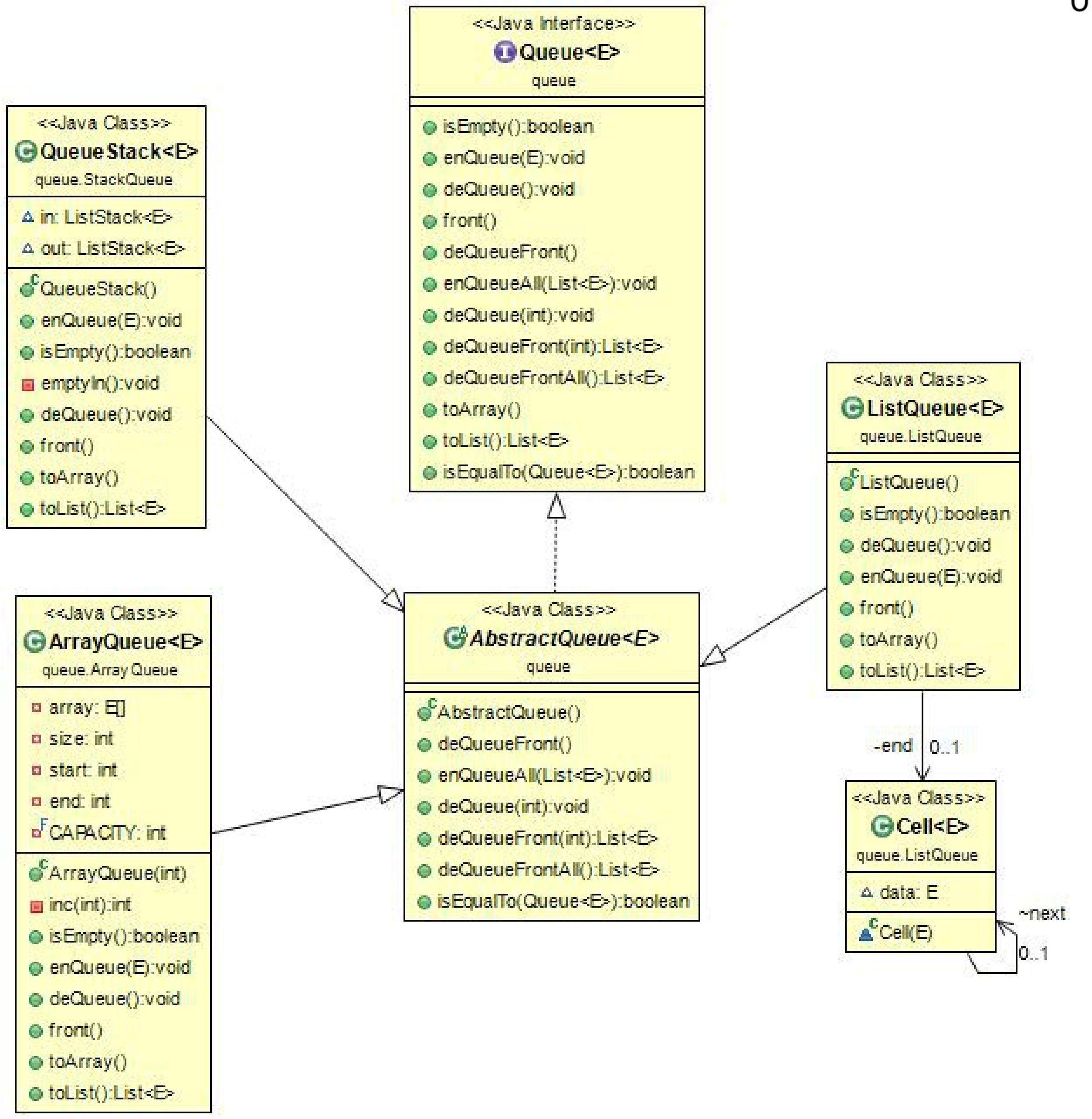
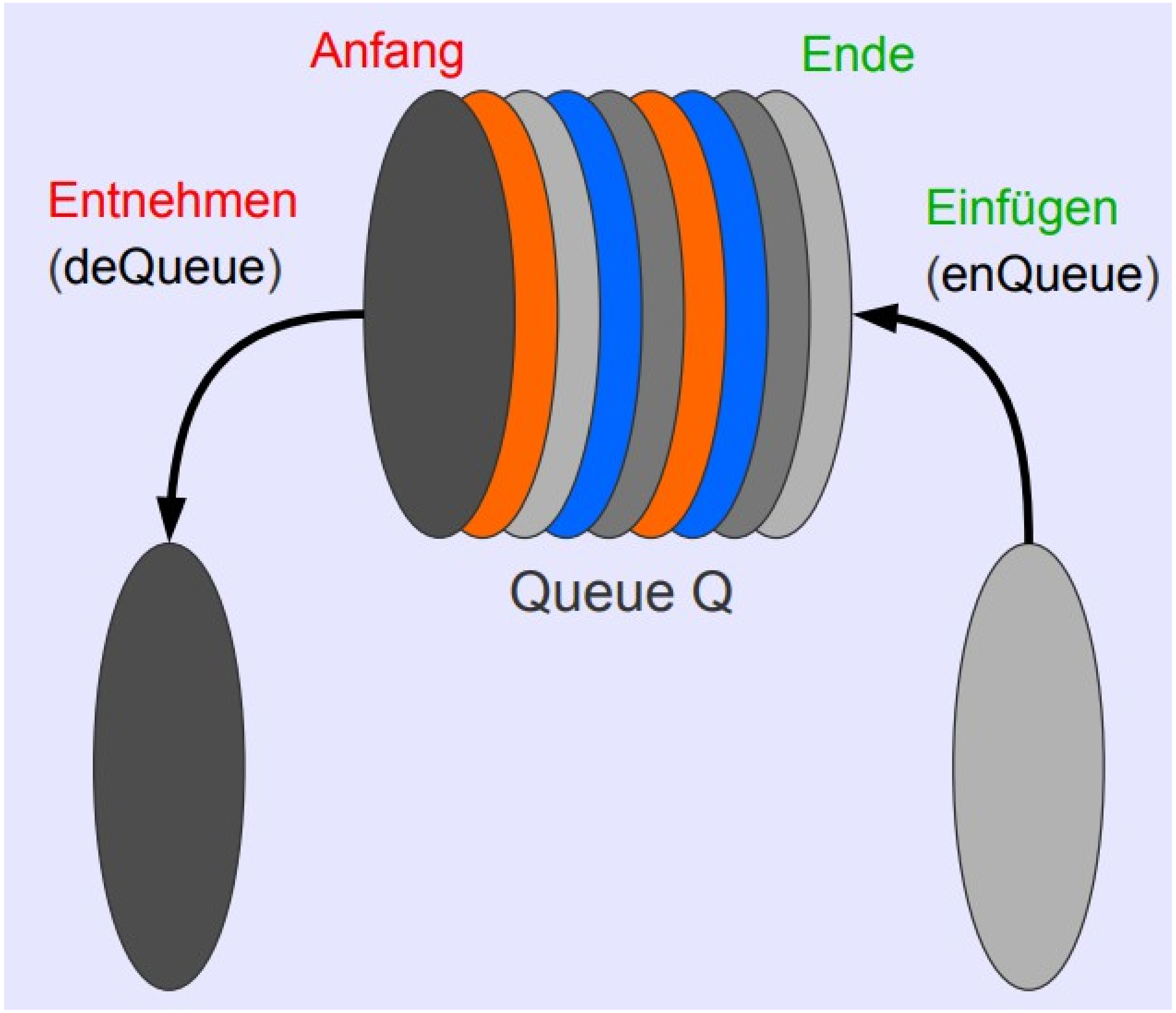
