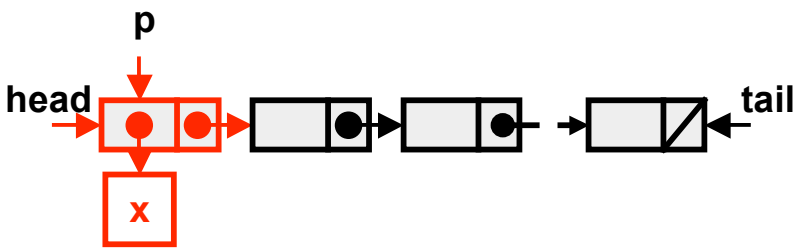
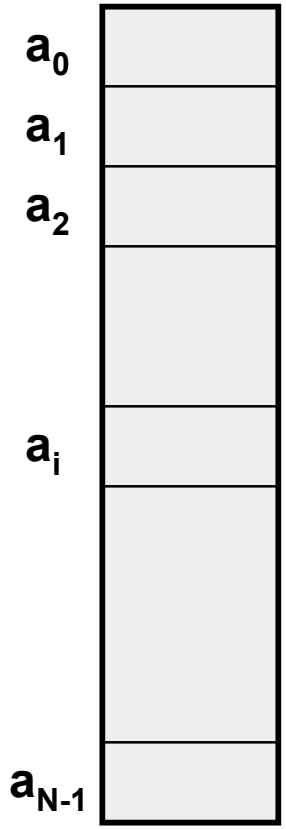
Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich



Listen, Stacks und Queues

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich



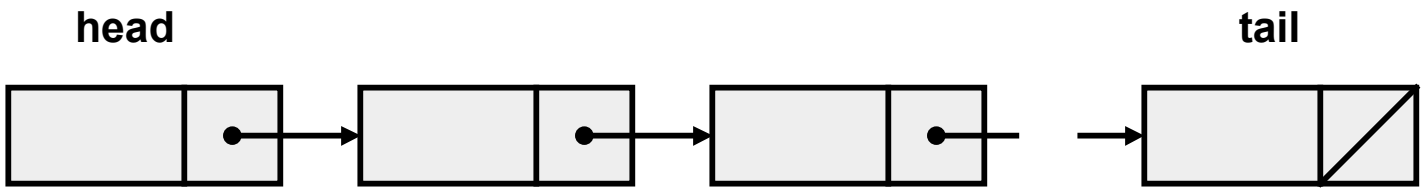
Arrays

- statisch
- wahlfreier Zugriff

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked Lists

- dynamisch
- sequentieller Zugriff



Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked List in Java (1)

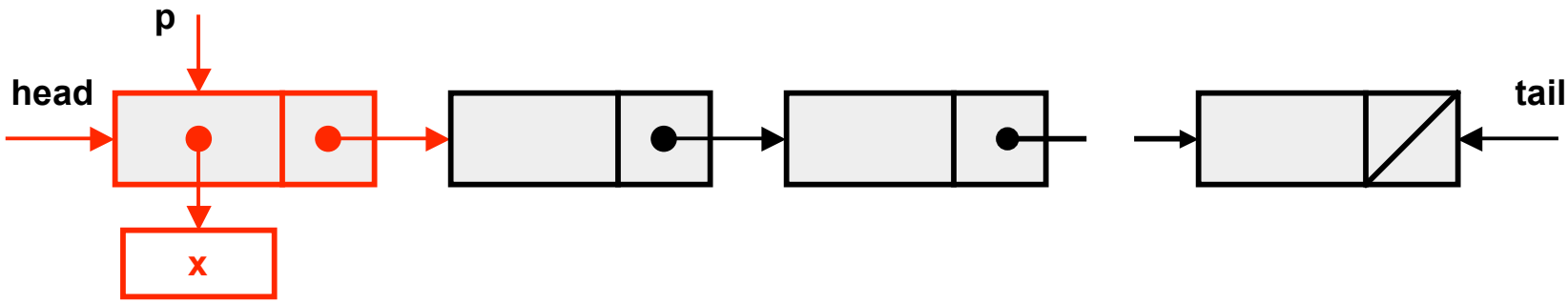
```
class LinkedList {
   class Node {
     Object info;
     Node next;
     Node(Object x) {info = x;}
   }

   Node head, tail;
   LinkedList() {head = null; tail = null;}
   boolean isEmpty() {return head == null;}
   ...
}
```

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked List in Java (2)

