CISC 3115 TY2

Set

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Module Outline

- Discussed
  - Concept of data structure
  - Use data structures
    - List
    - Sorting and searching in lists and arrays
  - Stack
  - Queue and priority queue
- To discuss
  - Set and map

Outline of This Lecture

- Concept of the Set data structure
- Set in Java
- HashSet, LinkedHashSet, and TreeSet
- Compare performance of Set and List
- Example programs
  - Word counting

Motivation

- In many applications, we do not allow duplications in a collection
  - Students enrolled in a class.
    - The class cannot have more than one objects of the same student.
  - Passengers on board an airplane
    - The passengers must also be unique.
- However, if we
  - ArrayList<Student> studentList = new ArrayList<>();
- We can do,
  - studentList.add(new Student(1, “John”));
  - studentList.add(new Student(1, “John”));
- Can we design a data structure that prevents this type of error?

Set

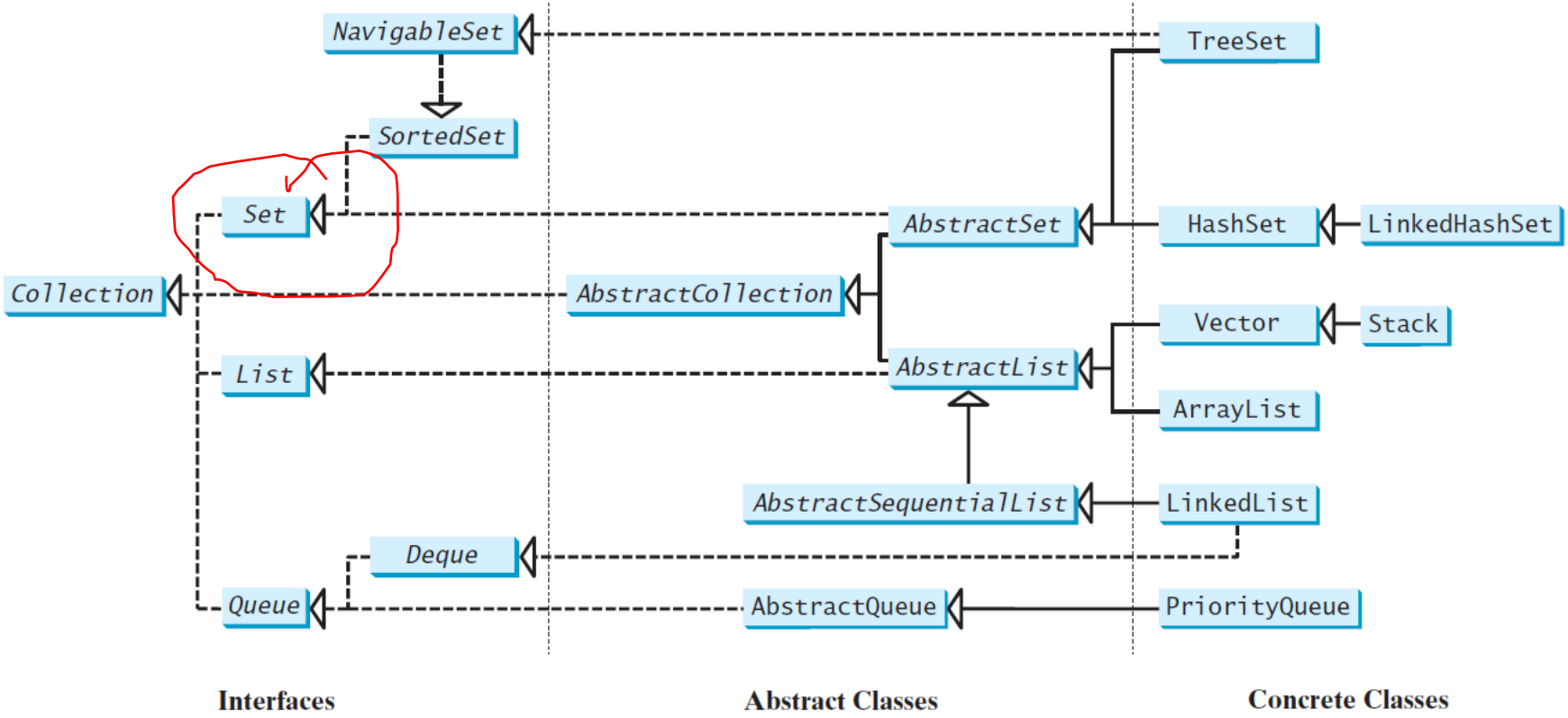
- A collection data structure where every item must be unique (no duplicates)

