Informatik II Woche 8

Bäume, Traversion, Heaps

Website: n.ethz.ch/~kvaratharaja/

Die Slides basieren auf den offiziellen Übungsslides der Kurswebsite: https://lec.inf.ethz.ch/mavt/informatik2/2025/

Heute

1. **Bäume**
2. **Traversierungsarten**
3. **Heaps**
4. **Inclass-Exercise**
5. **Hausaufgaben**

Einführung Datenstrukturen

- Implementation, um Daten zu speichern/manipulieren
- Verschiedene Implementierungen für unterschiedliche Anforderungen.
  - Binary search tree: effiziente Suche für sortierbare Daten
  - Heaps: effizienter Zugriff auf maximum/minimum

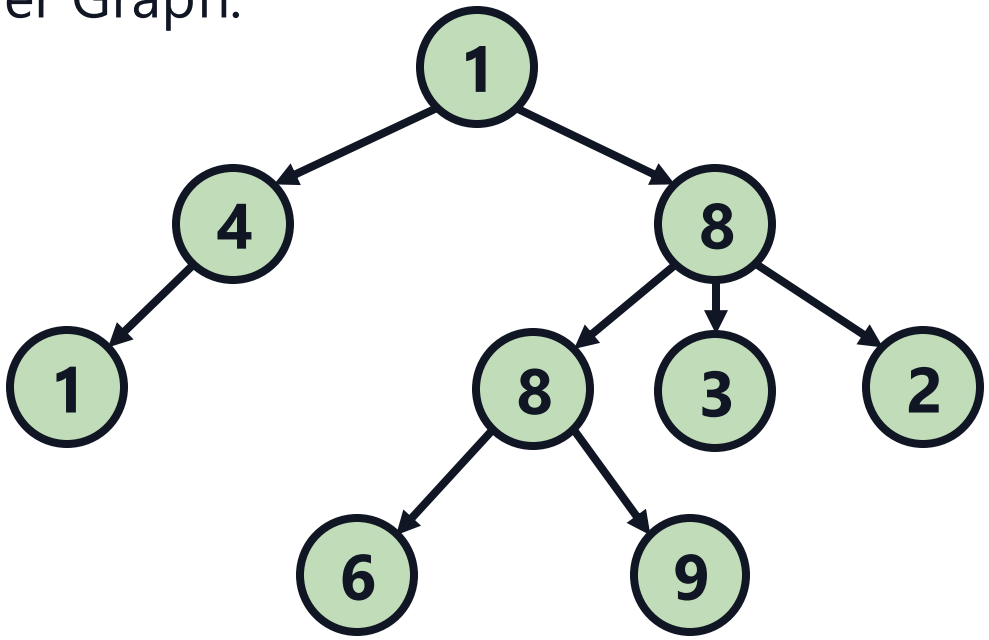
1. Bäume

Arten und Binäre Suchbäume (BST)

Was ist ein Baum?🌲

Ein **Baum** ist

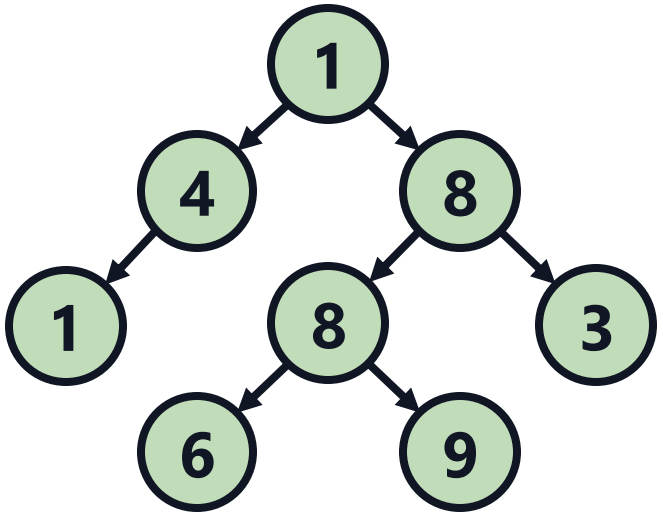
- Verallgemeinerte Liste: Knoten können mehrere Nachfolger haben.
- Spezieller Graph: Graphen bestehen aus Knoten und Kanten. Ein Baum ist ein zusammenhängender, gerichteter, azyklischer Graph.

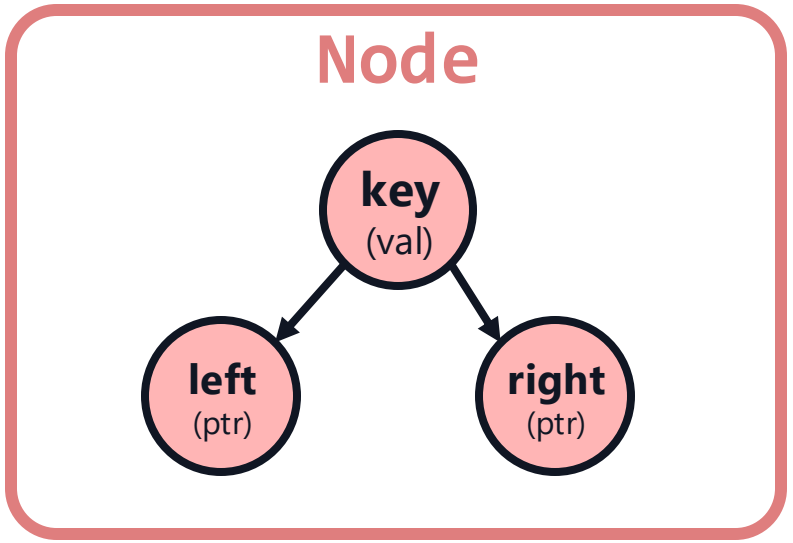


Was ist ein Binärbaum? 🌲

Ein Baum ist ein **Binärbaum**, falls:

- entweder ein Blatt, d.h. ein leerer Baum
- oder ein innerer Knoten mit zwei Bäumen T_l (linker Teilbaum) und T_r (rechter Teilbaum) als linken und rechten Nachfolger.

