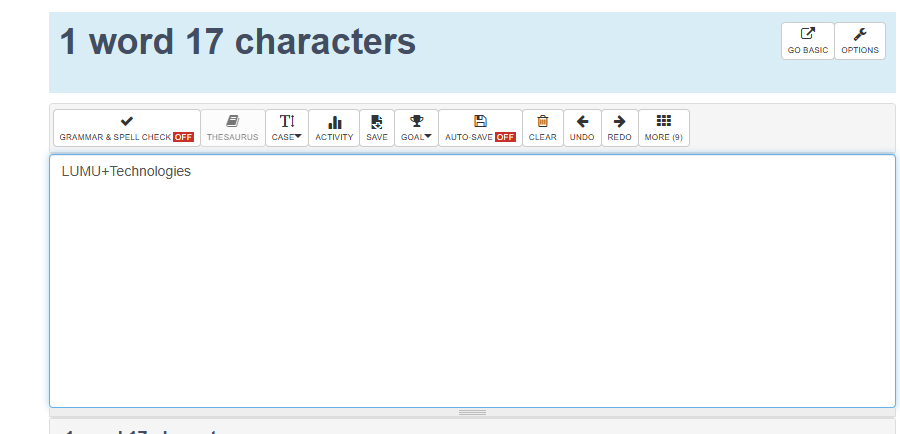
**NO hace una diferencia cuando dos palabras están separadas con carácter**



**import java.io.BufferedReader;** //Usado para leer el archivo

**import java.io.FileReader;**

**import java.io.IOException;**

**import java.util.ArrayList;**

**import java.util.Collections;**

**import java.util.Comparator;**

**import java.util.HashMap;**

**import java.util.List;**

**import java.util.Map;**

**public class WordCount {**

**public static void main(String[] args) {**

**String filePath = "path/to/your/file.txt";**

**countWordsAndCharacters(filePath);** //llama al metodo *countWordsAndCharacters*, pasando el parametro “filePath”

**}**

**public static void countWordsAndCharacters(String filePath) {**

**int wordCount = 0;** //variable de tipo entero que se incrementa con cada palabra encontrada

**int charCount = 0;** //variable de tipo entero que se incrementa con cada caracter encontrado, Incluyendo espacios

**Map<String, Integer> wordFrequency = new HashMap<>();** // Colección de tipo clave-valor, Mapa con claves tipo String y sus valores tipo Integer

**try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {** //lee el archive desde una ruta especificada

**String line;**

**while ((line = br.readLine()) != null) { //Ejecución del ciclo while hasta que devuelva un valor tipo null**

**charCount += line.length(); //cuenta el número total de caracteres**

**String[] words = line.split(" "); //creación de un array dividido por el caracter espacio (“ “).**

**wordCount += words.length; //total del número de palabras**

**for (String word : words) { //ciclo que recorre las palabra del array “words”, en cada iteración la variable de tipo String //Word, toma el valor de cada una de las palabra**

**String lowerWord = word.toLowerCase(); //convierte todos los caracteres de una palabra en minúscula**

**wordFrequency.put(lowerWord, wordFrequency.getOrDefault(lowerWord, 0) + 1); //actualiza el valor de la frecuencia de la palabra**

**}**

**}**

**} catch (IOException e) { //manejo de una excepcion**

**e.printStackTrace(); // captura la exepción**

**}**

**System.out.println(wordCount + " words"); //imprime numero de palabras**

**System.out.println(charCount + " characters"); //imprime numero de caracteres**

**List<Map.Entry<String, Integer>> sortedEntries = new ArrayList<>(wordFrequency.entrySet());**

**sortedEntries.sort((entry1, entry2) -> entry2.getValue().compareTo(entry1.getValue())); //ordena las palabras segun la frecuencia, el comparador organiza los valores en orden descendente**

**for (Map.Entry<String, Integer> entry : sortedEntries) {**

**System.out.println(entry.getKey() + ": " + entry.getValue()); //obtiene la palabra de la entrada actual y entry.getValue() obtiene la frecuencia de la palabra**

**}**

**}**

**}**

**2. Sugerencias de Mejora.**

1. **El codigo actual, divide las palabras por espacios, pero se pueden presentar otras situaciones como:**

* **Varios espacios**
* **Caracteres: , : . (varios signos de puntuacion)**

**Ejemplo: String word = word.replaceAll("[^a-zA-Z0-9 ]", "");**

1. **El bloque catch podría tener un mensaje más disiente:**
   1. **System.err.println("Error al leer el archivo: " + e.getMessage());**
2. **Cálculo del número de palabras y caracteres: Mejorar la forma de calcular los números de palabras y caracteres al mismo tiempo, sin necesidad de recorrer el archivo dos veces.**

***// Ordenar por frecuencia descendente***

***List<Map.Entry<String, Integer>> sortedEntries = new ArrayList<>(wordFrequency.entrySet());***

***sortedEntries.sort((entry1, entry2) -> entry2.getValue().compareTo(entry1.getValue()));***

***// Imprimir la frecuencia de las palabras más comunes***

***for (Map.Entry<String, Integer> entry : sortedEntries) {***

***System.out.printf("%s: %d\n", entry.getKey(), entry.getValue());***

***}***

1. ***Mejorar el manejo de las Excepciones, se pueden considerar las siguientes 2 opciones***
   1. ***Ruta del archivo errada***
      1. ***!Files.exists(Paths.get(filePath)***
   2. ***Error al leer el archivo***