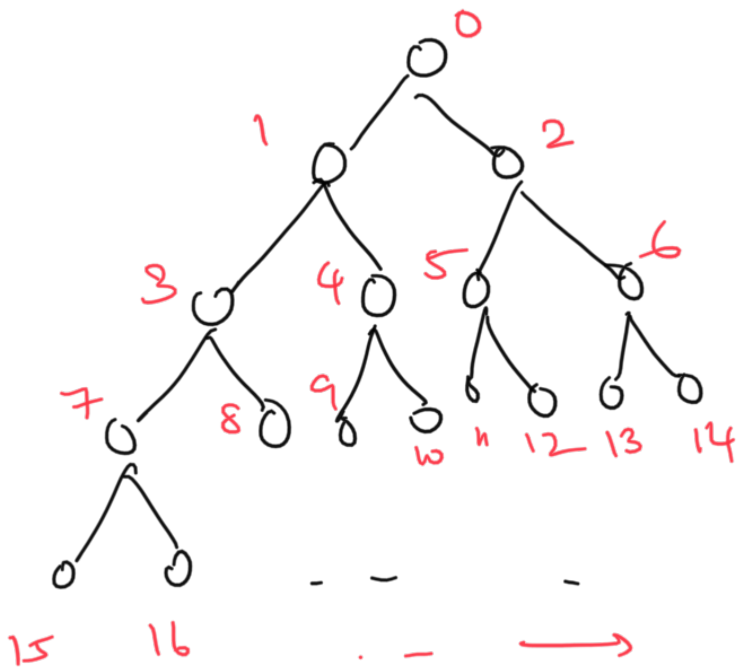
[Ref: https://nptel.ac.in/courses/106106131]

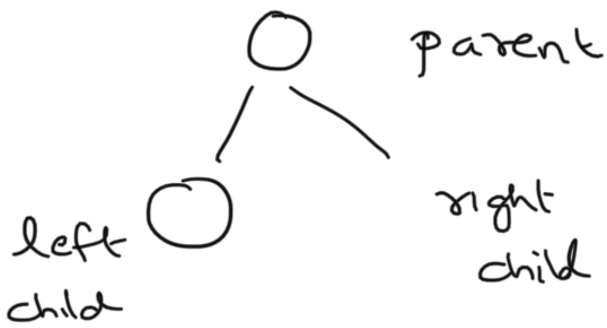
# Heaps

- Balanced binary tree
- shape determined once # of nodes given.




nodes must be inserted top to bottom left to right

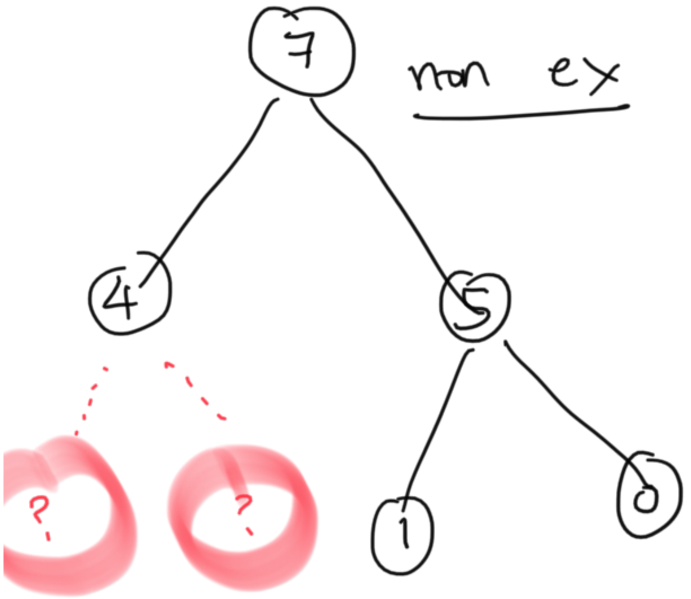
- local property of values




should be satisfied at every node { want val (parent) $\geq$ value (left child), value (right child)

