Práctica opcional

UNIVERSIDAD DE CANTABRIA

***Objetivos*: Practicar con tablas de tamaño variable**

***Descripción*: Se desea hacer un sistema para almacenar medidas del movimiento de un vehículo realizadas usando un giróscopo tridimensional**

Cada medida es un objeto de la clase `VelAngular`, que contiene los siguientes atributos:

- **`omegaX`, `omegaY`, `omegaZ`: velocidades angulares en los ejes X, Y, y Z, en radianes por segundo**
- **`tiempo`: instante en que se ha hecho la medida, en segundos desde el inicio del experimento**

La clase tiene un constructor al que se pasan los valores iniciales de los atributos, y un método observador para cada atributo

Práctica opcional (cont.)

UNIVERSIDAD DE CANTABRIA

Se desea escribir la clase `Medidas` que guarde un número variable de medidas. La clase debe tener como atributos:

- **`medida`: un array de objetos de la clase `VelAngular`**
- **`num`: un entero que dice cuántas medidas hay almacenadas en el array `medida`**
  - **las medidas se ponen en las primeras casillas del array (de la `0` a la `num-1`) y las demás se dejan sin usar**

Práctica opcional (cont.)

UNIVERSIDAD DE CANTABRIA

Los métodos de la clase son los siguientes:

- ***constructor***: crea al array `medida` del tamaño que se le pasa como parámetro al constructor; además, pone `num` a 0
- `anade`: añade a la lista una medida que se pasa como parámetro, si cabe (la mete en la casilla `num` e incrementa `num`); si no cabe, pone en pantalla un mensaje de error
- `mediaOmegaX`: retorna la media de la `omegaX`, a lo largo del tiempo
- `mediaOmegaY`: retorna la media de la `omegaY`, a lo largo del tiempo
- `mediaOmegaZ`: retorna la media de la `omegaZ`, a lo largo del tiempo

Práctica opcional (cont.)

UNIVERSIDAD DE CANTABRIA

Métodos de la clase (cont.):

- `maximo`: retorna el máximo del módulo de la velocidad angular; retorna menos infinito (valor `Double.NEGATIVE_INFINITY`) si no hay ninguna
- `minimo`: retorna el mínimo del módulo de la velocidad angular; retorna infinito (valor `Double.POSITIVE_INFINITY`) si no hay ninguna
- `tiempoEsMonotono`: retorna un booleano que será `true` si los tiempos de cada medida de la lista son iguales o mayores que los de la medida anterior, y `false` en caso contrario

Utilizar siempre que sea apropiado los esquemas de recorrido o búsqueda en tablas vistos en clase

Práctica opcional (cont.)

UNIVERSIDAD DE CANTABRIA

Para calcular la media en el tiempo:

- **si no hay ningún valor se retorna `Double.NaN`**
- **si sólo hay un valor, se retorna ese valor**
- **si hay más de un valor se usa esta expresión (siendo *n* el número de elementos de la lista, *val[i]* el valor almacenado en la posición *i*, y *t[i]* el instante de ese valor):**

$$media = \frac{\sum_{i=0}^{n-2} \left(\frac{val[i] + val[i+1]}{2} \right) \cdot (t[i+1] - t[i])}{t[n-1] - t[0]}$$

Práctica opcional (cont.)

UNIVERSIDAD DE CANTABRIA

Escribir también un programa principal de prueba de la clase, que utilice todos sus métodos, en diferentes situaciones (probar casos normales y de error)

- **leer los datos de las medidas de una caja de texto**

***Entregar*:**

- **los diagramas de las clases**
- **la especificación y diseño de los métodos `anade`, `mediaOmegaX`, `maximo` y `tiempoEsMonotono`**
- **el código fuente de las clases y del programa de prueba**