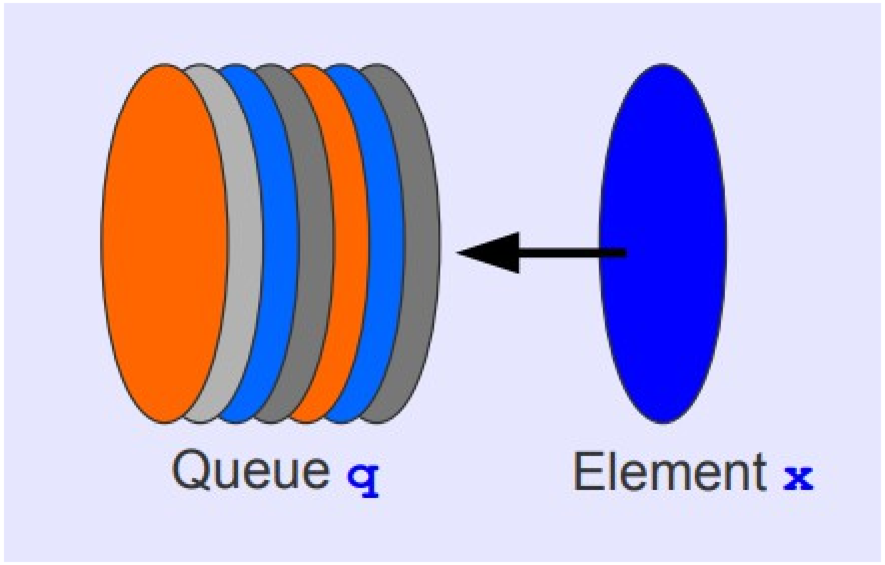
Informatik II

