

Informatik II - SS 2014

(Algorithmen & Datenstrukturen)

Vorlesung 10 (3.6.2014)

Binäre Suchbäume I



**UNI
FREIBURG**

Fabian Kuhn

Algorithmen und Komplexität

Dictionary:

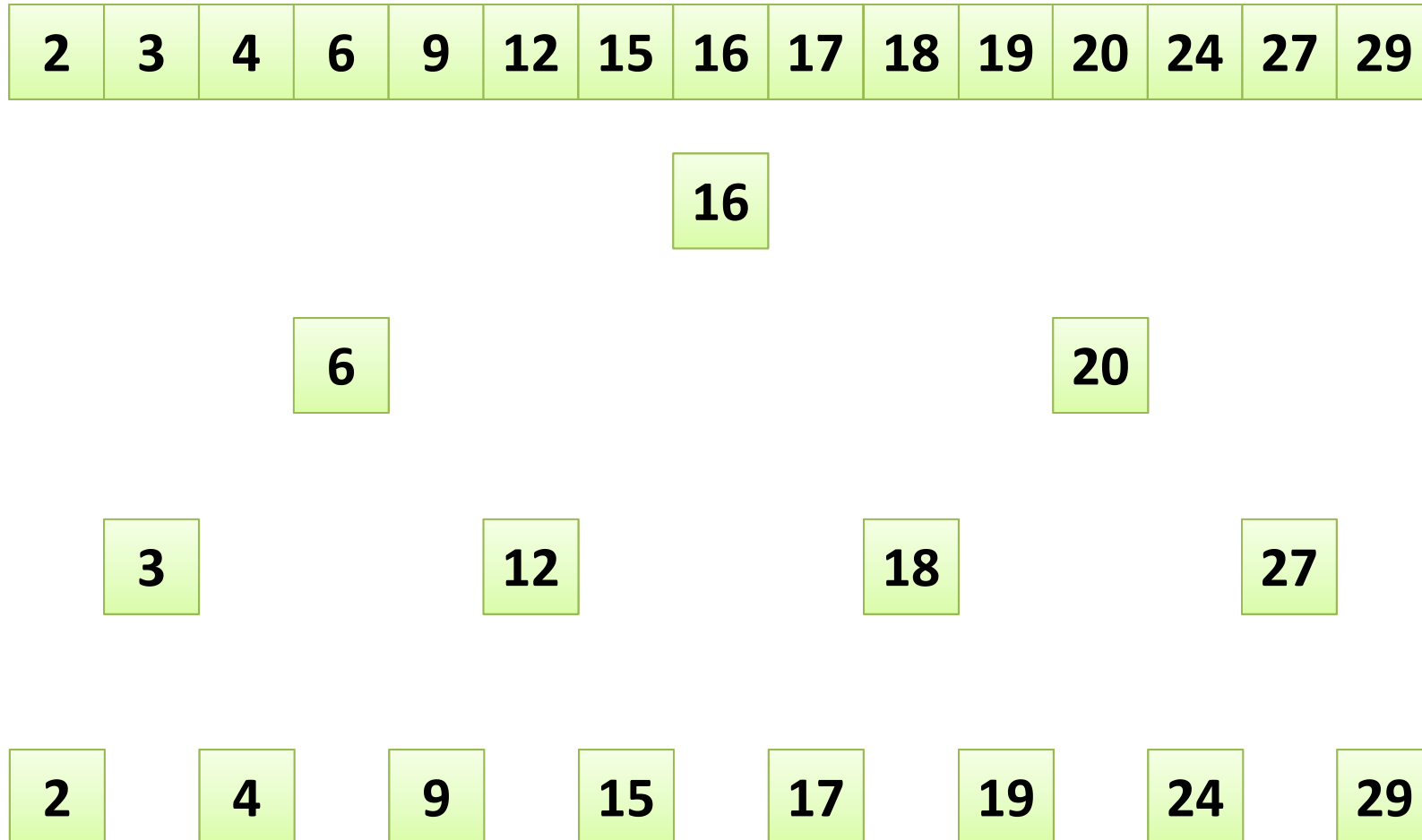
Zusätzliche mögliche Operationen:

- *D.minimum()* : gibt kleinsten *key* in der Datenstruktur zurück
- *D.maximum()* : gibt grössten *key* in der Datenstruktur zurück
- *D.successor(key)* : gibt nächstgrösseren *key* zurück
- *D.predecessor(key)* : gibt nächstkleineren *key* zurück
- *D.getRange(k1, k2)* : gibt alle Einträge mit Schlüsseln im Intervall $[k1, k2]$ zurück

- Lassen sich mit Hashtabellen nicht effizient implementieren!

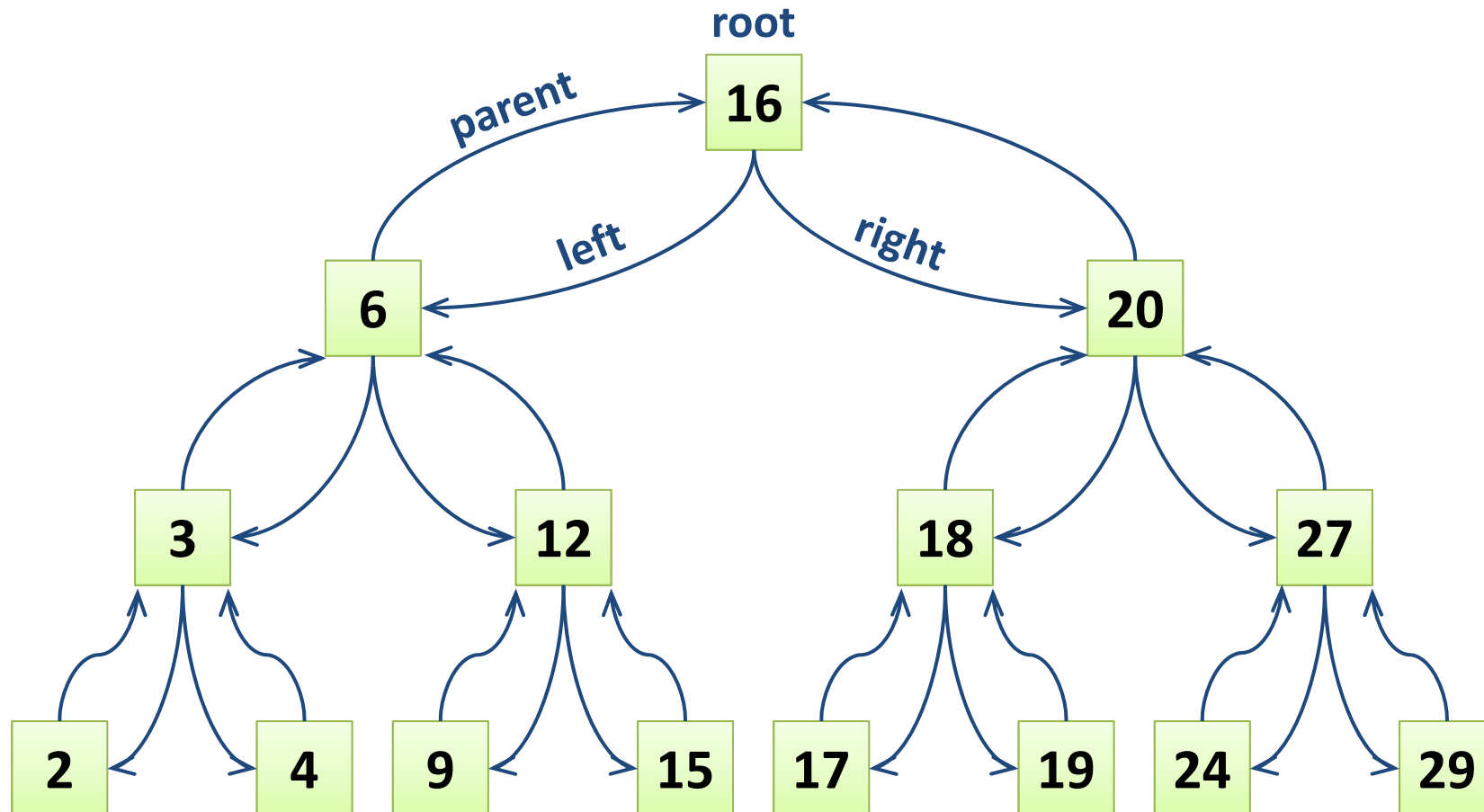
Binäre Suche Revisited...