we say a node violates local value property if

node.value $<$ value of atleast one of its children

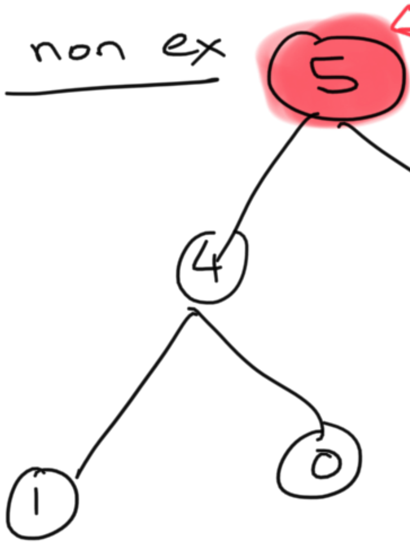


non ex

not a heap

wrong shape

non ex



violates local value property

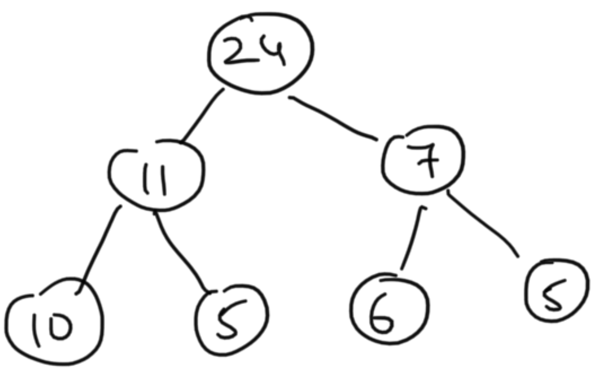
not a heap.

local property

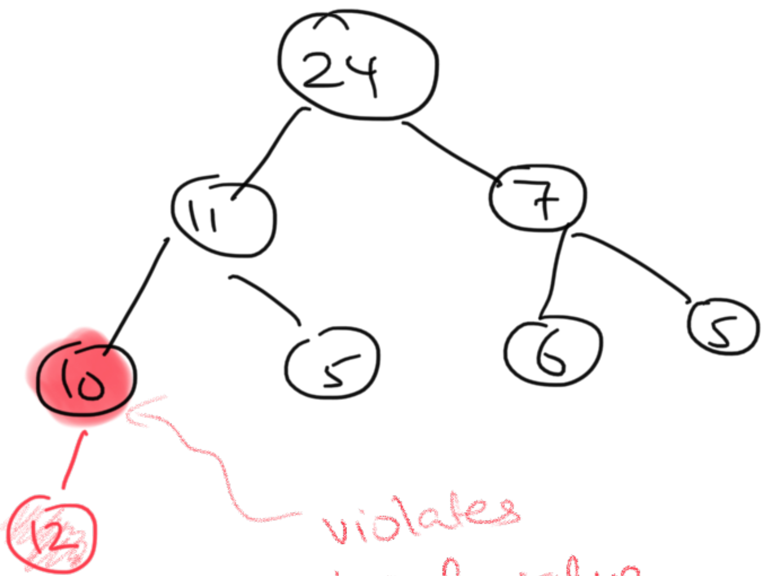
---

Insert operation

existing heap . (Insert 12)