FACH HOCHSCHULE LÜBECK
University of Applied Sciences

Queue
Thomas Jürgensen

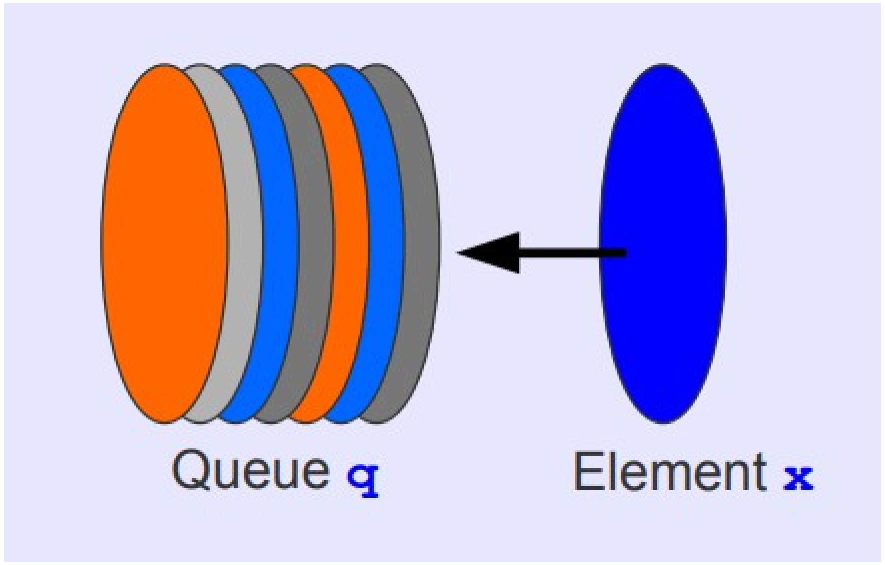

<<Java Interface>>
Queue<E>
queue
isEmpty():boolean
enQueue(E):void
deQueue():void
front()
deQueueFront()
enQueueAll(List<E>):void
deQueue(int):void
deQueueFront(int):List<E>
deQueueFrontAll():List<E>
toArray()
toList():List<E>
isEqualTo(Queue<E>):boolean
<<Java Class>>
QueueStack<E>
queue.StackQueue
in: ListStack<E>
out: ListStack<E>
QueueStack()
enQueue(E):void
isEmpty():boolean
emptyIn():void
deQueue():void
front()
toArray()
toList():List<E>
<<Java Class>>
ListQueue<E>
queue.ListQueue
ListQueue()
isEmpty():boolean
deQueue():void
enQueue(E):void
front()
toArray()
toList():List<E>
<<Java Class>>
AbstractQueue<E>
queue
AbstractQueue()
deQueueFront()
enQueueAll(List<E>):void
deQueue(int):void
deQueueFront(int):List<E>
deQueueFrontAll():List<E>
isEqualTo(Queue<E>):boolean
<<Java Class>>
ArrayQueue<E>
queue.ArrayQueue
array: E[]
size: int
start: int
end: int
CAPACITY: int
ArrayQueue(int)
inc(int):int
isEmpty():boolean
enQueue(E):void
deQueue():void
front()
toArray()
toList():List<E>
-end
0..1
<<Java Class>>
Cell<E>
queue.ListQueue
data: E
Cell(E)
~next
0..1