- Binäre Suche nach x in einem sortierten Array...



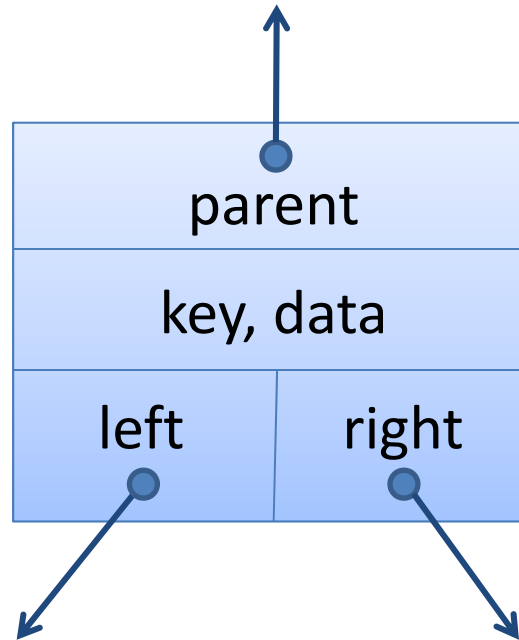
Binäre Suchbäume

- Benutze den Suchbaum der binären Suche als Datenstruktur



Binärer Suchbaum : Elemente

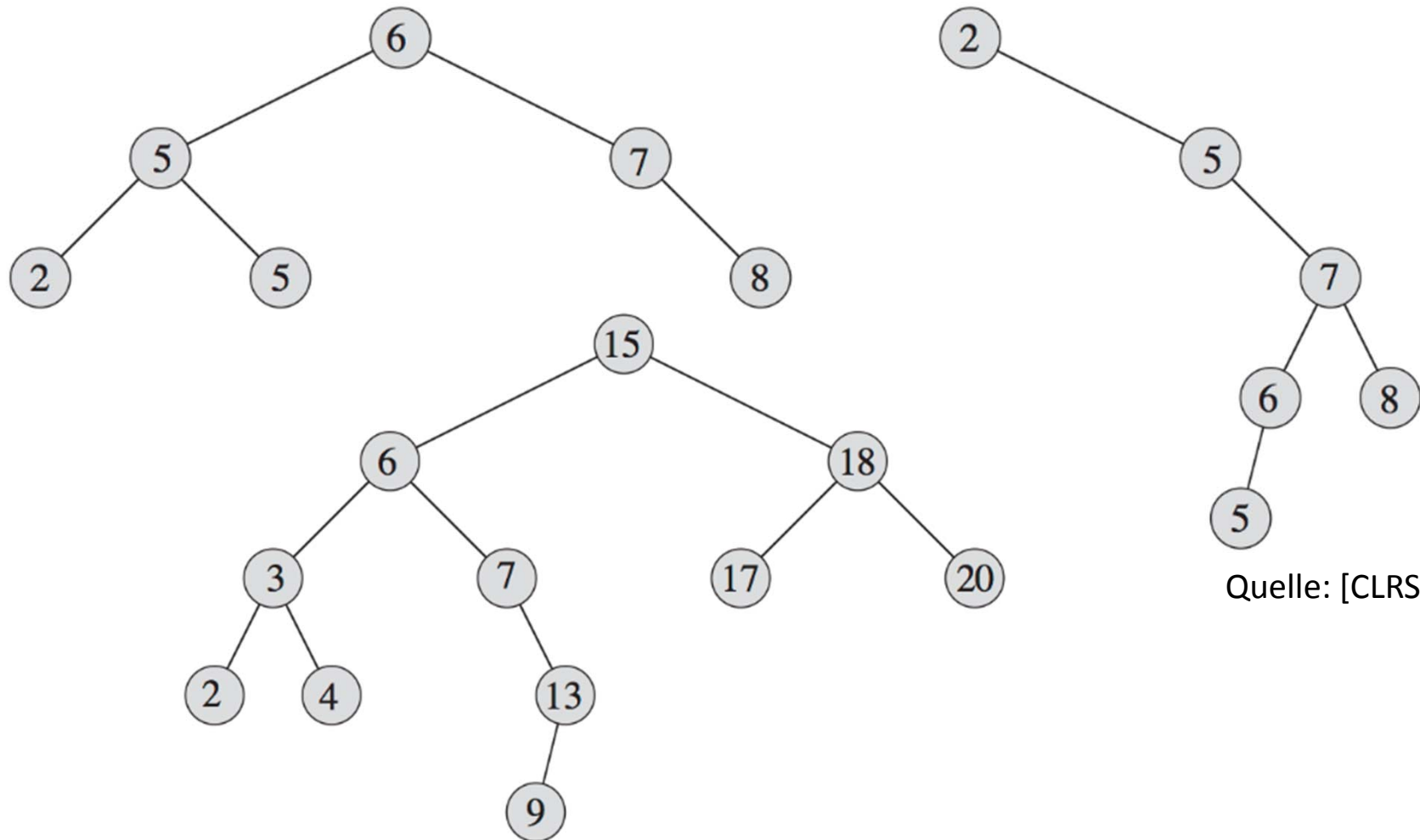
TreeElement:



Implementierung: gleich wie bei den Listen-Elementen

Binäre Suchbäume

- Binäre Suchbäume müssen nicht immer so schön symmetrisch sein...

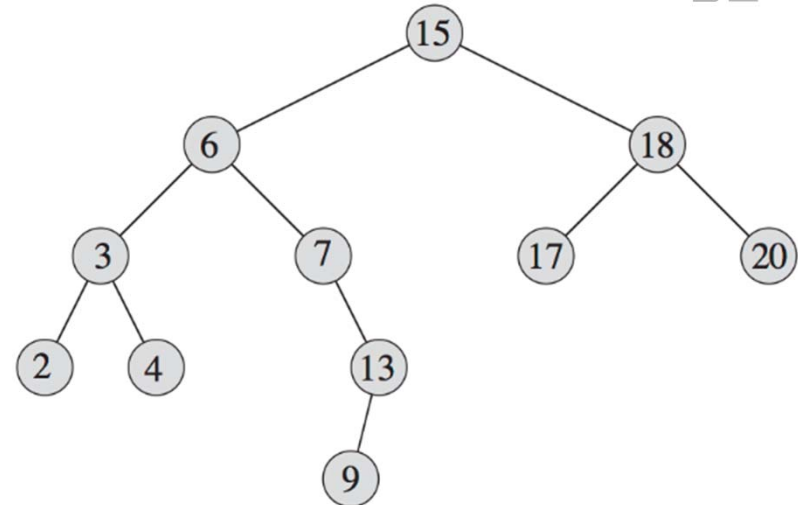


Quelle: [CLRS]

Suche in einem binären Suchbaum

Suche nach Schlüssel x

- Benutze binäre Suche
(darum heißt's binärer Suchbaum...)



```
current = root
```

```
while current != null and current.key != x do
```

```
  if current.key > x then
```

