Bài 5.8: Tìm hiểu lớp TreeSet

- ✓ Khái niệm và đặc điểm
- ✓ Các phương thức và mô tả
- ✓ Ví dụ minh họa

Khái niệm và đặc điểm

- Là một triển khai của NavigableSet dựa trên TreeMap.
- Các phần tử của tree set được sắp xếp theo thứ tự tự nhiên hoặc theo quy luật được chỉ định bởi comparator cho trước khi tạo đối tượng của TreeSet.
- Triển khai này đảm bảo cho các thao tác kiểm, thêm, xóa phần tử được thực hiện với chi phí thời gian bằng log(n).
- Triển khai này là không đồng bộ.
- TreeSet là một phần của Java Collections FrameWork.

Các phương thức và mô tả

Phương thức	Mô tả
TreeSet()	Khởi tạo một tree set mới rỗng, các phần tử của nó sẽ được sắp xếp sử dụng thứ tự tự nhiên.
TreeSet(Collection<? extend E> c)	Tạo một tree set mới từ các phần tử cho trước trong collection c trong tham số. Các phần tử của tree set sẽ được sắp xếp theo trật tự tự nhiên.
TreeSet(Comparator<? Super E> comparator)	Tạo một tree set mới rỗng. Thứ tự phần tử được sắp đặt theo trật tự được định ra trong comparator.
TreeSet(SortedSet<E> s)	Tạo một tree set mới chứa các phần tử được cho trong tham số. Sử dụng cùng quy tắc sắp xếp các phần tử được chỉ ra bởi tham số.
boolean add(E e)	Thêm mới phần tử e vào set nếu phần tử này chưa xuất hiện trong set.
boolean addAll(Collection<? extend E> c)	Thêm tất cả các phần tử trong collection c vào set hiện tại.
E ceiling(E e)	Trả về phần tử nhỏ nhất lớn hơn hoặc tương đương phần tử được chỉ định. Trả về null nếu không phần tử nào thỏa mãn.
void clear()	Xóa bỏ tất cả các phần tử trong set hiện tại.
Object clone()	Trả về một bản sao của đối tượng TreeSet.

Các phương thức và mô tả

Comparator<? Super K> comparator()	Trả về comparator sử dụng để sắp xếp các phần tử của tree set hiện thời. Trả về null nếu tree set hiện tại sử dụng thứ tự sắp xếp tự nhiên để sắp xếp các phần tử của nó.
boolean contain(Object o)	Trả về true nếu tree set chứa phần tử được chỉ định và false nếu ngược lại.
Iterator<E> descendingIterator()	Trả về một iterator của các phần tử trong set theo thứ tự duyệt giảm dần giá trị các phần tử.
NavigableSet<E> descendingSet()	Trả về một NavigableSet với các phần tử được đảo ngược thứ tự từ tree set hiện thời.
E first()	Trả về phần tử đầu tiên trong set hiện thời.
E floor(E e)	Trả về giá trị lớn nhất nhỏ hơn hoặc bằng e hoặc null nếu không có phần tử nào thỏa mãn.
SortedSet<E> headSet(E toElement)	Trả về một phần của set hiện tại chứa các phần tử nhỏ hơn toElement.
NavigableSet<E> headSet(E toElement, boolean inclusive)	Trả về một phàn cảu set trong đó các phần tử nhỏ hơn (hoặc bằng nếu inclusive = true) toElement.
E higher(E e)	Trả về phần tử nhỏ nhất lớn hơn phần tử được chỉ định hoặc null nếu không có phần tử thỏa mãn.
boolean isEmpty()	Trả về true nếu set rỗng và false trong trường hợp ngược lại.
Iterator<E> iterator()	Trả về một iterator lặp trên các phần tử của set hiện thời theo thứ tự tăng dần.

Các phương thức và mô tả

E last()	Trả về phần tử có giá trị lớn nhất trong set hiện thời.
E lower(E e)	Trả về giá trị lớn nhất nhỏ hơn phần tử được chỉ định hoặc null nếu không tồn tại phần tử thỏa mãn.
E pollFirst()	Xóa và trả về phần tử đầu tiên trong set hoặc trả về null nếu set rỗng.
E pollLast()	Xóa và trả về phần tử cuối trong set hoặc trả về null nếu set rỗng.
boolean remove(Object o)	Xóa phần tử được chỉ định khỏi set nếu nó tồn tại.
K lastKey()	Trả về key lớn nhất trong tree set hiện thời.
int size()	Trả về số lượng cặp ánh xạ key-value có trong tree set.
NavigableSet<E> (K from, boolean fromInclusive, K to, boolean toInclusive)	Trả về một phần của set hiện tại chứa các phần tử từ from đến to.
SortedSet<E> subSet(E from, E to)	Trả về một phần của set hiện tại năm trong khoảng [from, to).
SortedSet<E> tailSet(E fromElement)	Trả về một phần của set chứa các phần tử lớn honaw hoặc bằng fromElement.
NavigableSet<E> (K from, boolean fromInclusive)	Trả về một phần của set hiện tại chứa các phần tử lớn hơn(hoặc bằng fromElement nếu fromInclusive = true) from.

Nội dung tiếp theo

Tìm hiểu lớp TreeMap