APELLIDOS, Nombre	GRUPO	
	MÁQUINA	

Proyecto prCuentaPalabrasSimple

Se va a crear una aplicación para contar el número de veces que aparece cada palabra en un texto dado. Para ello se crearán las clases `PalabraEnTexto`, `ContadorPalabras`, `ContadorPalabrasSig` y `Main`.

A. (***2.5 ptos.***) La clase `PalabraEnTexto` mantiene información de una palabra (`String`) así como del número de veces que aparece en un determinado texto (`int`).
   1. (*0.5 ptos.)* La clase tendrá un constructor en el que se proporciona la palabra. Al construir el objeto, el número de veces que aparece la palabra se considera 1.
   2. (*1.5 ptos.)* Dos objetos de la clase `PalabraEnTexto` son iguales si coinciden las palabras que contiene, independientemente de que estén en mayúsculas o minúsculas. El número de apariciones no se tiene en cuenta.
   3. (*0.25 ptos.)* La representación de un objeto `PalabraEnTexto` debe mostrar la palabra que contiene y el número de veces que aparece.
   4. (*0.25 ptos.)* El método `void incrementa()` incrementa en uno el número de veces que aparece la palabra.

B. (***5.5 ptos.***) Crear la clase `ContadorPalabras` que almacena en un array las palabras que aparecen en un texto (array de `PalabraEnTexto`).

   Si en el transcurso de las operaciones que se hagan con objetos de esta clase, el array llega a tener un tamaño insuficiente, deberá crecer de manera que siempre quepan las palabras que se le proporcionen.

   5. (*0.5 ptos.)* La clase dispondrá de dos constructores; el primero de ellos, sin argumentos, que crea el array con un tamaño de 10; el segundo, con un argumento entero, que indicará el tamaño inicial del array.
   6. (*1 pto.*) El método privado `int esta(String pal)` que devuelve la posición en la que se encuentra la palabra que corresponde a `pal` en el array o -1 si no está.
   7. (*1 pto.*) El método `void incluye(String pal)` deberá incrementar el número de apariciones de la palabra que corresponda a la cadena `pal` en el contador de palabras si es que ya existía, o incluir una palabra nueva en caso contrario.
   8. (*0.5 ptos.*) El método `void incluyeTodas(String linea, String del)` permite extraer de `linea` las palabras usando los delimitadores incluidos en `del`. Cada una de las palabras obtenidas se irán acumulando en el contador.
   9. (*0.5 ptos.*) El método `void incluyeTodas(String [] texto, String del)` incluye todas las palabras que se encuentra en el array texto. Cada elemento del array será una línea de texto y en cada línea, las palabras se deben separar usando los delimitadores incluidos en `del`.
   10. (*1 pto.*) Crear el método `PalabraEnTexto encuentra(String pal)` que, dada una cadena de caracteres `pal` que representa una palabra, encuentra la instancia de `PalabraEnTexto` en el array que coincide con ella y la devuelve. Si la palabra no se encuentra en el texto deberá lanzar la excepción `NoSuchElementException`.

11. (*1 pto.*). La clase dispondrá de una representación de los objetos como la que se muestra en el ejemplo final. Usar `StringBuilder` para crear la representación, y obsérvese que, tras la última palabra, no hay coma).

C. (***2 ptos.***) La clase `ContadorPalabrasSig` representa objetos contadores de palabras que, en los procedimientos de inclusión, no contempla las palabras consideradas no significativas. Para ello, la clase deberá incluir un array de `String` que almacene estas palabras no significativas.

12. (*0.5 ptos.*) Definir constructores apropiados, que incluyan un argumento para proporcionar el array de palabras no significativas. Supóngase que las palabras de este array están todas en mayúsculas.
13. (*1.5 ptos.*) Conseguir que las instancias de la clase `ContadorPalabrasSig` se comporten como las de `ContadorPalabras`, a excepción de que los procedimientos de inclusión de palabras no realicen ninguna acción cuando éstas no sean significativas.

