Humboldt
Universität
Informatik

Übung
Algorithmen & Datenstrukturen

SS 2016

Kim Völlinger

Vorbereitung heute…

…zum 3. Übungsblatt

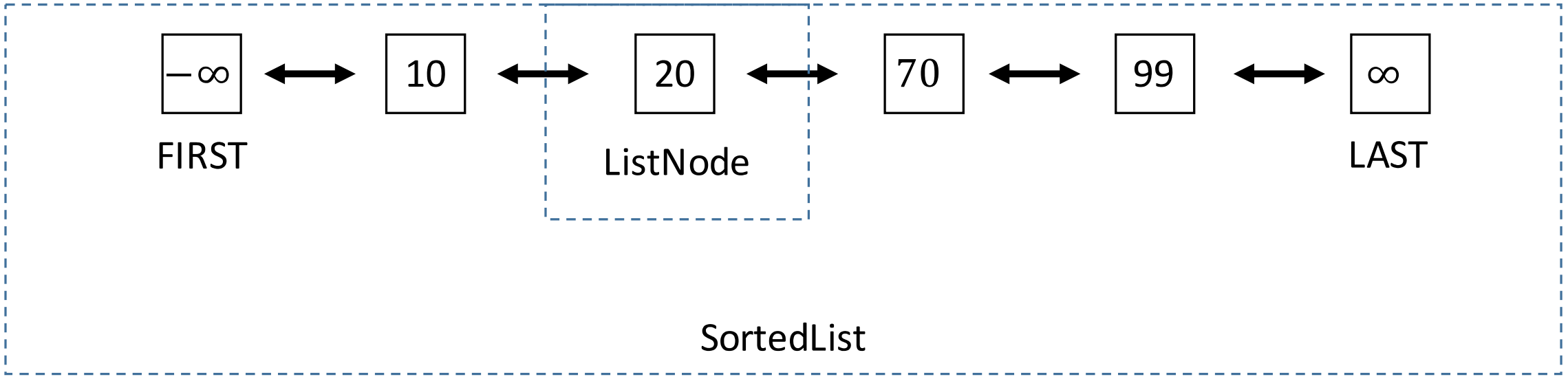
- Skip-Listen
- Listen
- Dynamische Programmierung
- Suchen