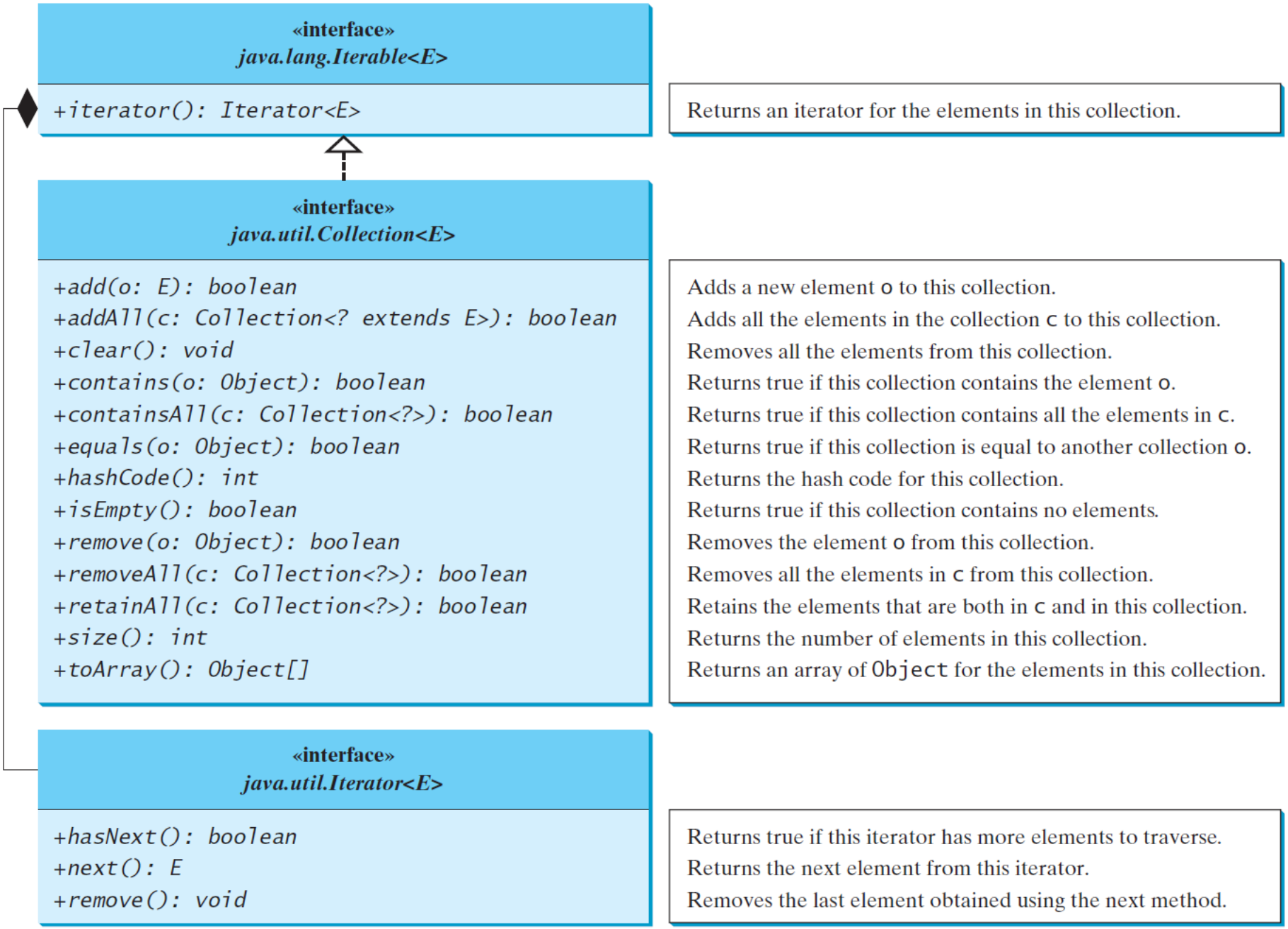
The Set Interface

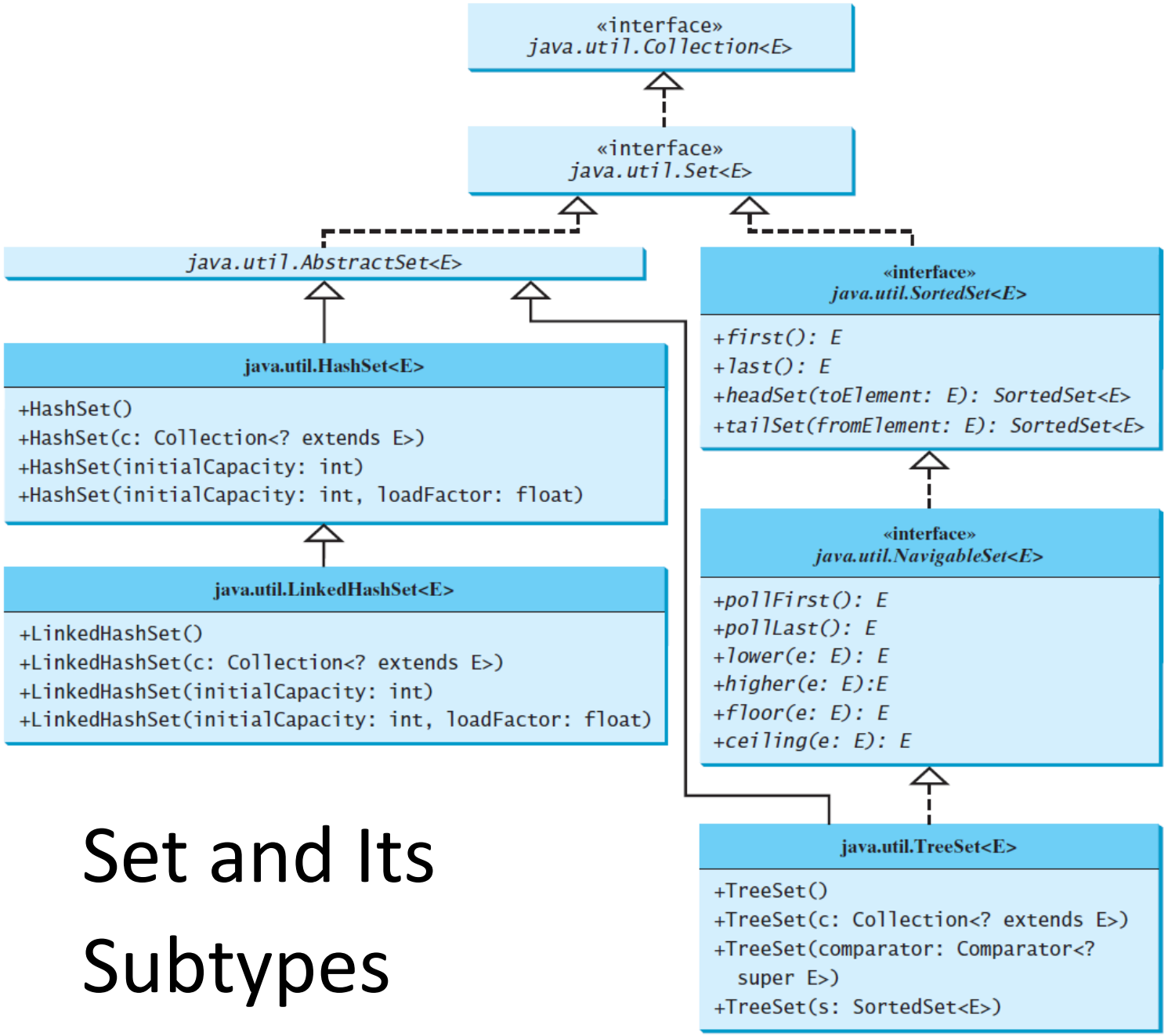
- The Set interface extends the Collection interface.
- It does not introduce new methods or constants, but it stipulates that an instance of Set contains no duplicate elements.
- The concrete classes that implement Set must ensure that no duplicate elements can be added to the set.
- Meaning of duplicates or uniqueness
  - No two elements e1 and e2 can be in the set such that e1.equals(e2) is true

