Heapsort

- $\Theta(n \lg n)$ like merge sort
- In-place like insertion sort
- Uses a data structure - **heap**

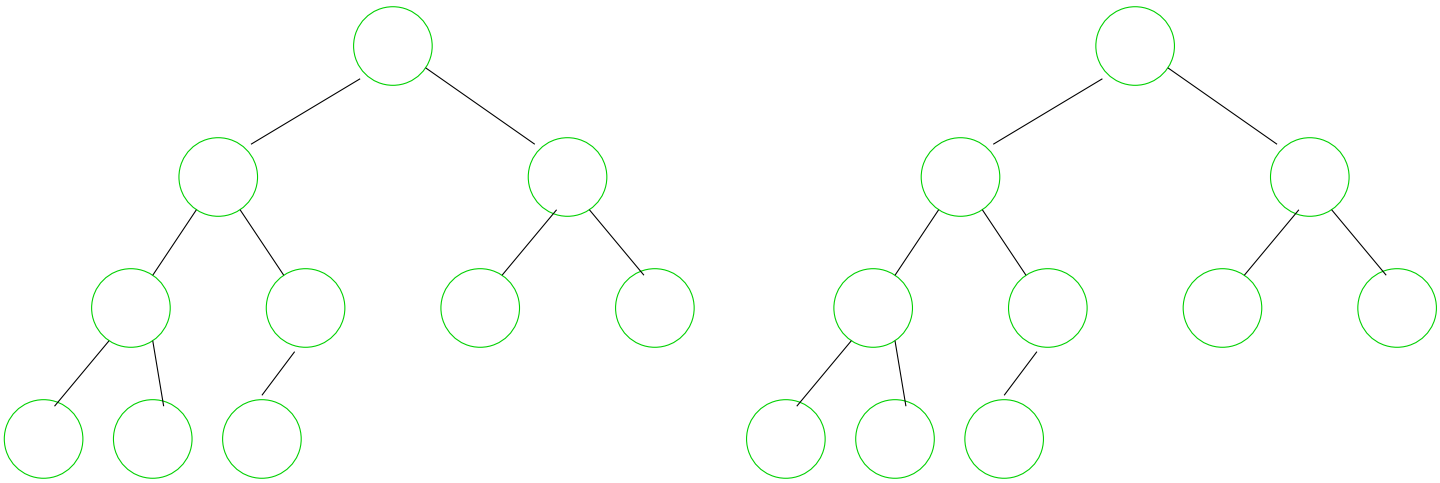
HEAP