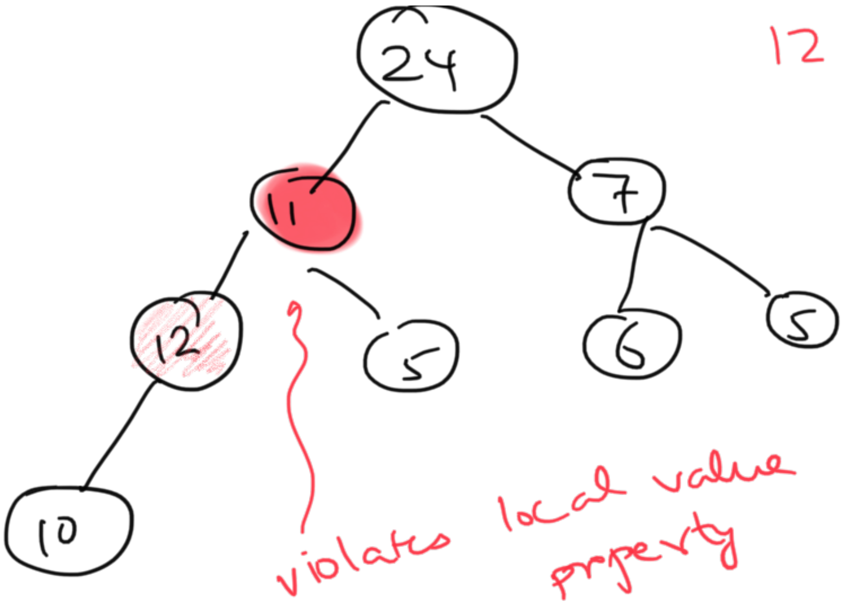
→ shape determined:



insert 12 in new node

violates local value property

→ switch 10, 12 . .



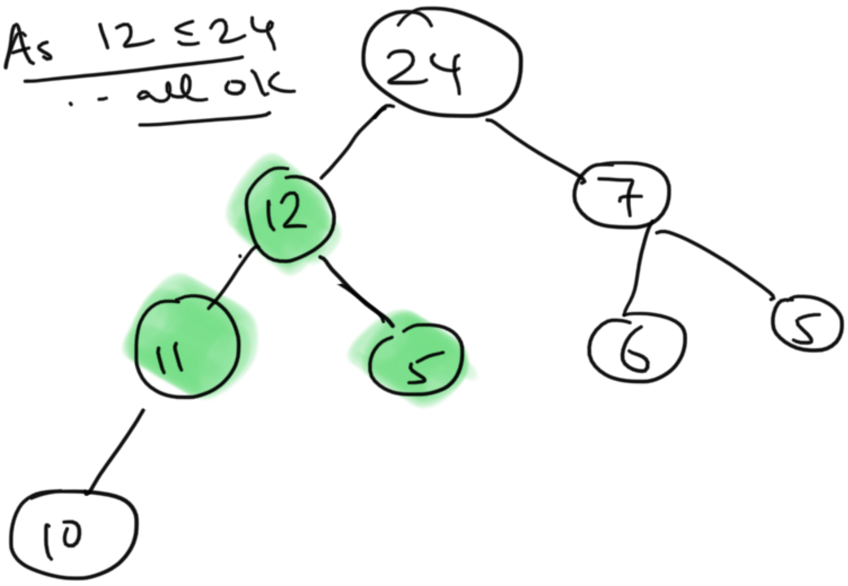

$12 \geq 11 \geq 5$

$11 \geq 10$

violates local value property

→ switch 12, 11

As $12 \leq 24$ ∴ all ok



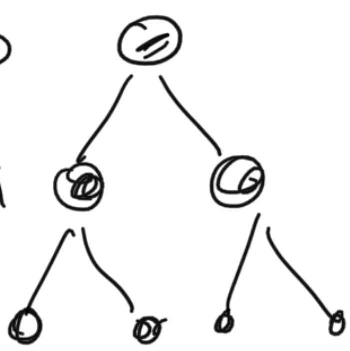

$12 \geq 11 \geq 5$ ✓

new heap

so $O(\log n)$

- swap operations as you go up the tree ...