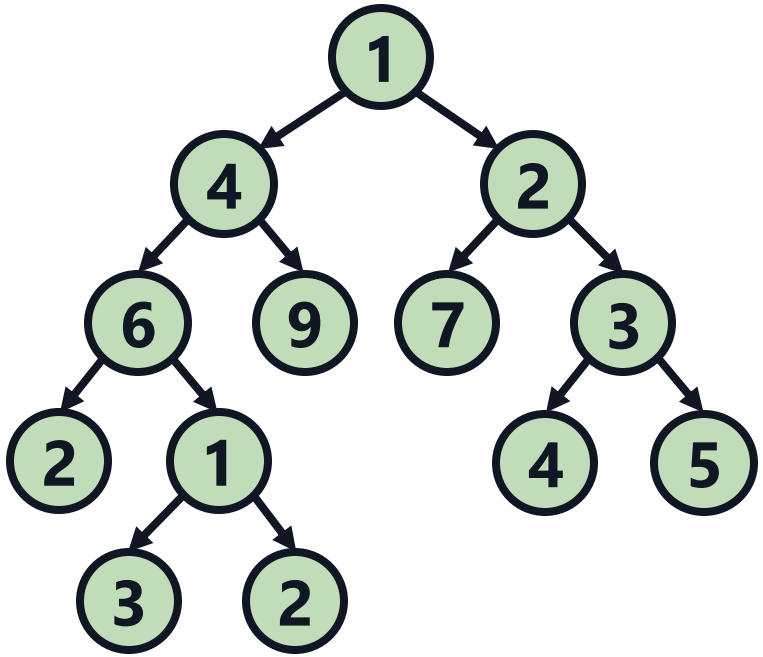




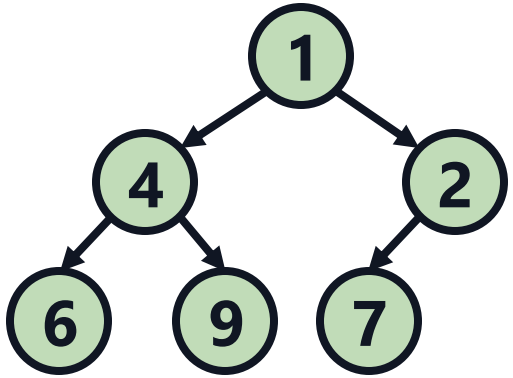
Arten von Binären Bäumen 🌲

- Es gibt drei Arten von Binärbäumen basierend auf der **Anzahl der Kinder**:

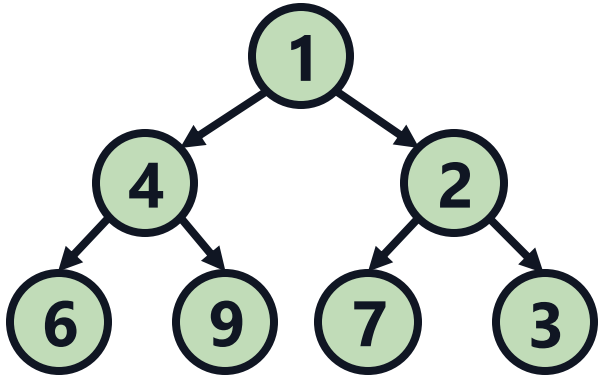
Voller Binärbaum



Vollständiger Binärbaum

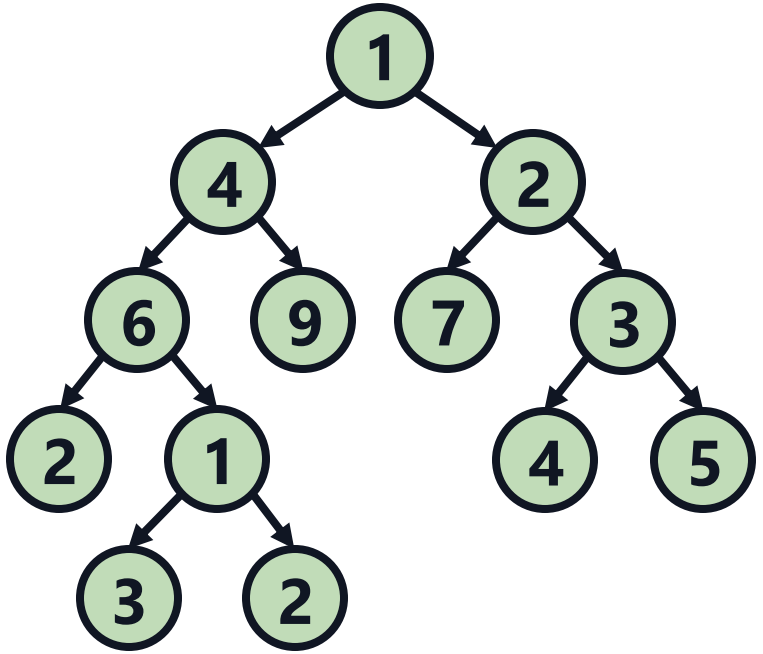


Perfekter Binärbaum



Voller Binärbaum 🌲

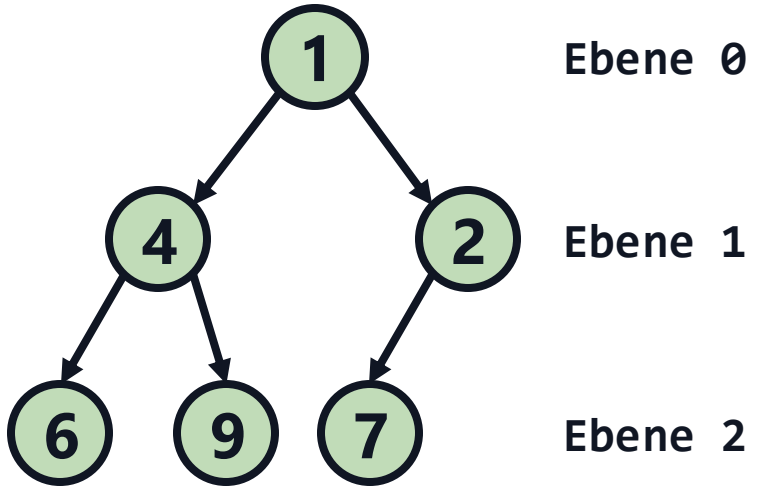
- Ein Binärbaum ist ein **voller** Binärbaum, wenn jeder Knoten 0 oder 2 Kinder hat.