Vorbereitung heute…

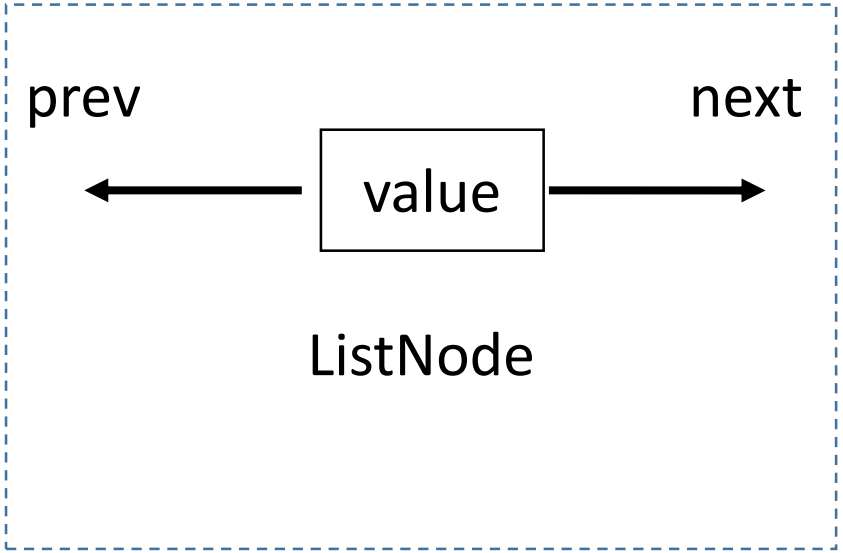
…zum 3. Übungsblatt

- Skip-Listen
- Listen
- Dynamische Programmierung
- Suchen

Doppelt verkettete sortierte Liste

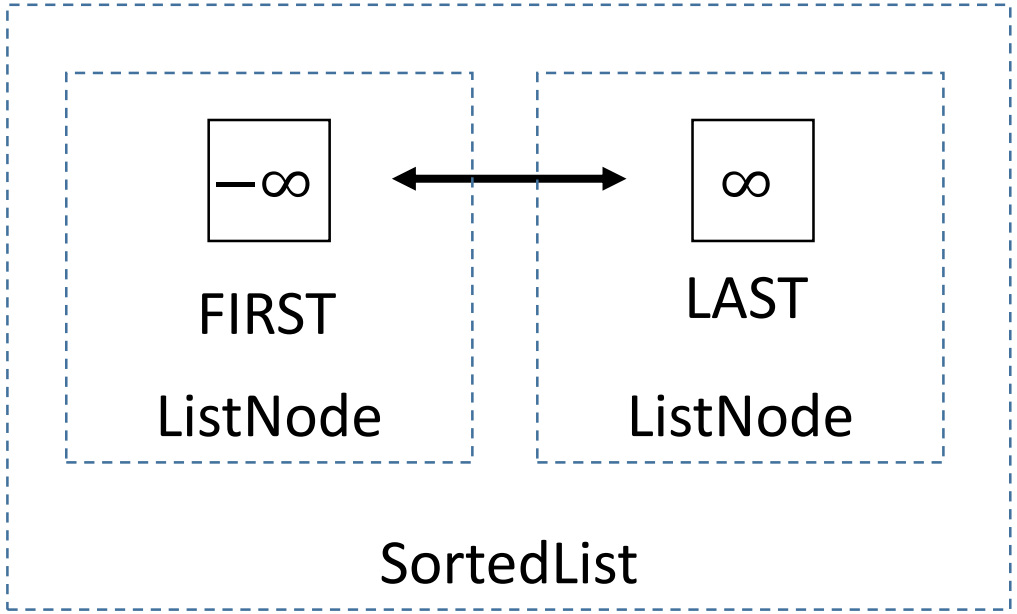


Knoten: ListNode



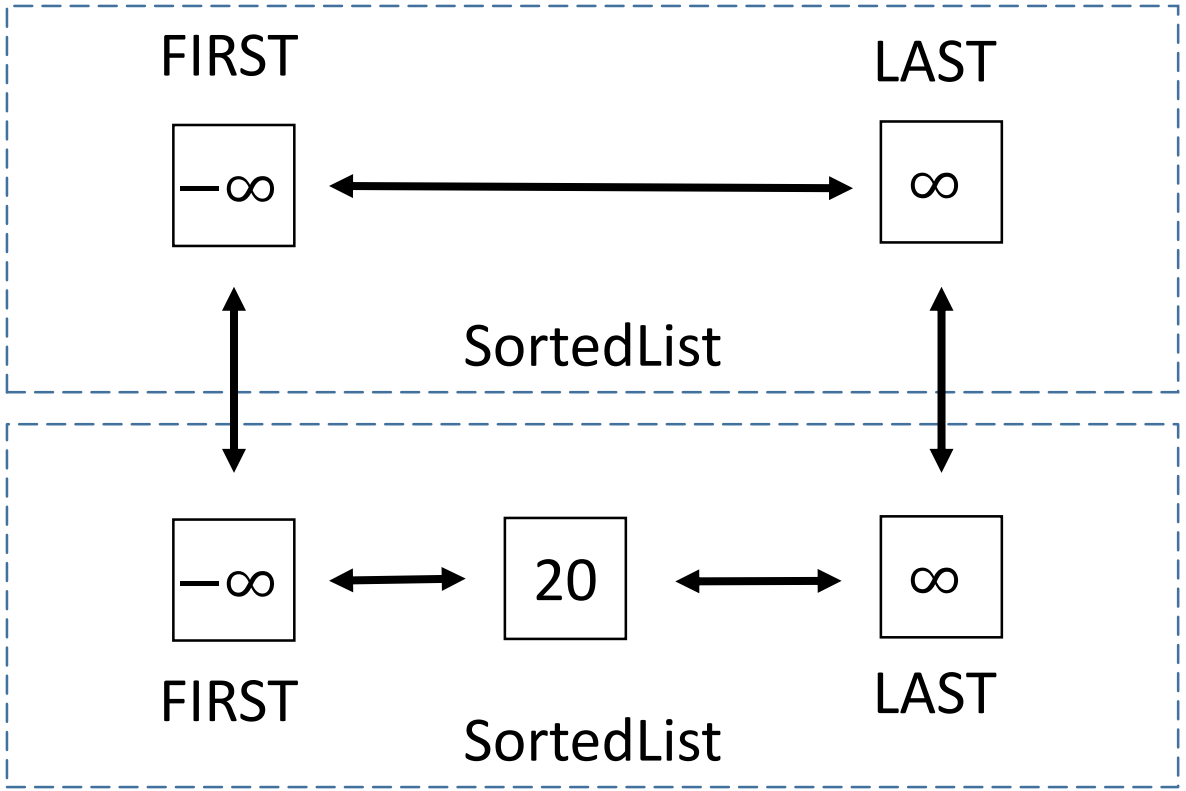

```
public class ListNode {
  // Der Wert des Knoten
  public int value;

  // Zeiger auf den Nachfolger
  // und den Vorgänger
  public ListNode next;
  public ListNode prev;
}
```

