Programación 2
Tecnicatura Universitaria en Desarrollo de Aplicaciones Informáticas
Práctica Nº 7 – 2021

Para cada uno de los siguientes problemas Implementarlos en Java.

1 - Diccionario

Un diccionario almacena una lista de palabras, ordenadas alfabéticamente. Cada palabra del diccionario posee la función gramatical que cumple la palabra (sustantivo, adverbio, adjetivo, etc), una o más definiciones, una lista de sinónimos y una lista de antónimos.
Implementar en Java una solución que permita:

- Agregar, modificar y eliminar palabras del diccionario.
- Dada una palabra, obtener la lista de sinónimos, ordenada ascendentemente
- Dada una palabra, obtener la lista de antónimos, ordenada ascendentemente
- Dada una palabra, obtener la lista de definiciones
- Dadas dos palabras, obtener la lista de palabras comprendidas entre ellas, en orden alfabético.

2 - Vocabulario

Se desean llevar las estadísticas del vocabulario de un texto. El constructor de esta clase recibe como parámetro un String y crea los objetos necesarios para saber qué palabras aparecen en el mismo y cuántas veces. Se debe permitir:

1. Conocer la cantidad de palabras diferentes que contiene el texto.
2. Retornar las N palabras más frecuentes.
3. Retornar las N palabras menos frecuentes.
4. Obtener la frecuencia de ocurrencia de una palabra.
5. Obtener un listado de palabras ordenadas ascendentemente.
6. Obtener un listado de palabras ordenadas por frecuencia.

Nota: para separar las palabras del string recibido como parámetro del constructor, se puede utilizar el método “split” de la clase String, que recibe como parámetro una expresión regular por la que dividir el string y retorna un arreglo con las paradas separadas por dicha expresión

```
String[] arreloDeStrings = texto.split(" ");
```

Programación 2
Tecnicatura Universitaria en Desarrollo de Aplicaciones Informáticas
Práctica Nº 7 – 2021

3 – Biblioteca

Una biblioteca posee un sistema para la administración de libros disponibles para sus socios. El sistema debe permitir ordenar el conjunto de todos los libros disponibles por diferentes criterios, acorde a lo que desee buscar en un determinado momento. Por defecto, los libros se ordenan por ISBN, un código identificatorio único de cada libro. Sin embargo, es posible que el administrador de la biblioteca desee ordenarlos por autor, año de edición o género principal, tanto ascendente como descendentemente. Resolver el problema utilizando el método `sort(.)`, de la clase `Collections`.

4. Búsqueda de documentos

Un historiador desea digitalizar sus documentos y organizarlo de acuerdo a palabras clave. Un documento tiene un título, una lista de autores, un contenido textual y una lista de palabras clave. El historiador necesita poder encontrar fácilmente documentos en su colección de acuerdo a diferentes criterios, por ejemplo:

a) Todos los documentos cuyo título sea exactamente igual a un título dado.
b) Todos los documentos cuyo título contenga una palabra o frase dada.
c) Todos los documentos que contengan una palabra clave dada.
d) Todos los documentos que no contengan ninguna palabra clave.
e) Todos los documentos de un autor determinado.
f) Todos los documentos cuyo contenido tenga una palabra o frase dada.
g) Todos los documentos cuyo contenido tenga al menos 20 palabras.
h) Cualquier combinación lógica de las formas anteriores.

Programación 2
Tecnicatura Universitaria en Desarrollo de Aplicaciones Informáticas
Práctica Nº 7 – 2021

5 - Vivero

Un vivero necesita un sistema que le permita cargar fichas de las diferentes plantas que vende. Cada planta posee un nombre científico, una lista de nombres vulgares, una clasificación superior, una familia y una clase. Las plantas pueden prosperar mejor en interior o en exterior, tienen un requerimiento de sol que se indica con un número entero del 1 al 10 (1 para sombra, hasta 10 para sol pleno) y un requerimiento de agua que se indica de la misma manera (1 para riesgo escaso y 10 para riego abundante).

