**PRACTICA 7 |ARREGLOS MULTIDIMENSIONALES**

# Introducción

Estamos comenzando a dejar de lado, en esta práctica, los arreglos de una sola dimensión, para subir un poco más de nivel y complejidad para ahora manejar matrices o arreglos multidimensionales. Tomaré en base el análisis y conocimiento adquirido en las prácticas de arreglos unidimensionales, para abordar el problema de este tipo de estructura de datos con la técnica sugerida en clase: visualizar cada dimensión de la matriz como si fuese un arreglo más simple.

De esta manera, espero poder resolver de una forma eficiente y efectiva cada procedimiento o método indicado en las instrucciones de la práctica.

# Desarrollo

Para la correcta implementación del código de cada método, pregunté junto con algunos compañeros algunas dudas que había sobre cómo hacer algunos métodos que en su descripción en el documento se tornaban un poco ambiguos.

Sobre la marcha, al momento de codificar cada método presenté pocas complicaciones necesitando pocas veces recurrir al análisis gráfico para poder razonar como iba a programar una función o para descubrir por qué no funcionaba la misma.

En este ejercicio se trabajó solamente sobre una nueva clase llamada **Matriz**, incluyendo comportamientos un poco parecidos a los ya hechos anteriormente en la clase **Arreglo,** pero adaptados a las dos dimensiones que maneja este nuevo tipo de datos abstracto.

A continuación, se muestran los archivos creados o modificados, junto con el nombre de los métodos incluidos:

## Matriz.java (paquete edlineal)

* public void aplicarTraspuesta().
* public Matriz copiar().
* public boolean esIgual(Matriz matriz2).
* private boolean validarDim(Matriz matriz2).
* private void copiarMatriz(Matriz matriz).
* public boolean vectorCol(int numRenglones, Object valor).
* public boolean vectorReng(int numColumnas, Object valor).
* public boolean definirMatriz(Matriz2D matriz2).
* public boolean agregarRenglon(Arreglo arreglo).
* public boolean agregarColumna(Arreglo arreglo).
* public boolean  agregarMatrizColumna(Matriz2D matriz2).
* public boolean  agregarMatrizRenglon(Matriz2D matriz2).
* public Matriz3D aMatriz3D(Arreglo matrices).

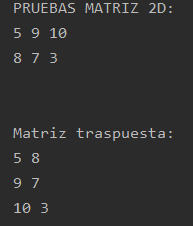
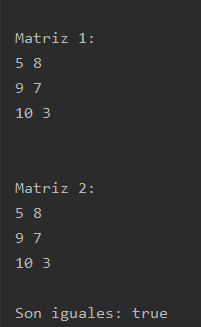
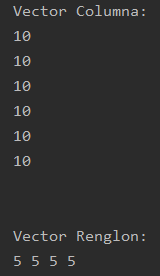
## PruebaMatriz2D.java (paquete interfaz).

La clase anterior es donde se encuentran las pruebas de esta práctica.

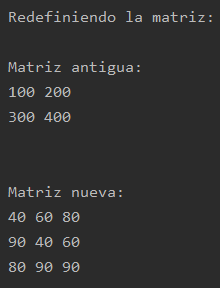
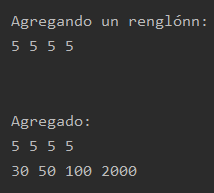
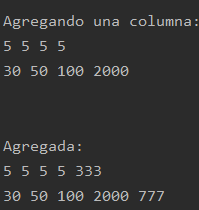
**Para mayor información sobre los métodos y la clase creada vaya a la ruta: /docs/index.html dentro del proyecto.**

# Capturas de pantalla del programa funcionando

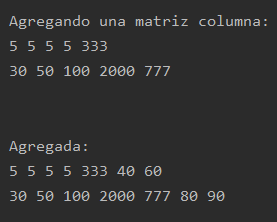
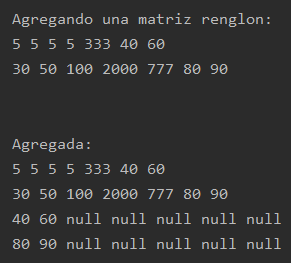
**Matriz traspuesta. Igualdad de matrices. Vectores renglón y columna**

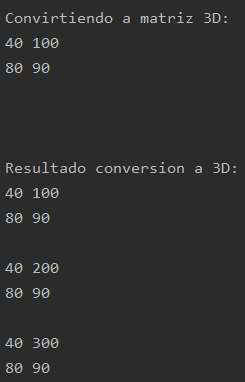
**Redefinir matriz. Agregar renglón. Agregar columna.**

**** **** 

**Agregar matriz columna. Agregar matriz renglón.**

**** ****

**Convertir a Matriz3D.**

****

# Conclusiones

Puedo decir satisfactoriamente que para esta práctica sentí mejor dominio de los arreglos unidimensionales y multidimensionales según cada caso. En base a las prácticas, lo aprendido en clase y el análisis propio puedo con gusto comentar que siento que mis habilidades sobre el manejo de éstas estructuras de datos están mejorando. Espero que realmente esto se siga manteniendo e incluso perfeccionando más.

Gracias a lo estudiado en clase, también logré manejar una matriz de tres dimensiones sin demasiadas complicaciones, partiendo de haber comprendido como interpretarla en mi mente y sobre el papel.