

Informatik I: Einführung in die Programmierung

10. Bäume

Albert-Ludwigs-Universität Freiburg



**UNI
FREIBURG**

Prof. Dr. Peter Thiemann

29. November 2022



Der Baum

Der Baum

Definition
Terminologie
Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Bäume sind in der Informatik allgegenwärtig.



Der Baum

Definition
Terminologie
Beispiele

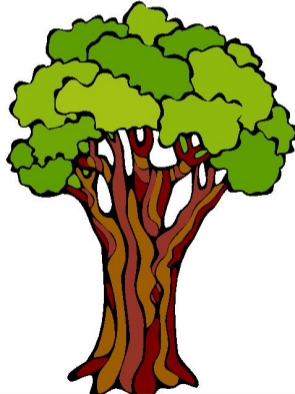
Binärbäume

Suchbäume

Zusammenfassung



- Bäume sind in der Informatik allgegenwärtig.
- Gezeichnet werden sie meistens mit der Wurzel nach oben!



Der Baum

Definition
Terminologie
Beispiele

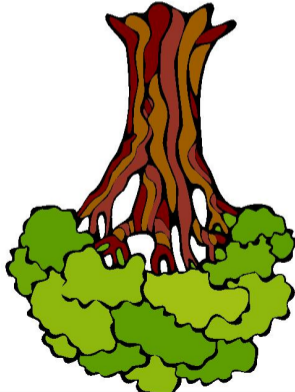
Binärbäume

Suchbäume

Zusammenfassung



- Bäume sind in der Informatik allgegenwärtig.
- Gezeichnet werden sie meistens mit der Wurzel nach oben!



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



Definition

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammen-
fassung



- Der **leere Baum** ist ein Baum.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammen-
fassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.
- Beispiel:

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.
- Beispiel:

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.
- Beispiel:



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.
- Beispiel:



...



Der Baum

Definition

Terminologie

Beispiele

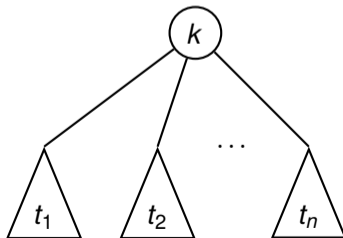
Binärbäume

Suchbäume

Zusammenfassung



- Der **leere Baum** ist ein Baum.
- Wenn t_1, \dots, t_n , $n \geq 0$ disjunkte Bäume sind und k ein **Knoten**, der nicht in t_1, \dots, t_n vorkommt, dann ist auch die Struktur bestehend aus der **Wurzel** k mit **zugeordneten Teilbäumen** t_1, \dots, t_n ein **Baum**.
- Nichts sonst ist ein Baum.
- Beispiel:



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



Terminologie

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammen-
fassung



- Alle Knoten, denen keine Teilbäume zugeordnet sind, heißen **Blätter**.

Der Baum

Definition

Terminologie

Beispiele

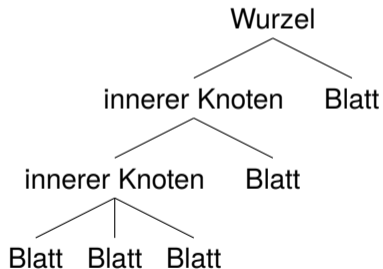
Binärbäume

Suchbäume

Zusammen-
fassung



- Alle Knoten, denen keine Teilbäume zugeordnet sind, heißen **Blätter**.
- Knoten, die keine Blätter sind, heißen **innere Knoten**.



Der Baum

Definition

Terminologie

Beispiele

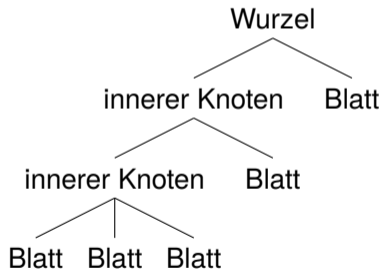
Binärbäume

Suchbäume

Zusammenfassung



- Alle Knoten, denen keine Teilbäume zugeordnet sind, heißen **Blätter**.
- Knoten, die keine Blätter sind, heißen **innere Knoten**.



- Die Wurzel kann also ein Blatt sein (keine weiteren Teilbäume) oder ein innerer Knoten.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,
 - Alle Elternknoten von k_2 , deren Elternknoten usw. sind **Vorgänger** von k_2 .

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,
 - Alle Elternknoten von k_2 , deren Elternknoten usw. sind **Vorgänger** von k_2 .
 - k_2 ist **Kind** von k_1 .