Structure:

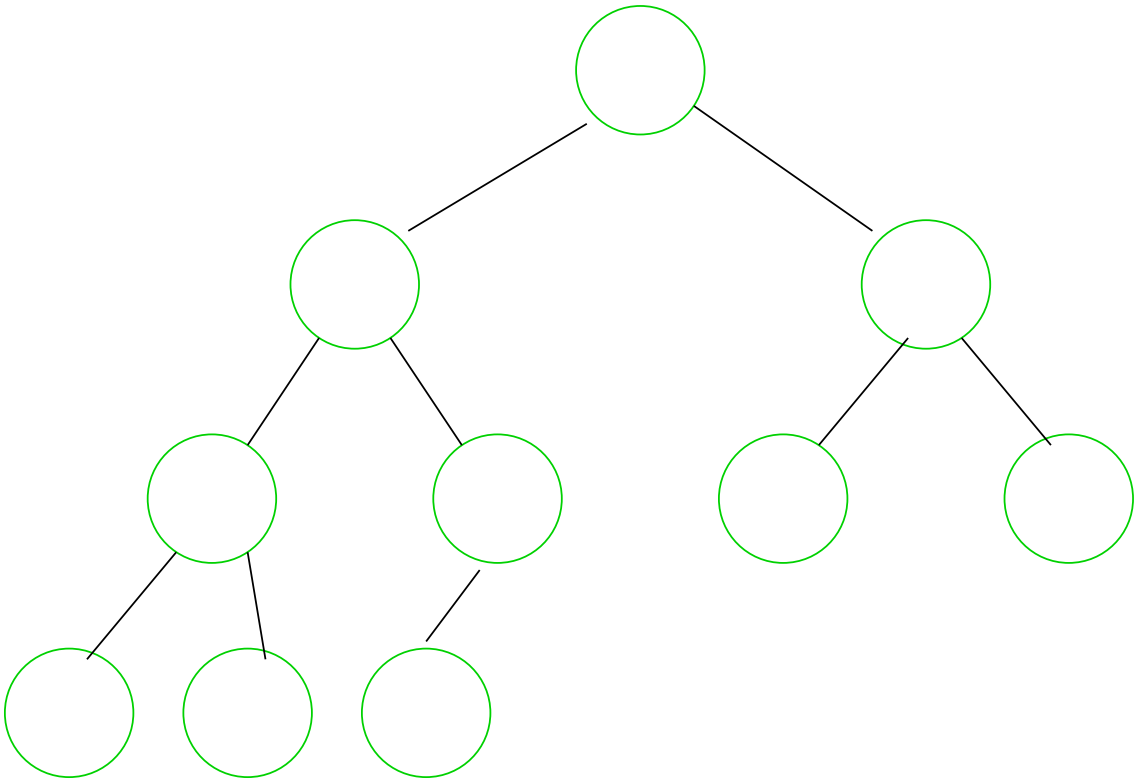
- labeled binary tree
- all levels full, except possibly last
- nodes in last level as far left as possible

