hhu.

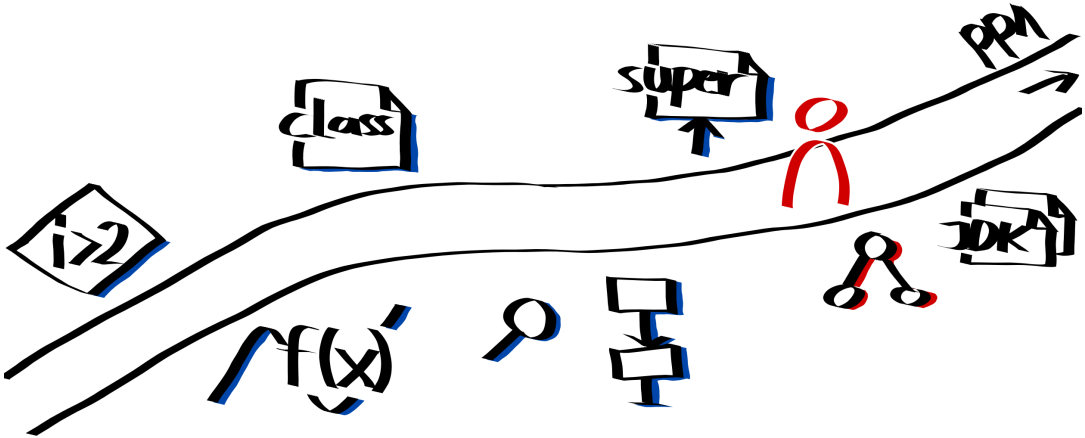
Kapitel 7: Datenstrukturen für effiziente Suche

VL 23: Binäre Suchbäume

Wintersemester 2023/24

Stand 9. Januar 2024, 09:05 Uhr

Wo stehen wir gerade?



Vorlesungsfeedback

Schnelle Suche bisher

- Binäre Suche in sortiertem Array

Schnelle Suche bisher

- Binäre Suche in sortiertem Array
  - müssen Sortierung sicherstellen
  - keine dynamische Datenstruktur (feste Größe)
- Sind sortierte Listen eine Lösung?

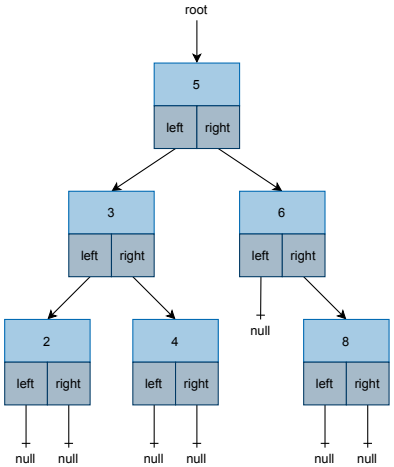
Schnelle Suche bisher

- Binäre Suche in sortiertem Array
  - müssen Sortierung sicherstellen
  - keine dynamische Datenstruktur (feste Größe)
- Sind sortierte Listen eine Lösung?
  - Zugriff auf beliebige Position langsam

⇒ neue dynamische Datenstruktur: Binärer Suchbaum

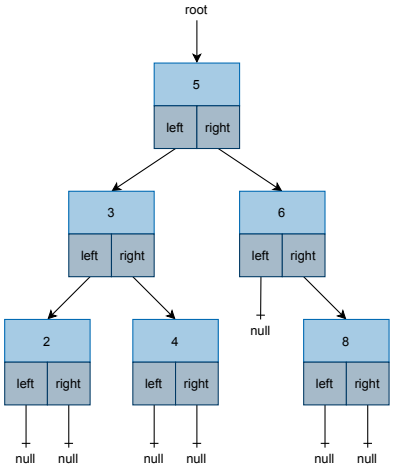
- \+ Wachsen/Schrumpfen nach Bedarf (wie Listen)
- \+ Automatische Sicherstellung der Sortierung
- \+ Suche (potentiell) genauso schnell wie in sortiertem Array