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,
 - Alle Elternknoten von k_2 , deren Elternknoten usw. sind **Vorgänger** von k_2 .
 - k_2 ist **Kind** von k_1 .
 - Alle Kinder von k_1 , deren Kinder, usw. sind **Nachfolger** von k_1 .

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,
 - Alle Elternknoten von k_2 , deren Elternknoten usw. sind **Vorgänger** von k_2 .
 - k_2 ist **Kind** von k_1 .
 - Alle Kinder von k_1 , deren Kinder, usw. sind **Nachfolger** von k_1 .
- Bäume sind oft **markiert**. Die Markierung weist jedem Knoten eine **Marke** zu.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Wenn k_1 ein Knoten und k_2 die Wurzel eines zugeordneten Teilbaums ist, dann gilt:
 - k_1 ist **Elternknoten** von k_2 ,
 - Alle Elternknoten von k_2 , deren Elternknoten usw. sind **Vorgänger** von k_2 .
 - k_2 ist **Kind** von k_1 .
 - Alle Kinder von k_1 , deren Kinder, usw. sind **Nachfolger** von k_1 .
- Bäume sind oft **markiert**. Die Markierung weist jedem Knoten eine **Marke** zu.
- Formal: Wenn K die Knotenmenge eines Baums ist und M eine Menge von Marken, dann ist die **Markierung eine Abbildung $\mu : K \rightarrow M$** .

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



Beispiele

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammen-
fassung

Beispiel: Verzeichnisbaum



In vielen Betriebssystemen ist die Verzeichnisstruktur im Wesentlichen baumartig.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

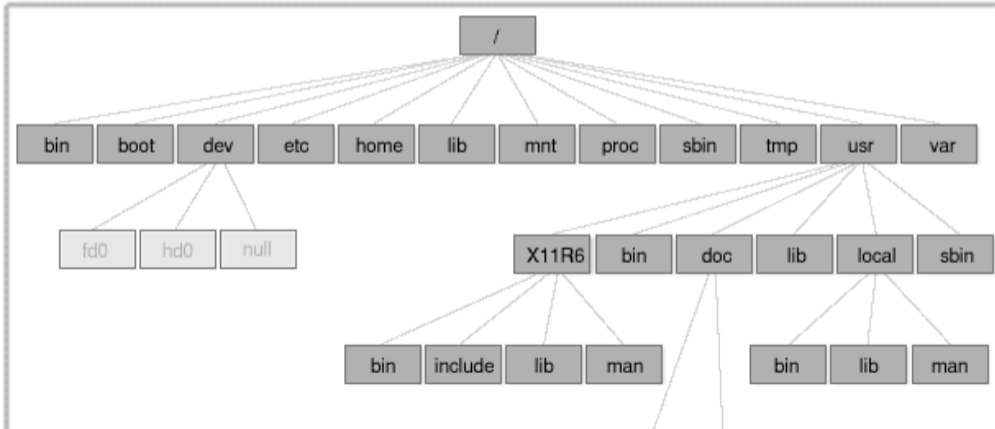
Suchbäume

Zusammen-
fassung

Beispiel: Verzeichnisbaum



In vielen Betriebssystemen ist die Verzeichnisstruktur im Wesentlichen baumartig.



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung

Beispiel: Syntaxbaum



Wenn die Struktur einer Sprache mit Hilfe einer formalen Grammatik spezifiziert ist, dann kann der Satzaufbau durch **Syntaxbäume** beschrieben werden.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

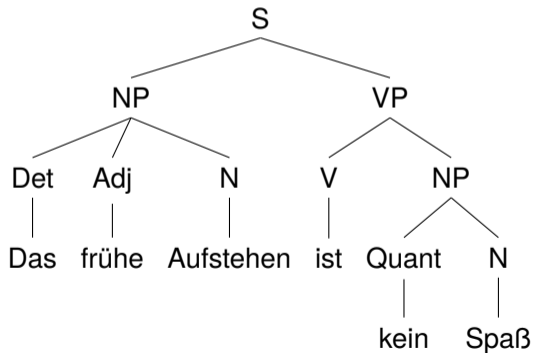
Suchbäume

Zusammen-
fassung

Beispiel: Syntaxbaum



Wenn die Struktur einer Sprache mit Hilfe einer formalen Grammatik spezifiziert ist, dann kann der Satzaufbau durch **Syntaxbäume** beschrieben werden.



Der Baum

Definition

Terminologie

Beispiele

Binäräume

Suchäume

Zusammenfassung



- Bäume können Ausdrücke so darstellen, dass ihre Auswertung eindeutig durchführbar ist, ohne dass Klammern notwendig sind.