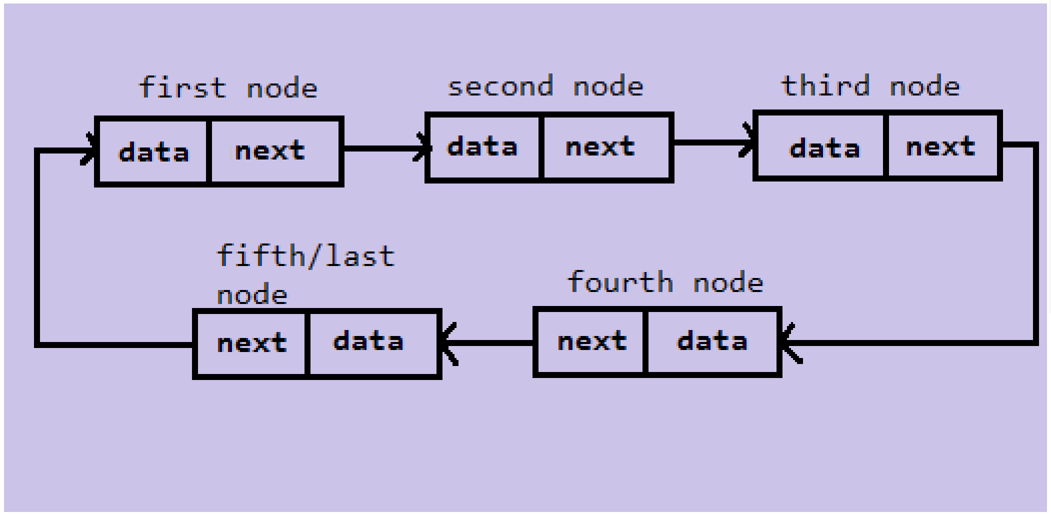

Anfang
Ende
Entnehmen
(deQueue)
Einfügen
(enQueue)
Queue Q

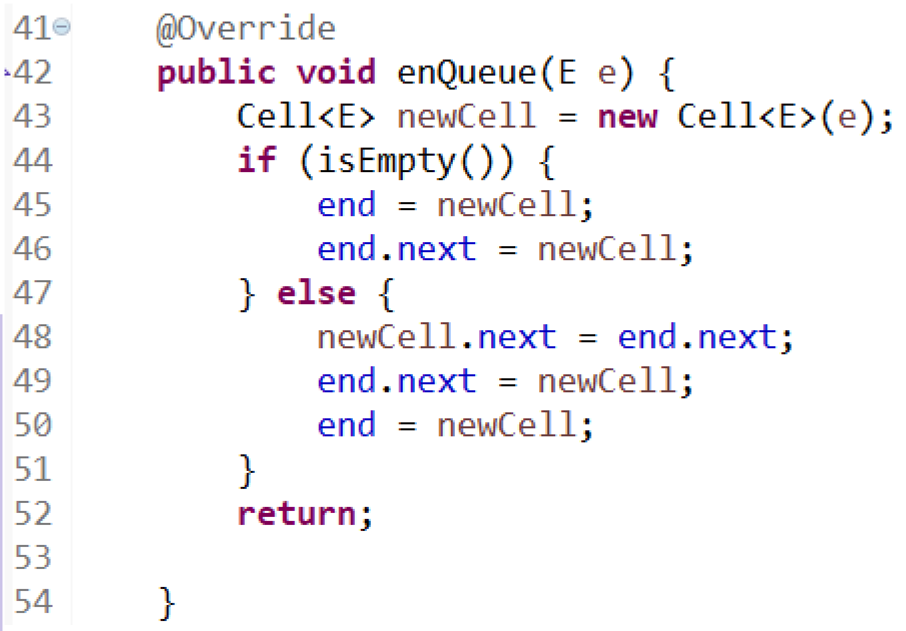


FACH HOCHSCHULE LÜBECK
University of Applied Sciences

FACH
HOCHSCHULE
LÜBECK
University of Applied Sciences

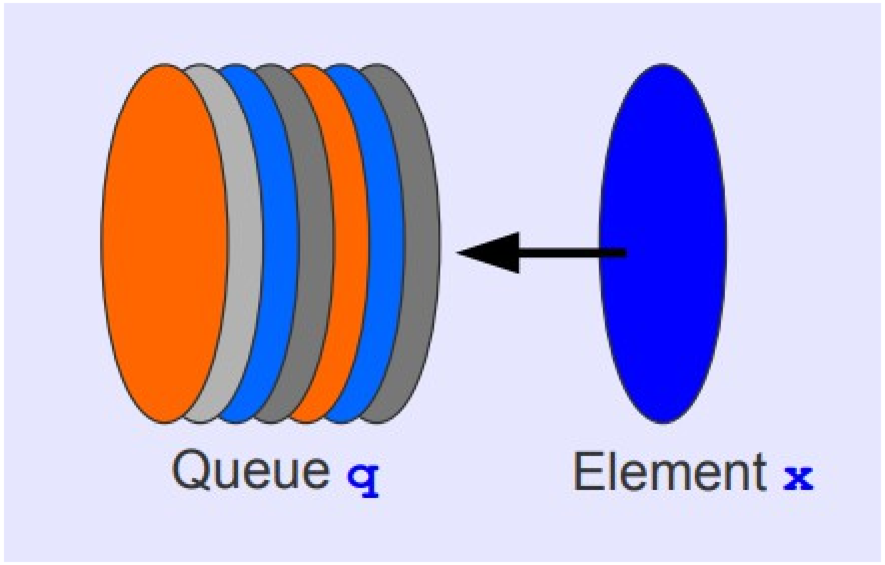


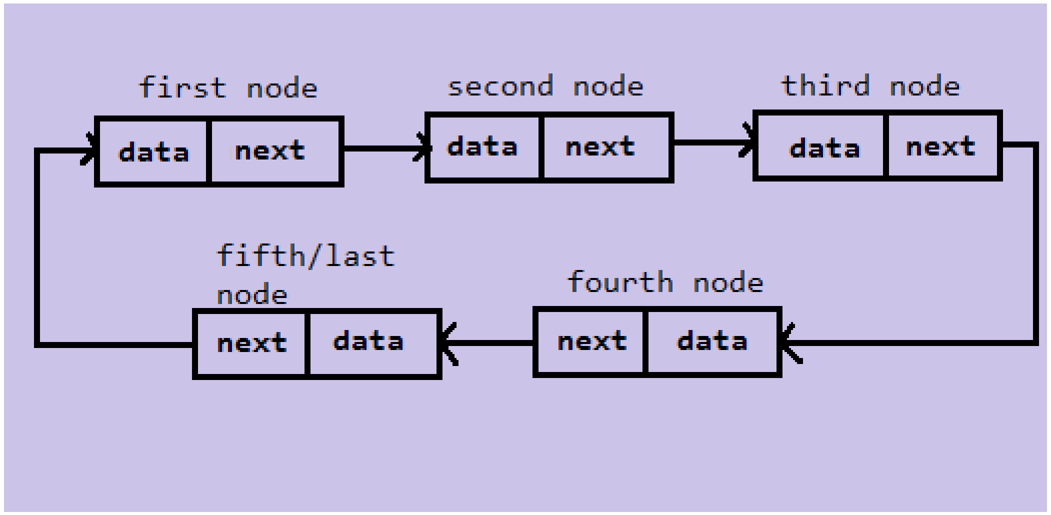




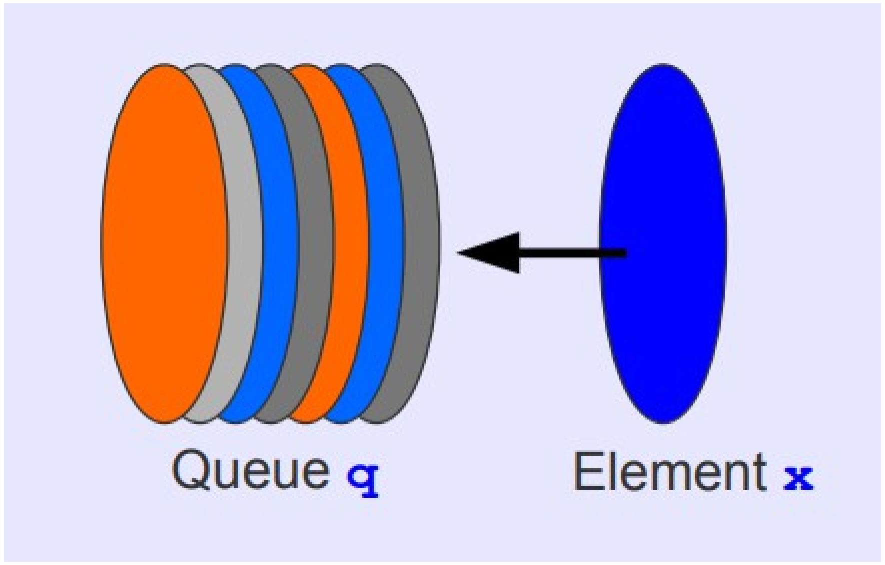
```java
    @Override
