**Cách dùng lớp java.util.TreeSet: Cây BST cân bằng**



* Lưu trữ dữ liệu dạng cây có thứ tự một tập phần tử phân biệt nhau (Set)
* Không có cơ chế đồng bộ (synchronized/ thread safe) 🡪 Hai luồng cùng đồng thời truy xuất một TreeSet có thể làm dữ liệu mất nhất quán.
* Thứ tự được dùng: thứ tự tự nhiên (nếu phần tử là số hoặc chuỗi) hoặc do người dùng tự định nghĩa thông qua một biến interface Comparator (hành vi *int compare(T obj1, T obj2)*) hoặc phần tử trong TreeSet có hiện thực interface Comparable (hành vi *int CompareTo (T obj)*).
* Độ phức tạp của các tác vụ add, remove, contains là **O(logn).**

**Các hành vi thông dụng:**

|  |  |
| --- | --- |
| **Constructors** | |
| **Constructor** | **Description** |
| [**TreeSet**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#%3Cinit%3E())() | Constructs a new, empty tree set, sorted according to the natural ordering of its elements. |
| [**TreeSet**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#%3Cinit%3E(java.util.Collection))​([**Collection**](file:///E:\Tools\JavaDocs\api\java.base\java\util\Collection.html)<? extends [**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> c) | Constructs a new tree set containing the elements in the specified collection, sorted according to the *natural ordering* of its elements. |
| [**TreeSet**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#%3Cinit%3E(java.util.Comparator))​([**Comparator**](file:///E:\Tools\JavaDocs\api\java.base\java\util\Comparator.html)<? super [**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> comparator) | Constructs a new, empty tree set, sorted according to the specified comparator. |
| [**TreeSet**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#%3Cinit%3E(java.util.SortedSet))​([**SortedSet**](file:///E:\Tools\JavaDocs\api\java.base\java\util\SortedSet.html)<[**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> s) | Constructs a new tree set containing the same elements and using the same ordering as the specified sorted set. |

**Add operations**

|  |  |  |
| --- | --- | --- |
| boolean | [**add**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#add(E))​([**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) e) | Adds the specified element to this set if it is not already present. |
| boolean | [**addAll**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#addAll(java.util.Collection))​([**Collection**](file:///E:\Tools\JavaDocs\api\java.base\java\util\Collection.html)<? extends [**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> c) | Adds all of the elements in the specified collection to this set. |

**Search operations:**

|  |  |  |
| --- | --- | --- |
| [**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) | [**ceiling**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#ceiling(E))​([**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) e) | Returns the least element in this set greater than or equal to the given element, or null if there is no such element. |
| [E](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) | [**floor**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#floor(E))**​(**[**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)**e)** | Returns the greatest element in this set less than or equal to the given element, or null if there is no such element. |
| boolean | [**contains**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#contains(java.lang.Object))**​(**[**Object**](file:///E:\Tools\JavaDocs\api\java.base\java\lang\Object.html)**o)** | Returns true if this set contains the specified element. |
| [E](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) | [**first**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#first())**()** | Returns the first (lowest) element currently in this set. |
| [E](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html) | [**last**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#last())**()** | Returns the last (highest) element currently in this set. |

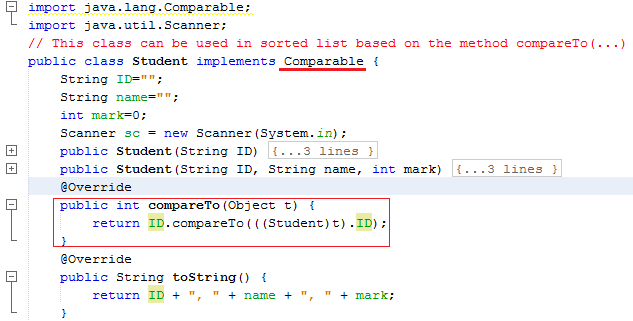
**Remove operations**

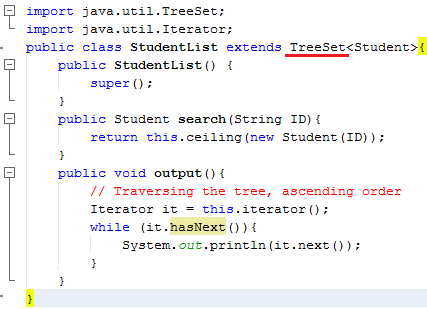
|  |  |  |
| --- | --- | --- |
| boolean | [**remove**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#remove(java.lang.Object))​([**Object**](file:///E:\Tools\JavaDocs\api\java.base\java\lang\Object.html) o) | Removes the specified element from this set if it is present. |

