PROGRAMACIÓN

EXAMEN MARZO 2025

SE VALORARÁ POSITIVAMENTE:
1. Los algoritmos y las estructuras de datos utilizados para la solución así como su justificación.
SE PENALIZARÁ:
2. Los errores en los requisitos y los algoritmos, teniendo en cuenta que un error leve descuenta 1 punto y un error grave 2 puntos. • Se considera un error leve aquel, que pese a su existencia, permite una solución al problema. • Un error grave es aquel que imposibilita una correcta y exacta solución al problema. • La no utilización del método, de cada ejercicio, en un **programa principal será un leve**.
Los ejercicios valen 2,5 puntos y se debe respetar el tiempo máximo establecido y las normas de permanencia. Recomendable utilizar, como método de resolución, el visto en clase pero se corregirá sobre Java.

Más de un 7,5 suma un punto al siguiente examen.

Nombre y apellidos: **07/03/2025** **Firma:**

1. Implementad en Java un método de ordenación para un array bidimensional (o matriz) de números enteros. Este método debe recibir una array bidimensional como parámetro y permitir la ordenación tanto ascendente como descendente. El tipo de ordenación (ascendente o descendente) se especificará mediante un parámetro booleano a tu criterio.

Ejemplos de uso:

• Dada esta matriz:

[[9, 8, 5],
[4, 2, 1]]

Al ordenarla descendentemente quedaría así:

[[1, 2, 4],
[5, 8, 9]]

• Dada esta matriz:

[[8, 1, 5]
[4, 2, 10]]

Al ordenarla ascendentemente quedaría así:

[[10, 8, 5]
[4, 2, 1]]

2. Implementa un método llamado verificarDiagonales que reciba una matriz (o array bidimensional) de números enteros y verifique si los elementos en la primera columna y la última fila son idénticos. Esto significa que todos los elementos de la primera columna deben ser iguales entre sí, y todos los elementos de la última fila también deben ser iguales entre sí.

Ejemplos de uso:

verificarDiagonales ([[1, 2, 3], [1, 2, 3], [1, 2, 3]]) → true

verificarDiagonales ([[1, 2, 3], [4, 5, 6], [7, 8, 9]] → false

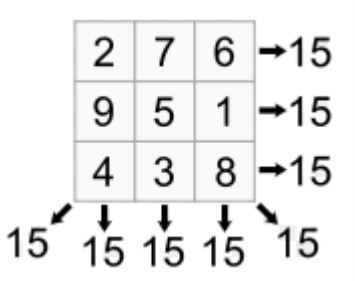
3. Escribe un método llamado **especiesDuplicadas** que recibe un array de cadenas, donde cada cadena representa el nombre de una especie de planta. El método debe devolver el número de especies únicas que aparecen más de una vez en el array. Los nombres de las especies no son key sensitive, es decir, no distinguen entre mayúsculas y minúsculas, por ejemplo, "rosa" y "Rosa" deben considerarse la misma especie.

Ejemplos de uso:

- especiesDuplicadas(["Rosa", "Lirio", "rosa", "Tulipan", "LIRIO", "Margarita"]) → 2
- especiesDuplicadas(["Orquidea", "Girasol", "Clavel", "Azucena"]) → 0
- especiesDuplicadas(["Bambu", "bambu", "BAMBU", "Bambu"]) → 1

4. Implementad un método en Java que solicite los elementos de una matriz de tamaño N x N. El método debe devolver si la matriz introducida corresponde a una matriz mágica, que es aquella donde la suma de los elementos de cualquier fila o de cualquier columna valen lo mismo.

Veamos un **ejemplo gráfico** de matriz mágica de tamaño 3 x 3:



Ejemplos de uso:

- isMagicSquare({ 6 }) → true
- isMagicSquare({ { 1, 2 },{ 3, 4 }) → false
- isMagicSquare({ 8, 1, 6 }, { 3, 5, 7 }, { 4, 9, 2 }) → true
- isMagicSquare({ 1, 15, 14, 4 }, { 12, 6, 7, 9 }, { 8, 10, 11, 5 }, { 13, 3, 2, 16 }) → 2
- isMagicSquare() → false