## Balanced binary tree



level 0 — 1 node $= 2^0$ nodes

level 1 — 2 nodes $= 2^1$ nodes

level 2 — 4 nodes $= 2^2$ nodes

⋮

level $k-1$ — $2^{k-1}$ nodes

# of levels = k = ht

$\Rightarrow$ # of nodes = $2^0 + 2^1 + \dots + 2^{k-1}$

$$= \frac{2^k - 1}{2 - 1} = 2^k - 1 = n$$

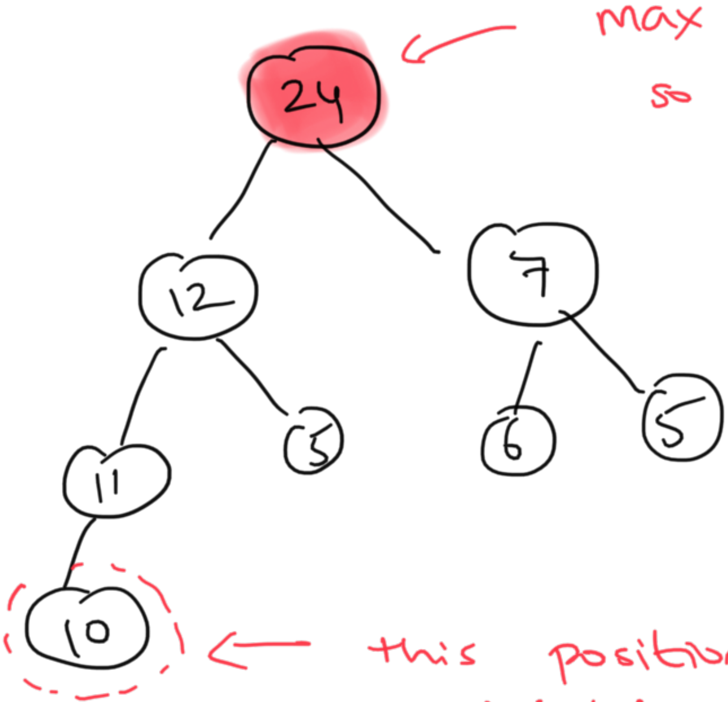
$\therefore$ ht = $\log_2 (n+1)$

$\uparrow$

$O(\log_2 n)$

---

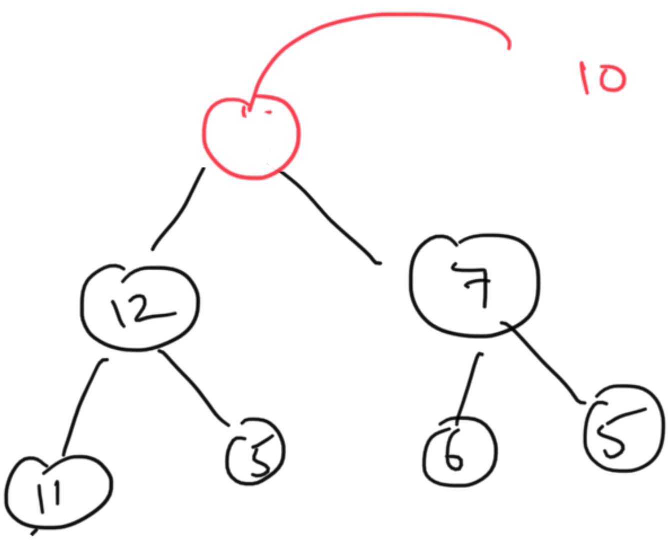
## delete_max operation



max value at top.
so need to delete 24
so top node is valueless

this position has to be deleted.

so 10 ← "homeless"