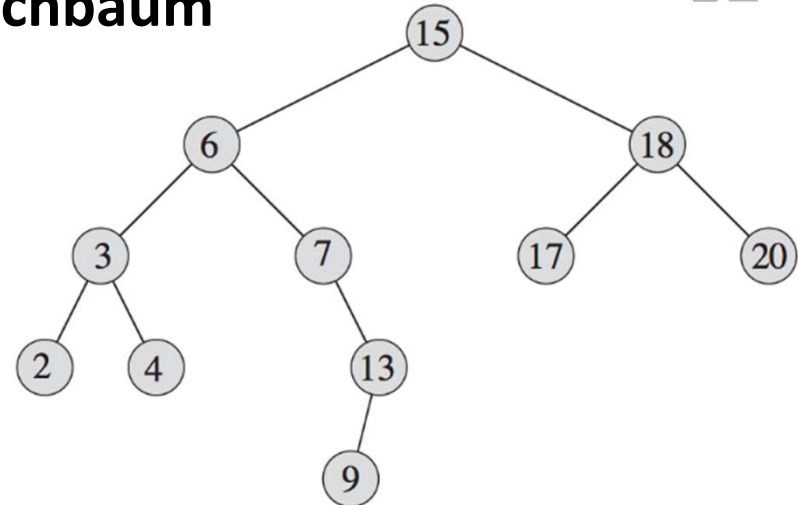
```
    current = current.left
```

```
  else
```

```
    current = current.right
```

