Árbol Binario De Búsqueda 🡪 ABB.

* No es lineal, **tiene una jerarquía.**
* Si el árbol es lineal se transforma en una lista.
* **Posee Búsqueda Binaria.** Son rápidos para la búsqueda si están ordenados y balanceados.
* Un árbol donde cada nodo no tiene más de dos hijos.
* Se dice que esta ordenado cuando cada nodo del árbol es mayor que los elementos de su subárbol izquierdo, y menor o igual que los de su subárbol derecho.
* Si no está balanceado tiene O(n). 🡪**Peor Caso**
* El nodo escondido es el ROOT, y a su izquierda (Hijo Izquierdo) empieza el árbol con datos.
* El nodo escondido sirve para usar coordenadas y representa el “END” (Siguiente del ultimo).
* El único punto de acceso al árbol es la raíz.
* Para **borrar un elemento** tenemos tres opciones:
  + - * + No hay ningún nodo con la clave igual a la que se desea eliminar. 🡪Retornamos False.
        + El nodo a eliminar tiene como máximo **un subárbol descendiente**:

Si tiene subárbol derecho, se cambia el puntero al nodo a eliminar, para que apunte al subárbol derecho del nodo a eliminar.

Si tiene subárbol izquierdo, se cambia el puntero al nodo a eliminar, para que apunte al subárbol izquierdo del nodo a eliminar.

* + - * + El nodo a eliminar tiene dos subárboles descendientes:

Cambiar el puntero al nodo a eliminar para que apunte al nodo que contiene LA MAYOR CLAVE, DEL SUBARBOL IZQUIERDO, DEL NODO A ELIMINAR.

Cambiar el puntero al nodo a eliminar para que apunte al nodo que contiene LA MENOR CLAVE, DEL SUBARBOL DERECHO, DEL NODO A ELIMINAR.

* Si se borra la raíz, hay que ir a la parte izquierda y buscar el mas grande para que siga balanceado. Ese nodo será la nueva raíz. Otra opción, es ir a la parte derecha y buscar el mas chico, y ese nodo seria la nueva raíz.
* En el BORRADO, **cada vez que se borra un nodo**, **hay que buscar un reemplazante**, ese reemplazante **será el hijo mayor, del hijo izquierdo del nodo borrado**.
* Se puede modificar el valor pero no a clave, ya que se desordena el árbol.
* **Coordenadas Livianas**: En el árbol, cada nodo conoce a su padre, por ende, la coordenada no necesita guardar información para retornar.
* **Coordenadas Pesadas**: La coordenada tiene la información necesaria para volver a la raíz desde el nodo donde este parado. Para ello, podemos usar una pila.
* En un árbol **no se puede insertar más rápido con coordenadas** porque desordena todo el árbol.
* La implementación de un ABB **no sirve para un árbol general** ya que el ABB se limita a 2 hijos por nodo. Para que sirva, en vez de tener los campos “LEFT” y “RIGHT”, hay que tener una lista con las referencias a los hijos de cada nodo.
* Recorrido “**INORDER**”: Voy al subárbol izquierdo, vuelvo a la raíz, y voy al subárbol derecho.