Doppelt verkettete sortierte Liste: SortedList



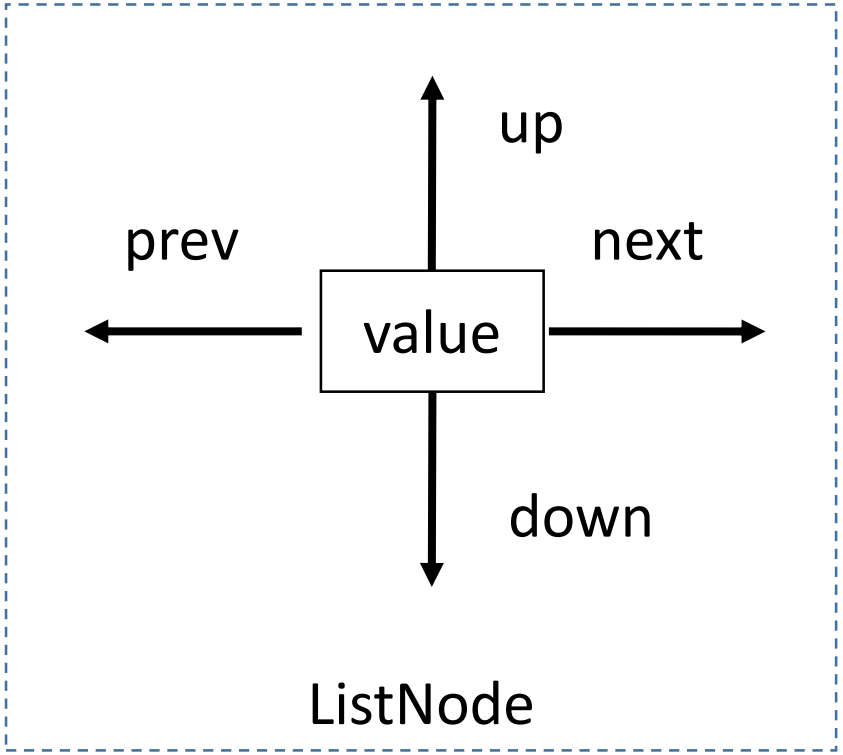
- Die leere Liste enthält den Start- und Endknoten.

