PROGRAMACIÓN JAVA **Nivel:** **Intermedio**
Puntos: **de 2.50 a 5**

Se pide o deseño da clase ***Tokenizer*** no *package* ***utils***.

Un obxecto ***Tokenizer*** é un obxecto que é capaz de dividir un ***String*** en un conxunto de Strings (*Tokens*) tomando unha cadea de texto como separador. Un ***Tokenizer*** ten as seguintes características:

- Si se emprega o construtor por defecto (sen parámetros) o separador será a cadea "**,**".
- Dispón dun construtor que recibe como argumento a cadea que se desexa utilizar como separador.
- Os *Token* non teñen espacios en branco nin comiñas o seu principio nin o final (ver ***trim*** na clase ***String***)
- A clase ***Tokenizer*** é *Iterable*, o iterador debe permitir percorrer os *Token* nos que se divide o ***String***.
- Ademais un ***Tokenizer*** terá os seguintes métodos públicos:
  - **void setString(String data);** : Permite indicar o ***String*** que se vai "tokenizar". O ***String*** a dividir non debe ter saltos de liña polo que inicialmente se debe "limpar" substituíndo os saltos de liña (\n) por un caracter nulo ("") (ver método ***replace()*** de String). Este método chamará o método *privado* ***void scanData()*** que se encargará de dividir o String e de gardar os Tokens. ***(ver método split() de String para unha solución simple)***
  - **void setSeparator(String separator);** : Cambia o divisor de tokens polo separator recibido como argumento.
  - **String getString();** : Retorna o String orixinal "Tokenizado". Retornará null si non temos ningún String dividido.
  - **public String[] getTokens();** : Retorna un String[] cos Token que compoñen o String.
  - **public String getToken(int idx);** : Retornará o Token número idx do String. Si non existe, retornará null
  - **public int numTokens();** : Retornará o número de Tokens nos que se dividiu o String

Este exercicio está clasificado segundo a súa capacidade de dividir o ***String*** en *tokens*, que **depende da implantación do método privado *void scanData();***

- *A "solución básica" divide correctamente* ***Strings*** *como:*
  frase, isto\n é, como\n non, unha frase da lugar a 4 tokens String
  varias, palabras, separadas por varias, comas da lugar a 4 tokens String
- *A "solución avanzada" divide correctamente ignorando os divisores dentro de comiñas:*
  frase, "isto é, como non, unha frase", de exemplo da lugar a 3 tokens String
- *A "solución completa" divide correctamente ignorando ademais as comiñas dentro de comiñas:*
  frase, "isto é ("imos, a ver..."), algo","estupendo" da lugar a 3 tokens String

A solución básica se pode realizar mediante o método ***split*** de String, as solucións avanzada e completa requiren un algoritmo máis complexo.

```
import utils.Tokenizer;

public class TokenizerTest {
   public static void main(String[] args) {
      Tokenizer tk=new Tokenizer();
      System.out.println("Test 1 - Basico");
      for(String s:tk) System.out.println("TOKEN: "+s);
      tk.setString("frase, isto\n é, como\n non, unha frase");
      for(String s:tk) System.out.println("\tTOKEN: "+s);
      tk.setString(null);
      for(String s:tk) System.out.println("\tTOKEN: "+s);
      System.out.println();
      tk.setSeparator("#;#");
      tk.setString("varias#;#palabras#;#separadas por  #;# separadores");
      tk.forEach(t->System.out.println("\tTOKEN: "+t));

      tk.setSeparator(",");
      System.out.println("\nTest 2 - Avanzado");
      tk.setString("frase, \"isto é, como non, unha frase\", de exemplo");
      tk.forEach(s->System.out.println("\tTOKEN: "+s));

      System.out.println("\nTest 3 - Completo");
      tk.setString("frase,\"isto é (\"imos, a ver...\"), algo\",\"estupendo\"");
      for(String s:tk) System.out.println("\tTOKEN: "+s);
      System.out.println();
      tk.setSeparator("#;#");
      tk.setString("frase#;#\"isto é (\"imos, a ver...\")#;# algo\"#;#\"estupendo\"");
      for(String s:tk) System.out.println("\tTOKEN: "+s);
   }
}
```

```
Test 1 - Basico
        TOKEN: frase
        TOKEN: isto é
        TOKEN: como non
        TOKEN: unha frase

        TOKEN: varias
        TOKEN: palabras
        TOKEN: separadas por
        TOKEN: separadores

Test 2 - Avanzado
        TOKEN: frase
        TOKEN: isto é, como non, unha frase
        TOKEN: de exemplo

Test 3 - Completo
        TOKEN: frase
        TOKEN: isto é ("imos, a ver..."), algo
        TOKEN: estupendo

        TOKEN: frase
        TOKEN: isto é ("imos, a ver...")#;# algo
        TOKEN: estupendo
```

Criterios de Avaliación

• A definición da clase e os construtores son correctos e os atributos apropiados	0.50
• ***setSeparator, setString, getString, getToken, getTokens e numTokens*** son correctos	1
• A clase é Iterable polos tokens do String	0.50
• scanData simple	0.50
• scanData avanzado	2
• scanData completo	3