ListIterator: More Practice

**E** it.next();

**E** it.previous();

**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.*

You can only remove after calling next or previous

Sometimes, next or previous has not been set, which causes run-time errors for remove. 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″); | X ^ A B C D | void |
| iter.next(); | X A ^ B C D | A |
| iter.remove(); | X ^ B C D | void |
| iter.remove(); | Illegal State Exception | void |

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

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.add(″A″); | A ^ B C D | void |
| iter.next(); | A B ^ C D | B |
| iter.previous(); | A ^ B C D | B |
| iter.remove(); | A ^ C D | void |
| iter.next(); | A ^ C D | C |
| iter.add(″Z″); | A C Z ^ D | void |
| iter.remove(); | B C D | void |

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

|  |  |  |
| --- | --- | --- |
|  | **new state** | **returns** |
| iter.previous(); | ^ B C | B |
| iter.add(″X″); | X ^ B C | void |
| iter.next(); | X B ^ C | B |
| iter.next(); | X B C ^ | C |
| iter.next(); | NoSuchElementException |  |

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

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

EXCEPTIONS YOU NEED TO KNOW

IllegalStateException: Preconditions of remove are not met, or if you don’t reset next();

When remove(); is called before next(); or previous(); is stored.

NoSuchElementException: When there is no element left when moving either next() or previous()/ no element where the pointer is at.

ConcurrentModificationException: When directly editing a list while using an iterator or a for-each loop.