Vollständiger Binärbaum 🌲

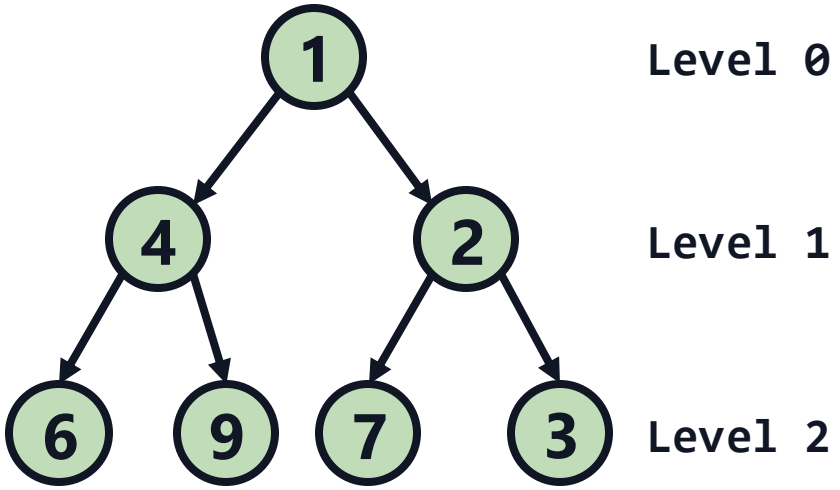
Ein Binärbaum ist **vollständig**, falls:

- alle Ebenen bis auf die letzte Ebene vollständig gefüllt sind
- die unterste Ebene von links nach rechts gefüllt ist



Perfekter Binärbaum 🌲

- Ein **Perfekter** Binärbaum ist ein Baum, bei dem alle inneren Knoten zwei Kinder haben und alle Blätter auf derselben Ebene sind.
- Einfach ausgedrückt: alle Ebenen sind perfekt gefüllt



Quiz: Binäre Bäume

Ein Binärbaum ist ein **voller** Binärbaum, falls jeder Knoten 0 oder 2 Kinder hat

Wie viele Knoten enthält ein **voller** Binärbaum mit 6 Nicht-Blatt-Knoten höchstens?

A. 6 Knoten

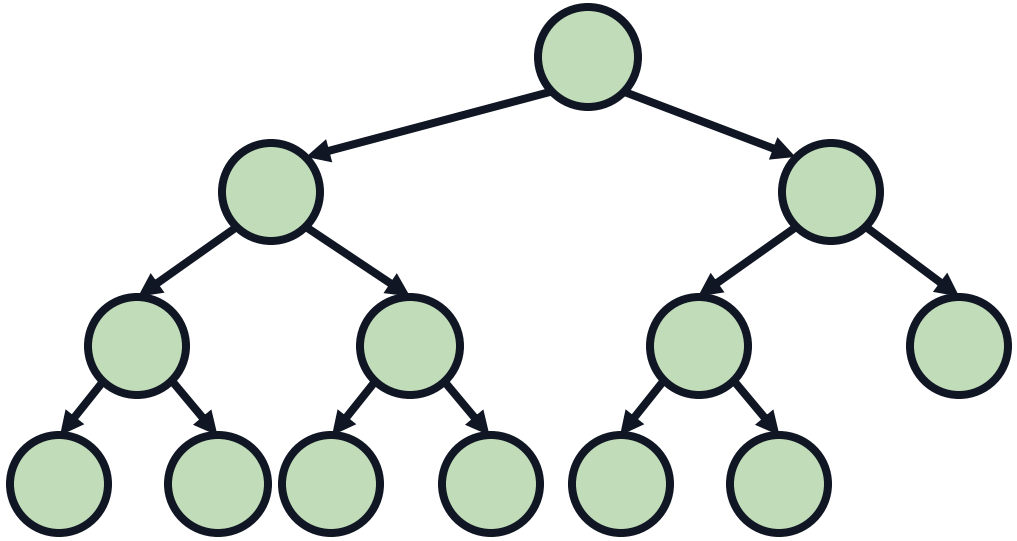
B. 9 Knoten

C. 11 Knoten

D. 13 Knoten

Quiz Lösung: Binäre Bäume

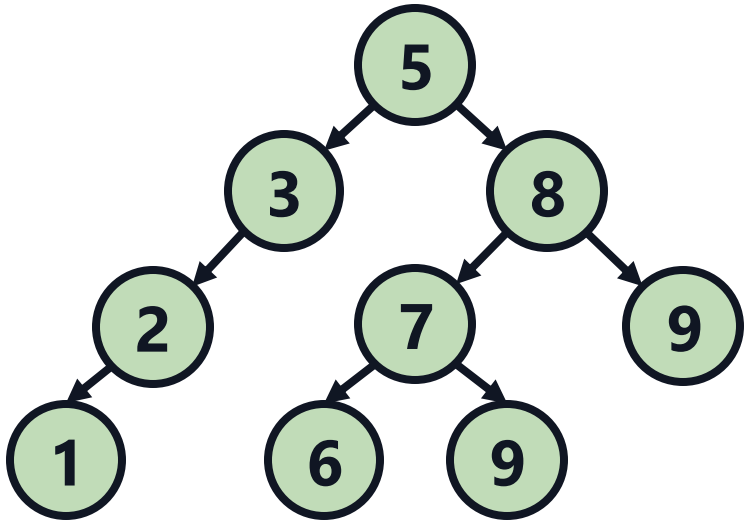
Ein voller Binärbaum mit n Nicht-Blatt-Knoten enthält $2n + 1$ Knoten. Also: $2 \times 6 + 1 = 13$.



Was ist ein Binärer Suchbaum? 🌲

Ein **binärer Suchbaum** ist ein binärer Baum, der die **Suchbaumeigenschaft** erfüllt:

- Jeder Knoten `v` speichert einen Schlüssel
- Schlüssel im linken Teilbaum `v.left` kleiner als `v.key`
- Schlüssel im rechten Teilbaum `v.right` grösser als `v.key`



Binäre Suchbäume: optimierte Suche

- Optimiert für effizientes Suchen von Zufallselementen: $\Theta(\log(n))$
- „Binäre Suche“ im BST ist kein Zufall → Rekursionsbaum der binären Suche