Max-heap property:

- For all nodes x, label(x) $\geq$ labels of children of x.



Implementation



10	9	7	8	4	6	1	3	5	2
1	2	3	4	5	6	7	8	9	10

If node x is in location i

$$\text{lchild}(x) \longrightarrow 2i$$

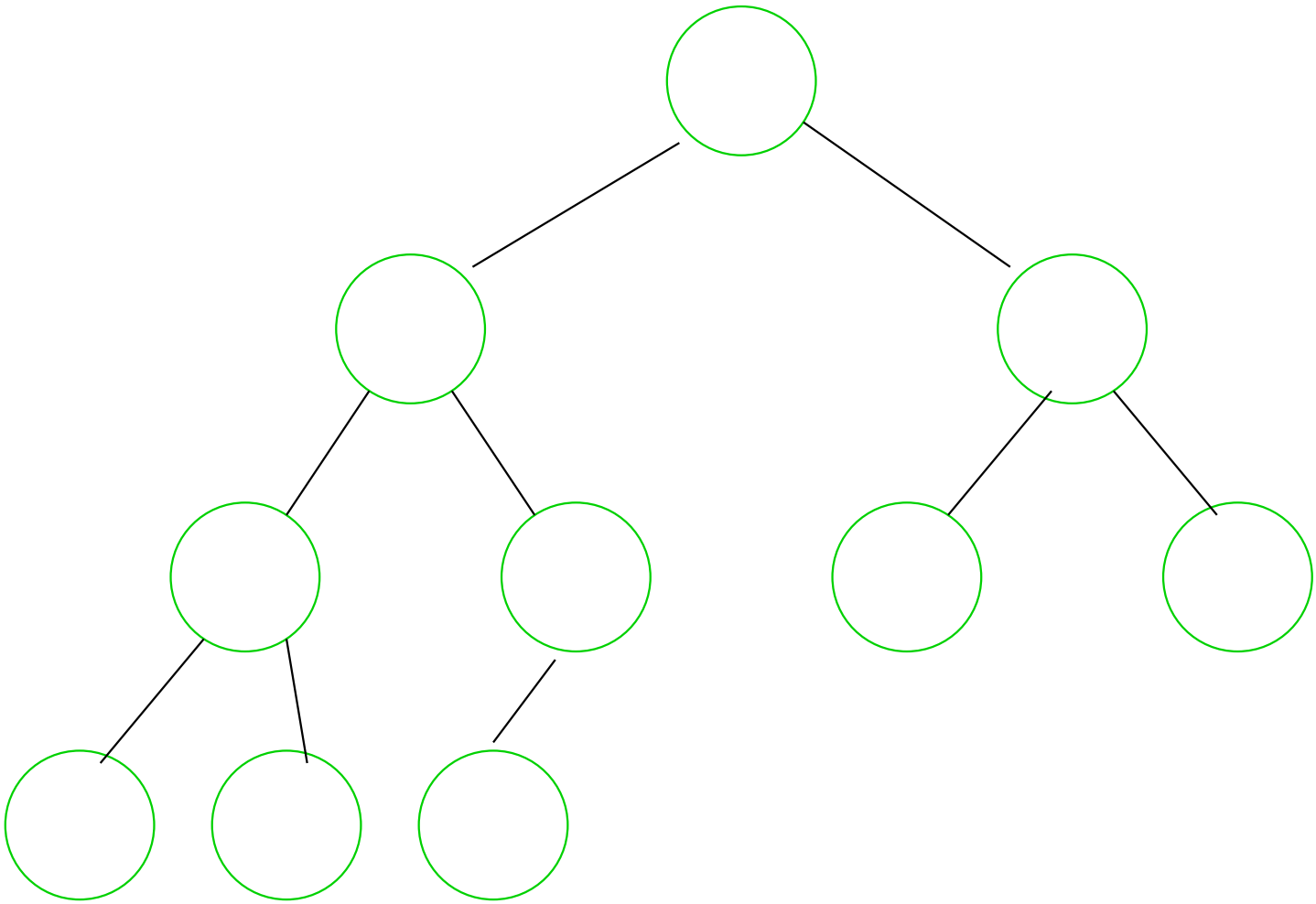
$$\text{rchild}(x) \longrightarrow 2i + 1$$