A continuación se presenta la salida correspondiente a la clase `Main` de la página siguiente:

```
Por defecto...
[GUERRA: 5, TEN¬íA: 2, UNA: 2, JARRA: 3, Y: 1, PARRA: 7, PERRA: 6,
PERO: 1, LA: 10, DE: 8, ROMPI¬ó: 1, PEG¬ó: 1, CON: 3, PORRA: 3, A:
3, OIGA: 1, USTED: 1, BUEN: 1, HOMBRE: 1, POR: 1, QU¬é: 1, HA: 1,
PEGADO: 2, PORQUE: 1, SI: 1, NO: 2, HUBIERA: 2, ROTO: 1]

[GUERRA: 5, TEN¬íA: 2, JARRA: 3, PARRA: 7, PERRA: 6, PERO: 1,
ROMPI¬ó: 1, PEG¬ó: 1, PORRA: 3, OIGA: 1, USTED: 1, BUEN: 1, HOMBRE:
1, POR: 1, QU¬é: 1, HA: 1, PEGADO: 2, PORQUE: 1, HUBIERA: 2, ROTO:
1]

PARRA: 7
No existe la palabra Gorra
```

```java
import java.util.NoSuchElementException;

public class Main {

   public static void main(String [] args) {
      String [] datos = {
        "Guerra tenía una jarra y Parra tenía una perra, "
        "pero la perra de Parra rompió la jarra de Guerra.",
        "Guerra pegó con la porra a la perra de Parra. ",
        "¡Oiga usted buen hombre de Parra! ",
        "Por qué ha pegado con la porra a la perra de Parra.",
        "Porque si la perra de Parra no hubiera roto la jarra de Guerra,",
        "Guerra no hubiera pegado con la porra a la perra de Parra."};
      String delimitadores = " .,:;-¡!¿?(){}[]";

      String [] noSig = {"CON","LA","A","DE","NO","SI","Y","UNA"};
      ContadorPalabras contador = null;
      // Si no se incluye un argumento numérico, se crea por defecto.
      try {
          int tam = Integer.parseInt(args[0]);
          System.out.println("Con argumento " + tam);
          contador = new ContadorPalabras(n);
          contadorSig = new ContadorPalabrasSig(n, noSig);
      } catch (RuntimeException e) {
          System.out.println("Por defecto...");
          contador = new ContadorPalabras();
          contadorSig = new ContadorPalabrasSig(noSig);
      }
      // Incluimos todas las palabras que hay en datos
      // teniendo en cuenta los delimitadores
      contador.incluyeTodas(datos, delimitadores);
      contadorSig.incluyeTodas(datos, delimitadores);

      System.out.println(contador + "\n");
      System.out.println(contadorSig + "\n");

      try {
          System.out.println(contador.encuentra("parra"));
          System.out.println(contador.encuentra("Gorra"));
      } catch (NoSuchElementException e) {
          System.err.println(e.getMessage());
      }
    }
  }
```

PROGRAMACIÓN ORIENTADA A OBJETOS

NOTAS PARA LA REALIZACIÓN DEL EJERCICIO

(Léase detenidamente antes de comenzar el ejercicio)

El ejercicio se almacenará en el directorio **C:\POO**. En caso de que no exista deberá crearse, y si ya existiese, deberá borrarse todo su contenido antes de comenzar.

CADA FICHERO DEBERÁ IDENTIFICARSE CON EL NOMBRE DEL ALUMNO, TITULACIÓN, GRUPO Y NÚMERO DE EQUIPO.

La evaluación tendrá en cuenta la claridad de los algoritmos, del código y la correcta elección de las estructuras de datos, así como los criterios de diseño que favorezcan la reutilización.

ESTÁ PERMITIDO:

- Consultar el API que se encuentra en el disco de red.

NO ESTÁ PERMITIDO:

- Utilizar otra documentación electrónica o impresa.
- Intercambiar documentación con otros compañeros.
- Utilizar soportes de almacenamiento.

IMPORTANTE: APAGAD LOS DISPOSITIVOS ELECTRÓNICOS DE COMUNICACIÓN

Una vez terminado el ejercicio subir un fichero comprimido (.jar, .zip o .rar) sólo con los **fuentes** que hayáis realizado a la tarea creada en el campus virtual para ello.