Informatik II

Übung 5

FS 2020

Program Today

1. Feedback of last exercise
2. Repetition Theory

Repetition: Binary Trees, Inserting a Key

Binary Search Trees

- Search for Key.
- Insert at the reached empty leaf (`null`).

MinHeap

- Insert at the very back of the Array.
- Restore Heap-Condition: `siftUp` (climb successively).

Repetition: Binary Trees, Inserting a Key

Binary Search Trees

- Search for Key.
- Insert at the reached empty leaf (`null`).

MinHeap

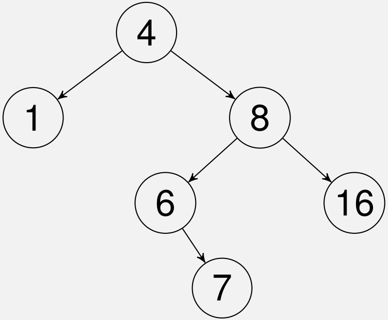
- Insert at the very back of the Array.
- Restore Heap-Condition: `siftUp` (climb successively).

Exercise: Insert $4, 8, 16, 1, 6, 7$ into empty Tree/Heap.

Repetition: Binary Trees, Inserting a Key

Binary Search Trees

- Search for Key.
- Insert at the reached empty leaf (`null`).



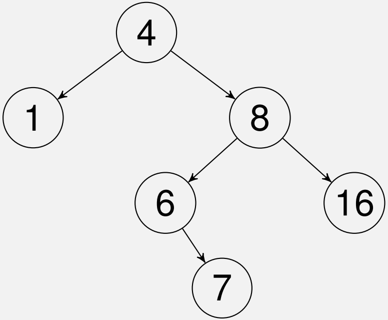
MinHeap

- Insert at the very back of the Array.
- Restore Heap-Condition: `siftUp` (climb successively).

Repetition: Binary Trees, Inserting a Key

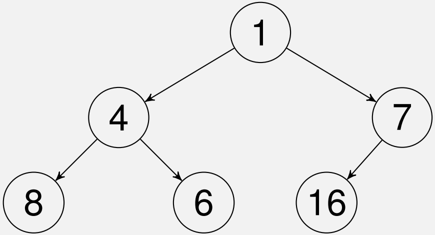
Binary Search Trees

- Search for Key.
- Insert at the reached empty leaf (`null`).



MinHeap

- Insert at the very back of the Array.
- Restore Heap-Condition: `siftUp` (climb successively).



Repetition: Binary Trees, Deleting a Key

Binary Search Trees

- Replace key k by symmetric successor n.
- Careful: What about right child of n?

MinHeap

- Replace key by last element of the array.
- Restore Heap-Condition: `siftDown` *or* `siftUp`.

Repetition: Binary Trees, Deleting a Key

Binary Search Trees

- Replace key k by symmetric successor n.
- Careful: What about right child of n?

MinHeap

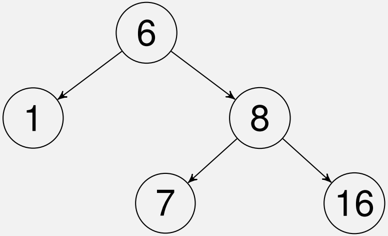
- Replace key by last element of the array.
- Restore Heap-Condition: `siftDown` *or* `siftUp`.

Exercise: Delete 4 from Example Tree/Heap.

Repetition: Binary Trees, Deleting a Key

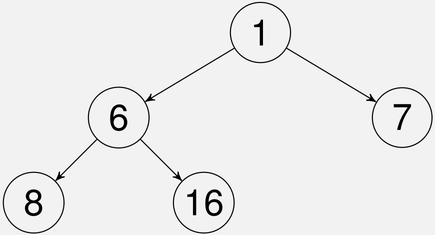
Binary Search Trees

- Replace key k by symmetric successor n.
- Careful: What about right child of n?



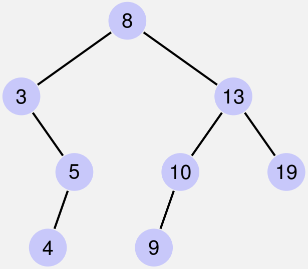
MinHeap

- Replace key by last element of the array.
- Restore Heap-Condition: `siftDown` *or* `siftUp`.



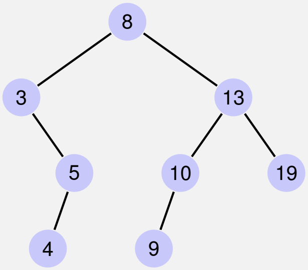
Traversal possibilities

- preorder: v, then $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$.
- postorder: $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$, then v.
- inorder: $T_{\text{left}}(v)$, then v, then $T_{\text{right}}(v)$.



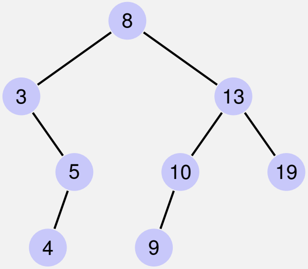
Traversal possibilities

- preorder: v, then $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$.
  8, 3, 5, 4, 13, 10, 9, 19
- postorder: $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$, then v.
- inorder: $T_{\text{left}}(v)$, then v, then $T_{\text{right}}(v)$.