    public void enQueue(E e) {
        Cell<E> newCell = new Cell<E>(e);
        if (isEmpty()) {
            end = newCell;
            end.next = newCell;
        } else {
            newCell.next = end.next;
            end.next = newCell;
            end = newCell;
        }
        return;

    }
```

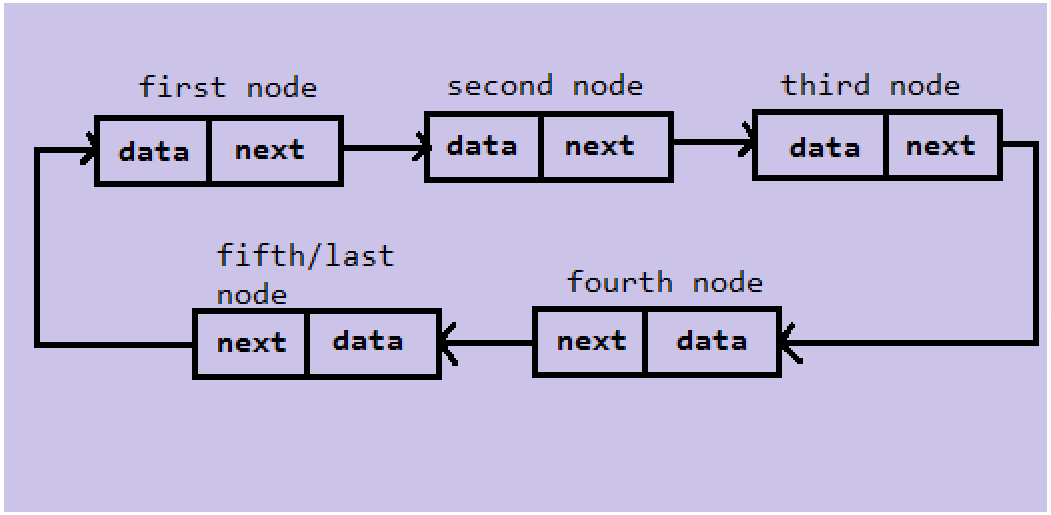
FACH HOCHSCHULE LÜBECK
University of Applied Sciences