Idee: Baumstruktur



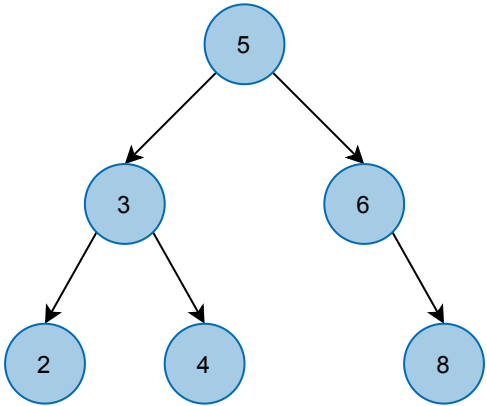
Idee: Baumstruktur

- Jedes Element (*Knoten*): linker und rechter Nachfolger (*Kind*) („Binärbaum“)
- Ordnung: Alle Knoten im linken *Teilbaum* kleiner als Knotenwert, alle Knoten im rechten Teilbaum größer („binärer Suchbaum“)
  - ⇒ keine doppelten Werte



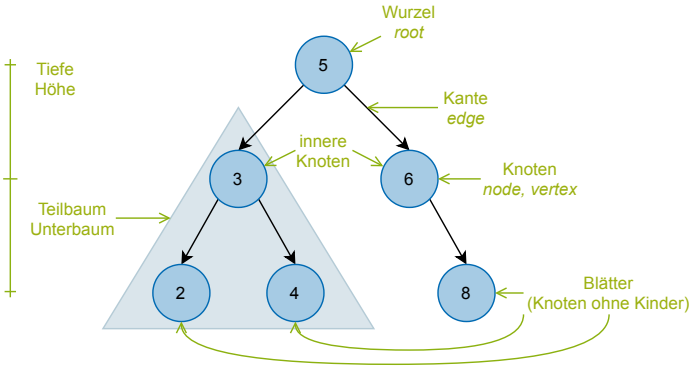
Idee: Baumstruktur

- Jedes Element (*Knoten*): linker und rechter Nachfolger (*Kind*) („Binärbaum“)
- Ordnung: Alle Knoten im linken *Teilbaum* kleiner als Knotenwert, alle Knoten im rechten Teilbaum größer („binärer Suchbaum“)
  - $\Rightarrow$ keine doppelten Werte



Binärer Suchbaum[1]: Vokabeln

- 2 ist linkes Kind von 3.
- 3 ist Elternknoten von 2 und 4.
- 3 ist die Wurzel des linken Teilbaums von 5.
- Der Baum hat die Tiefe 3.



[1] Binary Search Tree

Definition in Java

Heinrich Heine Universität Düsseldorf

```java
public class BinarySearchTree {

    private class BinaryNode {
        private int element;
        private BinaryNode left, right;

        private BinaryNode(int element) {
            this.element = element;
        }
    }

    private BinaryNode root;
```

Anmerkungen

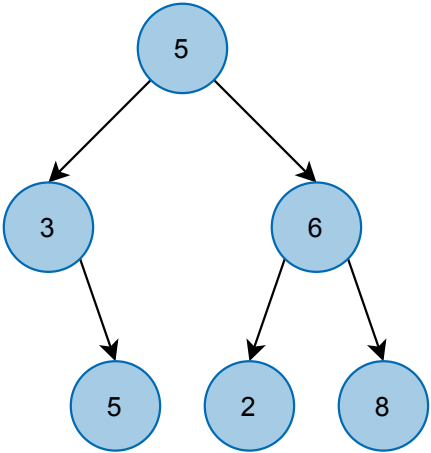
Heinrich Heine
Universität Düsseldorf

Rekursive Definition

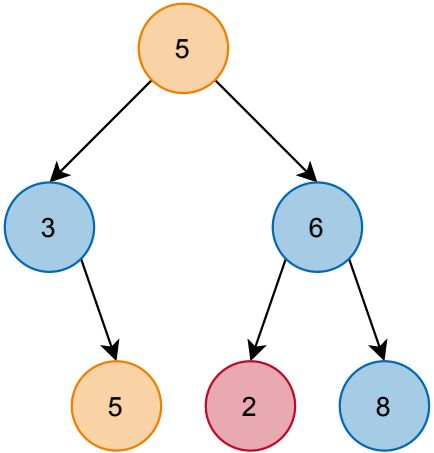
Ein Binärbaum ist entweder leer oder ein Wert mit zwei Binärbäumen.

- Andere Definitionen von Höhe und Blättern möglich
- nach unserer Defintion keine doppelten Elemente, andere Definitionen aber möglicht, z. B.:
  - pro Knoten mehrere Elemente gespeichert (z. B. mit Liste in jedem Knoten)
  - Zähler bei jedem Knoten, wie oft Element gespeichert ist

Ist das ein binärer Suchbaum?