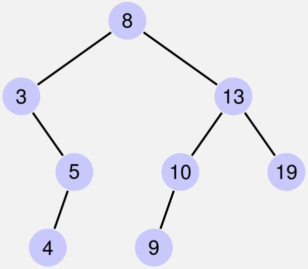
Traversal possibilities

- preorder: v, then $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$.
  8, 3, 5, 4, 13, 10, 9, 19
- postorder: $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$, then v.
  4, 5, 3, 9, 10, 19, 13, 8
- inorder: $T_{\text{left}}(v)$, then v, then $T_{\text{right}}(v)$.



Traversal possibilities

- preorder: v, then $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$.
  8, 3, 5, 4, 13, 10, 9, 19
- postorder: $T_{\text{left}}(v)$, then $T_{\text{right}}(v)$, then v.
  4, 5, 3, 9, 10, 19, 13, 8
- inorder: $T_{\text{left}}(v)$, then v, then $T_{\text{right}}(v)$.
  3, 4, 5, 8, 9, 10, 13, 19



Quiz (tricky)

Draw a binary search tree each that represents the following traversals. Is the tree unique?

inorder	1 2 3 4 5 6 7 8
preorder	4 3 1 2 8 6 5 7
postorder	1 3 2 5 6 8 7 4

Provide for each order a sequence of numbers from $\{1, \ldots, 4\}$ such that it cannot result from a valid binary search tree

Answers

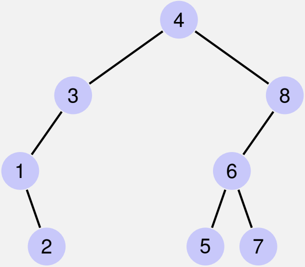
inorder: any binary search tree with numbers $\{1, \ldots, 8\}$ is valid.

The tree is not unique

There is no search tree for any non-sorted sequence.
Counterexample 1 2 4 3

Answers

preorder 4 3 1 2 8 6 5 7

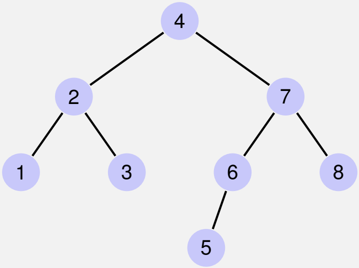


Tree is unique

It mus hold recursively that first there is a group of numbers with lower and then with higher number than the first value. Counterexample: 3 1 4 2

Answers

postorder 1 3 2 5 6 8 7 4

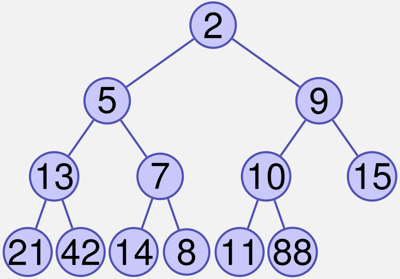


Tree is unique

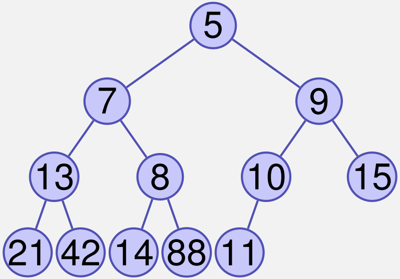
Construction here: `https://www.techiedelight.com/build-binary-search-tree-from-postorder-sequence/`, similar argument as before, but backwards. Counterexample 4 2 1 3

Heap

On the following Min-Heap, perform an extract-min operation, including re-establishing the heap-condition, as shown in class. What does the heap look like after the operation?



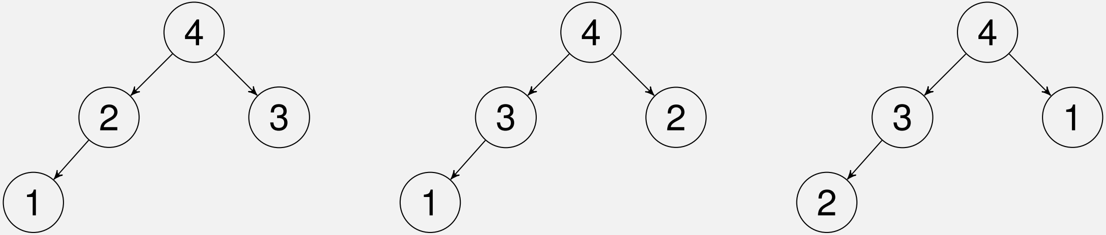
Solution



Number of MaxHeaps on n distinct keys

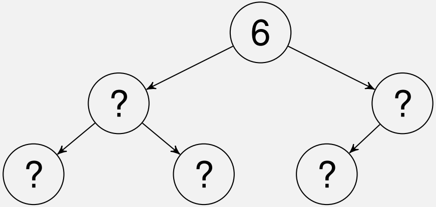
Let $N(n)$ denote the number of distinct Max-Heaps which can be built from all the keys $1, 2, \ldots, n$. For example we have $N(1) = 1,\ N(2) = 1,\ N(3) = 2,\ N(4) = 3$ und $N(5) = 8$.

Find the values $N(6)$ and $N(7)$.



Number of MaxHeaps on n distinct keys

A MaxHeap containing the elements $1, 2, 3, 4, 5, 6$ has the structure:



Number of combinations to choose elements for the left subtree: $\binom{5}{3}$.

$$\Rightarrow N(6) = \binom{5}{3} \cdot N(3) \cdot N(2) 10 \cdot 2 \cdot 1 = 20.$$

and $N(7) = \binom{6}{3} \cdot N(3) \cdot N(3) = 20 \cdot 2 \cdot 2 = 80.$

Questions / Suggestions?