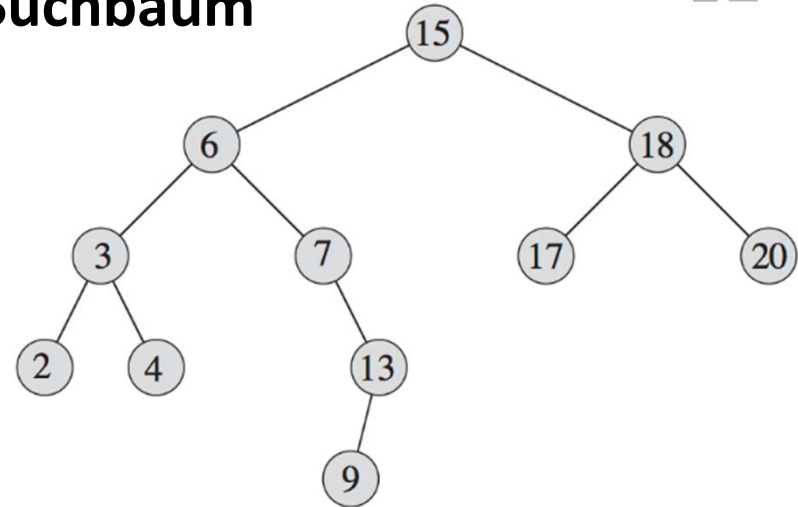
Suche in einem binären Suchbaum

Laufzeit der Suche in einem binären Suchbaum



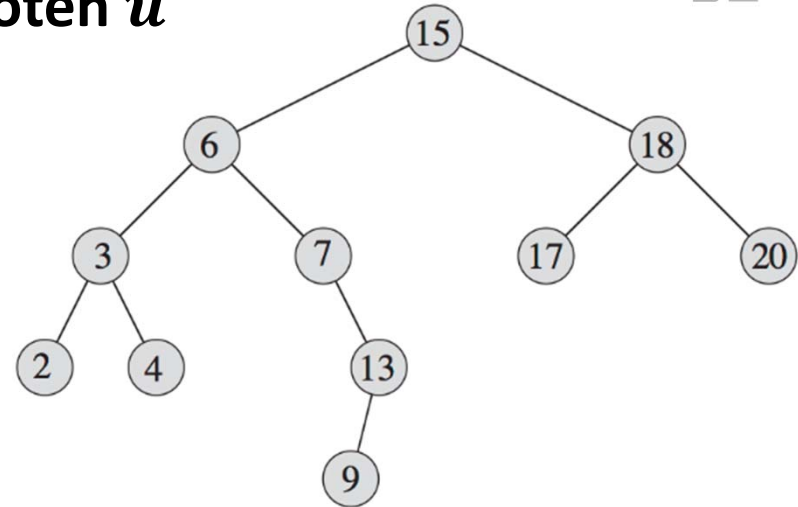
Suche Minimum / Maximum

Finde kleinstes Element in einem bin. Suchbaum



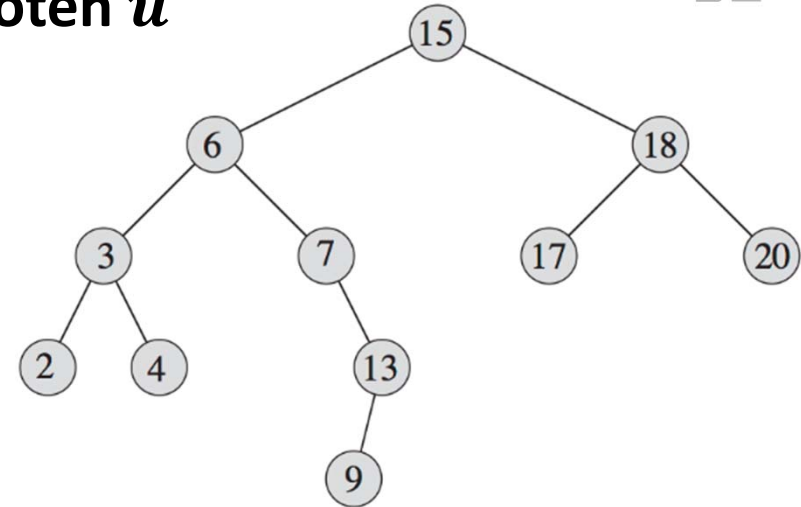
Suche Vorgänger / Nachfolger

Finde Vorgänger / Nachfolger eines Knoten u