2. Traversierungsarten

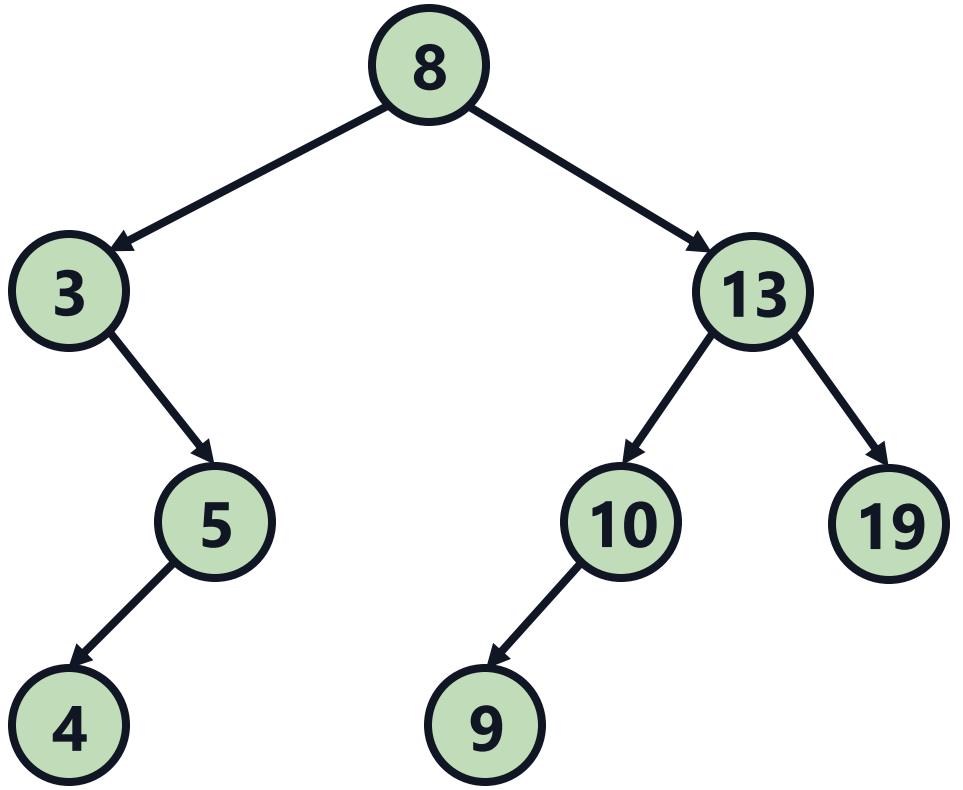
Überblick und ein Trick
kommt wahrscheinlich an der Prüfung

Traversierungsarten

- **Hauptreihenfolge** (**preorder**):
  v, then $T_{left}(v)$, then $T_{right}(v)$
  8, 3, 5, 4, 13, 10, 9, 19

- **Nebenreihenfolge** (**postorder**):
  $T_{left}(v)$, then $T_{right}(v)$, then v
  4, 5, 3, 9, 10, 19, 13, 8

- **Symmetrische Reihenfolge** (**inorder**):
  $T_{left}(v)$, then v , then $T_{right}(v)$
  3, 4, 5, 8, 9, 10, 13, 19

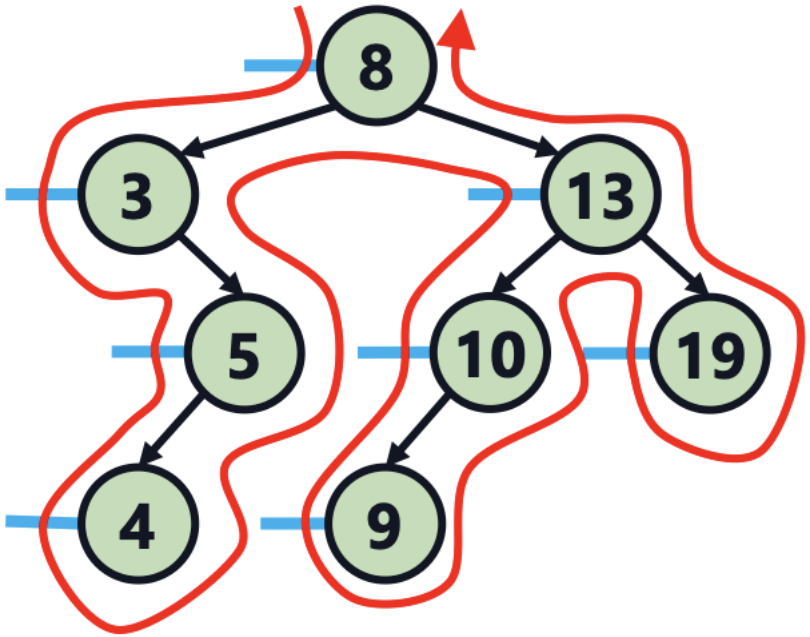


Traversierung – Trick

- **Schritt 1:** zeichne eine **Linie** an jeden Knoten

  links = preorder, unten = inorder, rechts = postorder

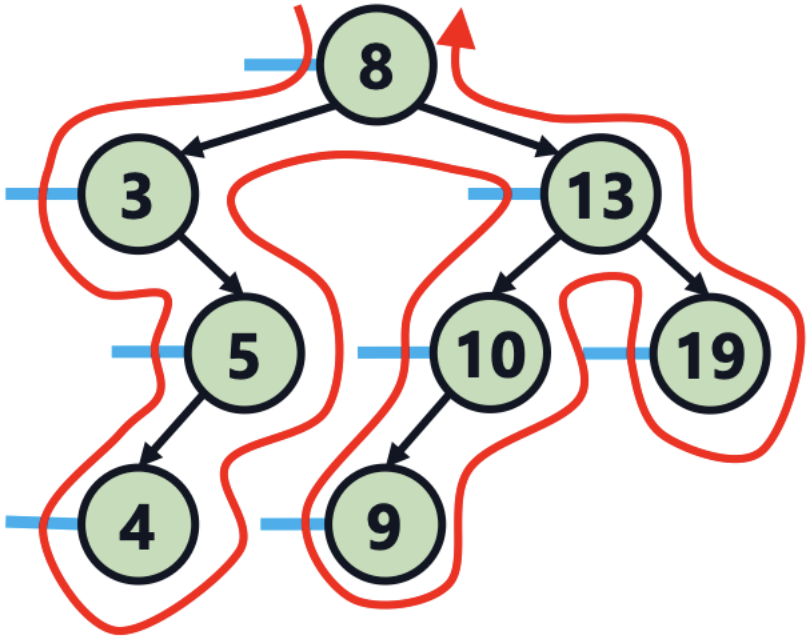
- **Schritt 2:** zeichne einen **Pfad**, der den gesamten Baum umgeht, beginnend links vom Wurzelknoten - der **Pfad** berührt die **Linie** -> nächste Schlüssel deines Traversals