Ist das ein binärer Suchbaum?

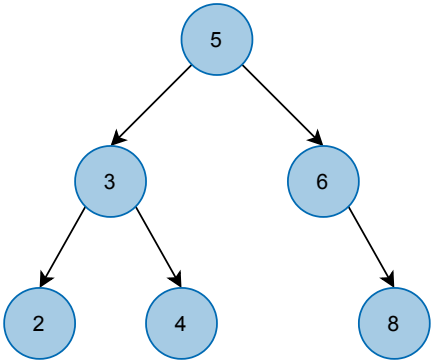


Nein, aber ein Binärbaum.

Einfügen

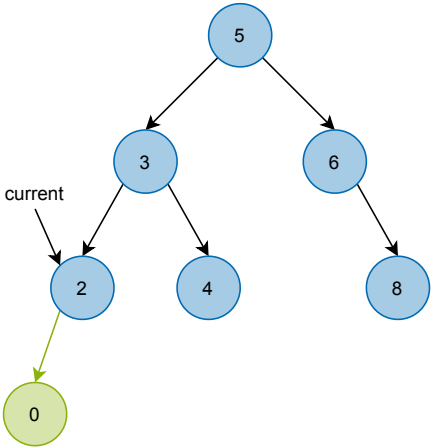
Heinrich Heine
Universität Düsseldorf

Ziel: „0“ einfügen



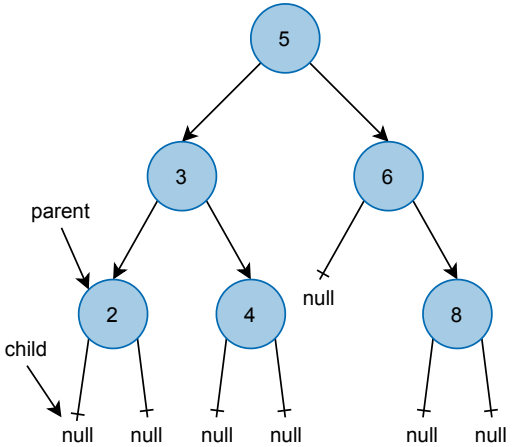
Einfügen

Ziel: „0“ einfügen



Einfügen

Ziel: „0“ einfügen



Codebeispiel: Einfügen – Einfügeposition finden

Heinrich Heine Universität Düsseldorf

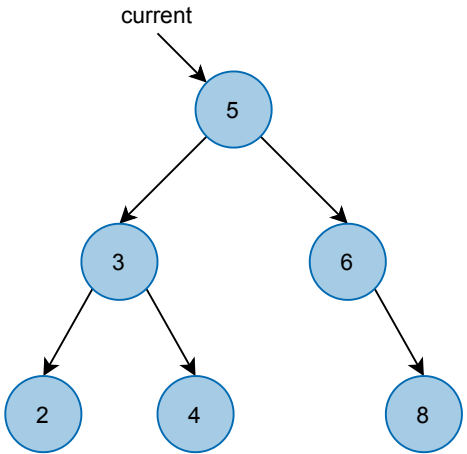
```
1   public void insert(int newNumber) {
2       // Sonderfall: leerer Baum
3       if (root == null) {
4           root = new BinaryNode(newNumber);
5           return;
6       }
7
8       BinaryNode parent = null;
9       BinaryNode child = root;
10      while (child != null) {
11          parent = child;
12          if (newNumber == child.element) {
13              // Zahl bereits im Baum vorhanden
14              return;
15          } else if (newNumber < child.element) {
16              child = child.left;
17          } else {
18              child = child.right;
19          }
20      }
21
```

Codebeispiel: Einfügen – Neuen Knoten einhängen

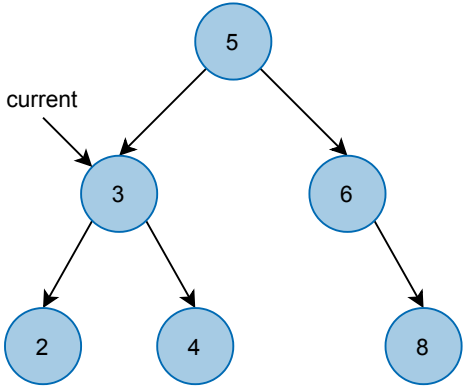
Heinrich Heine Universität Düsseldorf

```
22          if (newNumber < parent.element) {
23              parent.left = new BinaryNode(newNumber);
24          } else {
25              parent.right = new BinaryNode(newNumber);
26          }
27      }
```