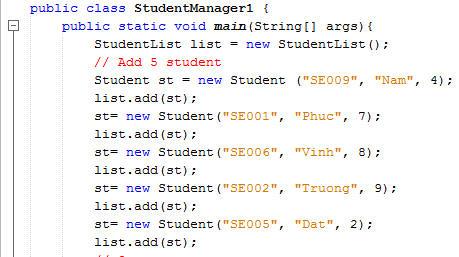
Methods for Traversing:

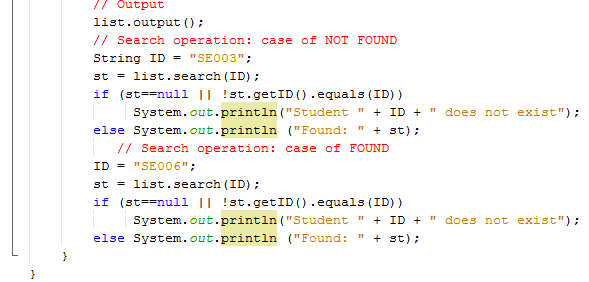
|  |  |  |
| --- | --- | --- |
| [Iterator](file:///E:\Tools\JavaDocs\api\java.base\java\util\Iterator.html)<[E](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> | [**iterator**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#iterator())**()** | Returns an iterator over the elements in this set in ascending order. |
| [**Iterator**](file:///E:\Tools\JavaDocs\api\java.base\java\util\Iterator.html)<[**E**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html)> | [**descendingIterator**](file:///E:\Tools\JavaDocs\api\java.base\java\util\TreeSet.html#descendingIterator())() | Returns an iterator over the elements in this set in descending order. |

**Demonstration 1: Using Comparable interface**



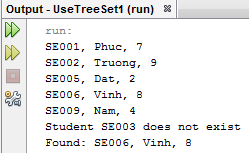




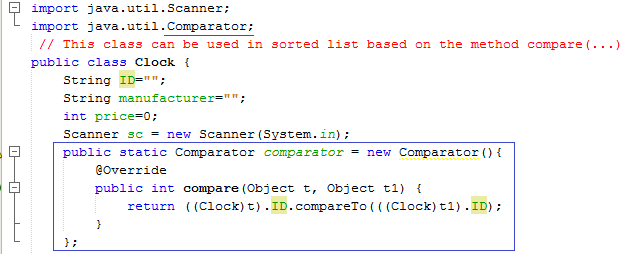


**Result:**

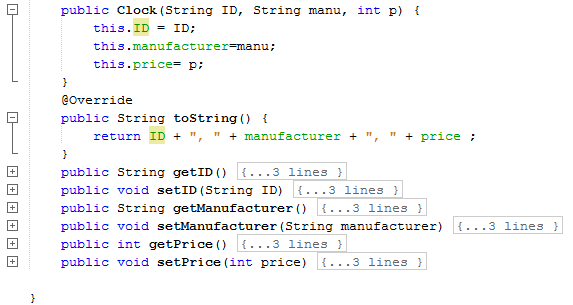
**hgf**

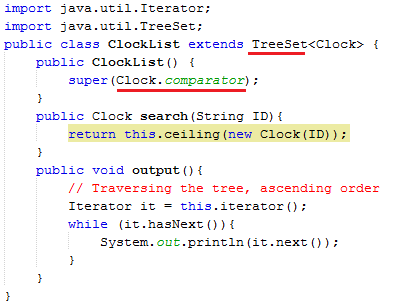
****

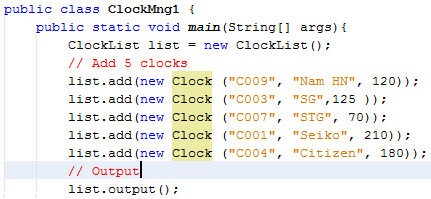
**Demonstration 2: Using Comparator interface**

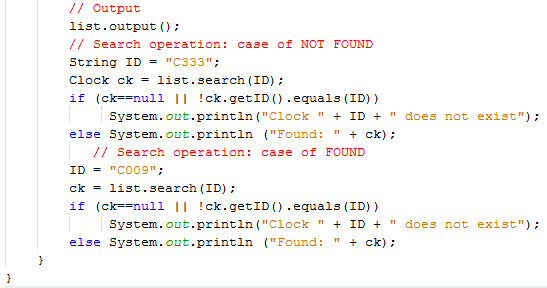




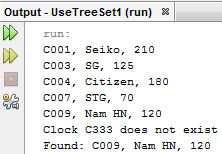








**Result:**



**Làm thêm Menu: Thêm/ Tìm/ Xóa/ Cập nhật/ Xuất**