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Bäume können Ausdrücke so darstellen, dass ihre Auswertung eindeutig durchführbar ist, ohne dass Klammern notwendig sind.
- Beispiel: $(5 + 6) * 3 * 2$

Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



- Bäume können Ausdrücke so darstellen, dass ihre Auswertung eindeutig durchführbar ist, ohne dass Klammern notwendig sind.
- Beispiel: $(5 + 6) * 3 * 2$
- Entspricht: $((5 + 6) * 3) * 2$

Der Baum

Definition

Terminologie

Beispiele

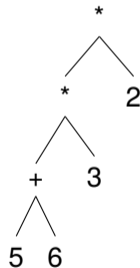
Binärbäume

Suchbäume

Zusammenfassung



- Bäume können Ausdrücke so darstellen, dass ihre Auswertung eindeutig durchführbar ist, ohne dass Klammern notwendig sind.
- Beispiel: $(5 + 6) * 3 * 2$
- Entspricht: $((5 + 6) * 3) * 2$
- Operatoren als Markierung innerer Knoten, Zahlen als Markierung der Blätter:



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

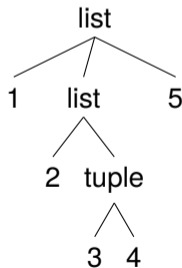
Suchbäume

Zusammenfassung

Beispiel: Listen und Tupel als Bäume



- Jede Liste und jedes Tupel kann als Baum angesehen werden, bei dem der Typ die Knotenmarkierung ist und die Elemente die Teilbäume sind.
- Beispiel: $[1, [2, (3, 4)], 5]$



Der Baum

Definition

Terminologie

Beispiele

Binärbäume

Suchbäume

Zusammenfassung



Binärbäume

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Der Binärbaum ist ein **Spezialfall eines Baumes**.
- Ein Binärbaum ist entweder **leer** oder besteht aus einem (Wurzel-) Knoten und zwei Teilbäumen.
- Für viele Anwendungsfälle angemessen.
- Funktionen über solchen Bäumen sind einfach definierbar.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Repräsentation

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.
 - Das Attribut `left` enthält den **linken Teilbaum**.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.
 - Das Attribut `left` enthält den **linken Teilbaum**.
 - Das Attribut `right` enthält den **rechten Teilbaum**.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.
 - Das Attribut `left` enthält den **linken Teilbaum**.
 - Das Attribut `right` enthält den **rechten Teilbaum**.
- Beispiele:

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.
 - Das Attribut `left` enthält den **linken Teilbaum**.
 - Das Attribut `right` enthält den **rechten Teilbaum**.
- Beispiele:
 - Der Baum bestehend aus dem einzigen Knoten mit der Markierung 8:
`Node (8, None, None)`

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



- Der **leere Baum** wird durch `None` repräsentiert.
- Jeder andere **Knoten** wird durch ein `Node`-Objekt repräsentiert.
 - Das Attribut `mark` enthält die **Markierung**.
 - Das Attribut `left` enthält den **linken Teilbaum**.
 - Das Attribut `right` enthält den **rechten Teilbaum**.
- Beispiele:
 - Der Baum bestehend aus dem einzigen Knoten mit der Markierung 8:
`Node(8, None, None)`
 - Der Baum mit Wurzel '+', linkem Teilbaum mit Blatt 5, rechtem Teilbaum mit Blatt 6:
`Node('+', Node(5, None, None), Node(6, None, None))`

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



```
from typing import Any, Optional
@dataclass
class Node:
    mark : Any
    left : Optional['Node']
    right : Optional['Node']
```

Bemerkung zu den Typannotationen

- Any: ein Objekt von beliebigem Typ
- Optional[t]: entweder t oder None (aber nichts anderes)
- Der Typ Node existiert erst **nach** Ausführung der class-Anweisung. Der String 'Node' in der Typannotation wird rückwirkend durch den Typ Node ersetzt.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Beispiel

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

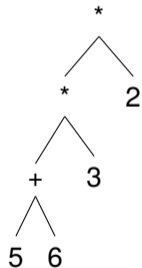
Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung

Beispiel: Der Ausdrucksbaum



wird folgendermaßen mit Node Objekten dargestellt:

```
Node('*', Node('*', Node('+', Node(5, None, None),  
                             Node(6, None, None)),  
         Node(3, None, None)),  
Node(2, None, None))
```

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Funktionen auf Bäumen

Der Baum

Binärbäume

Repräsentation

Beispiel

**Funktionen auf
Bäumen**

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



```
def tree_str(tree : Optional[Node]) -> str:
    match tree:
        case None:
            return "fill in"
        case Node (m, l, r):
            l_str = tree_str(l)
            r_str = tree_str(r)
            return "fill in"
```