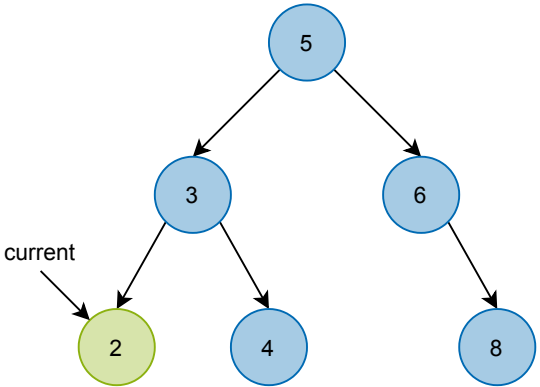
Minimum suchen: iterativ



Minimum suchen: iterativ

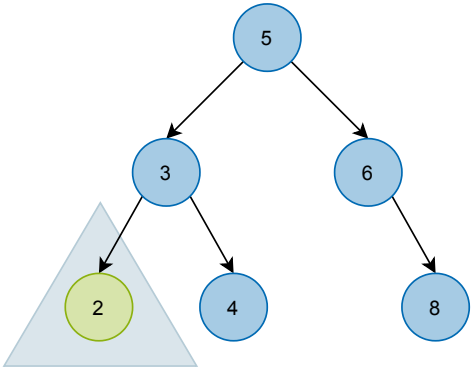


Minimum suchen: iterativ



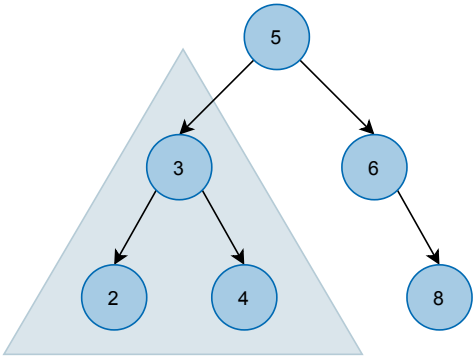
Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = ?



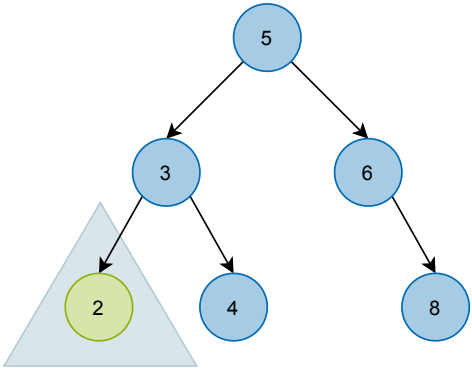
Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = Wurzel
- Minimum eines Binären Suchbaums = ?



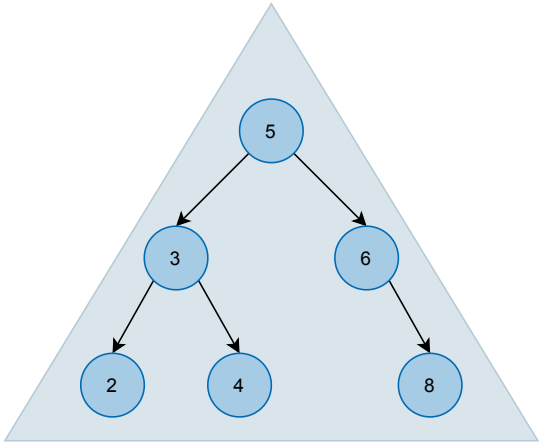
Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = Wurzel
- Minimum eines Binären Suchbaums = Minimum des linken Teilbaums



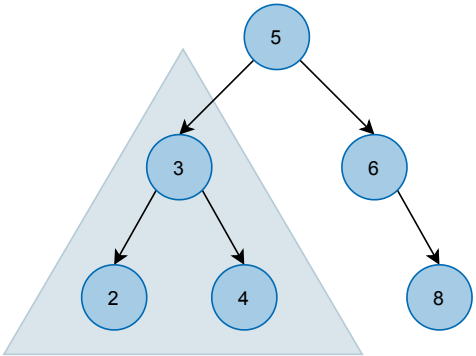
Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = Wurzel
- Minimum eines Binären Suchbaums = Minimum des linken Teilbaums