$$\text{parent}(x) \longrightarrow \lfloor i/2 \rfloor$$

How to construct heap?
How to use heap?

Heapify:

"Fixes" tree which is heap except possibly at root.



(Root value filters down.)

Heapify(x)

▷ Assumes tree rooted at x is heap except possibly at root

if x is not a leaf **then**
  let y be child of x with maximum value
  if value(x) $<$ value(y) **then**
    swap values of x and y
    Heapify(y)

Max. # of compares for Heapify(x) in tree with n nodes?

(2 * height of x in tree)

Height h of heap with n nodes

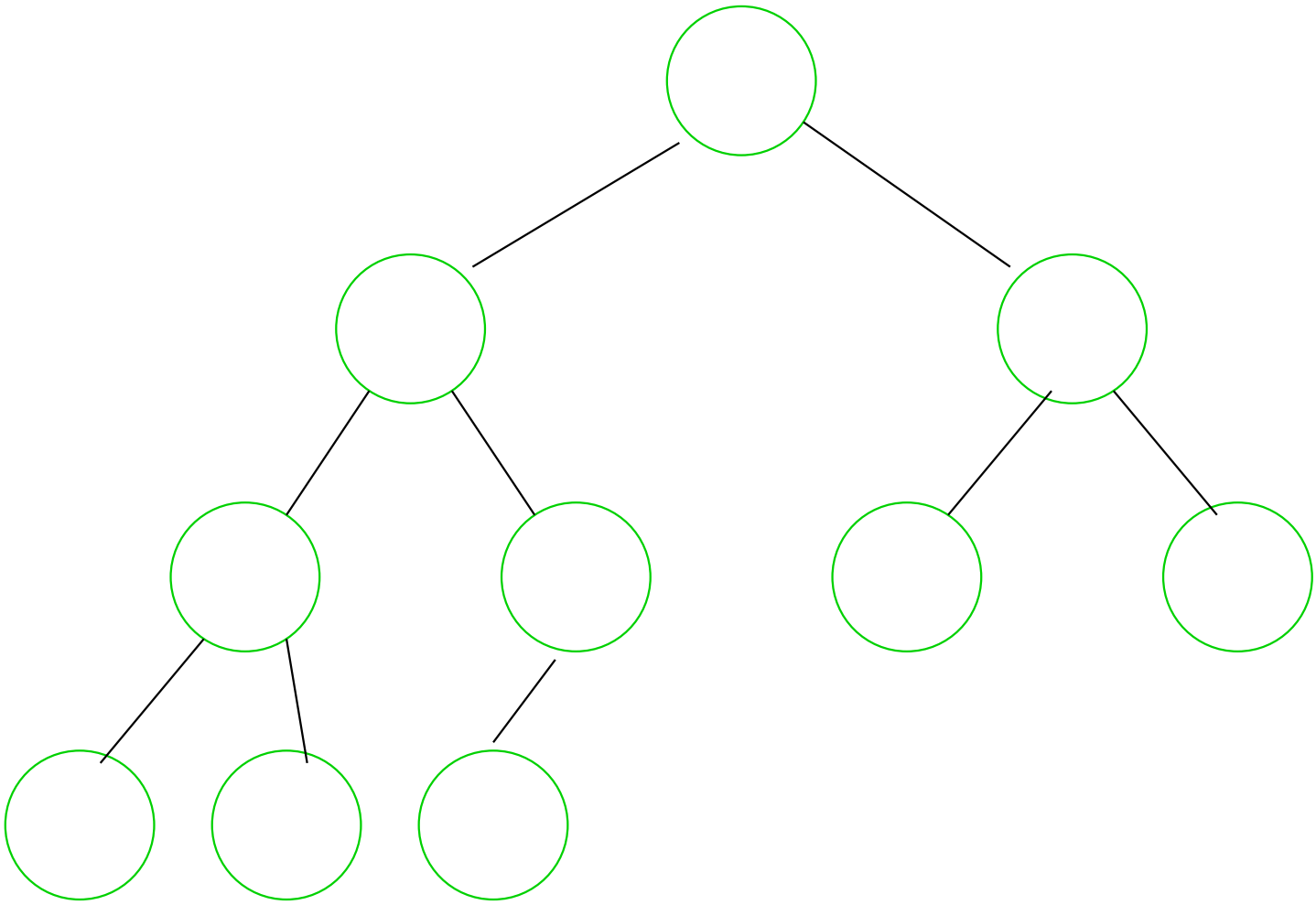
$$1 + \sum_{i=0}^{h-1} 2^i \;\leq\; n \;\leq\; \sum_{i=0}^{h} 2^i$$

$$2^h \;\leq\; n \;\leq\; 2^{h+1} - 1$$

How to organize n elements into a heap?

1. Place elements arbitrarily in tree.

2. **Buildheap:** Heapify(x) as x runs through all nodes, from bottom to top level, right-to-left across levels.



Array implementation

Place elements in $A[1..n]$, then:

Buildheap

for $i \leftarrow n$ **downto** 1 **do** (*)

Heapify(i)

(*) could start at $\lfloor n/2 \rfloor$

Total # of compares:

$$\leq \sum_{i=1}^{n} (2 * \text{ height of node } i)$$

Upper Bound:

$$\sum_{i=1}^{n} 2 * \lfloor \lg n \rfloor = 2n \lfloor \lg n \rfloor$$

Better Bound?

Better Bound

Let $h = \lfloor \lg n \rfloor$.

