# Lösung Selbsttest (1)

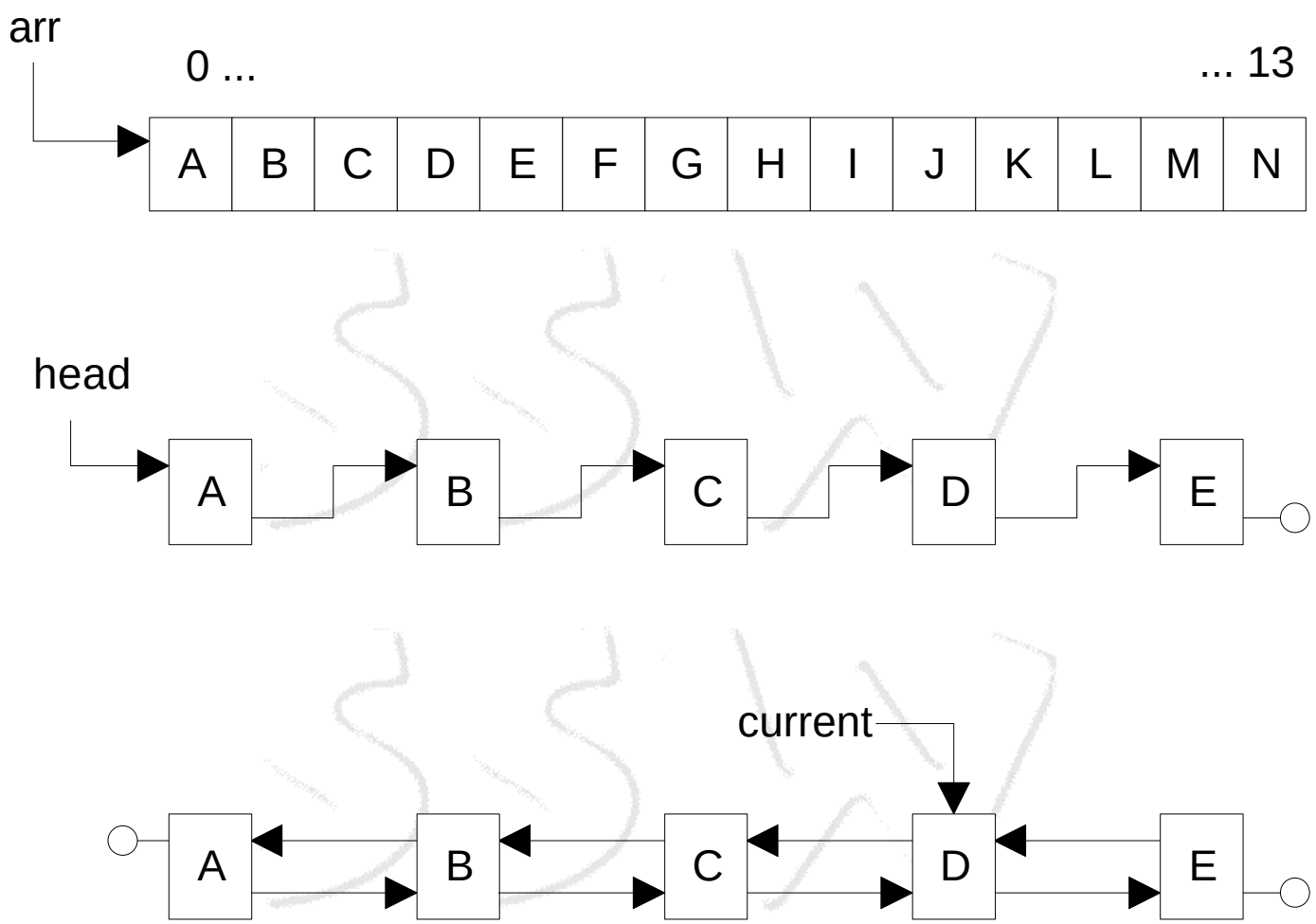
```
public class Queue { // Kein Formatierungsvorbild!!!
private QNode head, tail;
private int nOfElems;

public Queue() {
  head = new QNode(0);
  tail = head;
  nOfElems = 0;
}
public void insert(int val) {
  QNode p = new QNode(val);
  tail.next = p;
  tail = p;
  nOfElems ++;
}
public boolean hasNext() {
  return nOfElems != 0;
//return head != tail;
}
```

# Lösung Selbsttest (2)

```
public int remove() {
  if (head != tail) {
    head = head.next;
    nOfElems --;
    return head.val;
  } else {
    throw new NoSuchElementException("Queue empty");
  }
}
public int getSize() {
  return nOfElems;
}
public void clear() {
  head = tail;
  nOfElems = 0;
}
}
```

# Array vs. Listen



# Auto-Resize bei Queue (Array) 1

```
public void insert(int x) {
  queue[tail] = x;
  int tail1 = (tail + 1) % queue.length;
  if (tail1 == head) { // resize variant 1
    int[] newQueue = new int[queue.length * 2];
    int j = 0;
    for (int i = head; i < queue.length; i ++) {
      newQueue[j] = queue[i];
      j ++;
    }
    for (int i = 0; i < tail1; i ++) {
      newQueue[j] = queue[i];
      j ++;
    }
    tail = j;
    head = 0;
    queue = newQueue;
  }
  else {
    tail = tail1;
  }
}
```

# Auto-Resize bei Queue (Array) 2

```
public void insert(int x) {
  queue[tail] = x;
  int tail1 = (tail + 1) % queue.length;
  if (tail1 == head) { // resize variant 2
    int[] newQueue = new int[queue.length * 2];
    System.arraycopy(queue, head,
                     newQueue, 0, queue.length - head);
    System.arraycopy(queue, 0,
                     newQueue, head, tail1);
    tail = queue.length;
    head = 0;
    queue = newQueue;
  }
  else {
    tail = tail1;
  }
}
```