

Nombre \_\_\_\_\_

*Se respetará escrupulosamente las reglas de estilo del lenguaje Java (además de los comentarios, reglas de indentación, nombres descriptivos para variables y métodos, capitalización de nombres, blancos de separación entre los operadores, .....).*

## **Ejercicio 2 UT5 (para entregar)**

---

### **Objetivos**

#### *Saber:*

- *declarar y crear un array unidimensional de objetos String*
- *manejar array como atributos*
- *borrar varios elemento de un array*
- *declarar y manejar arrays de dos dimensiones*
- *utilizar métodos de la clase String*
- *utilizar objetos StringBuilder como apoyo a ciertas operaciones con String*
- *usar argumentos del main()*

#### *Repasar:*

- *sentencias de control : if / while / for*
- *llamadas a métodos*
- *toString()*
- *proceso de arrays parcialmente completos*

Abre el proyecto **Entrega 02 UT5 AL**.

Pon tu nombre en la clase **Texto** que has de completar.

Esta clase incluye el método **main()** con código para probar todos los demás métodos y un método **leerDeFichero()** ya completo para poder leer las frases que forman el texto desde un fichero.

Un objeto de la clase **Texto** guarda un texto que está formado por varias frases. Cada frase consta de varias palabras (asumimos que toda frase tiene como mínimo una palabra) separadas por exactamente un espacio.

Los atributos de la clase definen:

- a) el array que permite guardar todas las frases del texto
- b) el n.º real de frases almacenadas en un momento dado
- c) un par de constantes

Completa en la clase:

- el constructor que crea e inicializa adecuadamente los atributos. El array *frases* se crea con el tamaño máximo indicado por el argumento
- **public void addFrase(String frase)** – añade una nueva frase al texto siempre al final y solo si el texto no está completo. Antes de guardar la frase se eliminarán espacios a izquierda y derecha de frase (consulta la API para ver qué método de la clase **String** has de usar)
- **public boolean textoCompleto()** – indica si el texto está o no completo
- **public int getNumeroFrases()** – devuelve el nº de frases del texto

- **public boolean posicionCorrecta(int p)** – devuelve *true* si *p* representa una posición correcta del array, es decir, *p* está en el rango de posiciones del array en los que hay algún valor
- **public int palabrasDeFrase(int p)** – dada una posición correcta devuelve el nº de palabras de la frase que está en dicha posición. Si la posición es incorrecta devuelve -1. Usa únicamente `indexOf()` y `substring()` de la clase `String`
- **public int borrarFrases(int n)** – borra del texto las frases que tienen menos de *n* palabras. Devuelve la cantidad de frases borradas
- **public int[] frecuenciaCaracteres()** - calcula y devuelve la frecuencia de aparición de cada una de las letras del alfabeto en el texto.
- **char[][] toArray2D(int p)** - dada una posición correcta crea y devuelve un *ragged array* de caracteres con tantas filas como nº de palabras tenga la frase de esa posición. Cada fila se asocia a una palabra y tendrá tantas columnas como caracteres tenga la palabra. Si la posición es incorrecta se devuelve *null*. Se permite utilizar `split()`.