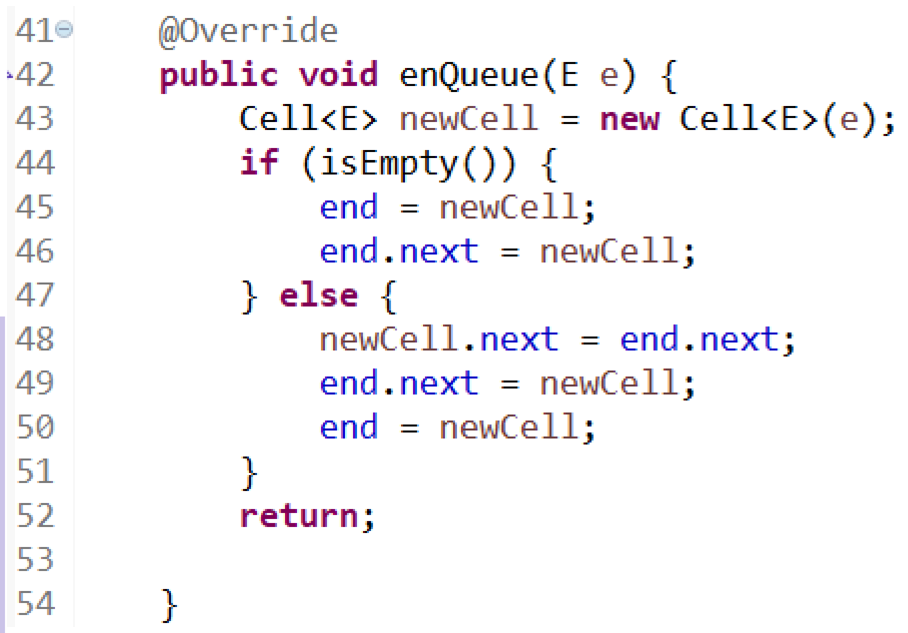




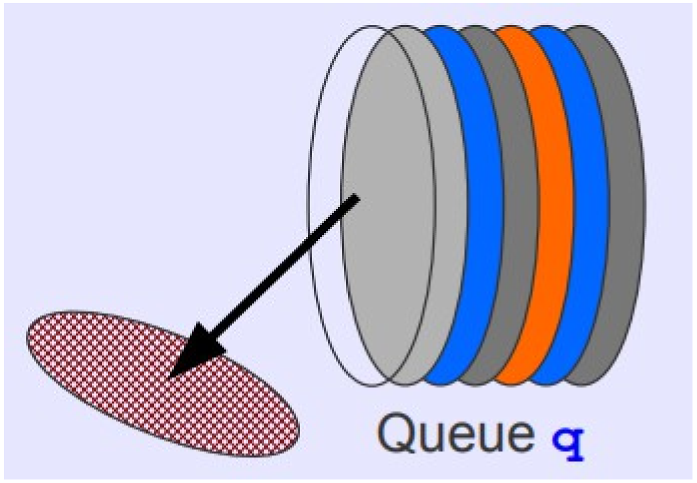
FACH HOCHSCHULE LÜBECK
University of Applied Sciences

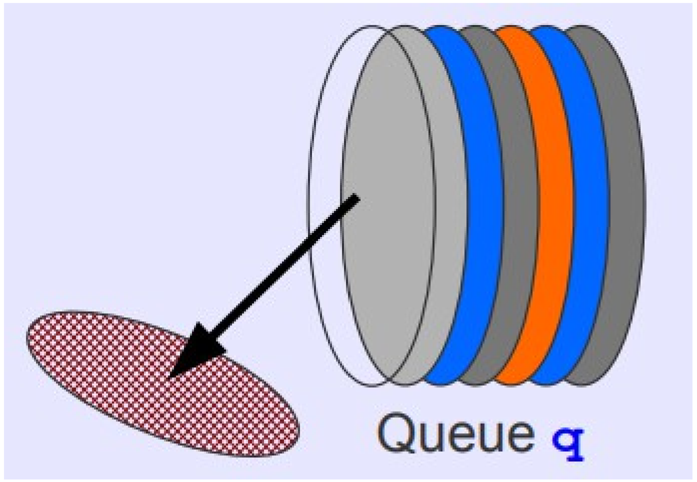


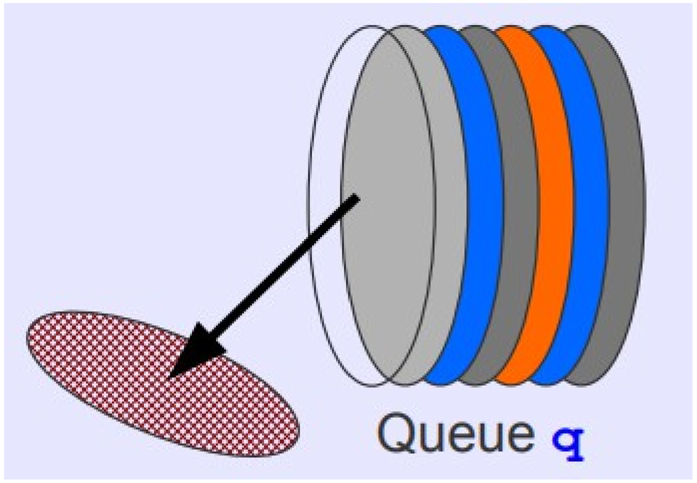


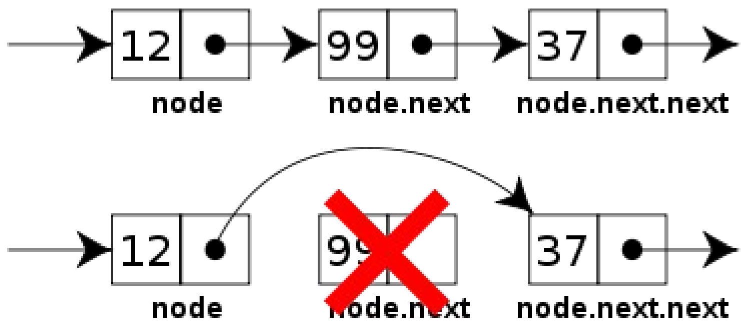


```java
    @Override
    public void enQueue(E e) {
        Cell<E> newCell = new Cell<E>(e);
        if (isEmpty()) {
            end = newCell;
            end.next = newCell;
        } else {
            newCell.next = end.next;
            end.next = newCell;
            end = newCell;
        }
        return;

    }
```

