Algoritmos y Estructuras de Datos I
Primer Cuatrimestre 2018

Guía Práctica 9
Algoritmos sobre matrices

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ejercicio 1. ★ Dada una matriz cuadrada triangular (es decir, todos los elementos por debajo de la diagonal son cero) de números enteros, escribir un programa que calcule su determinante. Recordar que el determinante de una matriz triangular es el producto de los elementos de su diagonal.

Ejercicio 2. Dadas dos matrices A y B, implementar un programa que retorne AB. ¿Qué precondición tiene este problema?. Escribir luego un programa que calcule A^n siendo n un número natural.

Ejercicio 3. ★ Dada una matriz cuadrada A, escribir un programa que retorne $B = AA^t$ siendo A^t la matriz transpuesta de A.

Ejercicio 4. *Búsqueda en Matrices*: Dada una matriz cuadrada de enteros y un entero x,

1. Especificar el problema de retornar la fila y columna que contiene al elemento x (o $\langle -1, -1 \rangle$ si no existe ese elemento)
2. Escribir un programa que implemente la especificación del problema.
3. ¿Qué cantidad de iteraciones realiza el programa en **peor caso**?

Ejercicio 5. Dada una matriz de números enteros con las filas ordenadas de menor a mayor y las columnas ordenadas de menor a mayor, en la que se sabe que ninguna fila ni columna tiene elementos repetidos:

1. Escribir un programa que cuente la cantidad de veces que aparece un número en la matriz leyendo a lo sumo una única vez toda la matriz.
2. Adaptar el programa anterior si se asume que puede haber repetidos en las columnas (y no hay repetidos en las filas).
3. Adaptar el programa si tanto las filas como las columnas pueden tener elementos repetidos.

Ejercicio 6. Se tiene una matriz de ceros y unos de n filas y m columnas con n impar. Se sabe que hay una fila que no está repetida, y el resto de las filas están repetidas exactamente una vez (es decir, hay dos filas iguales para cada una de las filas, excepto para una que no tiene otra fila igual). Contamos además con una fila extra, que empieza toda en ceros. Escribir un programa que guarde en la fila extra la fila que no aparece repetida en la matriz.

Ejercicio 7. Escribir un programa que resuelva el problema del ejercicio anterior si le agregamos la restricción de que no tenemos más variables disponibles además de la fila extra y que podemos leer cada elemento de la matriz a lo sumo una vez.

Ejercicio 8. ★ Dada una matriz cuadrada de enteros que representa una imagen de $2n \times 2n$:

1. Escribir un programa que compute el promedio entre las 4 celdas adyacentes, y escriba el valor en una nueva matriz de $n \times n$.
2. ¿Qué cantidad de iteraciones realiza el programa en **peor caso**?

Ejercicio 9. *Saddleback Search*: Dada una matriz cuadrada de dimensión $n > 0$ tal que sus filas y columnas tienen sus elementos ordenados de manera ascendente.

1. Especificar el problema de retornar la fila y columna que contiene al elemento buscado (o $\langle -1, -1 \rangle$ si no existe ese elemento)
2. Escribir un programa que implemente la especificación del problema y que efectúe en **peor caso** $2 \times n$ iteraciones.
3. ¿Es posible aplicar búsqueda binaria para resolver este problema? ¿Cómo lo haría?

Ejercicio 10. ★ *Hill Climbing:* Dada una matriz cuadrada de enteros sin repetidos que representa la elevación de un terreno, un máximo local es una celda tal que ninguna de las celdas adyacentes tiene un valor mayor.

1. Escribir un programa que busque algún máximo local a partir de la celda $\langle 0, 0 \rangle$ de la matriz.
2. Adaptar el programa para que encuentre alguún máximo local únicamente escalando (i.e., sólo subiendo).

Ejercicio 11. Dada una matriz cuadrada de enteros sin repetidos que representa la elevación de un terreno,

1. Especificar e implementar un programa que encuentre, todos los caminos desde un punto dado $\langle i, j \rangle$ hasta un valle (mínimo local).
2. Implementar versiones iterativas y recursivas para el problema anterior.

Ejercicio 12. ★ *Wall Follower:* Dada una matriz cuadrada de enteros que representa un laberinto tal que:

- el valor de celda 0 indica que en esa celda no hay pared,
- el 1 indica que esa celda está ocupada por una pared, y
- el valor -1 indica que esa celda es la salida del laberinto.

Asumiendo que todos los muros del laberinto están conectados entre sí, escribir el programa que a partir de la entrada del laberinto (la celda $\langle 0, 0 \rangle$) busca el camino para salir del laberinto, retornando la ubicación de la salida y la cantidad de celdas que debió recorrer. **Importante:** El algoritmo no puede *atravesar* los muros.