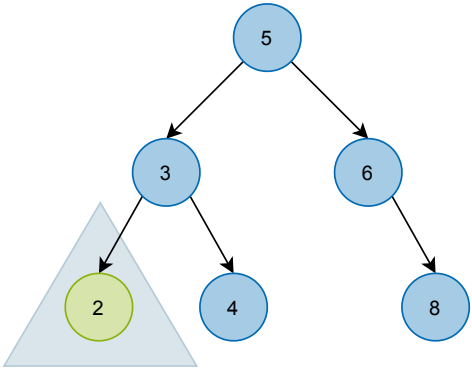
Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = Wurzel
- Minimum eines Binären Suchbaums = Minimum des linken Teilbaums



Minimum suchen: rekursiv

- Minimum eines Baums ohne linken Teilbaum = Wurzel
- Minimum eines Binären Suchbaums = Minimum des linken Teilbaums



Praxis: iterativ oder rekursiv?

```
ublic int minimum() {
   if(root == null) {
       throw new java.util.NoSuchElementException();
   }

   BinaryNode current = root;
   while(current.left != null) {
       current = current.left;
   }

   return current.element;

```

```
ublic int minimumRecursive() {
   if(root == null) {
       throw new java.util.NoSuchElementException();
   }

   return minimumRecursive(root);


rivate int minimumRecursive(BinaryNode current) {
   if(current.left == null) {
       return current.element;
   }
   return minimumRecursive(current.left);

```

Praxis: iterativ oder rekursiv?

Heinrich Heine Universität Düsseldorf

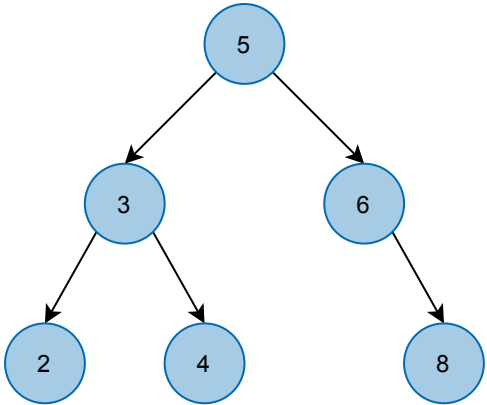
```
ublic int minimum() {
   if(root == null) {
       throw new java.util.NoSuchElementException();
   }

   BinaryNode current = root;
   while(current.left != null) {
       current = current.left;
   }

   return current.element;

```

```
ublic int minimumRecursive() {
   if(root == null) {
       throw new java.util.NoSuchElementException();
   }

   return minimumRecursive(root);


rivate int minimumRecursive(BinaryNode current) {
   if(current.left == null) {
       return current.element;
   }
   return minimumRecursive(current.left);

```

Iterative Lösung zu bevorzugen

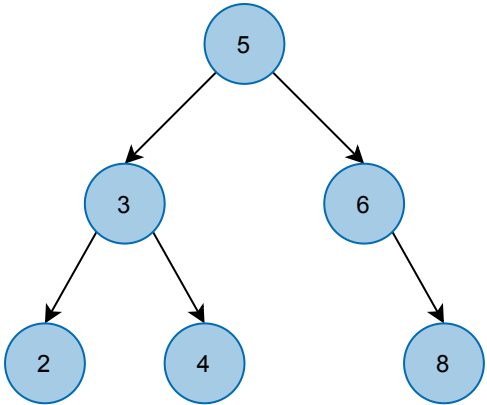
- nicht wesentlich komplizierter
- kein unnötiger Aufbau der Stackframes für jeden Methodenaufruf
- kein Stackoverflow möglich

Suche eines beliebigen Werts *n*

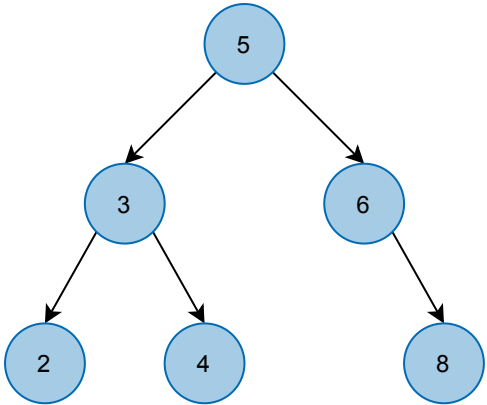


Suche eines beliebigen Werts n

- Wert im aktuellen Knoten $= n$?
  - Element gefunden
- Wert im aktuellen Knoten $< n$?
  - $\rightarrow$ im rechten Teilbaum weitersuchen
- Wert im aktuellen Knoten $> n$?
  - $\rightarrow$ im linken Teilbaum weitersuchen
- aktueller Knoten `null`?
  - Element nicht gefunden



