Mini Proyecto 3 - Implementación de tablas con árboles binarios de búsqueda

En este proyecto os pedimos que implementéis las operaciones básicas sobre tablas usando árboles binarios de búsqueda.

Interfaz *Map<K, V>*

Esta será la interfaz que declarará las operaciones que podremos realizar sobre una tabla:

```
public interface Map<K, V> {
    boolean isEmpty();
    void put(K key, V value);
    V get(K key);
    void remove(K key);
}
```

Las operaciones de la interfaz se comportan de la siguiente manera:

- *isEmpty()* devuelve true si la tabla está vacía y false, en caso contrario.
- *put(key, value)* asocia la clave *key* al valor *value*. Si *key* o *value* son *null*, lanzará *NullPointerException*.
- *get(key)* devuelve el valor asociado a la clave *key* o *null*, en caso de que la clave no tenga ningún valor asociado.
- *remove(key)* elimina la asociación que pudiera tener la clave *key*.

Clase *BSTMap<K, V>*

Esta será la clase que implementará el `Map` con un árbol binario de búsqueda. Como en la implementación será necesario que las claves sean comparables, la declaración de la clase será:

```
public class BSTMap<K extends Comparable<? super K>, V>
       implements Map<K, V> {

   ¿?
}
```

- Internamente el BST estará formado por nodos similares a los utilizados para implementar la clase *LinkedBinaryTree* de la práctica anterior.
- Estudiad con detenimiento la parte de la teoría que describe las operaciones sobre árboles binarios de búsqueda.
- Haced diagramas de todos los casos de todas las operaciones y después, una vez tenéis claro el comportamiento de las operaciones, implementadlas y probadlas.
- La recursividad es vuestra amiga.
- Especialmente en la operación `remove`, no os preocupéis de si se hacen dos recorridos del mismo árbol, uno para encontrar el elemento mínimo y otro para borrarlo.

Entrega

La entrega de este ejercicio consiste en un proyecto Netbeans con los ficheros que contienen vuestra implementación y un **informe en formato PDF** que contenga, como mínimo, los diagramas que muestren el funcionamiento de las operaciones sobre los árboles.

El resultado obtenido debe ser entregado por medio de un único **fichero comprimido en formato ZIP**, **uno por cada grupo,** con el nombre "MP3_NombreIntegrantes.zip".

Consideraciones de Evaluación

A la hora de evaluar el laboratorio se tendrán en cuenta los siguientes aspectos:

- Desgraciadamente hemos de indicar que una solución que se detecte como copiada tendrá **una nota igual a 0**.
- Corrección de la solución (una solución incorrecta tendrá una **nota inferior a 4**).
  - Es conveniente que hagáis diferentes pruebas sobre diferentes situaciones.
- Corrección del informe entregado y justificación de los diseños de los métodos.
  - No hace falta que hagáis los diagramas electrónicamente: podéis hacerlos a mano y después escanearlos.
- Calidad y limpieza del código (nombres de variables y métodos, descomposición funcional, etc., etc.).
- Diseño orientado a objetos y buen uso de los genéricos:
  - Recordaos de configurar `-Xlint:unchecked` (y de desconectar la compilación automática) para que se os muestren los errores en el uso de genéricos.

La nota de esta práctica, por cuestiones de calendario, no será recuperable.

Universitat de Lleida
Departament d'Informàtica
i Enginyeria Industrial