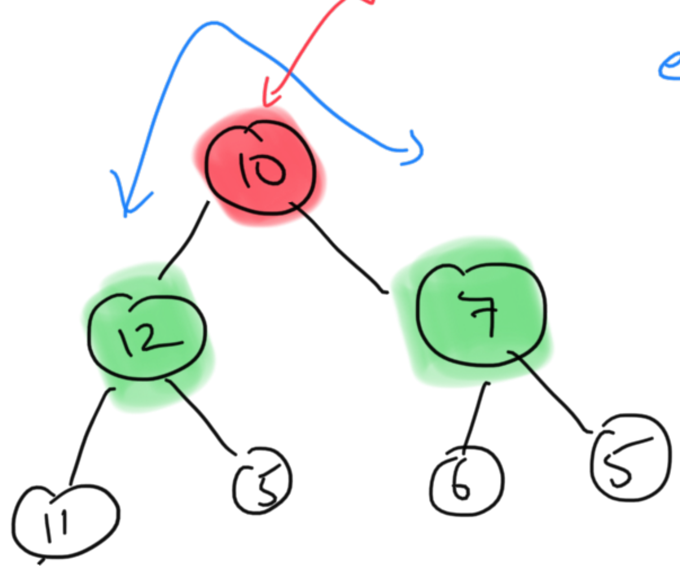


10

Put homeless value into empty node

violates local value property

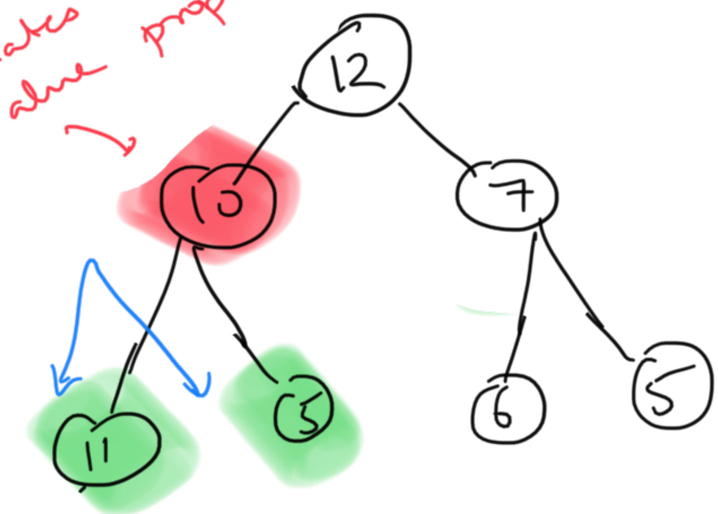
↪




exchange 10 with its biggest child ... (so 12)

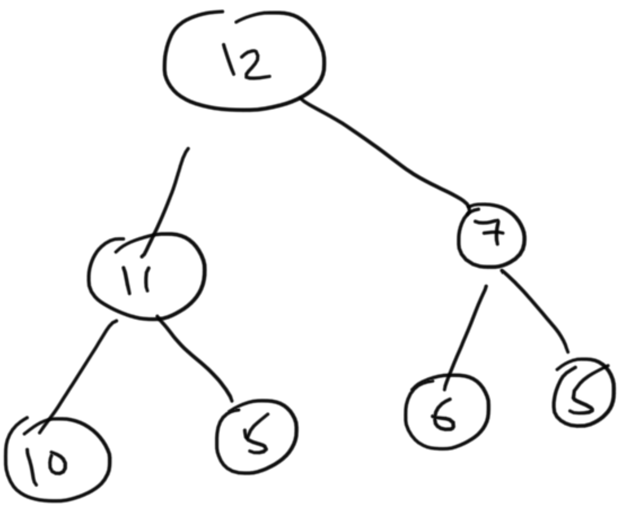
violates local value property

↪




swap 10 with its biggest child (so 11)

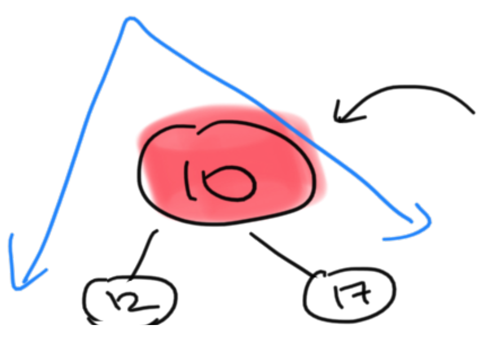
↪



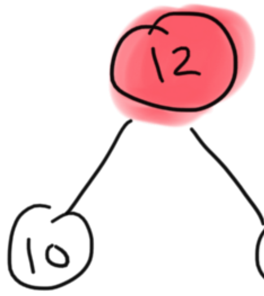

OK ✓

why swap with biggest child?