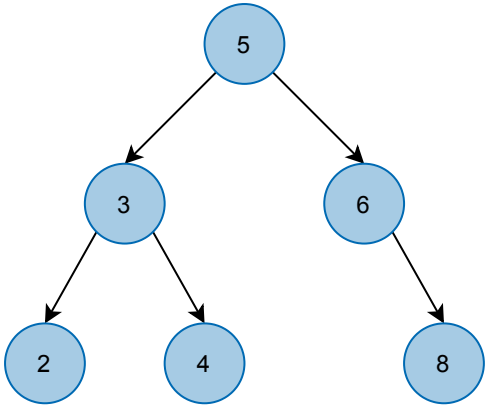
Wie lange dauert die Suche maximal?



Wie lange dauert die Suche maximal?

- max. Anzahl Vergleiche = Höhe des Baums

$\Rightarrow$ Höhe sollte möglichst klein sein

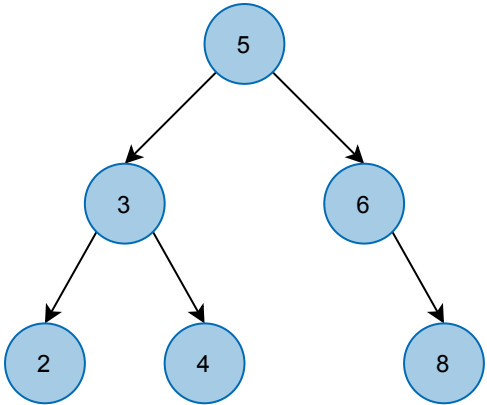


Wie lange dauert die Suche maximal?

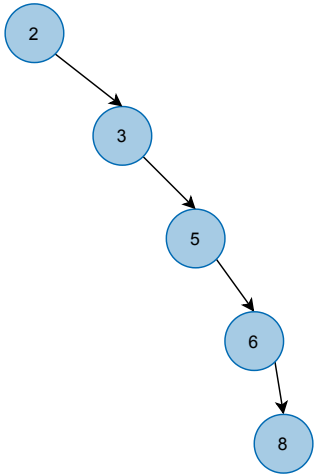
- max. Anzahl Vergleiche = Höhe des Baums

$\Rightarrow$ Höhe sollte möglichst klein sein

- vollständig ausgeglichener Binärbaum:
  - alle *Niveaus* bis auf das unterste voll
  - hat minimale Höhe ($\sim \log_2(\#Knoten)$)
  - $\Rightarrow$ Suche genauso schnell wie Binäre Suche



Wie schnell ist Suche hier?

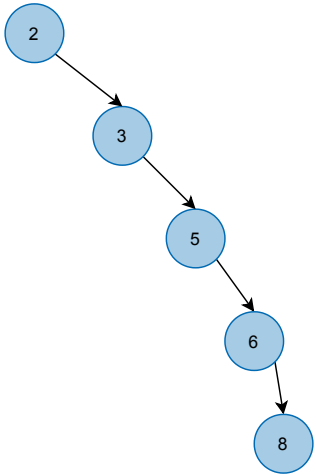


Wie schnell ist Suche hier?

- Baum zu linearer Liste *entartet*

⇒ Suche nicht schneller als in linearer Liste

- Wie kann ein solcher Baum entstehen?



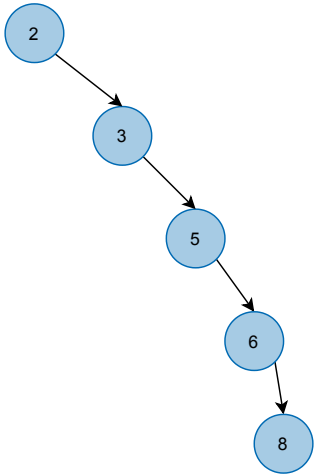
Wie schnell ist Suche hier?