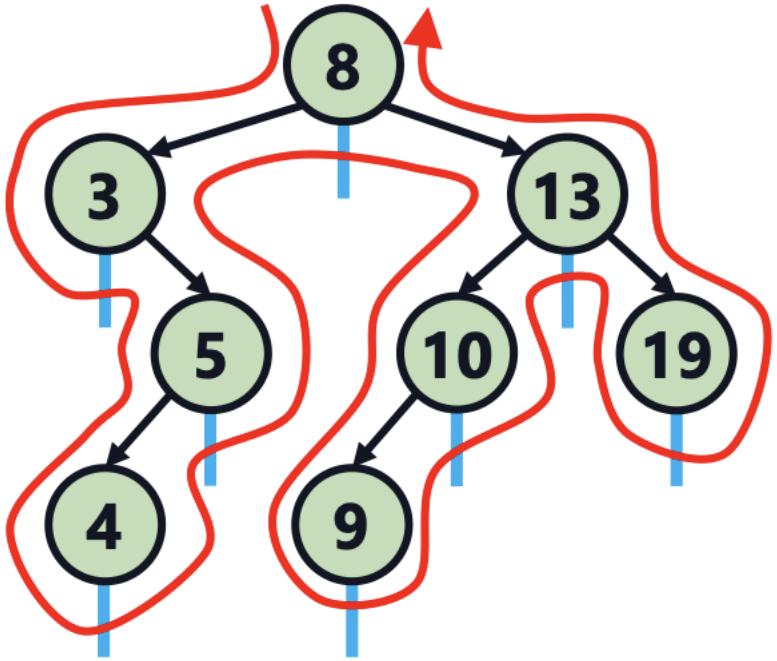
Beispiel: preorder, **Pfad** berührt **Linie** in folgender Reihenfolge:

8, 3, 5, 4, 13, 10, 9, 19

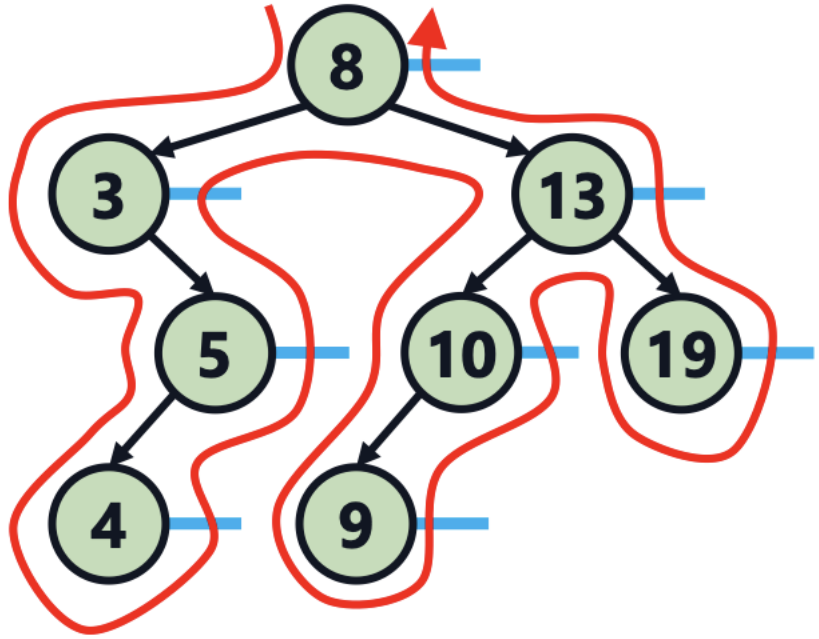
Traversierung– Trick



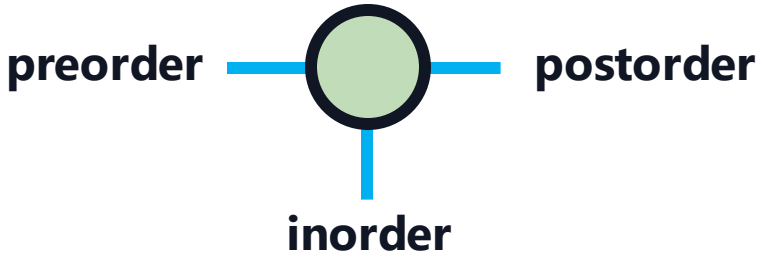
Preorder: **Pfad** berührt **Linie** in folgender Reihenfolge:
8, 3, 5, 4, 13, 10, 9, 19



Inorder: **Pfad** berührt **Linie** in folgender Reihenfolge:
3, 4, 5, 8, 9, 10, 13, 19



Postorder: **Pfad** berührt **Linie** in folgender Reihenfolge:
4, 5, 3, 9, 10, 19, 13, 8



Traversierungsarten: Quiz

- Zeichnen Sie jeweils einen binären Suchbaum, der die folgenden Traversierungen erzeugt. Ist der Baum eindeutig?

- Preorder: 4, 3, 1, 2, 8, 6, 5, 7

- Inorder: 1, 2, 3, 4, 5, 6, 7, 8

- Postorder: 1, 3, 2, 5, 6, 8, 7, 4

Geben Sie zu jeder Reihenfolge eine Zahlensequenz aus {1, . . . , 4}, die nicht aus einem gültigen binären Suchbaum stammen kann.

Exercise: Preorder

- Preorder: 4, 3, 1, 2, 8, 6, 5, 7

Erster Wert = Wurzel
Alles was < Wurzel gehört zum linken Teilbaum
Alles was > Wurzel gehört zum rechten Teilbaum
Rekursiv auf Teilbäume anwenden

- Tree ist _____ (eindeutig/nicht eindeutig?)
- Gegenbeispiel:

Exercise: Inorder

- Inorder: 1, 2, 3, 4, 5, 6, 7, 8

- Tree ist _____ (eindeutig/nicht eindeutig?
- Gegenbeispiel:

Exercise: Postorder

- Postorder: 1, 3, 2, 5, 6, 8, 7, 4

Letzter Wert = Wurzel
Alles was < Wurzel gehört zum linken Teilbaum
Alles was > Wurzel gehört zum rechten Teilbaum
Rekursiv auf Teilbäume anwenden

- Tree ist _____ (eindeutig/nicht eindeutig?)
- Gegenbeispiel:

3. Heaps

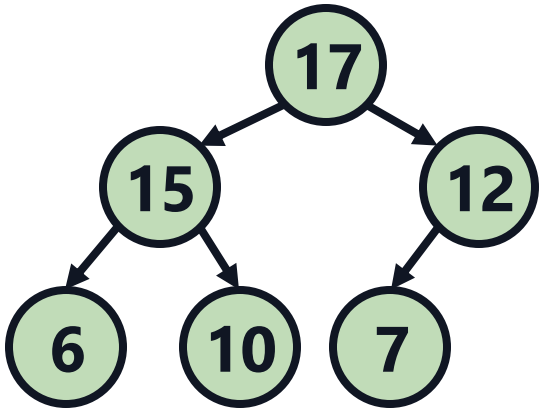
Einfügen/Löschen, Intuition

Was ist ein Heap?