Skip-Liste: SkipList



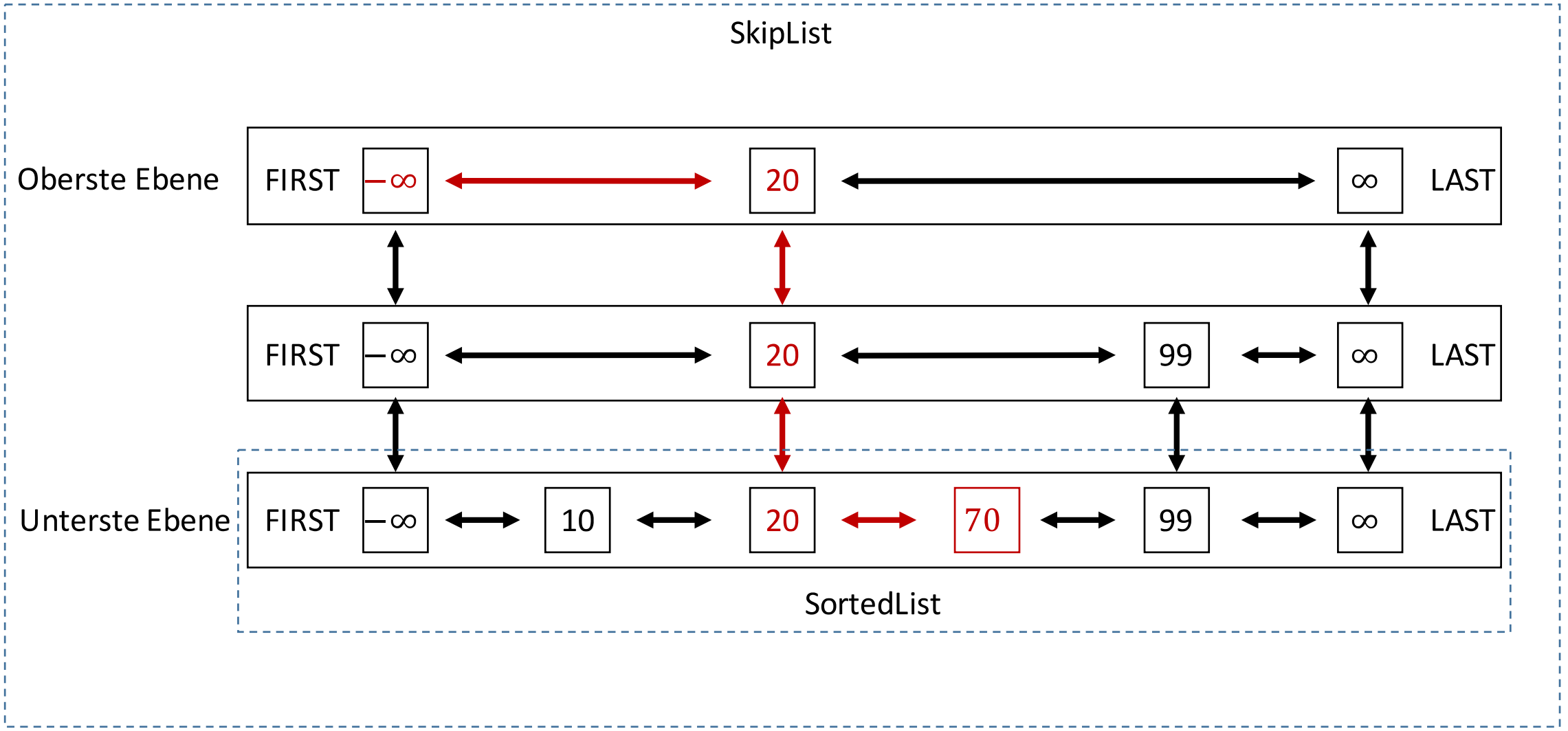
- Jede Ebene repräsentiert eine sortierte doppelt verkettete Liste.
- Kommt ein Knoten auf einer oberen Ebene vor, so kommt er auch auf allen darunter liegenden Ebenen vor.
- Alle Knoten kommen auf der untersten Ebene vor.