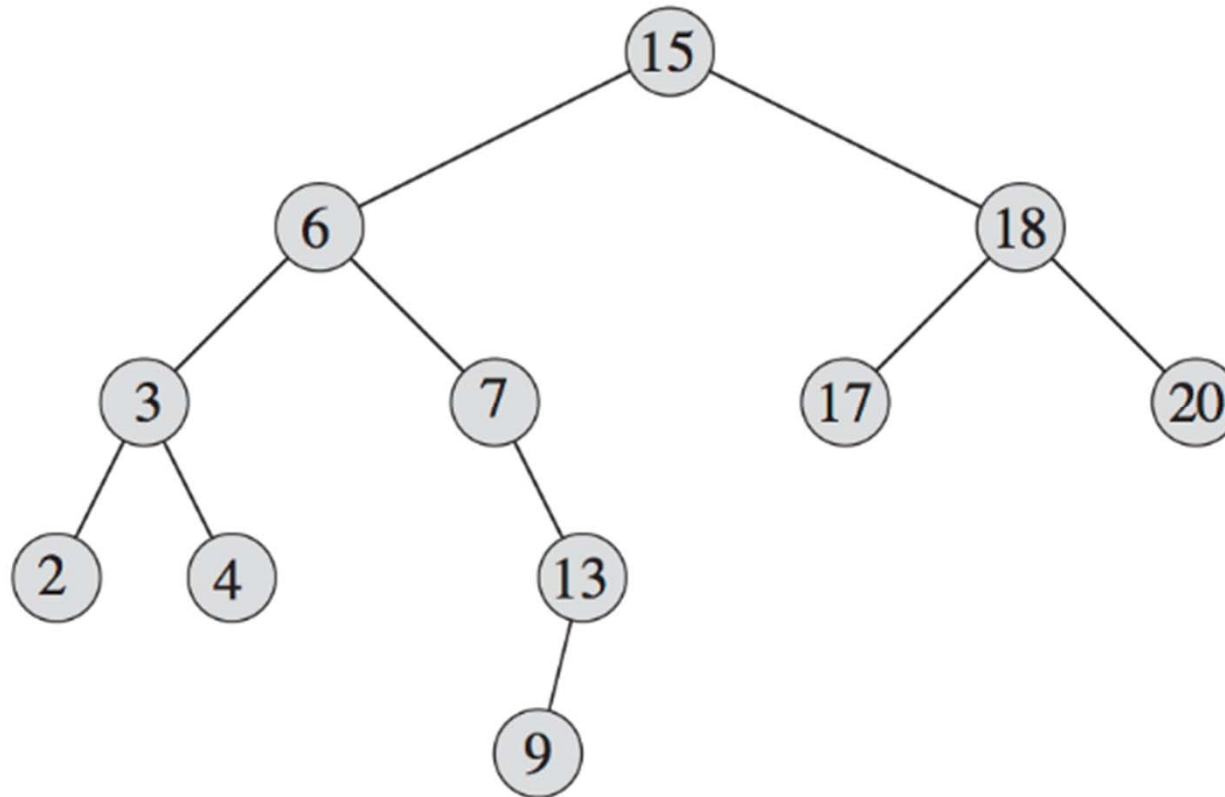
Suche Vorgänger / Nachfolger

Finde Vorgänger / Nachfolger eines Knoten u



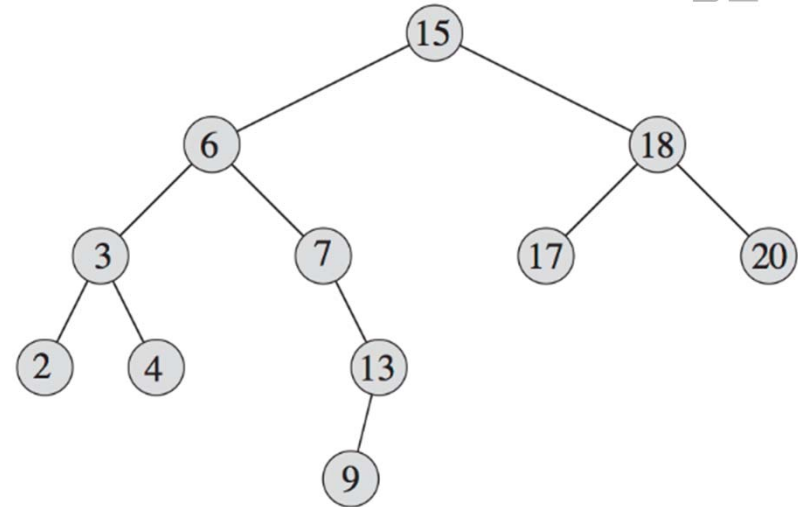
Einfügen eines Schlüssels

Füge Schlüssel 1, 5, 14, 6.5, 19 ein...



Einfügen eines Schlüssels

Füge Schlüssel x ein



Löschen eines Schlüssels I

Lösche Schlüssel x , einfache Fälle:

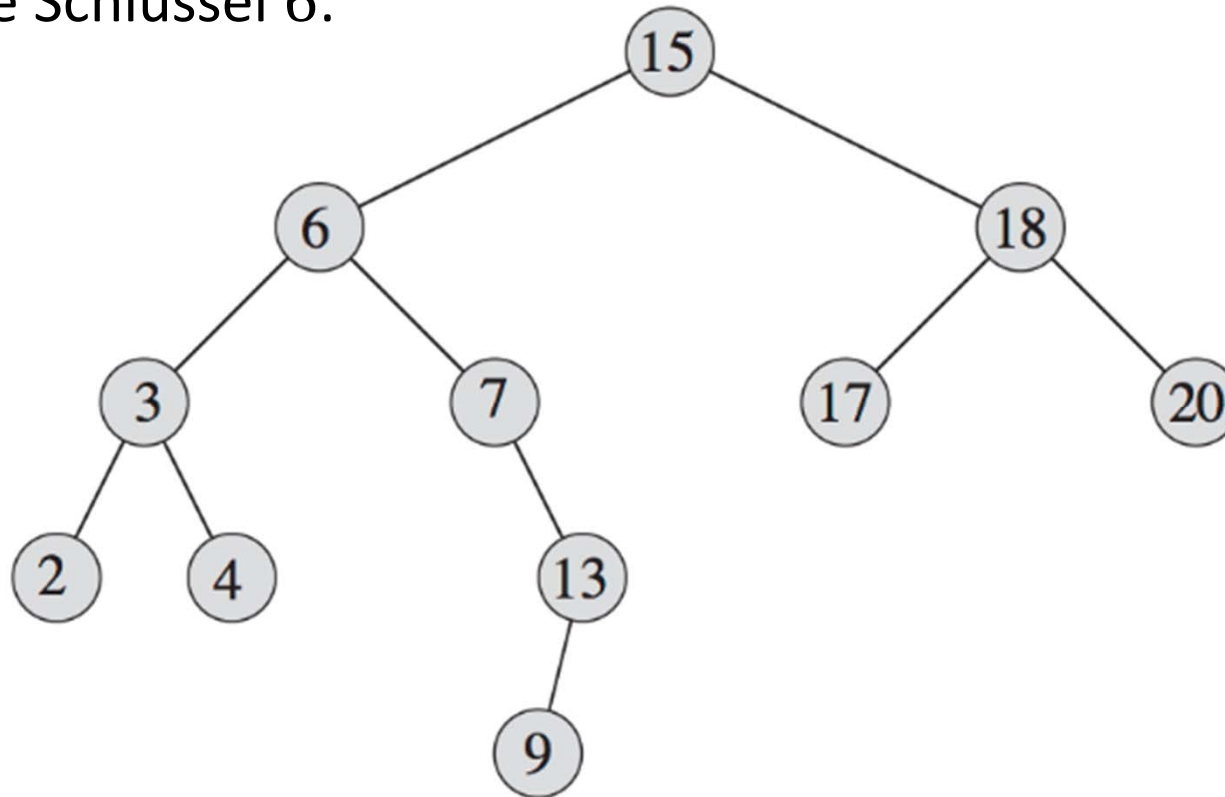
- Schlüssel x ist in einem Blatt des Baums
 - Blatt = Knoten hat keine Kinder

- Knoten mit Schlüssel x hat nur 1 Kind

Löschen eines Schlüssels II

Lösche Schlüssel x , Knoten hat zwei Kinder:

- Lösche Schlüssel 6:



Lösche Schlüssel x , Knoten hat zwei Kinder:

- Vorgänger ist grösster Knoten im linken Teilbaum
 - Vorgänger hat kein rechtes Kind
- Nachfolger ist kleinster Knoten im rechten Teilbaum
 - Nachfolger hat kein linkes Kind
- Schreibe Schlüssel und Daten des Vorgängers (oder alternativ Nachfolgers) in den Knoten von x
- Lösche Vorgänger/Nachfolger-Knoten
 - Vorgänger/Nachfolger ist entweder ein Blatt oder hat nur ein Kind

Lösche Schlüssel x :

1. Finde Knoten u mit $u.key = x$
 - wie üblich mit binärer Suche
2. Falls u nicht 2 Kinder hat, lösche Knoten u
 - Annahme: v ist Parent von u , u ist linkes Kind von v (anderer Fall analog)
 - Fall u ein Blatt ist, wird $v.parent.left = \text{null}$
 - Falls u ein Kind w hat, wird $v.parent.left = w$
3. Falls u zwei Kinder hat, dann bestimme Vorgängerknoten v
 - Funktioniert auch mit Nachfolgerknoten
4. Setze $u.key = v.key$ und $u.data = v.data$
5. Lösche Knoten v (gleich, wie oben u gelöscht wird)
 - Knoten v hat höchstens 1 Kind!

Laufzeit Binärer Suchbaum

Die Operationen

find, min, max, predecessor, successor, insert, delete

haben alle **Laufzeit $O(\text{Tiefe des Baums})$** .

Was ist die Tiefe eines binären Suchbaums?

Praktische Übungsaufgabe

Programmieren einer Binary Search Tree Klasse

- Sprache: C++, Java, Python
- Operationen: find, insert, delete, minimum, maximum
- Zusätzlich:
 - toArray (gibt sortiertes Array zurück)
 - avgDepth (gibt durchschnittliche Blatttiefe zurück)
- Vorgabe: Struktur der Klasse
 - Signatur aller public Methoden
- Wir werden jetzt gleich mal die C++-Vorgabe anschauen und eine erste Methode programmieren...