**Enunciado**

Un programa que sea capaz de procesar un conjunto de archivos de texto y construir un vocabulario con todas las palabras diferentes descubiertas en los archivos, de forma de determinar además la frecuencia de aparición de cada palabra en el conjunto de archivos.

* Capaz de procesar de a uno los archivos indicados
* Se deberá analizar (o "parsear") el documento, limpiar el texto de los signos de puntuación, palabras numéricas o alfanuméricas, dígitos y todo otro símbolo que no forme parte de una palabra normalmente entendida
* Cada palabra agregándola al vocabulario y contando su frecuencia de aparición
* Además de la frecuencia de aparición de cada palabra, informar el/los documentos del conjunto de documentos procesados donde la palabra fue encontrada (es decir que cada palabra debe mantener información acerca de en qué documentos apareció).

**Requerimientos**

* Interfaz de usuario
* Permitir elegir el/los archivos a procesar
* Diseño eficiente
* Presentar una pantalla que liste todas las palabras del en una grilla mostrando la palabra, la frecuencia total de aparición y la cantidad de documentos de la muestra donde fue observada
* Soportar la funcionalidad de filtrar el vocabulario en base a una raíz de palabra escrita en un cuadro de texto, a única posibilidad de filtro es mediante la búsqueda de los primeros caracteres de la palabra y no una sub cadena contenida en cualquier parte de la palabra
* La posibilidad de ir filtrando la lista a medida que se van agregando letras en el cuadro de texto de filtro del vocabulario
* El vocabulario y la relación entre las palabras y los documentos en los que se encontró la misma deberán ser almacenados en una base de datos
* Debe ser posible agregar documentos nuevos a una base de datos donde ya hay un vocabulario construido

**Tareas**

* **Investigar e Implementar Hashing**

**Ficha 17 Catedra**

* **Investigar e Implementar parseo de Strings**

[**http://pages.cs.wisc.edu/~hasti/cs302/examples/Parsing/parseString.html**](http://pages.cs.wisc.edu/~hasti/cs302/examples/Parsing/parseString.html)

* **Leer los archivos para buscar los separadores; los signos de puntuación, palabras numéricas o alfanuméricas, dígitos y todo otro símbolo que no forme parte de una palabra normalmente entendida**
* **Crear Base de Datos; 2 Tablas palabra y documento**
* **Implementar Persistencia; Investigar Transacciones**

**JDBC**

* **Implementar Entidades**
* **Diseñar Interfaz Grafica**
* **Investigar e Implementar Cuadro de dialogo**

**JFileChooser (ver material Catedra)**

[**https://docs.oracle.com/javase/7/docs/api/javax/swing/JFileChooser.html**](https://docs.oracle.com/javase/7/docs/api/javax/swing/JFileChooser.html)

[**https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html**](https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html)

* **Implementar Cola de tareas**

**LinkedList/Queue**

* **Investigar e Implementar las Grillas**
* **Implementar filtrado de grilla por letras raíz**
* **Implementar avance de Proceso**

**SwingWorker (ver material Catedra)**

[**https://docs.oracle.com/javase/7/docs/api/javax/swing/SwingWorker.html**](https://docs.oracle.com/javase/7/docs/api/javax/swing/SwingWorker.html)

[**https://docs.oracle.com/javase/tutorial/uiswing/concurrency/worker.html**](https://docs.oracle.com/javase/tutorial/uiswing/concurrency/worker.html)

* **Implementar Bonus extra**
* **Ver el tema de rutas relativas; y que pasa cuando el proyecto cambia de pc; probar intercambios con pendrive; y de cargar más archivos una vez cambiado punto h)**