BSTREE

Árbol binario de búsqueda, donde la información de la raíz siempre es más grande que todos los nodos que se encuentran por su rama izquierda y más pequeña que toda la información que se encuentra en los nodos de la rama derecha. Siempre que haga un BSTree de cualquier tipo de objeto, esa clase tiene que implementar la interfaz comparable.

ADD:

Método público y booleano que le pasamos por parámetro un elemento del tipo parametrizado T y devuelve true o false. Este método nos permite insertar un nuevo elemento en el árbol.

Busca el lugar donde se va a insertar el nuevo elemento, cumpliendo la propiedad de menores a la izquierda, mayores a la derecha (siempre insertando en hojas).

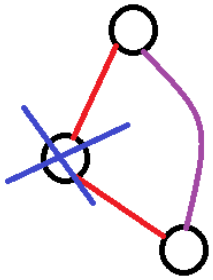
Si el elemento ya está, no se hace nada el método devuelve false. En caso contrario, se inserta en el lugar donde acaba la búsqueda devolviéndonos true. La búsqueda acaba sin éxito al llegar a un subárbol izquierdo o derecho que está vacío obtenemos un false como en el caso de estar ya el elemento.

REMOVE:

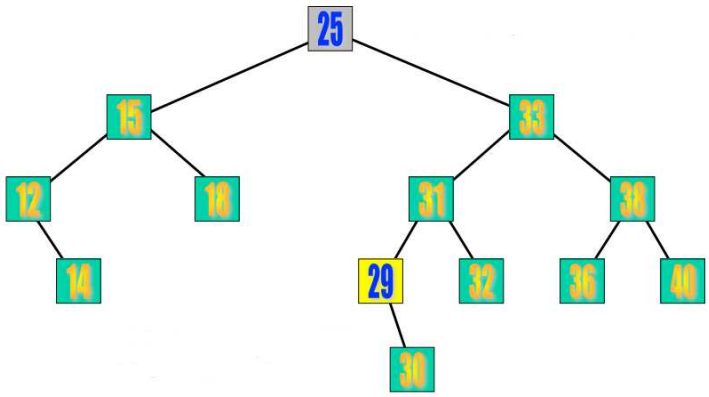
Método público y booleano que le pasamos por parámetro un elemento del tipo parametrizado T y devuelve true o false.

Hay tres posibilidades a la hora de eliminar un elemento/nodo:

- Nodo sin hijos: Elimina el nodo sin ningún problema.
- Nodo con un solo hijo: El padre del nodo a eliminar pasa a apuntar al hijo del nodo que está siendo eliminado.



-
- Nodo con dos hijos: Reemplazamos el nodo a eliminar por el nodo mínimo de su subárbol derecho (o máximo de su subárbol izquierdo). Elimina el nodo mínimo (o máximo).
  Dicho nodo será una hoja o tendrá un solo hijo.



El 29 que es el menor de los mayores pasara a ser la raíz cuando eliminemos el 25.

CLEAR:

Método público que no devuelve nada. Este método elimina todo los elementos del árbol, por lo que obtenemos un árbol vacío después de ejecutar el método.

CONTAINS:

Método publico abstracto booleano, nos devuelve true si el árbol contiene el elemento que le especificamos (se lo pasamos como parámetro de tipo Objeto). Como tienen una propiedad de orden no es necesario buscar en todo el árbol.

ISEMPTY:

Método público abstracto booleano, nos devuelve true si el árbol no contiene elementos, es decir está vacío.

ITERATOR:

Método público y abstracto que nos devuelve un iterador sobre los elementos del árbol. Los elementos se devuelven en orden ascendente. Dentro de la clase BSTree hemos definido una clase que se llama TreeIterator, este hace un recorrido por el árbol INORDEN (Izquierda-Raiz-Derecha).

SIZE:

Método público y abstracto que nos devuelve el número de elementos que hay en el árbol.

FIND:

Método público de tipo T que le pasamos un elemento de tipo T que sirve como clave para localizar el elemento en el árbol. Busca el elemento especificado en el árbol y devuelve el valor del nodo, cuyo

valor coincide con el elemento clave. Nos devuelve el valor del nodo que buscábamos o null en caso de que no lo encuentre.

Si usamos un BSTree y este está aproximadamente equilibrado, la complejidad en las búsquedas, las inserciones y las eliminaciones son de orden logarítmica en vez de complejidad n, pero si el árbol no está equilibrado no hay ventaja. Otra ventaja es que tendremos listados ordenados.