



LABORATORIO N^{RO} 8
CLASE TEXTO

Dada la siguiente definición de la clase *Texto*, parcialmente implementada en Java:

Texto	
<< atributos de instancia >>	
tex : char[][]	
<< constructor >>	
Texto (s: String)	←
<< consultas >>	
caracterEnPosicion (i, j: entero): caracter	←
toString(): String	
empiezaDosVocales(): entero	
palabraMasLarga(): String	

El String recibido contiene sólo letras y espacios. Comienza y termina con una letra. Para simplificar el problema, no considere caracteres especiales (á,é,í,ó,ú,ñ,Ñ,etc).

Requiere que i y j sean posiciones válidas. El cliente asume las posiciones en el rango [0..49][0..39]

Un objeto de tipo *Texto* permite modelar una secuencia de líneas de texto de las siguientes características:

- Cada línea tiene un ancho fijo de 50 caracteres, y el texto admite hasta 40 líneas.
- Cada línea contiene palabras separadas por **un** espacio y signos: **coma, punto y coma y punto final**. Sólo hay un punto final en todo el texto.
- El constructor se encarga de distribuir las palabras en la fila y columna que corresponda, garantizando que las palabras no se corten de una línea a otra.
- Los espacios sobrantes en una fila, no ocupados con los caracteres permitidos, se completan con blancos.
- Si la palabra pasada al constructor es demasiado grande para ser contenida en la matriz, se almacena hasta donde alcance y se agrega punto final.
- Un String que inicializa el texto es válido si supera los mismos filtros de una oración válida.

Ejercicio 1: implementar el constructor de clase *Texto* según la especificación anterior.

Ejercicio 2: desarrollar un método *empiezaDosVocales():entero* que cuente cuántas palabras de la oración comienzan con dos vocales seguidas.

Ejercicio 3: desarrollar *palabraMasLarga():String*. Si hay dos o más palabras con la máxima longitud, se devuelve la primera.

Complete la definición de la clase *Tester*, agregando casos de prueba significativos, para probar los métodos implementados.

Tip:

- Antes de implementar el constructor, realizar en papel un ejemplo que ayude al diseño del algoritmo. Plantear todos los posibles casos que puedan ocurrir (que entre todo sin inconvenientes, que una palabra no entre en una fila, que no alcance la matriz para almacenar todo el string, etc.).