Knoten: ListNode

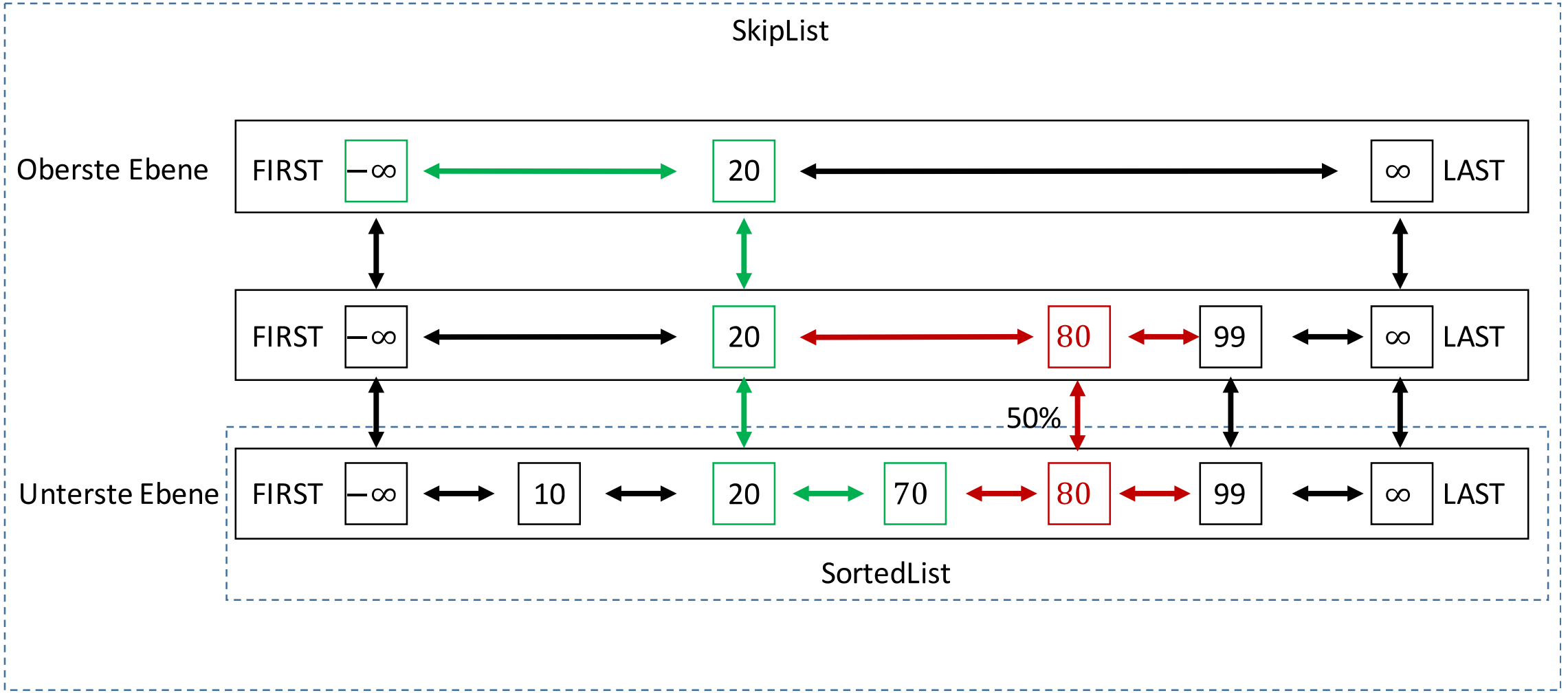



```
public class ListNode {
  // Der Wert des Knoten
  public int value;

  // Zeiger auf den Nachfolger
  // und den Vorgänger
  public ListNode next;
  public ListNode prev;

  // Zeiger auf die obere
  // und untere Ebene
  public ListNode up;
  public ListNode down;
}
```