- Baum zu linearer Liste *entartet*

⇒ Suche nicht schneller als in linearer Liste

- Wie kann ein solcher Baum entstehen?
  - ungünstige Einfügereihenfolge
- Wie kann man das (trotz ungünstiger Einfügereihenfolge) verhindern?

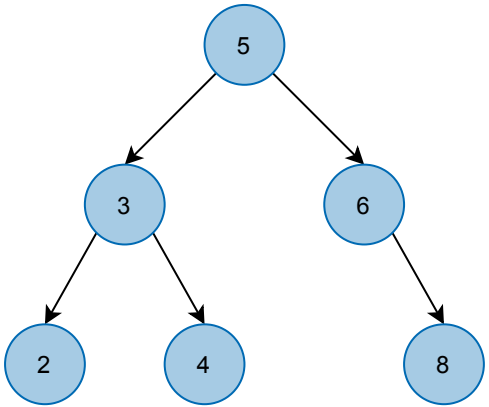
  → AlDat



Knoten zählen

Rekursiver Ansatz:

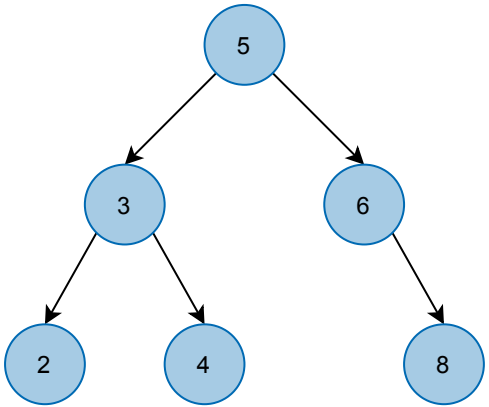
- Wie viele Knoten hat ein leerer Baum?



Knoten zählen

Rekursiver Ansatz:

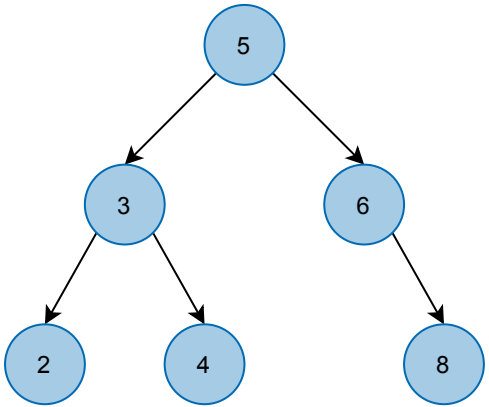
- Wie viele Knoten hat ein leerer Baum?
  - 0



Knoten zählen

Rekursiver Ansatz:

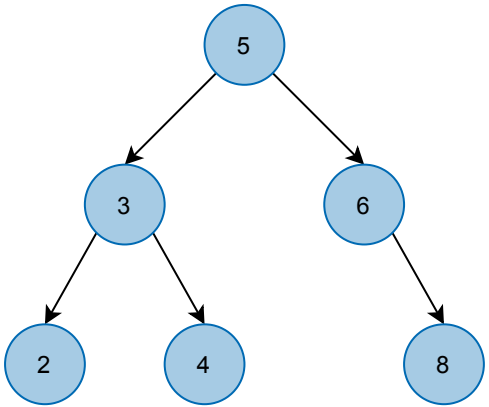
- Wie viele Knoten hat ein leerer Baum?
  - 0
- Angenommen, Teilbaum mit Wurzel r hat im linken Teilbaum n_l Knoten und im rechten Teilbaum n_r Knoten: Wie viele Knoten hat Teilbaum mit Wurzel r?



Knoten zählen

Rekursiver Ansatz:

- Wie viele Knoten hat ein leerer Baum?
  - 0
- Angenommen, Teilbaum mit Wurzel r hat im linken Teilbaum n_l Knoten und im rechten Teilbaum n_r Knoten: Wie viele Knoten hat Teilbaum mit Wurzel r?
  - $n_l + n_r + 1$



Lernziele

Heinrich Heine
Universität Düsseldorf

Sie können am Ende der Woche ...

- neue Operationen für binäre Suchbäume **implementieren**.
- **erklären**, wann die Suche in einem binären Suchbaum schnell ist.

Wichtige Begriffe

Heinrich Heine
Universität Düsseldorf

Binärer Suchbaum	Knoten	Wurzel
Teilbaum	Höhe	Entartung