Práctica 3: Mapas y Diccionarios Ordenados

Normas:

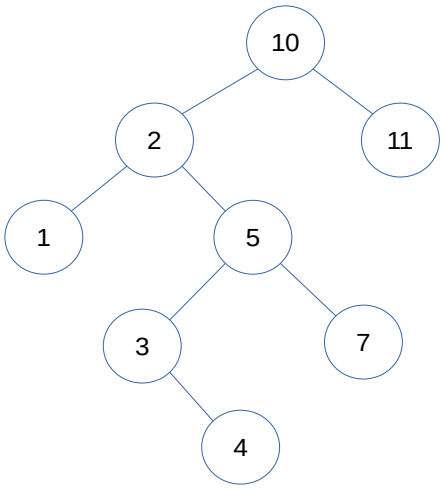
- Cada alumno/a debe realizar los ejercicios de manera individual, aunque pueden compartir información oral. Existe un detector anticopias.
- Este enunciado va acompañado de un fichero ZIP que contiene el código necesario para hacer la práctica. Los test que acompañan las prácticas son orientativos y pueden ser ampliados en la fase de evaluación.

Ejercicio 1: Encontrar el elemento más cercano en árboles binarios de búsqueda.

En la clase `ClosestInteger` implementar el método:

```
public Position<Integer> closest (BinarySearchTree<Integer> tree, int i);
```

Este método recibe un árbol binario de búsqueda de números enteros y un valor entero. Devolverá el `Position` del árbol binario de búsqueda más próximo en valor al elemento buscado.
Si el elemento está en el árbol devuelve su `Position`.
Si el árbol es vacío o nulo debe lanzar una excepción.



Para el árbol de la figura con un valor i = 13 obtendríamos el `Position` de valor 11, puesto que es el más cercano a 13.

Ejercicio 2: Averiguar si es o no binario de búsqueda

En la clase `CheckTree` implementar el método:

```
public boolean isBinarySearchTree(BinaryTree<E> tree);
```

Que devuelve `true` si el árbol que se le pasa cumple las características del árbol binario de búsqueda, en caso contrario devuelve `false`.

Ejercicio 3: El problema de las referencias cruzadas

Dado un texto se desea conocer la posición en la que aparece cada palabra en él. Para ello se crea una clase `ReferenciasCruzadas` que almacena cada palabra y la posición que ocupa dicha palabra en el texto.

La clase `ReferenciasCruzadas` tiene dos métodos, el constructor, que recibe el fichero de texto que se desea analizar. Y el método `apariciones`, que recibe una palabra y devuelve una lista de enteros ordenados de menor a mayor, que indica las posiciones del texto en las que aparece dicha palabra (suponiendo que la primera palabra del texto ocupa la primera posición).

Para resolver el problema se desea que se haga uso de la clase `TreeMap` de Java. Esta clase es un mapa que implementa internamente el árbol Rojo-Negro estudiado en la asignatura.

Se recomiendo usar la clase `Scanner` de Java para procesar el fichero, así como un patrón para eliminar los signos de puntuación. Aquí os dejamos un ejemplo:

```
Scanner sc = new Scanner(fichero).useDelimiter("\\`|\\
~|\\!|\\@|\\#|\\$|\\%|\\^|\\&|\\*|\\(|\\)|\\+|\\=|\\[|\\
{|\\]|\\}|\\|\\|\\'|\\<|\\,|\\.|\\
>|\\?|\\/|\\"\"|\\;|\\:|\\s+");
```