Skip-Listen: Suchen



Skip-Listen: Einfügen



Vorbereitung heute…

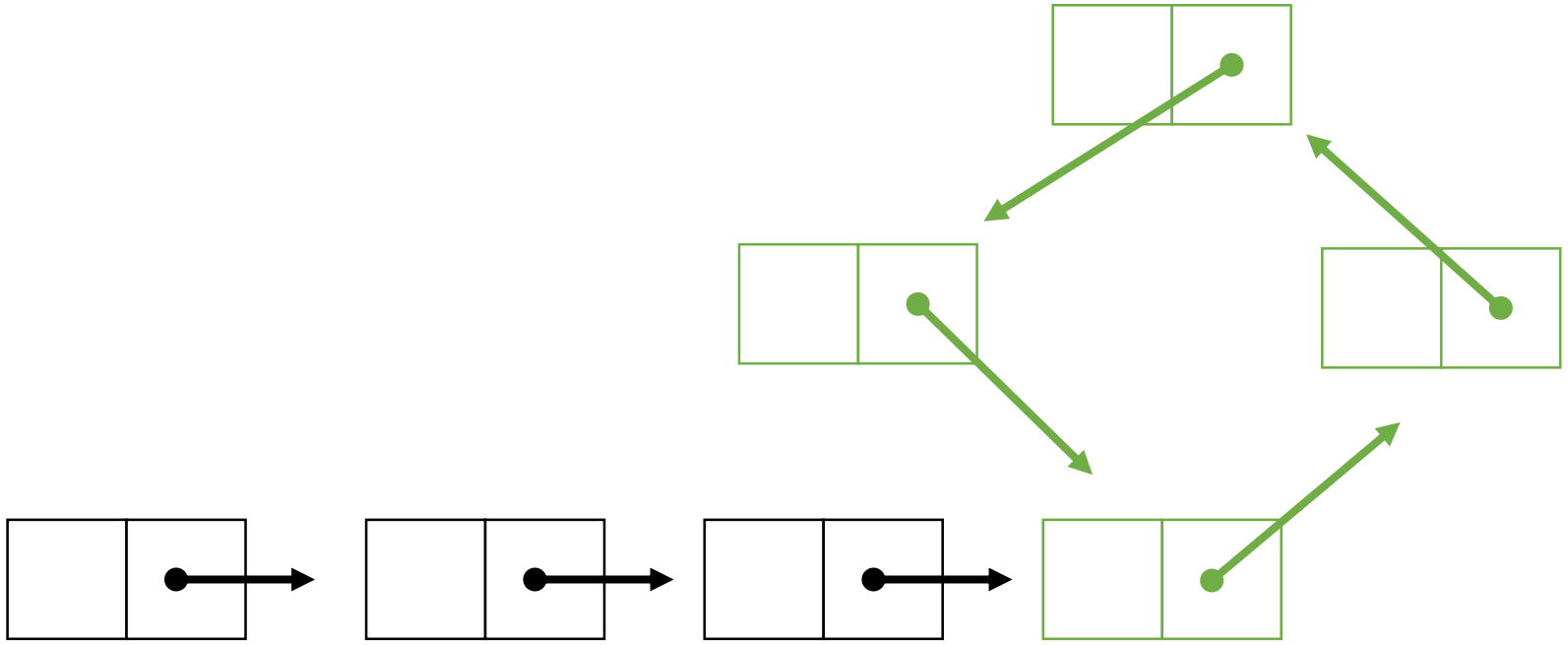
…zum 3. Übungsblatt

- Skip-Listen
- Listen
- Dynamische Programmierung
- Suchen

Existiert ein Kreis?

Gegeben sei eine einfach verkettete Liste.

Gibt es einen Kreis?