$$\sum_{i=1}^{n} (2 * \text{ height of node } i)$$

$$= \sum_{j=0}^{h} 2 * j * \#\text{ of nodes of height } j$$

$$\leq 2 \sum_{j=0}^{h} j * 2^{h-j}$$

$$= 2^{h+1} \sum_{j=0}^{h} j * (1/2)^j$$

$$\leq 2^{h+1} \sum_{j=0}^{\infty} j * (1/2)^j = 2^{h+1} \frac{1/2}{(1-1/2)^2}$$

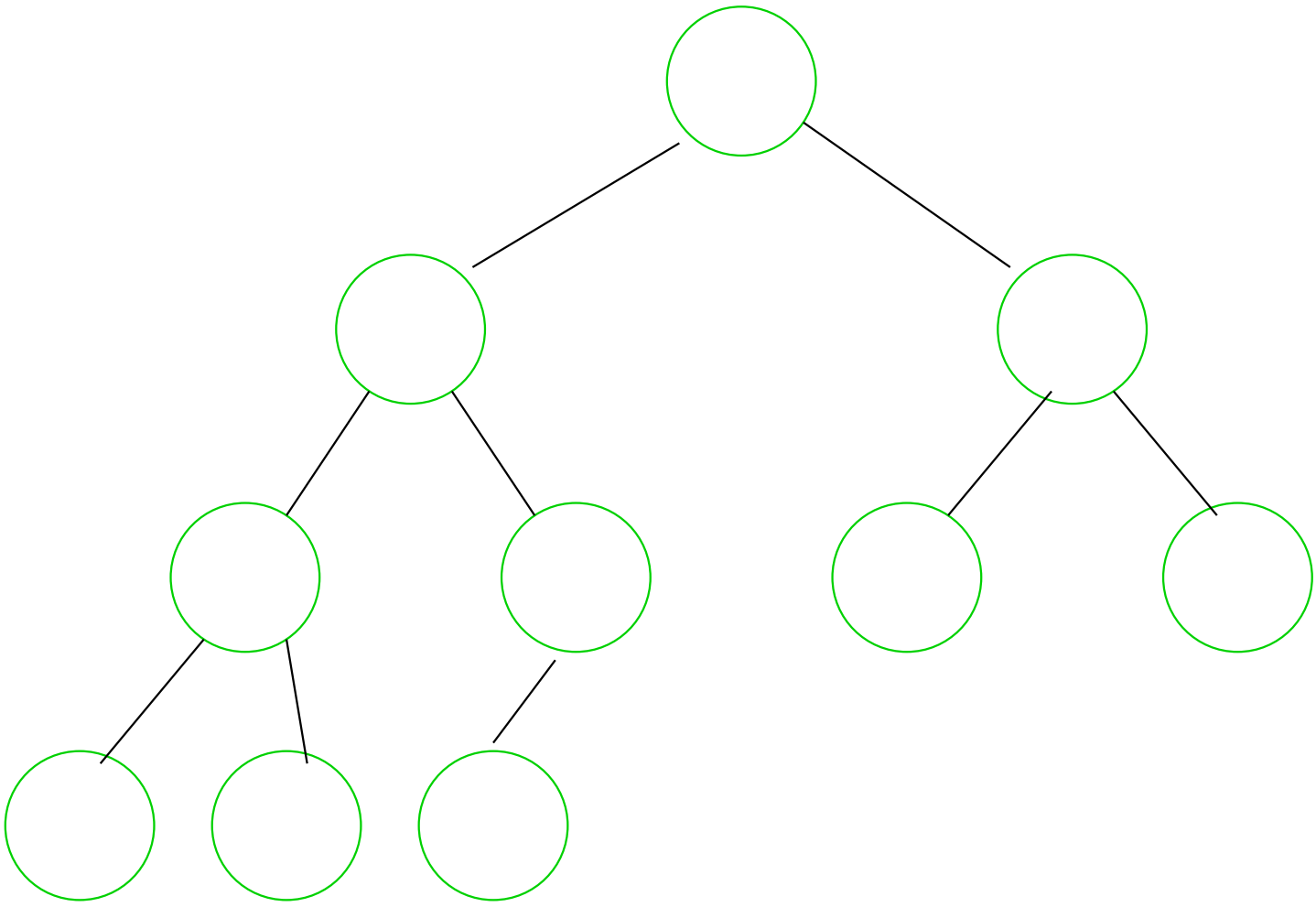
$$= 2^{h+2} = 4 * 2^h \leq 4n$$

linear!

Heapsort

1. Builheap
2. **while** heap not empty **do**
   - Remove maximum element (root)
   - Move last element at bottom level to root
   - Heapify

(# of compares $\leq 4n + 2n \lg n$)



Priority Queue

Abstract data structure

Objects: Set S of records, each with a key from a totally ordered set.

Operations:

- **Maximum**(S)**:** returns element of S with maximum key
- **Extract-Max**(S)**:** returns and deletes element of S with maximum key
- **Insert**(S, x)**:** adds element x to S
- **Increase-Key**(S, x, k)**:** increases the key value of x in S to k; assumes $k \geq$ current key of x and that a pointer is given to x.

Implement with **HEAP**

- how to implement each operation?
- w.c. time for each operation?