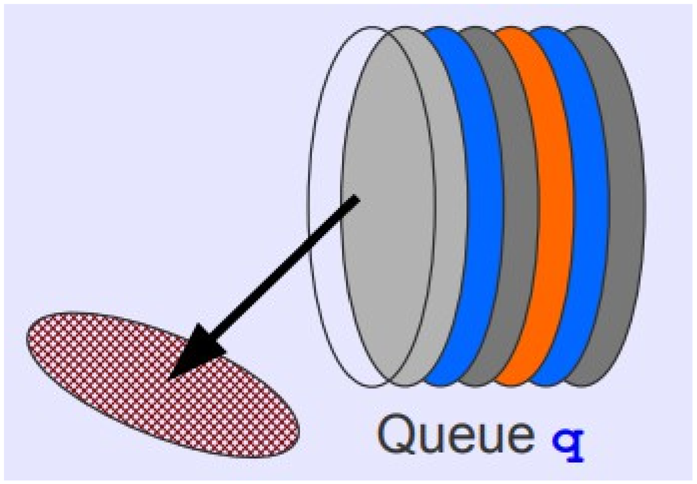

Queue q

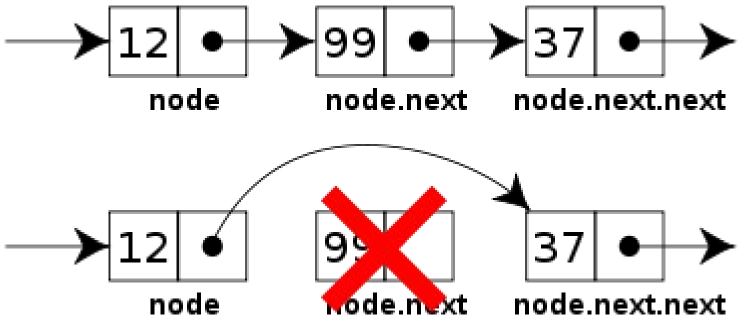

Queue q


Queue q

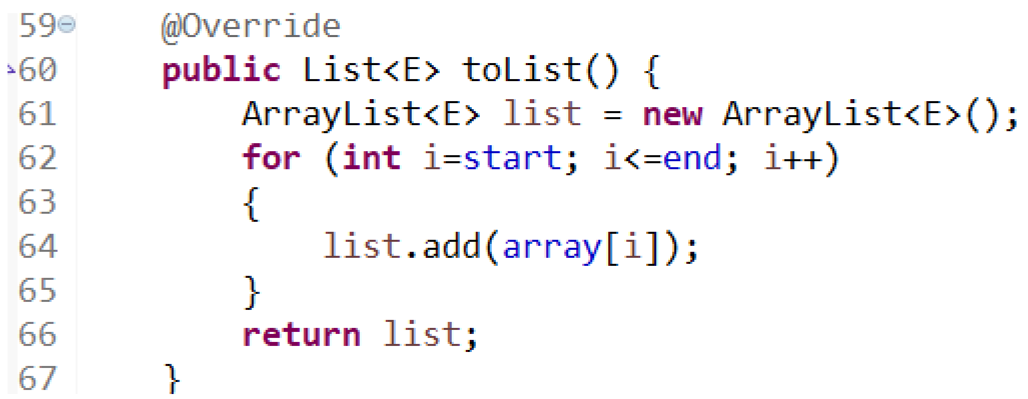

12
99
37
node
node.next
node.next.next
12
99
37
node
node.next
node.next.next

FACH HOCHSCHULE LÜBECK
University of Applied Sciences

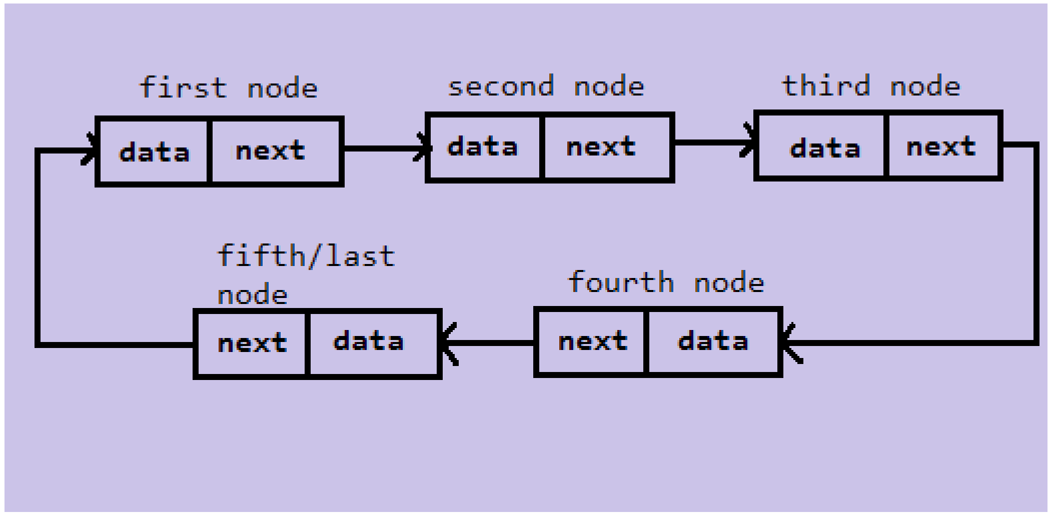




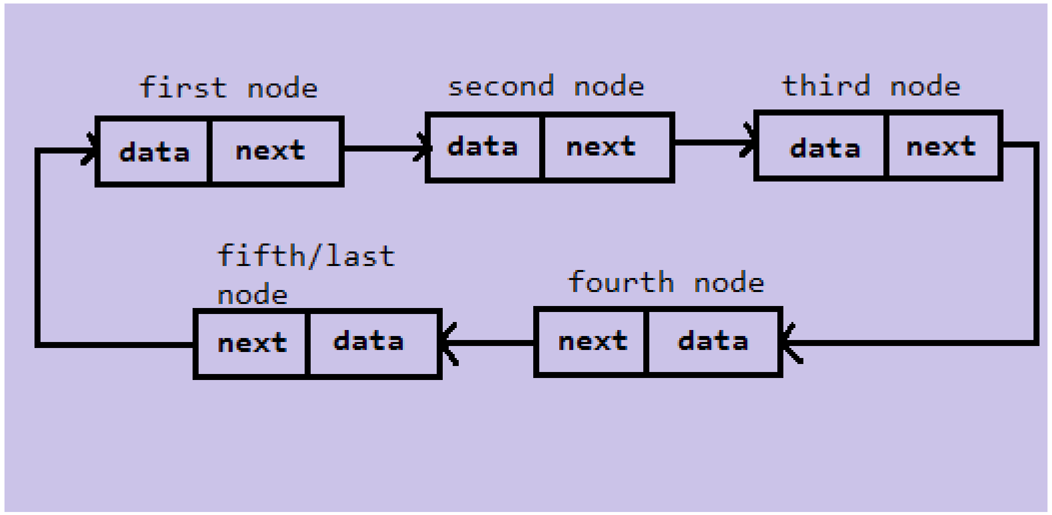
```java
    public void deQueue() {
        if (isEmpty())
            throw new QueueError("Queue is empty!");
        if (end.next == end) {
            end = null;
        } else {
            end.next = end.next.next;
        }
        return;
    }
```



```java
    @Override
    public List<E> toList() {
        ArrayList<E> list = new ArrayList<E>();
        for (int i=start; i<=end; i++)
        {
            list.add(array[i]);