Por ejemplo:

Nombre Científico: Epipremnum aureum
Nombres vulgares: potus, pothos o potos
Clasificación superior: Epipremnum
Familia: Araceae
Clase: Monocotyledoneae
Planta de interior
Riego: 3
Sol: 4

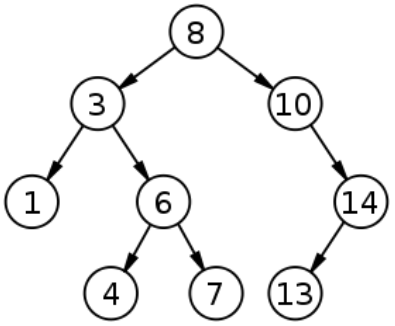
El sistema debe ser útil para los administradores del vivero para, por ejemplo, cuando llega un cliente poder ofrecerle las plantas que cumplen con sus requerimientos:

a) Todas las plantas cuyo nombre científico incluya la palabra "auereum"
b) Todas las plantas a las que se conozca vulgarmente como "lengua de suegra"
c) Todas las plantas cuya clasificación sea "Crassula"
d) Todas las plantas que requieran un nivel de sol superior a 5 y riego inferior a 3
e) Todas las plantas que requieran un nivel de sol inferior a 4 y riego superior a 4
f) Todas las plantas de interior que necesiten poco riego (inferior a 3)
g) Cualquier combinación lógica de las formas anteriores

Los listados anteriores se deben retornar ordenados por el criterio de preferencia de cada cliente. Por ejemplo, un cliente puede estar interesado en "crassulas", ordenadas por el nivel de sol requerido (primero las que requieran más sol), y si hay varias que requieren el mismo nivel de sol, las que requieran menos agua deben mostrarse primero y, finalmente, por orden alfabético del nombre científico. El cliente puede proporcionar un criterio de orden (ascendente o descendente) de cualquiera de los atributos que se guardan de las plantas.

Programación 2
Tecnicatura Universitaria en Desarrollo de Aplicaciones Informáticas
Práctica Nº 7 – 2021

6 - Árbol binario de búsqueda



Un árbol binario es una estructura de datos formada por nodos que contienen un determinado valor. El primer elemento agregado a la estructura se conoce con el nombre de "*raíz*" y es el único punto de acceso a la misma. Cada nodo, puede tener un nodo "*hijo*" a su izquierda y un nodo hijo a su derecha cumpliendo con la restricción que los valores a su izquierda son valores menores que su propio valor, y los valores a su derecha son valores mayores (no se almacenan valores repetidos). Los nodos sin hijos se conocen como "*hojas*". Normalmente, para facilitar el recorrido de la estructura, cada nodo tiene una referencia a su nodo "*padre*".

A. Implementar la funcionalidad para agregar un nuevo objeto a la estructura. Para poder trabajar con cualquier objeto es necesario que el mismo pueda ser comparable, es decir, implementar la interfaz `Comparable` de Java.
B. Implementar un método que permita recorrer la estructura en orden, es decir, todos los elementos a la izquierda, luego la raíz y después todos los elementos a la derecha. Al recorrer los elementos es necesario que se defina una acción la cual se va a ejecutar. Para poder trabajar de forma transparente y que se pueda extender la funcionalidad definir una interfaz `AccionEjecutable`. La misma posee un método public void `ejecutarNodo(Object nodo)`. Una posible implementación sería:

```
public class ImprimirEnPantalla implements AccionEjecutable {

     public void ejecutarNodo(Object nodo) {
          System.out.println(nodo.toString());
     }
}
```

- Crear una acción que permita incorporar los elementos de forma ordenada ascendentemente a un Vector.
- Crear una acción que permita incorporar los elementos de forma ordenada descendentemente a un Vector.
- Crear una acción que cuente la cantidad de elementos visitados.