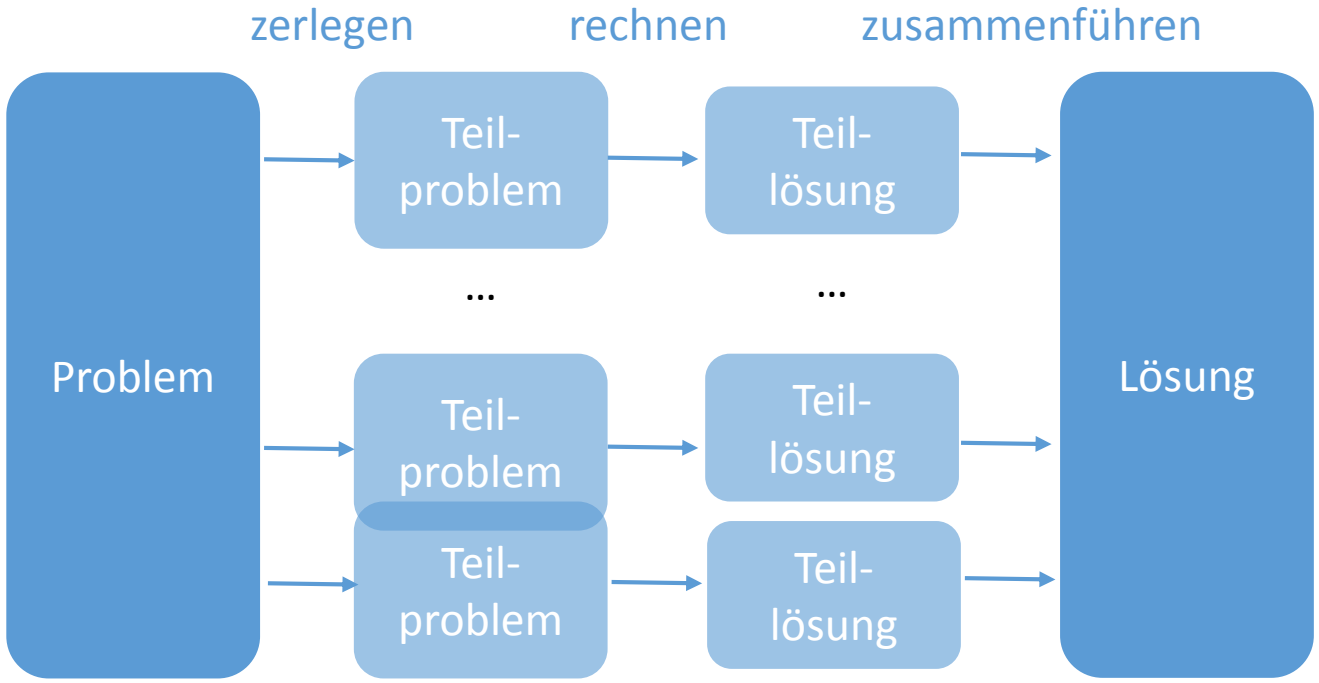
Vorbereitung heute…

…zum 3. Übungsblatt

- Skip-Listen
- Listen
- Dynamische Programmierung
- Suchen

Dynamische Programmierung

Problem hat optimale Substruktur.
Teilprobleme überlappen.
Teillösungen merken und wiederverwenden.



Beispiel:
Fibonacci-Zahlen

Vorbereitung heute…

…zum 3. Übungsblatt

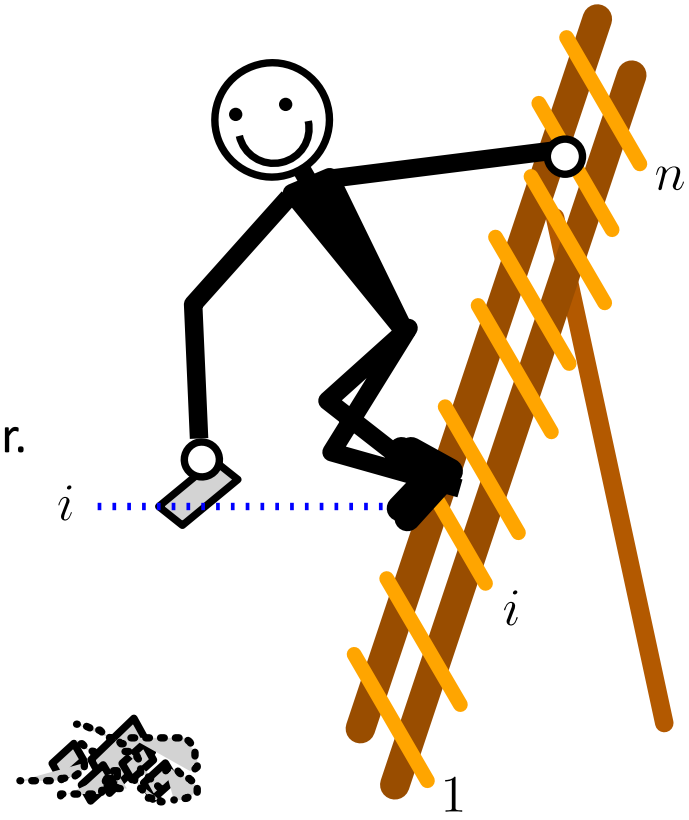
- Skip-Listen
- Listen
- Dynamische Programmierung
- Suchen

Suche

- Redakteur einer Handy-Zeitschrift testet neues Smartphone auf Bruchtauglichkeit
- Er hat k Smartphones und eine Leiter mit n Sprossen
- In jedem Versuch steigt er auf eine Sprosse und lässt ein Smartphone fallen.
- Entweder das Smartphone bleibt unbeschadet oder es bricht auseinander.

Er möchte die höchste Sprosse ermitteln, bei der das Smartphone nicht bricht und dabei außerdem die Anzahl der Versuche minimieren.

Gib einen Algorithmus an, der die Anzahl der Testversuche im Worst-Case minimiert und höchstens k Smartphones verwendet:



a) $k = 1$ b) $k = \infty$ c) $k = 2$