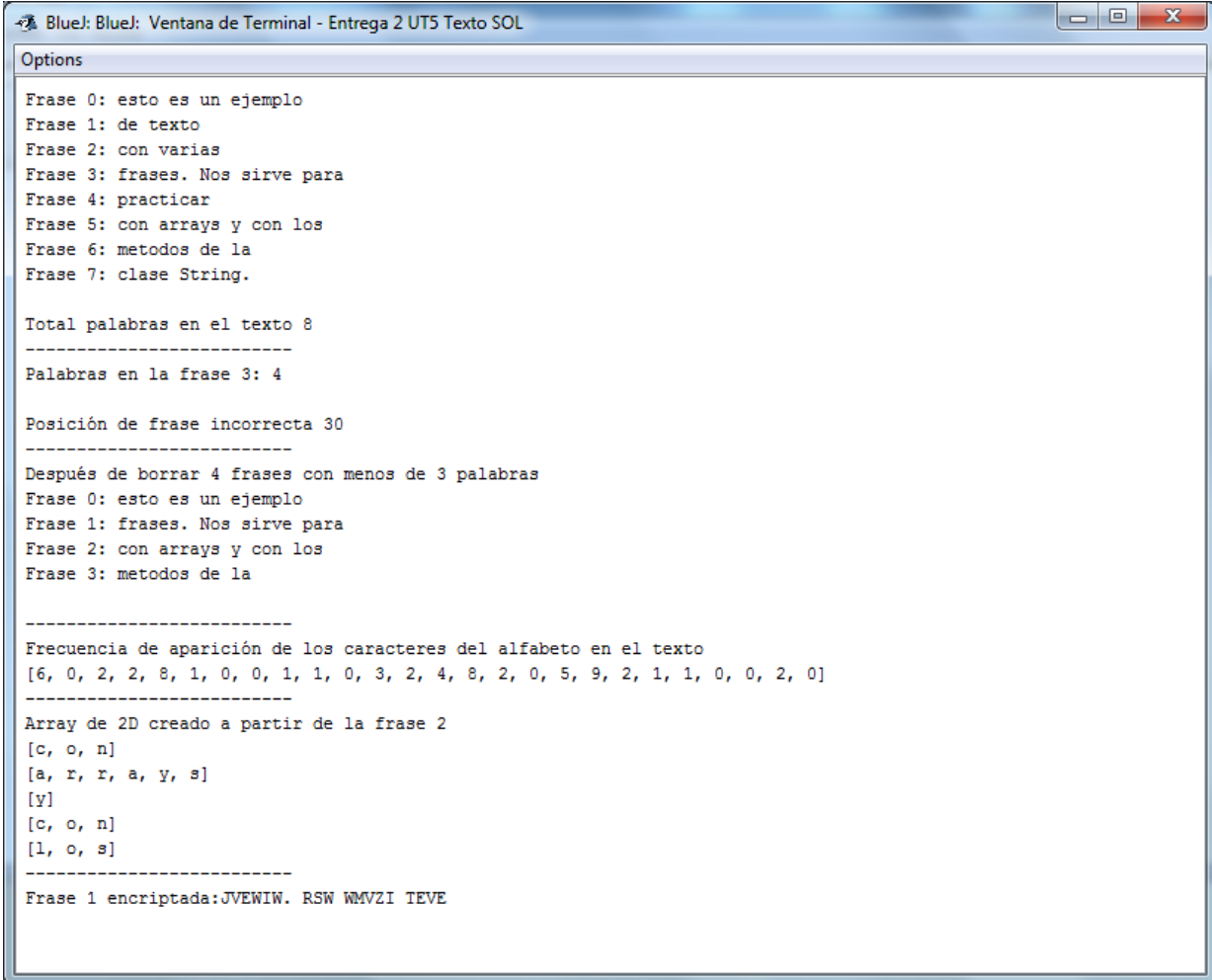
Si en el array *frases* hay: *"esto es un ejemplo"* *"de texto"* *"con varias"*  
*"frases. Nos sirve para"* *"practicar"*  
*"con arrays y con los"*

y la posición es 3 se devuelve el array de 2D de 4 filas y en cada fila queda cada una de las palabras de la frase

```
frases.  
nos  
sirve  
para
```

- **public String encriptar(int p)** – dada una posición (que asumimos correcta) devuelve la frase de esa posición encriptada. Para encriptar se sustituye cada carácter letra de la frase por el que está 4 veces más a la derecha en el alfabeto definido como constante. El resto de caracteres se quedan igual. La frase encriptada se devuelve en mayúsculas. Se puede hacer usando como objeto de apoyo `StringBuilder`
- **public String toString()** – devuelve una representación textual del array. Utiliza como apoyo un objeto `StringBuilder`. (ver resultados en la figura)

## Ejemplo de ejecución



```
Options
Frase 0: esto es un ejemplo
Frase 1: de texto
Frase 2: con varias
Frase 3: frases. Nos sirve para
Frase 4: practicar
Frase 5: con arrays y con los
Frase 6: metodos de la
Frase 7: clase String.

Total palabras en el texto 8
-----
Palabras en la frase 3: 4

Posición de frase incorrecta 30
-----
Después de borrar 4 frases con menos de 3 palabras
Frase 0: esto es un ejemplo
Frase 1: frases. Nos sirve para
Frase 2: con arrays y con los
Frase 3: metodos de la

-----
Frecuencia de aparición de los caracteres del alfabeto en el texto
[6, 0, 2, 2, 8, 1, 0, 0, 1, 1, 0, 3, 2, 4, 8, 2, 0, 5, 9, 2, 1, 1, 0, 0, 2, 0]
-----
Array de 2D creado a partir de la frase 2
[c, o, n]
[a, r, r, a, y, s]
[y]
[c, o, n]
[l, o, s]
-----
Frase 1 encriptada:JVEWIW. RSW WMVZI TEVE
```