Review: Java Collection Framework Type Hierarchy

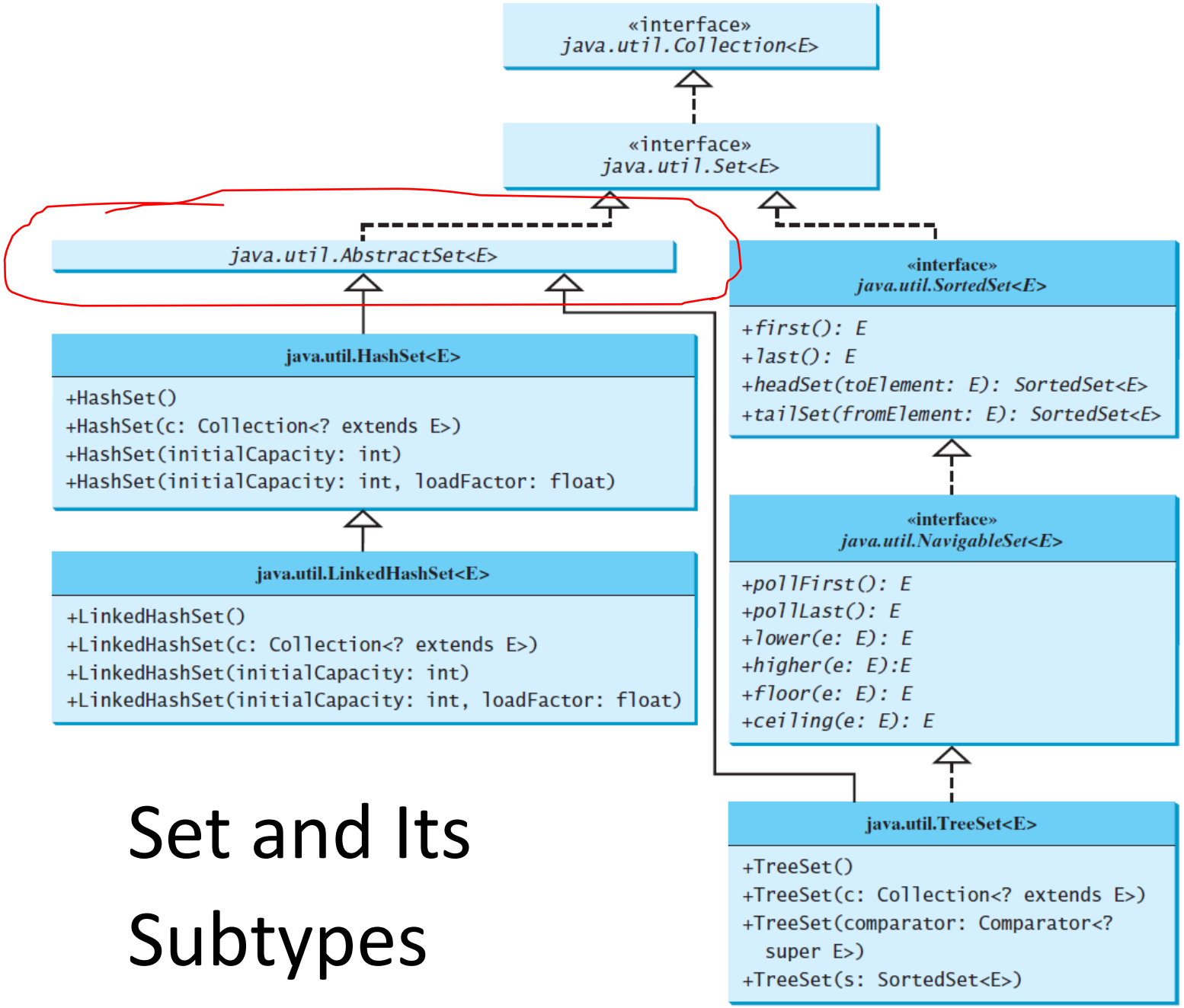
- Set is a subinterface of Collection




«interface»
java.lang.Iterable<E>
+iterator(): Iterator<E>
Returns an iterator for the elements in this collection.
«interface»
java.util.Collection<E>
+add(o: E): boolean
+addAll(c: Collection<? extends E>): boolean
+clear(): void
+contains(o: Object): boolean
+containsAll(c: Collection<?>): boolean
+equals(o: Object): boolean
+hashCode(): int
+isEmpty(): boolean
+remove(o: Object): boolean
+removeAll(c: Collection<?>): boolean
+retainAll(c: Collection<?>): boolean
+size(): int
+toArray(): Object[]
Adds a new element o to this collection.
Adds all the elements in the collection c to this collection.
Removes all the elements from this collection.
Returns true if this collection contains the element o.
Returns true if this collection contains all the elements in c.
Returns true if this collection is equal to another collection o.
Returns the hash code for this collection.
Returns true if this collection contains no elements.
Removes the element o from this collection.
Removes all the elements in c from this collection.
Retains the elements that are both in c and in this collection.
Returns the number of elements in this collection.
Returns an array of Object for the elements in this collection.
«interface»
java.util.Iterator<E>
+hasNext(): boolean
+next(): E
+remove(): void
Returns true if this iterator has more elements to traverse.
Returns the next element from this iterator.
Removes the last element obtained using the next method.