- Node Objekte enthalten selbst wieder Node Objekte (oder None) in den Attributen left und right.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



```
def tree_str(tree : Optional[Node]) -> str:
    match tree:
        case None:
            return "fill in"
        case Node (m, l, r):
            l_str = tree_str(l)
            r_str = tree_str(r)
            return "fill in"
```

- Node Objekte enthalten selbst wieder Node Objekte (oder None) in den Attributen left und right.
- Zum Ausdrucken eines Node Objekts müssen auch die enthaltenen Node Objekte ausgedruckt werden.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



```
def tree_str(tree : Optional[Node]) -> str:
    match tree:
        case None:
            return "fill in"
        case Node (m, l, r):
            l_str = tree_str(l)
            r_str = tree_str(r)
            return "fill in"
```

- Node Objekte enthalten selbst wieder Node Objekte (oder None) in den Attributen left und right.
- Zum Ausdrucken eines Node Objekts müssen auch die enthaltenen Node Objekte ausgedruckt werden.
- tree_str ist **rekursiv**, es wird in seiner eigenen Definition aufgerufen!

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



- Die **rekursiven Aufrufe** `tree_str (tree.left)` und `tree_str (tree.right)` erfolgen **nur auf den Kindern des Knotens**.
- Ergibt sich zwangsläufig aus der induktiven Definition!
- **Rekursive Aufrufe auf den Teilbäumen** sind Teil des Funktionsgerüsts, sobald eine baumartige Struktur bearbeitet werden soll.

- Die **Alternative** “`case None`” ergibt sich zwangsläufig aus dem Typ `tree:Optional[Node]`: `tree` ist **entweder `None` oder eine `Node`-Instanz**.

- Alle Funktionen auf Binärbäumen verwenden dieses Gerüst.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



```
def tree_str(tree : Optional[Node]) -> string:
  match tree:
    case None:
      return "None"
    case Node (m, l, r):
      return ("Node("
        + repr(m) + ", "
        + tree_str (l) + ", "
        + tree_str (r) + ")")
```

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Baumeigenschaften

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

**Baumeigen-
schaften**

Traversierung

Suchbäume

Zusammen-
fassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

**Baumeigen-
schaften**

Traversierung

Suchbäume

Zusammen-
fassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:
 - -1 für den leeren Baum,

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:
 - -1 für den leeren Baum,
 - $m + 1$, wenn m die maximale Höhe aller der Wurzel zugeordneten Teilbäume ist.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:
 - -1 für den leeren Baum,
 - $m + 1$, wenn m die maximale Höhe aller der Wurzel zugeordneten Teilbäume ist.
- Die **Größe eines Baumes** ist die Anzahl seiner Knoten.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



UNI
FREIBURG

- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:
 - -1 für den leeren Baum,
 - $m + 1$, wenn m die maximale Höhe aller der Wurzel zugeordneten Teilbäume ist.
- Die **Größe eines Baumes** ist die Anzahl seiner Knoten.
 - 0 für den leeren Baum,

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung

Tiefe von Knoten, Höhe und Größe von (Binär-)Bäumen

induktiv definiert



- Die **Tiefe eines Knotens** k (Abstand zur Wurzel) ist
 - 0, falls k die Wurzel ist,
 - $i + 1$, wenn i die Tiefe des Elternknotens ist.
- Die **Höhe eines Baumes** ist die maximale Tiefe über alle Blätter:
 - -1 für den leeren Baum,
 - $m + 1$, wenn m die maximale Höhe aller der Wurzel zugeordneten Teilbäume ist.
- Die **Größe eines Baumes** ist die Anzahl seiner Knoten.
 - 0 für den leeren Baum,
 - $s + 1$, wenn s die Summe der Größen der Teilbäume ist.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



$$\text{height}(tree) = \begin{cases} -1, & \text{if } tree \text{ is empty} \\ 1 + \max(\text{height}(tree.left), \text{height}(tree.right)), & \text{otherwise.} \end{cases}$$

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



$$\text{height}(tree) = \begin{cases} -1, & \text{if } tree \text{ is empty} \\ 1 + \max(\text{height}(tree.left), \text{height}(tree.right)), & \text{otherwise.} \end{cases}$$

$$\text{size}(tree) = \begin{cases} 0, & \text{if } tree \text{ is empty;} \\ 1 + \text{size}(tree.left) + \text{size}(tree.right), & \text{otherwise.} \end{cases}$$