        }
        return list;
    }
```

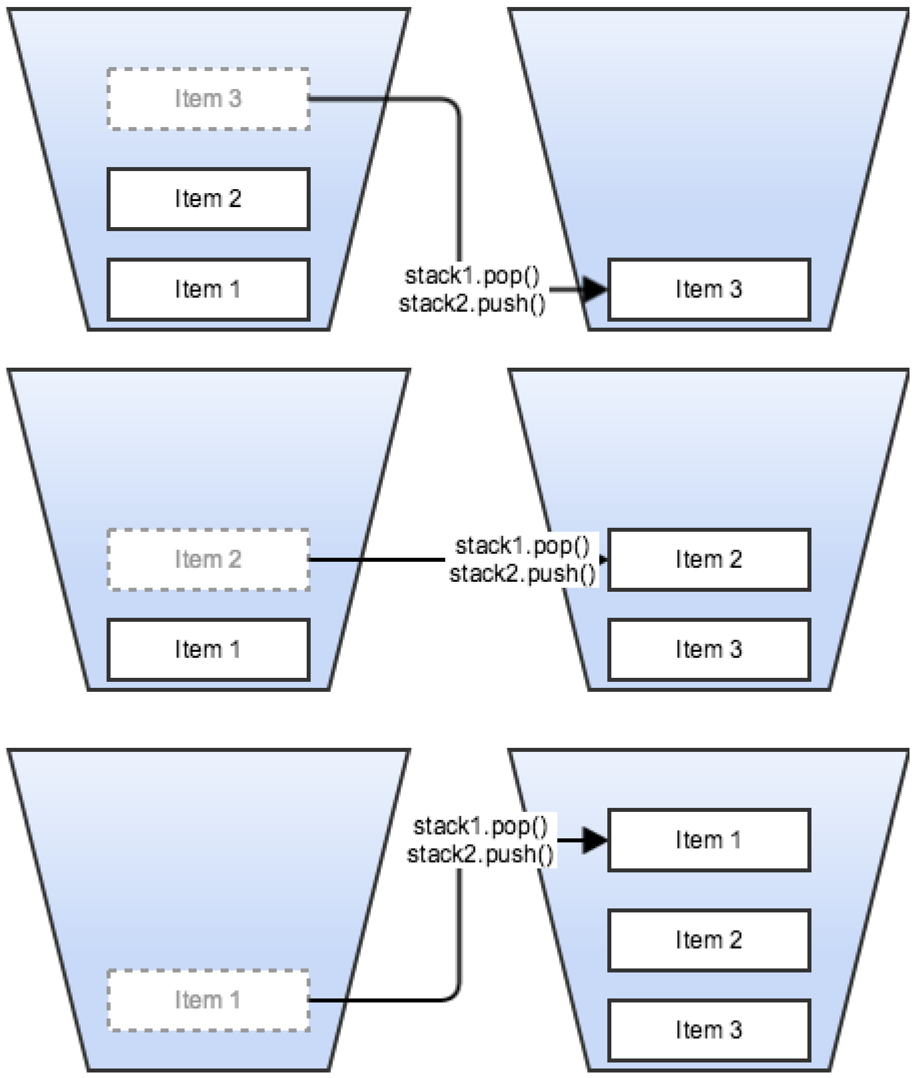

first node
data
next
second node
data
next
third node
data
next
fifth/last node
next
data
fourth node
next
data

FACH HOCHSCHULE LÜBECK
University of Applied Sciences



```java
@Override
public List<E> toList() {
    List<E> list = new LinkedList<E>();
    if (!isEmpty()) {
        Cell<E> tmp = end;
        while (tmp.next != end) {
            list.add(tmp.next.data);
            tmp = tmp.next;

        }
        // add last (missing) element
        list.add(tmp.next.data);
    }
    return list;
}
```

FACH
HOCHSCHULE
LÜBECK
University of Applied Sciences


Item 3
Item 2
Item 1
stack1.pop()
stack2.push()
Item 3
Item 2
Item 1
stack1.pop()
stack2.push()
Item 2
Item 3
Item 1
stack1.pop()
stack2.push()
Item 1
Item 2
Item 3