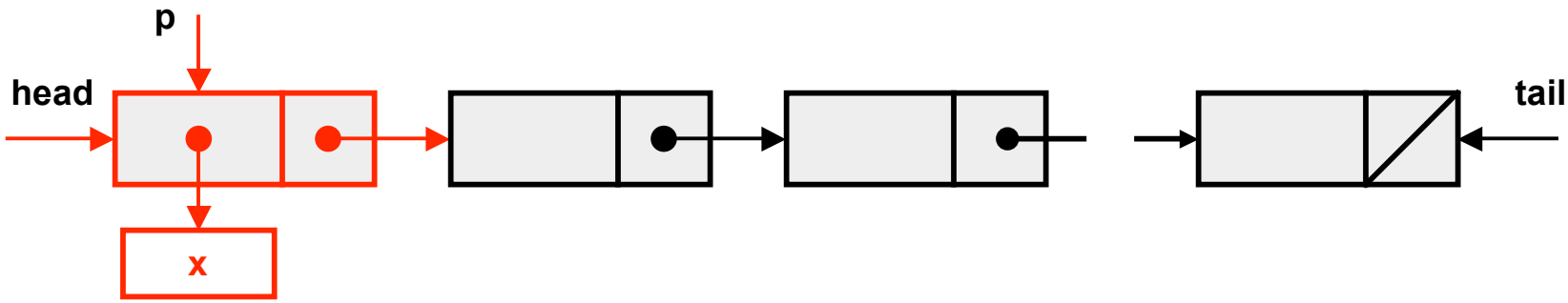
```
void insertFirst(Object x) {
   Node p = new Node(x);
   if (isEmpty()) tail = p;
   else p.next = head;
   head = p;
}
```



Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked List in Java (2)

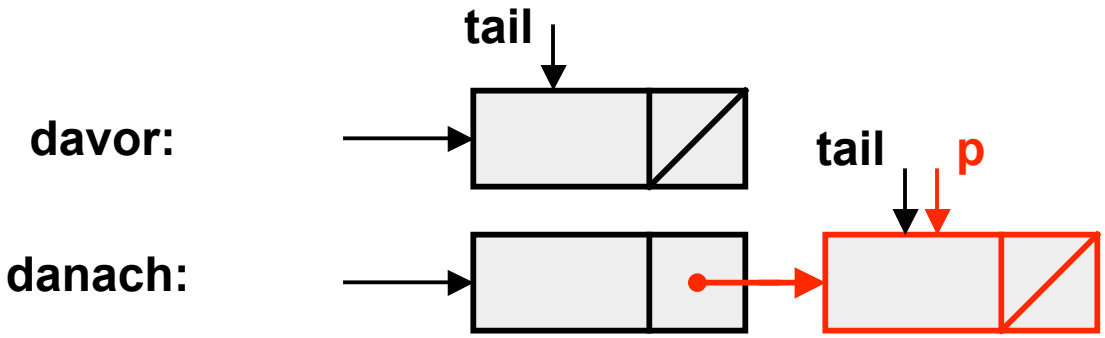
```
Object removeFirst() {
   if (isEmpty()) return null;
   Node p = head;
   head = p.next;
   if (isEmpty()) tail = null;
   return p.info;
}
```



Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked List in Java (3)

```
void insertLast(Object x) {
  if (isEmpty())
    insertFirst(x);
  else {
    Node p = new Node(x);
    tail.next = p;
    tail = p;
  }
}
```



Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Linked List in Java (4)

```
int size() {
    int count = 0;
    for (Node p = head; p != null; p = p.next)
      count++;
    return count;
}

boolean contains(Object x) {
    for (Node p = head; p != null; p = p.next)
      if (p.info.equals(x)) return true;
    return false;
}
```

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Adapter für String List

```
class StringList {
   private LinkedList list;
   StringList() {list = new LinkedList();}
   boolean isEmpty() {return list.isEmpty();}
   void insertFirst(String x) {list.insertFirst(x);}
   String removeFirst() {return (String)list.removeFirst();}
   ...
}
```

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Stack in Java

```
class Stack extends LinkedList {
   Stack() {super();}
   void push(Object x) {insertFirst(x);}
   Object pop() {return removeFirst();}
   Object top() {
     if (isEmpty())
       return null;
     else
       return head.info;
   }
}
```

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Ordered List in Java

```
class OrderedList extends LinkedList {
   OrderedList() {super();}
   void insert(Comparable x) {
     if (isEmpty() || (x.compareTo((Comparable) head.info)<0))
       insertFirst(x);
     else {
       Node p = head;
       while((p.next != null) && (x.compareTo((Comparable) p.next.info)>=0))
         p = p.next;
       if (p.next == null)
         insertLast(x);
       else {
         Node q = new Node(x);
         q.next = p.next;
         p.next = q;
       }
     }
   }
}
```

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Assertions für Stacks

für jeden Stack s und jedes Objekt x muss gelten:

s = new Stack(); {s.size() = 0, s.isEmpty() = true}
{s.size() = n} s.push(x); {s.size() = n+1, s.isEmpty() = false}
s.push(x); {s.top() = x, s.pop() = x}
{P} s.push(x); s.pop(); {P}

Helmut Schauer
Educational Engineering Lab
Department for Informatics
University of Zurich

Assertions für Queues

für jede Queue q und jedes Objekt x muss gelten:

q = new Queue(); {q.size() = 0, q.isEmpty() = true}
{q.size() = n} q.put(x); {q.size() = n+1, q.isEmpty() = false}

({Q, not q.isEmpty()} q.get(); q.put(x); {R}) impliziert
({Q, not q.isEmpty()} q.put(x); q.get(); {R}) und umgekehrt