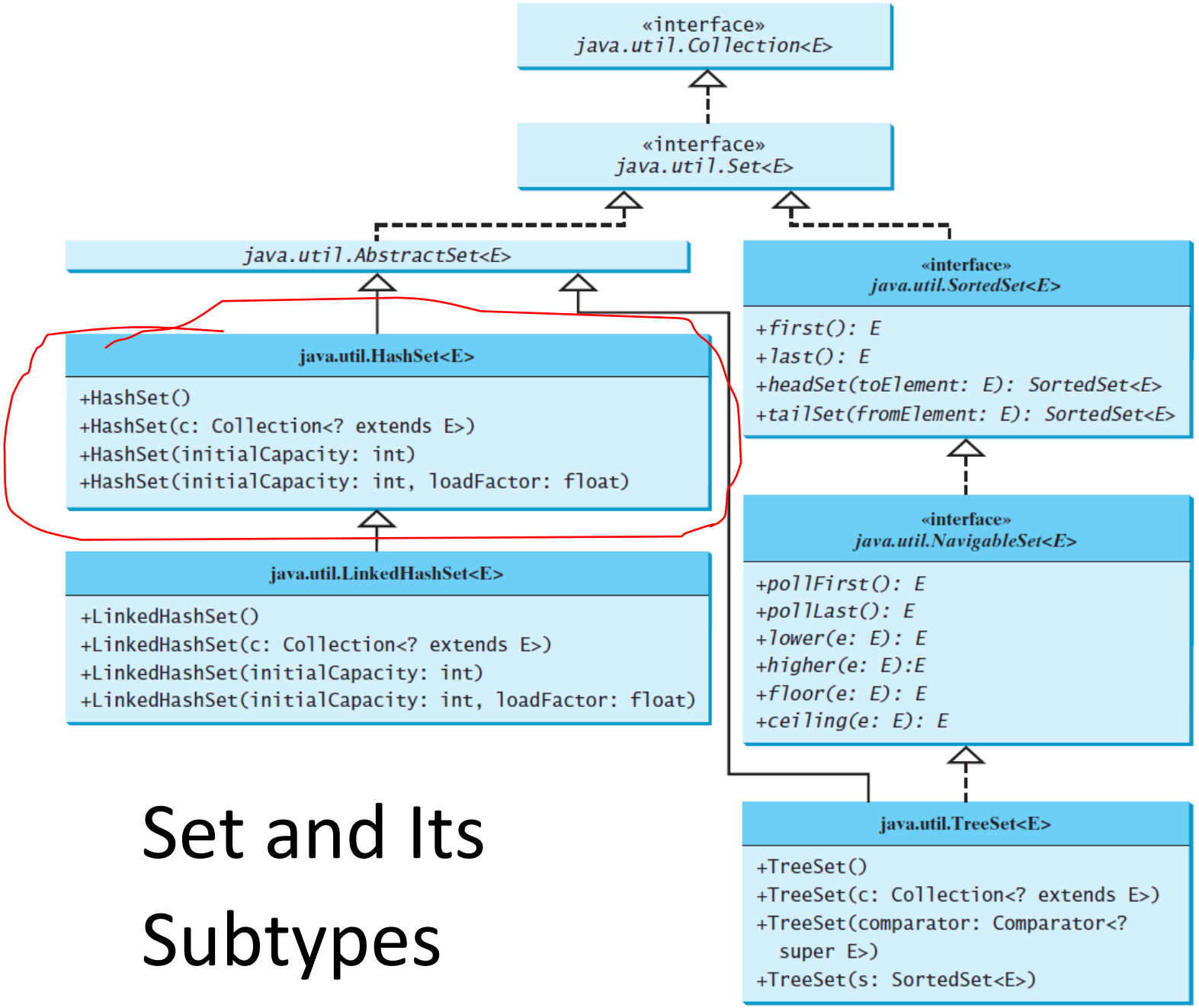
Set and Its Subtypes



Set and Its Subtypes

AbstractSet

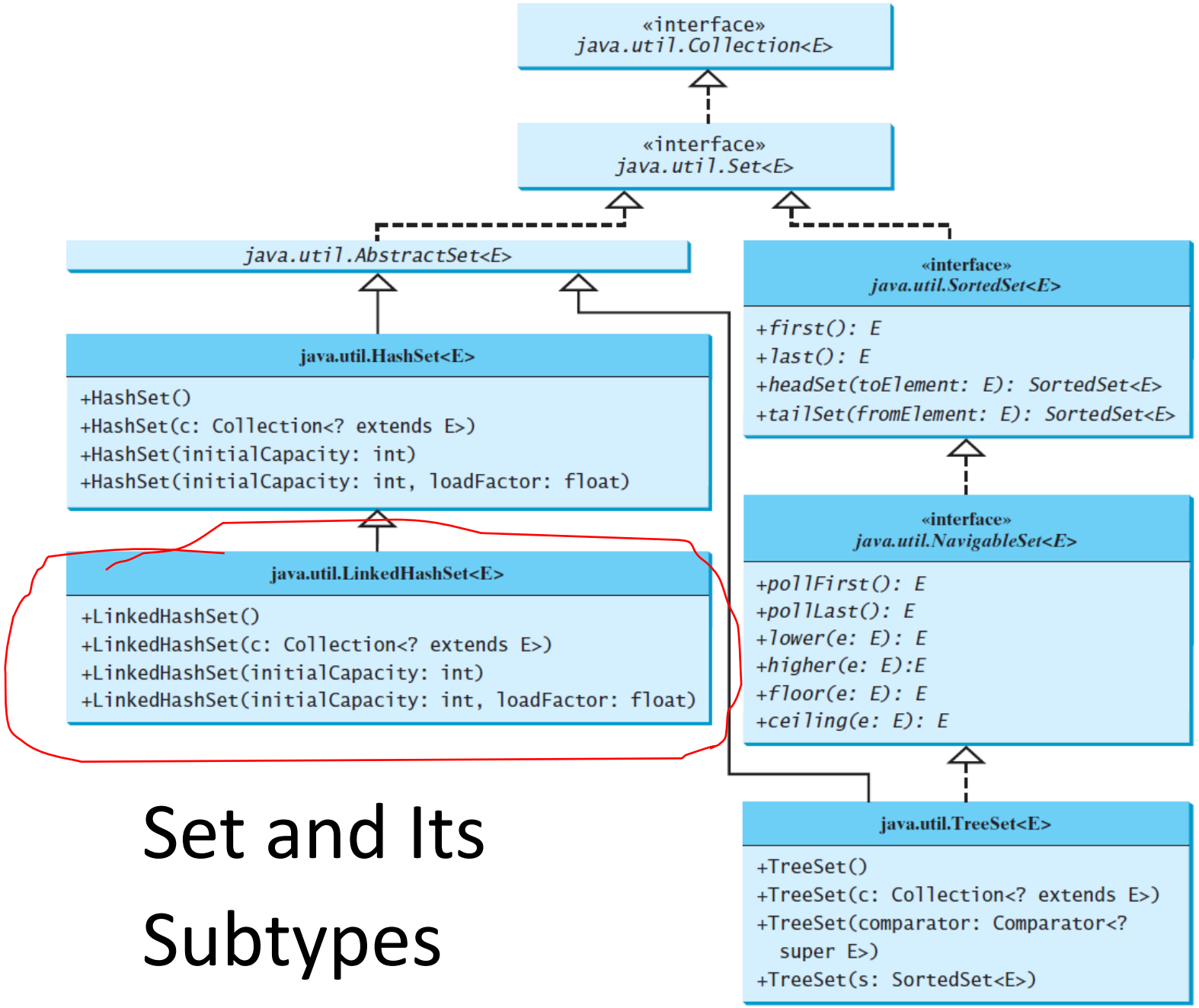
- The AbstractSet class is a convenience class that extends AbstractCollection and implements Set.
- The AbstractSet class provides concrete implementations for the equals method and the hashCode method.
  - You may use it to determine if two sets are equal
  - The hash code of a set is the sum of the hash code of all the elements in the set.
- Since the size method and iterator method are not implemented in the AbstractSet class, AbstractSet is an abstract class



