada elemento exactamente una vez. Por lo tanto, esta operación también tiene una complejidad de $O(n \cdot m)$.

Por lo tanto, la complejidad total del programa es $O(n \cdot m)$, ya que tanto la lectura de la matriz como la impresión en espiral requieren recorrer todos los elementos de la matriz.

La complejidad total de este programa es $O(n \cdot m)$ debido al proceso de recorrer toda la matriz para realizar las operaciones de lectura e impresión.