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



```
def height(tree : Optional[Node]) -> int:
  match tree:
    case None:
      return -1
    case Node (m, l, r):
      return(max(height(l), height(r)) + 1)

def size(tree : Optional[Node]) -> int:
  match tree:
    case None:
      return 0
    case Node (m, l, r):
      return(size(l) + size(r) + 1)
```

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Traversierung

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

Traversierung

Suchbäume

Zusammen-
fassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:
 - **Pre-Order** (Hauptreihenfolge): Bearbeite zuerst den Knoten selbst, dann besuche den linken, danach den rechten Teilbaum

- Der Baum
- Binärbäume
 - Repräsentation
 - Beispiel
 - Funktionen auf Bäumen
 - Baumeigenschaften
 - Traversierung**
- Suchbäume
- Zusammenfassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:
 - **Pre-Order** (Hauptreihenfolge): Bearbeite zuerst den Knoten selbst, dann besuche den linken, danach den rechten Teilbaum
 - **Post-Order** (Nebenreihenfolge): Besuche zuerst den linken, danach den rechten Teilbaum, zum Schluss bearbeite den Knoten selbst

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:
 - **Pre-Order** (Hauptreihenfolge): Bearbeite zuerst den Knoten selbst, dann besuche den linken, danach den rechten Teilbaum
 - **Post-Order** (Nebenreihenfolge): Besuche zuerst den linken, danach den rechten Teilbaum, zum Schluss bearbeite den Knoten selbst
 - **In-Order** (symmetrische Reihenfolge): Besuche zuerst den linken Teilbaum, dann bearbeite den Knoten selbst, danach besuche den rechten Teilbaum

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:
 - **Pre-Order** (Hauptreihenfolge): Bearbeite zuerst den Knoten selbst, dann besuche den linken, danach den rechten Teilbaum
 - **Post-Order** (Nebenreihenfolge): Besuche zuerst den linken, danach den rechten Teilbaum, zum Schluss bearbeite den Knoten selbst
 - **In-Order** (symmetrische Reihenfolge): Besuche zuerst den linken Teilbaum, dann bearbeite den Knoten selbst, danach besuche den rechten Teilbaum
- Manchmal auch **Reverse In-Order** (anti-symmetrische Reihenfolge): Rechter Teilbaum, Knoten, dann linker Teilbaum

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



- Oft sollen alle Knoten eines Baumes besucht und bearbeitet werden.
- 3 Vorgehensweisen (**Traversierungen**) sind üblich:
 - **Pre-Order** (Hauptreihenfolge): Bearbeite zuerst den Knoten selbst, dann besuche den linken, danach den rechten Teilbaum
 - **Post-Order** (Nebenreihenfolge): Besuche zuerst den linken, danach den rechten Teilbaum, zum Schluss bearbeite den Knoten selbst
 - **In-Order** (symmetrische Reihenfolge): Besuche zuerst den linken Teilbaum, dann bearbeite den Knoten selbst, danach besuche den rechten Teilbaum
- Manchmal auch **Reverse In-Order** (anti-symmetrische Reihenfolge): Rechter Teilbaum, Knoten, dann linker Teilbaum
- Auch das Besuchen nach Tiefenlevel von links nach rechts (**level-order**) ist denkbar

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

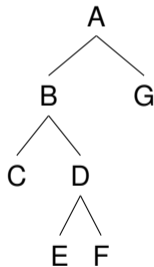
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

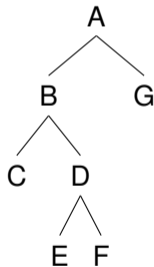
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

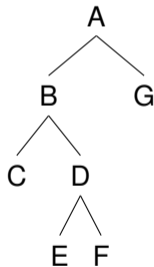
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

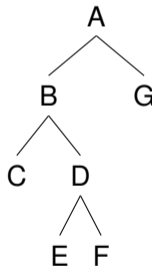
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

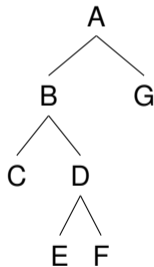
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B C

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

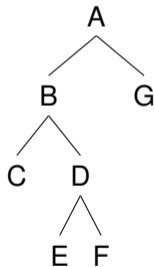
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B C D

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

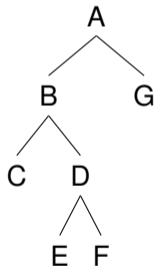
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B C D E

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

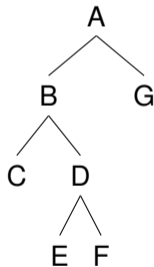
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B C D E F

