ListIterator: remove and next

**E** it.next(); *//advances the “pointer” to the right and returns what it passed over.*

**E** it.previous(); *//retreats the “pointer” to the left and returns what it passed over.*

**void** it.add(E) *//adds obj before the element that will be returned by next.*

**void** it.remove() *//removes the last element that was returned by next or previous*

If next or previous has not been set, remove throws an IllegalStateException. Predict the state of each iterator below.

1. Given a ListIterator in this initial state: ^A B C D

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.add(″X″); |  |  |
| iter.next(); |  |  |
| iter.remove(); |  |  |
| iter.remove(); |  |  |

1. Given a ListIterator in this initial state: ^A B

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.remove(); |  |  |

The rule is: The remove method throws an IllegalStateException if the next method has not yet been called or if the remove method is called twice after the last call to  next.

1. Given a ListIterator in this initial state: ^B C D , predict each new state of the iterator.

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.add(″A″); |  |  |
| iter.next(); |  |  |
| iter.previous(); |  |  |
| iter.remove(); |  |  |
| iter.next(); |  |  |
| iter.add(″Z″); |  |  |
| iter.remove(); |  |  |

1. Given a ListIterator in this initial state: B^C , predict each new state of the iterator.

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.previous(); |  |  |
| iter.add(″X″); |  |  |
| iter.next(); |  |  |
| iter.next(); |  |  |
| iter.next(); |  |  |

1. What condition causes each Exception:?

IllegalStateException

NoSuchElementException

ConcurrentModificationException