

Primero desarrollamos un **programa/clase** llamada “**lenguaje**” que escriba palabras, generadas/formadas de forma aleatoria, en el interior de un archivo de texto con extensión .txt. Para su implementación desarrollamos el siguiente código:

```

13 public class Lenguaje {
14
15     public static void main(String[] args) {
16
17         String letras = "abcdefghijklmnopqrstuvwxyz";
18         String nombreFichero;
19         FileLock bloqueo = null;
20
21         if (args.length == 2) {
22             try {
23                 int numLineas = Integer.parseInt(args[0]);
24                 String osName = System.getProperty("os.name");
25                 if (osName.toUpperCase().contains("WIN")) {nombreFichero = args[1].replace("\\", "\\\\");}
26                 else {nombreFichero = args[1];}
27                 File archivo = new File(nombreFichero);
28                 if (!archivo.exists()) {
29                     archivo.createNewFile();
30                 }
31
32                 RandomAccessFile raf = new RandomAccessFile(archivo, "rwd");
33                 bloqueo = raf.getChannel().lock();
34                 raf.seek(archivo.length());
35
36                 for (int i = 0; i < numLineas; i++) {
37                     String linea = "";
38                     int numCaracteres = generarNumeroAleatorio(1, 10);
39                     for (int j = 0; j < numCaracteres; j++) {
40                         linea += letras.charAt(generarNumeroAleatorio(0, letras.length() - 1));
41                     }
42                     raf.writeChars(linea + "\n");
43                 }
44                 bloqueo.release(); bloqueo = null; raf.close();
45             } catch (IOException ex) {Logger.getLogger(Lenguaje.class.getName()).log(Level.SEVERE, null, ex);}
46         } else {
47             System.out.println("El programa debe tener dos parametros");
48         }
49     }
50
51     public static int generarNumeroAleatorio(int minimo, int maximo) {
52         int num = (int) (Math.random() * (maximo - minimo + 1) + (minimo));
53         return num;
54     }
55 }

```

Como se puede apreciar en la captura adjuntada:

- Primeramente creamos e inicializamos las variables principales, como:
 - Una **variable** llamada “**letras**” (a modo de ejemplo). Dicha variable contendrá las letras del alfabeto a usar como base; en este caso usaremos el abecedario español sin la letra ñ, para evitar errores de compilación y ejecución. Dichas letras servirán para generar las diferentes palabras aleatorias, solicitadas en el enunciado, y poderlas introducir así en el archivo resultante con extensión .txt.
 - Otra **variable** llamada “**nombreFichero**” para alojar el nombre del fichero a llenar o modificar.
 - Y otra variable de tipo ‘FileLock’ llamada “**bloqueo**”, para evitar que se escriban y sobre escriban todas las letras de forma simultanea.
- Seguidamente definimos un **condicional ‘if else’** para lograr que nuestro programe sólo se ejecute cuando se presenten dos parámetros.
- Envolvemos todo el contenido del ‘if else’ con el **método de control de excepciones ‘try catch’**, para capturar así los posibles errores durante su ejecución.
- Por consiguiente, cuando se presenten dos parámetros y se cumpla la condición necesaria, **se ejecutaran las sentencias o acciones requeridas / deseadas**. Dichas acciones realizadas resultan las siguientes:
 - Convertimos a entero el número de líneas del archivo introducido.
 - Detectamos el **nombre del sistema operativo** (u OS).

- Realizamos las “correcciones” adecuadas y necesarias para lograr la correcta creación, lectura y escritura de nuestro archivo.
 - Creamos la **variable “archivo”**.
 - **Comprobamos si existe un archivo con el nombre introducido** mediante la ejecución del comando correspondiente.
 - **Creamos una variable realizar el acceso aleatorio sobre el archivo y bloqueamos** la posibilidad de **que varios procesos puedan escribir sobre este de forma simultanea**.
 - **Forzamos que** nuestro proceso **se posicione después de la última línea** del archivo y siga escribiendo a partir de ese punto.
 - Abrimos un **bucle ‘for’** (para).
 - **Inicializamos una nueva línea** mediante un ‘String (o cadena de texto) vacío’.
 - Creamos e inicializamos la **variable “numCaracteres”** para ir generando y guardando diferentes longitudes (números enteros) con la que poder generar palabras de forma aleatoria.
 - **Anidamos otro bucle ‘for’** en el interior del primero,
 - Generamos una **nueva línea** en el interior del archivo y esta contendrá el **número de caracteres generado** previamente.
 - Y antes de salir del segundo bucle ‘for’, **escribimos la nueva línea generada**, añadiéndole un ‘intro’ al final de esta.
 - **Limpiamos la variable ‘bloqueo’ y le volvemos a asignar un valor ‘nulo’**.
 - **Cerramos la conexión** que nos brinda el acceso aleatorio al archivo.
 - Y por último, **definimos el método auxiliar** dedicado a generar un número aleatorio tomando como base unos valores mínimo y máximo preestablecidos.
- Para finalizar el primer apartado del presente ejercicio, **ejecutamos el siguiente comando** desde nuestro terminal de comandos:

java -jar lenguaje.jar 40 miFichero.txt

Seguidamente, pasamos a desarrollar la **clase Colaborar**, tal y como se solicita en el enunciado y la ubicamos en otro proyecto/proyecto independiente, como desarrollo del propio enunciado.

```
1  /* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
2  */
3  package com.kevinzamora.Ejercicio2;
4
5
6  import java.io.IOException;
7  import java.util.logging.Level;
8  import java.util.logging.Logger;
9
10 /**
11  *
12  * @author kzdesigner
13  */
14 public class Colaborar {
15
16     public static void main(String[] args) {
17
18         if (args.length == 1) {
19
20             try {
21                 for (int i = 1; i <= 10; i++) {
22
23                     System.out.println("Lanzado el proceso: " + i);
24
25                     String comando = "java -jar lenguaje.jar " + (i * 10) + " " + args[0];
26                     System.out.println("Lanzamos el siguiente comando: " + comando);
27
28                     Runtime.getRuntime().exec(comando);
29                 }
30             } catch (SecurityException ex) {
31                 System.out.println("Problema de seguridad: \n" + ex.getMessage());
32             } catch (IOException ex) {
33                 Logger.getLogger(Colaborar.class.getName()).log(Level.SEVERE, null, ex);
34             }
35         }
36     }
37 }
38
39 }
40 }
```

A
continuación
abrimos y

modificamos los archivos “pom.xml” (pertenecientes a ambos proyectos y relacionados directamente con el gestor de dependencias llamado ‘Maven’), para así poder **añadir el ‘plugin’ dedicado a exportar un ejecutable**, partiendo de los programas desarrollados y con extensión de archivo .jar. Para ello, copiaremos y pegaremos el código de instalación, cuya base/punto de partida hemos obtenido del siguiente [sitio web](#).

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5
6      <modelVersion>4.0.0</modelVersion>
7      <groupId>com.kevinzamora</groupId>
8      <artifactId>PSP01_ej2_p2</artifactId>
9      <version>1.0-SNAPSHOT</version>
10     <packaging>jar</packaging>
11
12     <properties>
13         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
14         <maven.compiler.source>21</maven.compiler.source>
15         <maven.compiler.target>21</maven.compiler.target>
16         <exec.mainClass>com.kevinzamora.psp01_ej2_p2.Colaborar</exec.mainClass>
17     </properties>
18
19     <build>
20         <plugins>
21             <plugin>
22                 <!-- Build an executable JAR -->
23                 <groupId>org.apache.maven.plugins</groupId>
24                 <artifactId>maven-jar-plugin</artifactId>
25                 <version>3.1.0</version>
26                 <configuration>
27                     <archive>
28                         <manifest>
29                             <mainClass>com.kevinzamora.Ejercicio2.Colaborar</mainClass>
30                         </manifest>
31                     </archive>
32                 </configuration>
33             </plugin>
34         </plugins>
35     </build>
36 </project>
```

Una vez realizados los pasos y procedimientos anteriores, **añadimos las anotaciones/comentarios** (para documentar los métodos y clases utilizados), haciendo uso de la herramienta llamada **Java Doc**. Para ello utilizamos la sintaxis siguiente:

```

68  /**
69  * Método para generar números aleatorios contenidos entre un valor mínimo y
70  * un máximo introducidos.
71  * @param min
72  * @param max
73  * @return int
74  */
75  public static int generarNumeroAleatorio(int minimo, int maximo) {
76      int num = (int) (Math.random() * (maximo - minimo + 1) + (minimo));
77      return num;
78  }
79
80  }

```

Finalmente, para completar el presente (y segundo) ejercicio, procedemos a ejecutar y poner a prueba el siguiente comando:

**java -jar colaborar.jar
miFichero.txt**

```

kzdesigner@kzdesigner-PC: ~/Downloads
File Edit View Search Terminal Help
kzdesigner@kzdesigner-PC:~/Downloads$ java -jar colaborar.jar miFichero.txt
Lanzado el proceso: 1
Lanzamos el siguiente comando: java -jar lenguaje.jar 10 miFichero.txt
Lanzado el proceso: 2
Lanzamos el siguiente comando: java -jar lenguaje.jar 20 miFichero.txt
Lanzado el proceso: 3
Lanzamos el siguiente comando: java -jar lenguaje.jar 30 miFichero.txt
Lanzado el proceso: 4
Lanzamos el siguiente comando: java -jar lenguaje.jar 40 miFichero.txt
Lanzado el proceso: 5
Lanzamos el siguiente comando: java -jar lenguaje.jar 50 miFichero.txt
Lanzado el proceso: 6
Lanzamos el siguiente comando: java -jar lenguaje.jar 60 miFichero.txt
Lanzado el proceso: 7
Lanzamos el siguiente comando: java -jar lenguaje.jar 70 miFichero.txt
Lanzado el proceso: 8
Lanzamos el siguiente comando: java -jar lenguaje.jar 80 miFichero.txt
Lanzado el proceso: 9
Lanzamos el siguiente comando: java -jar lenguaje.jar 90 miFichero.txt
Lanzado el proceso: 10
Lanzamos el siguiente comando: java -jar lenguaje.jar 100 miFichero.txt
kzdesigner@kzdesigner-PC:~/Downloads$

```