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

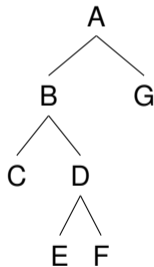
Traversierung

Suchbäume

Zusammenfassung



- Gebe den Baum *pre-order* aus



- Ausgabe: A B C D E F G

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

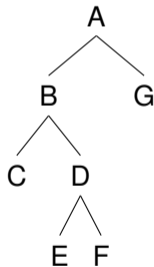
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



Der Baum

Binär**b**äume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

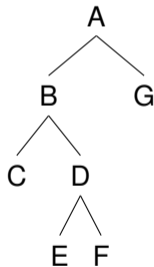
Traversierung

Such**b**äume

Zusammen-
fassung



- Gebe Baum *post-order* aus



- Ausgabe: C

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

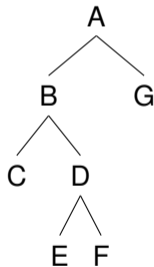
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

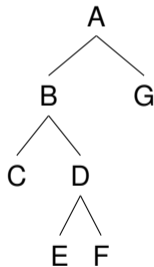
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C E

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

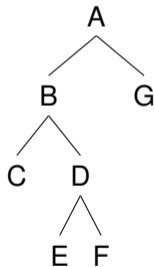
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C E F

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaf-
ten

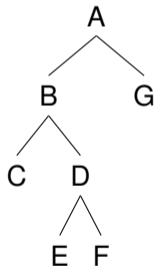
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *post-order* aus



- Ausgabe: C E F D

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

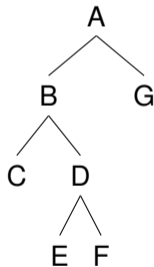
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C E F D B

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

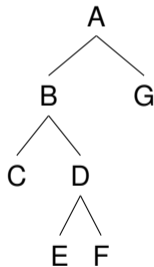
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C E F D B G

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

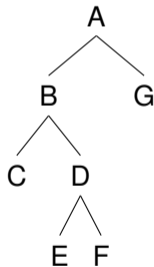
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *post-order* aus



- Ausgabe: C E F D B G A

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

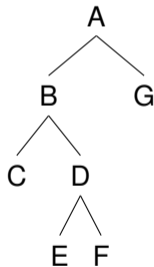
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *in-order* aus.



Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

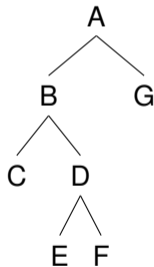
Traversierung

Suchbäume

Zusammenfassung



- Gebe Baum *in-order* aus.



- Ausgabe: C

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

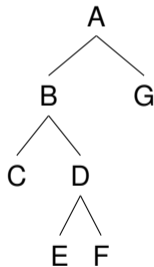
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

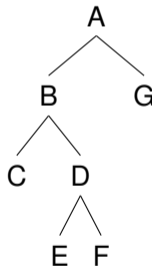
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

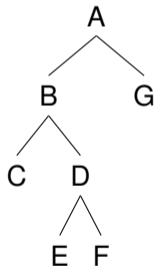
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B E

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

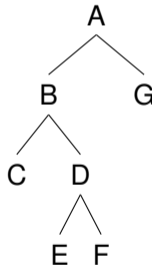
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B E D

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

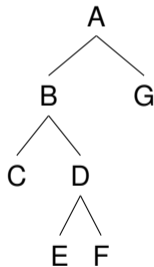
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B E D F

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

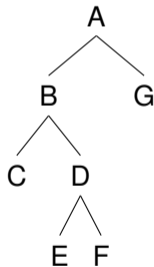
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B E D F A

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

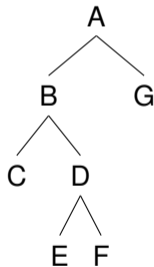
Traversierung

Suchbäume

Zusammen-
fassung



- Gebe Baum *in-order* aus.



- Ausgabe: C B E D F A G

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammen-
fassung



```
def postorder(tree : Optional[Node]):  
  match tree:  
    case Node (m, l, r):  
      postorder(l)  
      postorder(r)  
      print(m)  
def leaf(m : Any) -> Node:  
  return Node(m, None, None)  
tree = Node('*', Node('+', leaf(6), leaf(5)),  
             leaf(1))  
postorder(tree)
```

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



```
def postorder(tree : Optional[Node]):  
  match tree:  
    case Node (m, l, r):  
      postorder(l)  
      postorder(r)  
      print(m)  
def leaf(m : Any) -> Node:  
  return Node(m, None, None)  
tree = Node('*', Node('+', leaf(6), leaf(5)),  
             leaf(1))  
postorder(tree)
```