Set and Its Subtypes

HashSet

- The HashSet class is a concrete class that implements Set.
- It can be used to store duplicate-free elements.
- For efficiency, objects added to a hash set need to implement the hashCode method in a manner that properly disperses the hash code
  - Recall the contract of the HashCode, and how the equals method should be overridden.



Set and Its Subtypes

LinkedHashSet

- LinkedHashSet retains the order of elements as they are inserted (insertion order) via a linked list
  - Note that insertion order is not affected if an element is re-inserted into the set.
  - Meaning of re-insertion
    - if (set.contains(element)) { set.add(element); }
- However, in a HashSet, the order of elements are unspecified and generally chaotic.

HashSet: Example

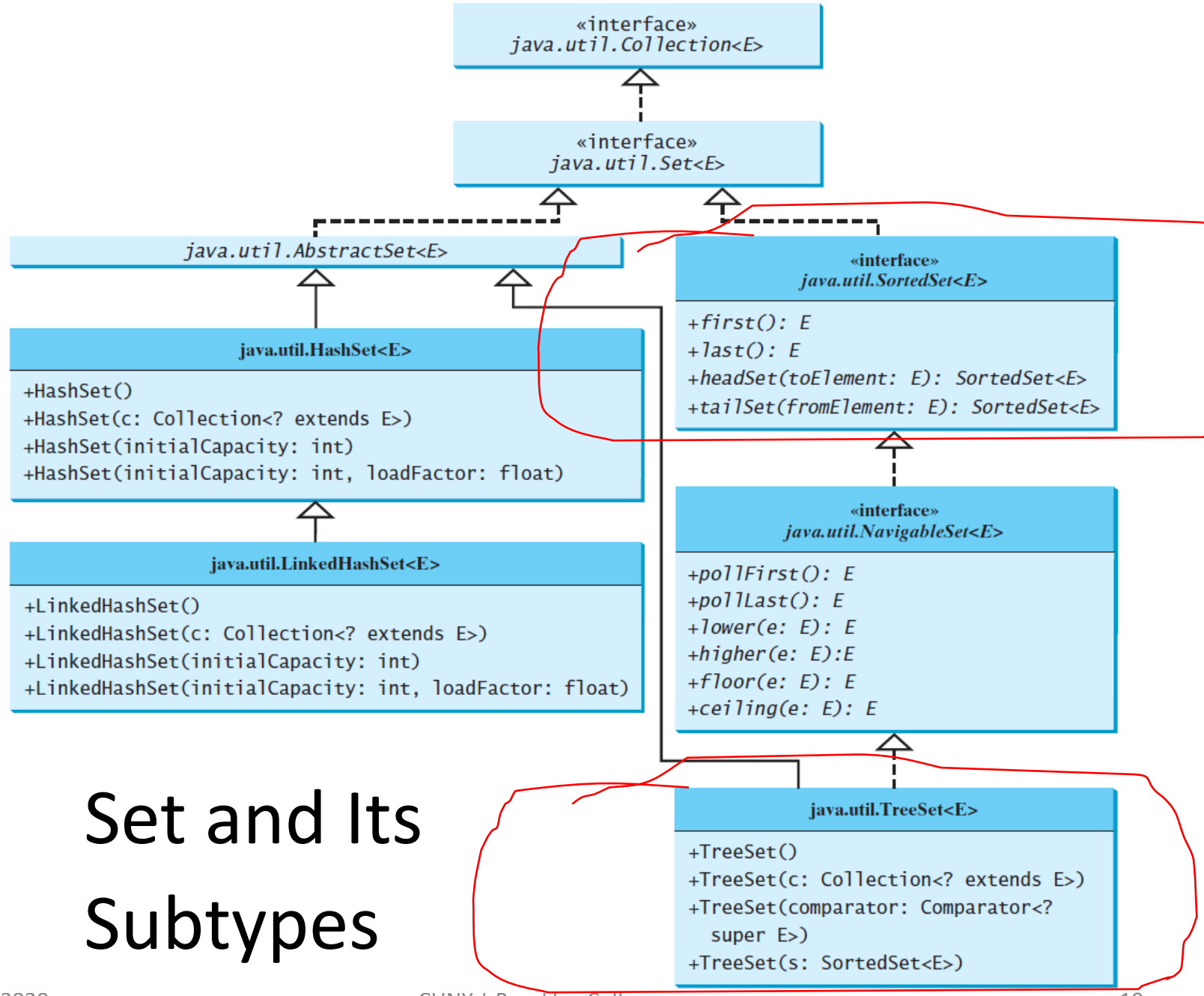
- This example creates a hash set filled with strings, and traverse the elements in the list.
  - Use the enhanced for loop
  - Use the Iterable::forEach method and the Consumer interface
- Note: pay attention to the order of the elements

LinkedHashSet: Example

- This example creates a linked hash set filled with strings, and traverse the elements in the list.
  - Use the enhanced for loop
  - Use the Iterable::forEach method and the Consumer interface
- Note: pay attention to the order of the elements

Questions?

- Concept of set
- Set in Java
- HashSet and LinkedHashSet, and difference?



Set and Its Subtypes

SortedSet and TreeSet

- SortedSet is a subinterface of Set, which guarantees that the elements in the set are sorted.
- TreeSet is a concrete class that implements the SortedSet interface.
  - You can use an iterator to traverse the elements in the sorted order.

Sorting in TreeSet

- The elements can be sorted in two ways
  - One way is to use the Comparable interface, which means elements (objects) needs to be comparable.
  - The other way is to specify a comparator for the elements, i.e., *order by comparator*.
    - if the class for the elements does not implement the Comparable interface, or
    - you don't want to use the compareTo method in the class that implements the Comparable interface

TreeSet: Example 1

- This example creates a hash set filled with strings, and then creates a tree set for the same strings.
  - The strings are sorted in the tree set using the compareTo method in the Comparable interface.

TreeSet: Example 2

- The example also creates a tree set of geometric objects.
  - The geometric objects are sorted using the compare method in the Comparator interface

Questions?

- SortedSet and TreeSet
- Based on what are SortedSet and TreeSet sorted?

List and Set

- Compare List and Set
- How much time does it take to do the following?
  - Test if the collection contains an element
  - Remove a given element
- See the example

Set: Example

- Write an application that counts the number of the keywords in a Java source file

Questions?

- Count keywords using Set