Komplexität des Heap-Aufbaus

- Wir haben $O(n)$ Aufrufe von heapify mit jeweils einem Aufwand von $O(\log n)$.
- Ergibt einen Gesamtaufwand von $O(n \log n)$.
- Bei genauerem Hinsehen stellt man jedoch fest, dass heapify zunächst auf kleine Unterbäume angewendet wird.
- Da es in einem Heap mit n Elementen höchstens $\lceil n/2^{h+1} \rceil$ Elemente der Höhe h gibt:
- $\sum_{h=0}^{\lfloor \log_2 n \rfloor} \lceil n/2^{h+1} \rceil O(h) = O(n \sum_{h=0}^{\lfloor \log_2 n \rfloor} \frac{h}{2^h})$
- Die letzte Summe kann als Konstante abgeschätzt werden:
- $\sum_{h=0}^{\lfloor \log_2 n \rfloor} \frac{h}{2^h} \leq \sum_{h=0}^{\infty} \frac{h}{2^h} = \frac{1/2}{(1-1/2)^2} = 2$
- $O(n \sum_{h=0}^{\lfloor \log_2 n \rfloor} \frac{h}{2^h}) \leq O(2n) = O(n)$