Die *post-order* Ausgabe eines Ausdrucks heißt auch **umgekehrt polnische** oder **Postfix**-Notation (HP-Taschenrechner, Programmiersprachen *Forth* und *PostScript*)

Der Baum

Binärbäume

Repräsentation

Beispiel

Funktionen auf
Bäumen

Baumeigenschaften

Traversierung

Suchbäume

Zusammenfassung



Suchbäume

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



Definition

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.
- **Suchen nach einem Objekt m beginnend beim Knoten k :**
Vergleiche m mit Markierung des aktuellen Knotens k ,

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.
- **Suchen nach einem Objekt m beginnend beim Knoten k :**
Vergleiche m mit Markierung des aktuellen Knotens k ,
 - wenn gleich, stoppe und gebe True zurück,

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.
- **Suchen nach einem Objekt m beginnend beim Knoten k :**
Vergleiche m mit Markierung des aktuellen Knotens k ,
 - wenn gleich, stoppe und gebe True zurück,
 - wenn m kleiner ist, suche im linken Teilbaum,

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.
- **Suchen nach einem Objekt m beginnend beim Knoten k :**
Vergleiche m mit Markierung des aktuellen Knotens k ,
 - wenn gleich, stoppe und gebe True zurück,
 - wenn m kleiner ist, suche im linken Teilbaum,
 - wenn m größer ist, such im rechten Teilbaum.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



- *Suchbäume* dienen dazu, Objekte schnell wieder zu finden.
- Ein **Suchbaum** ist ein binärer Baum, bei dem jeder Knoten k die **Suchbaumeigenschaft** erfüllt:
 - Alle Markierungen im linken Teilbaum sind *kleiner* als die Markierung von k , alle Markierungen im rechten Teilbaum sind *größer*.
- **Suchen nach einem Objekt m beginnend beim Knoten k :**
Vergleiche m mit Markierung des aktuellen Knotens k ,
 - wenn gleich, stoppe und gebe True zurück,
 - wenn m kleiner ist, suche im linken Teilbaum,
 - wenn m größer ist, such im rechten Teilbaum.
- Suchzeit ist proportional zur **Höhe des Baums!**
Im besten Fall *logarithmisch in der Größe des Baums*.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



Lemma

Ist h die Höhe eines Binärbaums, so ist seine Größe kleiner gleich $2^{h+1} - 1$.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



Lemma

Ist h die Höhe eines Binärbaums, so ist seine Größe kleiner gleich $2^{h+1} - 1$.

Beweis (Induktion)

Ist der Baum leer, so ist seine Höhe -1 und seine Größe 0 .

Besteht ein Baum t aus einem Knoten und zwei Teilbäumen l und r mit Höhen $h(l)$ und $h(r)$, so gilt nach Induktionsvoraussetzung $s(l) \leq 2^{h(l)+1} - 1$ und $s(r) \leq 2^{h(r)+1} - 1$.

Wegen $s(t) = 1 + s(l) + s(r)$ und $h(t) = 1 + \max(h(l), h(r))$ gilt

$$\blacksquare \quad s(t) = 1 + s(l) + s(r) \leq 1 + (2^{h(l)+1} - 1) + (2^{h(r)+1} - 1) \leq 2 \cdot 2^{\max(h(l)+1, h(r)+1)} - 1 = 2^{h(t)+1} - 1$$

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



Suche

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung



```
def search(tree : Optional[Node], item : Any) -> bool:
    if tree is None:
        return False
    elif tree.mark == item:
        return True
    elif tree.mark > item:
        return search(tree.left, item)
    else:
        return search(tree.right, item)
```

```
# smaller values left, bigger values in right subtree
nums = Node(10, Node(5, leaf(1), None),
            Node(15, leaf(12), leaf(20)))
print(search(nums, 12))
```

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



Aufbau

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammen-
fassung

Aufbauen eines Suchbaums

Immutable — unveränderlich



UNI
FREIBURG

- Aufruf `insert(tree, item)` für das Einsortieren von `item` in `tree`

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung

Aufbauen eines Suchbaums

Immutable — unveränderlich



UNI
FREIBURG

- Aufruf `insert(tree, item)` für das Einsortieren von `item` in `tree`
- Ist `tree` leer, so wird der Knoten `leaf(item)` zurückgegeben.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung

Aufbauen eines Suchbaums

Immutable — unveränderlich



UNI
FREIBURG

- Aufruf `insert(tree, item)` für das Einsortieren von `item` in `tree`
- Ist `tree` leer, so wird der Knoten `leaf(item)` zurückgegeben.
- Wenn die Markierung `tree.mark` größer als `item` ist, wird `item` in den linken Teilbaum eingesetzt und der Baum rekonstruiert (das erhält die Suchbaumeigenschaft!).