Ein Binärbaum ist vollständig, falls:
- alle Ebenen bis auf die letzte Ebene vollständig gefüllt sind
- die unterste Ebene von links nach rechts gefüllt ist

- Ein **vollständiger** Binärbaum.
- Optimiert für schnelle Extraktion von min/max.
- Höhe: $\log(N+1)$.
- Implementiert als Array, Bsp: [17,15,12,6,10,7]

Es gibt zwei Arten von Heap Implementierungen:
- Max-Heaps
- Min-Heaps

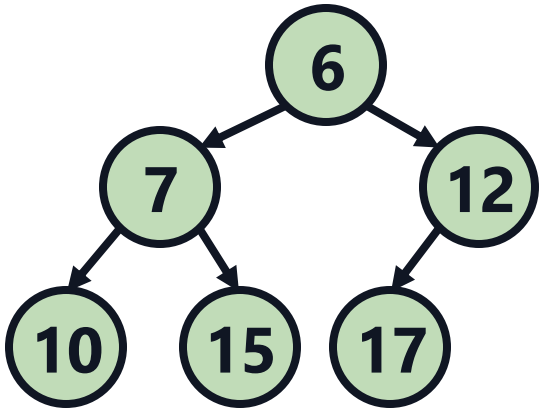


max heap

Was ist ein Min-Heap?

Ein **Min-Heap** ist ein binärer Baum wo:

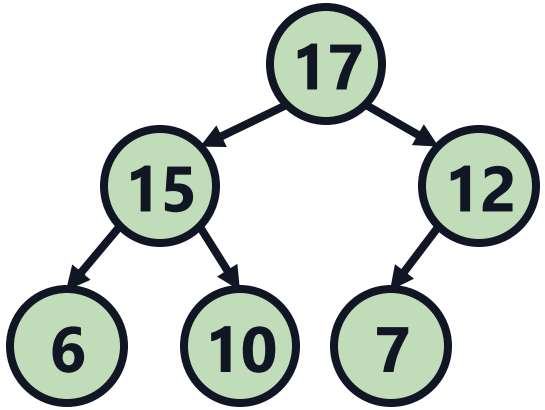
- Der am Wurzelknoten vorhandene Schlüssel ist immer **kleiner oder gleich** den Schlüsseln aller seiner Kinder.
- Dieselbe Eigenschaft muss für alle Teilbäume in diesem Binärbaum rekursiv wahr sein.
- Das **kleinste** Schlüsselelement ist daher immer an der Wurzel vorhanden.



Was ist ein Max-Heap?

Ein **Max-Heap** ist ein binärer Baum wo:

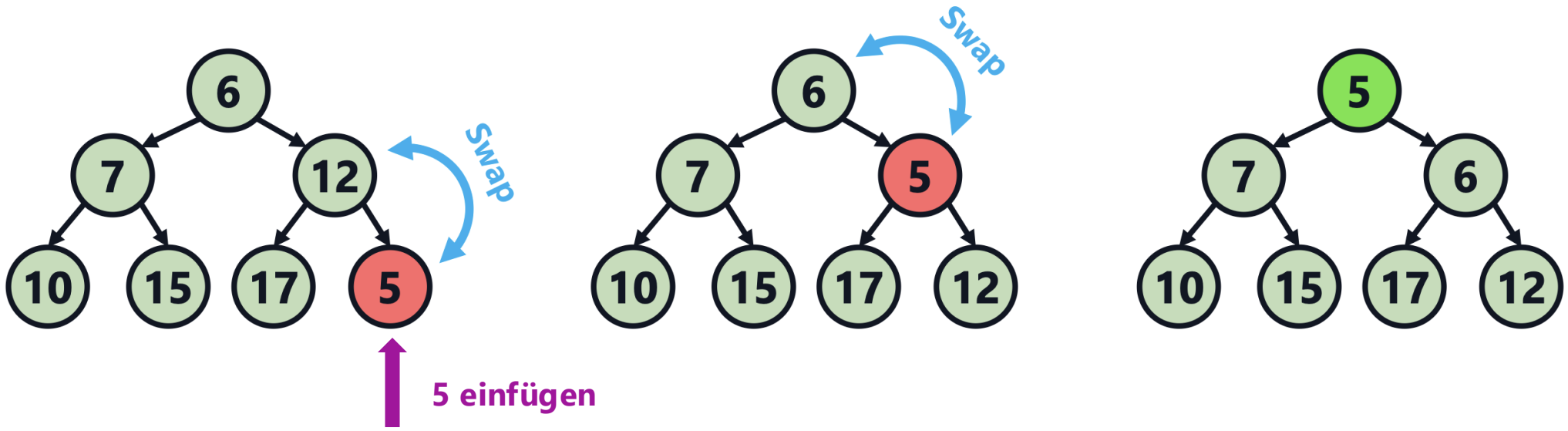
- Der am Wurzelknoten vorhandene Schlüssel ist immer **größer oder gleich** den Schlüsseln aller seiner Kinder.
- Dieselbe Eigenschaft muss für alle Teilbäume in diesem Binärbaum rekursiv wahr sein.
- Das **grösste** Schlüsselelement ist daher immer an der Wurzel vorhanden.



Element einfügen in ein Heap (SiftUp)

- **Einfügen** eines Elements in einen freien Platz unten rechts im Baum
- **Verletzt** die Heap-Eigenschaft -> **nach oben swappen**, bis es **korrekt** ist
- Funktion zum Swappen nach oben ist **„SiftUp“**.

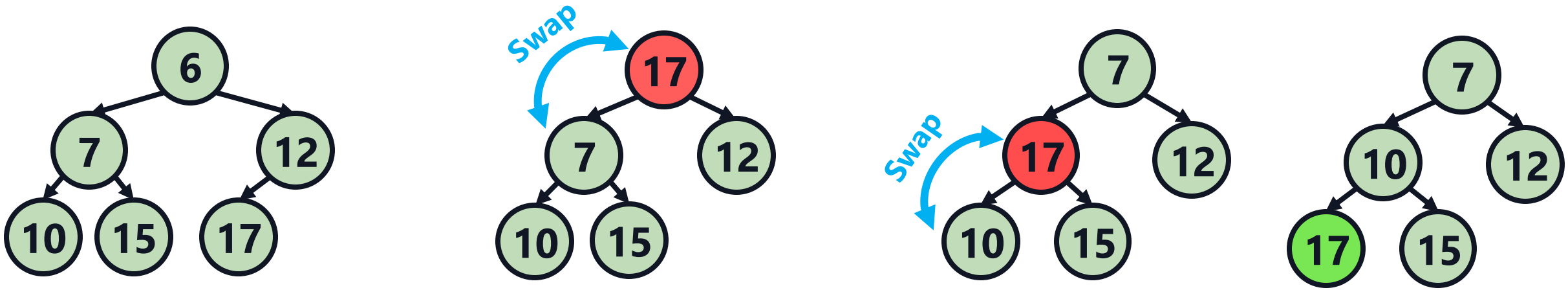
Beispiel: füge 5 zum Min-Heap hinzu:



Max/Min löschen (SiftDown)

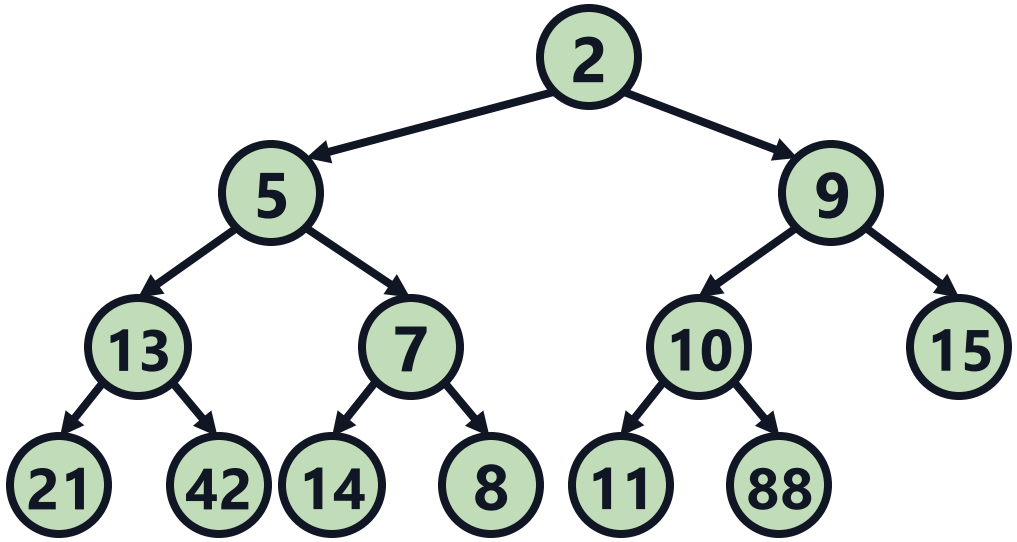
- **Ersetze** die Wurzel des Heaps durch das Element unten rechts im Baum.
- Dies **verletzt** die Heap-Eigenschaft -> **Swap nach unten**, bis es **stimmt**.
- Die Funktion zum Swappen nach unten heisst **„SiftDown“.**
- SiftDown für max-heap: immer in **Richtung des grösseren Childs** tauschen.
- SiftDown für min-heap: immer in **Richtung des kleineren Childs** tauschen.

Beispiel: Entferne 6 vom Min-Heap:



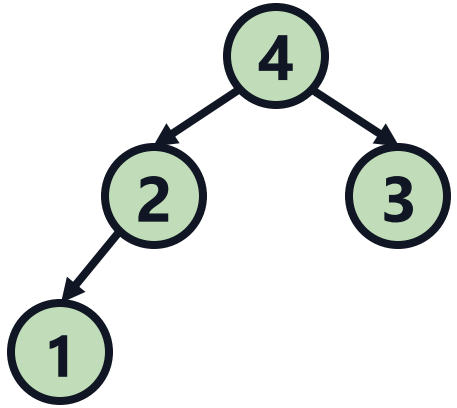
Quiz: Min-Heaps

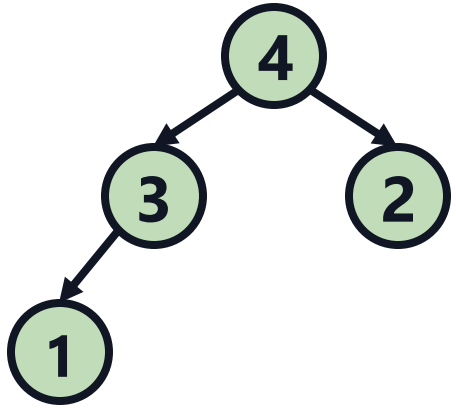
Führen Sie auf folgendem Min-Heap eine Extract-Min (entferne den kleinsten Schlüssel) Operation aus, wie in der Vorlesung vorgestellt, einschliesslich der Wiederherstellung der Heap-Bedingung. Wie sieht der Heap nach der Operation aus?

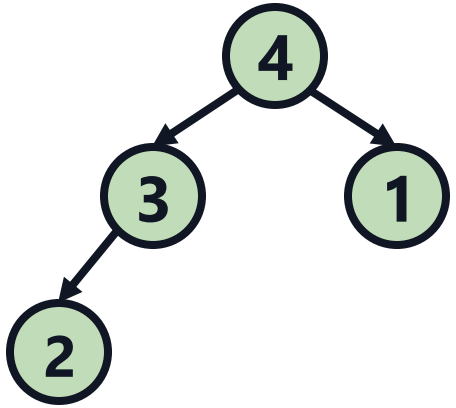


Anzahl Max-Heaps

- Sei $N(n)$ die Anzahl verschiedener Max-Heaps, welche aus allen Schlüsseln $1, 2, \ldots, n$ gebildet werden können. Beispielsweise ist $N(1) = 1,\ N(2) = 1,$ $N(3) = 2,\ N(4) = 3$ und $N(5) = 8.$
  Finde die Werte $N(6)$ und $N(7)$.
- Beispiel: 4 Knoten

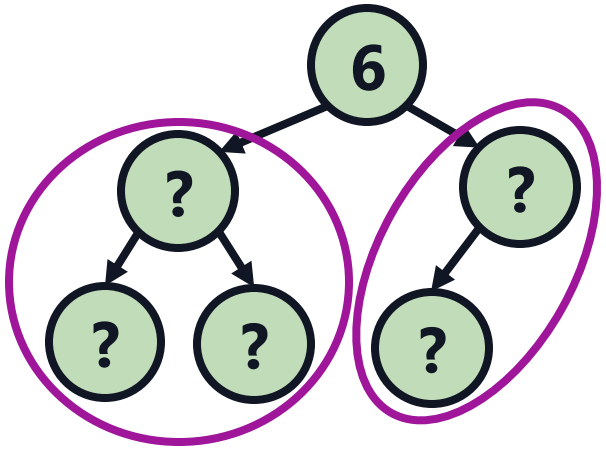






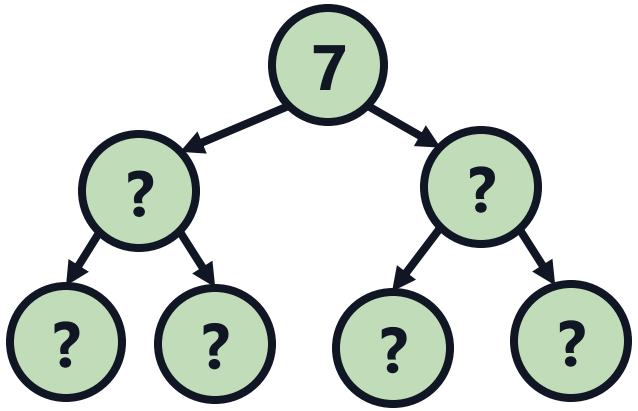
Beispiel für 6 Knoten

- Ein die Elemente 1, 2, 3, 4, 5, 6 enthaltender Max-Heap sieht so aus:



Beispiel für 7 Knoten

- Ein die Elemente 1, 2, 3, 4, 5, 6, 7 enthaltender Max-Heap sieht so aus:



Komplexitätsanalyse von Min-Heap und Max-Heap

- Erhalten des grössten oder kleinsten Elements: $\mathbf{O(1)}$
- Element in Max-Heap oder Min-Heap einfügen: $\mathbf{O(\log N)}$
- Grösstes oder kleinstes Element entfernen: $\mathbf{O(\log N)}$

Überblick: Vergleich zw. BSTs und Heaps

Operation	Binary Search Tree	Max/min-Heap
Einfügen	Nach Schlüssel suchen, Bei erreichtem leeren Blatt (null) einfügen	Zuhinterst im Array einfügen, Heap-Bedingung wiederherstellen: siftUp (Aufsteigen lassen)
Löschen	Schlüssel k durch symm. Nachfolger n ersetzen	Schlüssel durch hinterstes Arrayelement ersetzen. Heap-Bedingung wiederherstellen: siftDown oder siftUp.

Übung: einfügen in BSTs vs. Heaps

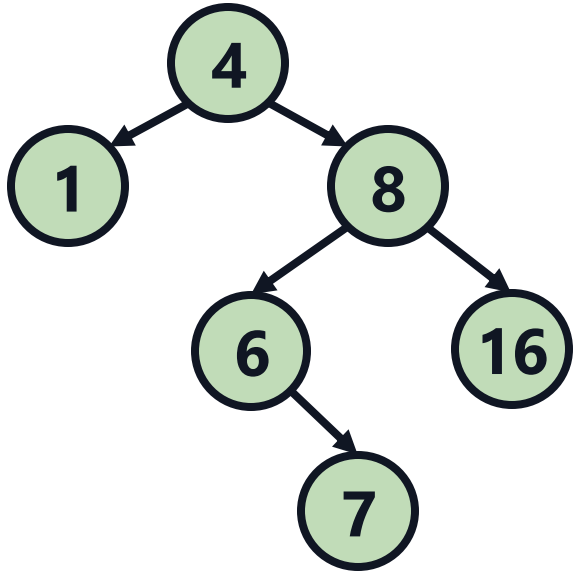
Übung: füge 4, 8, 16, 1, 6, 7 in einen leeren Baum/Min-Heap ein.

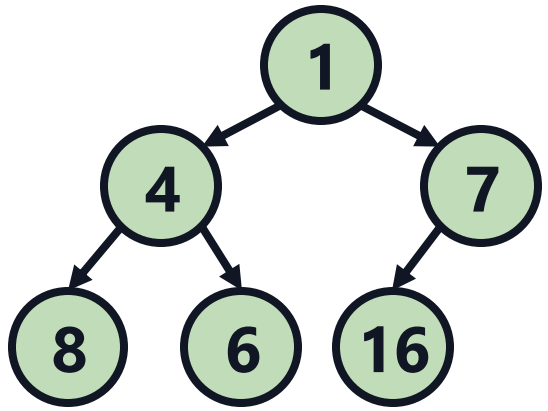
Operation	Binary Search Tree	Max/min-Heap
Insertion	Suche nach Element im Baum, Einfügen bei erreichtem leaf	Ganz hinten rechts einfügen und Element nach oben swappen (SiftUp)

Übung: BSTs vs. Heaps delete

Übung: Lösche 4 vom Tree/Min-Heap.

Operation	Binary Search Tree	Max/min-Heap
Deletion	Ersetze Knoten mit symmetrischen Nachfolger (2 Kinder)	Ersetze mit dem letzten Element und swape runter (SiftDown)



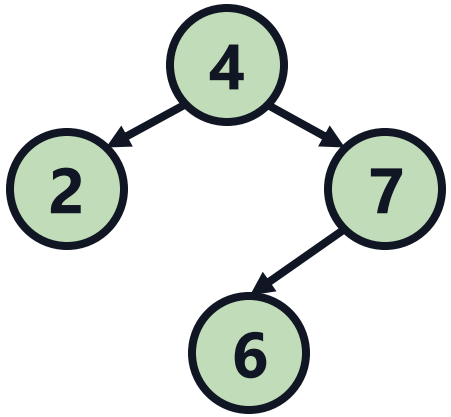


Quiz: Schlüssel löschen

Wenn Sie zwei Schlüssel aus einem Binären Suchbaum löschen wollen, spielt es dann eine Rolle, in welcher Reihenfolge Sie dies tun? Mit anderen Worten, ist das Löschen kommutativ?

A. Reihenfolge spielt eine Rolle

B. Reihenfolge spielt keine Rolle



4. In-Class Exercise

- Kinder (Node i): {2i, 2i+1}
- Eltern (Node i): i//2

In-class Exercise: Alte Prüfungsaufgabe

In der folgenden Tabelle ist ein Min-Heap in seiner üblichen Form gespeichert.

Wie sieht die Tabelle aus, nachdem die Zahl 17 eingefügt wurde?

1	2	3	4	5	6	7	8	9	10	11	12
2	9	7	25	23	32	26	48	37	39	39	37

1	2	3	4	5	6	7	8	9	10	11	12	13

5. Hausaufgaben

Exercise 7: Trees

Auf https://expert.ethz.ch/mycourses/SS25/mavt2/exercises

- Binary Search Trees and Heaps
- Implementing a Binary Search Tree
- Concatenating Heaps
- Heapsort

Fällig bis Montag 14.04.2025, 20:00 CET

NO HARDCODING

Fragen?

Feedback?

Zu schnell? Zu langsam? Weniger Theorie, mehr Aufgaben?
Dankbar für Feedback am besten mir direkt sagen oder Mail schreiben

Credits

Die Slide(-templates) stammen ursprünglich von Julian Lotzer und Daniel Steinhauser, vielen Dank!

→Checkt ihre Websites ab für zusätzliches Material in Informatik I, Informatik II und Stochastik & Machine Learning.

- https://n.ethz.ch/~jlotzer/
- https://n.ethz.ch/~dsteinhauser/