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung

Aufbauen eines Suchbaums

Immutable — unveränderlich



UNI
FREIBURG

- Aufruf `insert(tree, item)` für das Einsortieren von `item` in `tree`
- Ist `tree` leer, so wird der Knoten `leaf(item)` zurückgegeben.
- Wenn die Markierung `tree.mark` größer als `item` ist, wird `item` in den linken Teilbaum eingesetzt und der Baum rekonstruiert (das erhält die Suchbaumeigenschaft!).
- Falls `tree.mark` kleiner als `item` ist, entsprechend.

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung

Aufbauen eines Suchbaums

Immutable — unveränderlich



UNI
FREIBURG

- Aufruf `insert(tree, item)` für das Einsortieren von `item` in `tree`
- Ist `tree` leer, so wird der Knoten `leaf(item)` zurückgegeben.
- Wenn die Markierung `tree.mark` größer als `item` ist, wird `item` in den linken Teilbaum eingesetzt und der Baum rekonstruiert (das erhält die Suchbaumeigenschaft!).
- Falls `tree.mark` kleiner als `item` ist, entsprechend.
- Falls `tree.mark == item` ist nichts zu tun!

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



```
def insert(  
    tree : Optional[Node], item : Any  
    ) -> Node:  
    if tree is None:  
        return leaf(item)  
    elif tree.mark > item:  
        return Node(tree.mark,  
                    insert(tree.left, item),  
                    tree.right)  
    elif tree.mark < item:  
        return Node(tree.mark,  
                    tree.left,  
                    insert(tree.right, item))  
    else:  
        return tree
```

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



```
def insertall(tree : Optional[Node],  
             lst : list[Any]  
             ) -> Optional[Node]:  
    for key in lst:  
        tree = insert(tree, key)  
    return tree  
  
bst = insertall(None, [10, 15, 20, 12, 5, 1])
```

Der Baum

Binärbäume

Suchbäume

Definition

Suche

Aufbau

Zusammenfassung



Zusammenfassung

Der Baum

Binärbäume

Suchbäume

Zusammen-
fassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.

Der Baum

Binärbäume

Suchbäume

Zusammen-
fassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.
- Operationen über Bäumen lassen sich einfach als **rekursive Funktionen** implementieren.

Der Baum

Binärbäume

Suchbäume

Zusammen-
fassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.
- Operationen über Bäumen lassen sich einfach als **rekursive Funktionen** implementieren.
- In einem **Binärbaum** besitzt jeder Knoten genau zwei Teilbäume.

Der Baum

Binärbäume

Suchbäume

Zusammenfassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.
- Operationen über Bäumen lassen sich einfach als **rekursive Funktionen** implementieren.
- In einem **Binärbaum** besitzt jeder Knoten genau zwei Teilbäume.
- Es gibt drei Hauptarten der **Traversierung** von Binärbäumen: pre-order, post-order, in-order.

Der Baum

Binärbäume

Suchbäume

Zusammenfassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.
- Operationen über Bäumen lassen sich einfach als **rekursive Funktionen** implementieren.
- In einem **Binärbaum** besitzt jeder Knoten genau zwei Teilbäume.
- Es gibt drei Hauptarten der **Traversierung** von Binärbäumen: pre-order, post-order, in-order.
- **Suchbäume** sind Binärbäume, die die Suchbaumeigenschaft besitzen, d.h. im linken Teilbaum sind nur kleinere, im rechten nur größere Markierungen als an der Wurzel.

Der Baum

Binärbäume

Suchbäume

Zusammenfassung



- Der **Baum** ist eine Struktur, die in der Informatik allgegenwärtig ist.
- Operationen über Bäumen lassen sich einfach als **rekursive Funktionen** implementieren.
- In einem **Binärbaum** besitzt jeder Knoten genau zwei Teilbäume.
- Es gibt drei Hauptarten der **Traversierung** von Binärbäumen: pre-order, post-order, in-order.
- **Suchbäume** sind Binärbäume, die die Suchbaumeigenschaft besitzen, d.h. im linken Teilbaum sind nur kleinere, im rechten nur größere Markierungen als an der Wurzel.
- Das **Suchen** und **Einfügen** kann durch einfache rekursive Funktionen realisiert werden. **Sortierte Ausgabe** ist auch sehr einfach!

Der Baum

Binärbäume

Suchbäume

Zusammenfassung