(ex) if



if you swap 10, 12

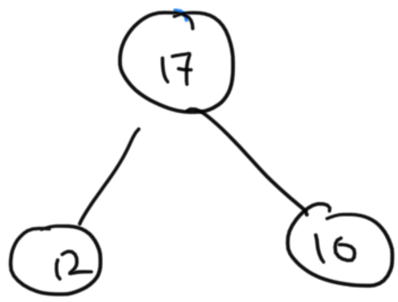



so swap 10, 17.

...

17 $\geq$ 12, 17 $\geq$ 10

biggest child




so walk down tree

$O(\log n)$

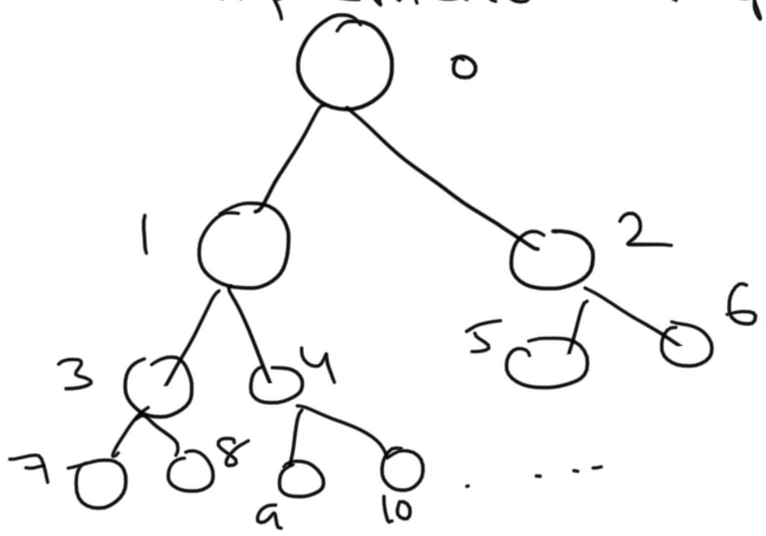
---

So processing n jobs $\rightsquigarrow$ $O(n \log n)$

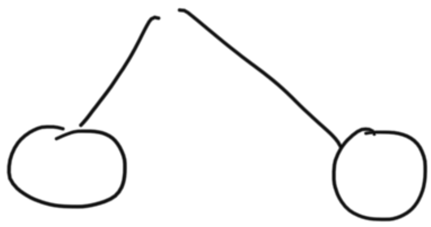
- n inserts
- n deletes

---

How to implement heaps?



parent = i



right child

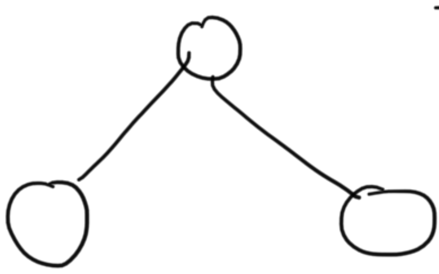
left child

2i+1

2i+2

j odd     j' even

parent = $\frac{j-1}{2}$ or $\frac{j'-2}{2} = \frac{j'}{2} - 2$



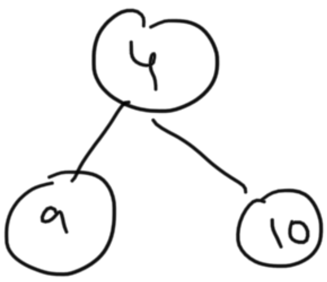
left child

right child

$2i+1 = \underbrace{j}_{\text{odd}}$

$2i+2 = \underbrace{j'}_{\text{even}}$

(eg)




if child # is 9

⇝ parent is $\frac{9-1}{2}$

$= 4$

If child # is 10

⇝ parent is $\frac{10-2}{2}$

$= 4$

uniform formula :

$$\text{parent}(j) = \left\lfloor \frac{j-1}{2} \right\rfloor$$

(floor)

irrespective of $j$ odd/even

-- $\frac{10-1}{2} = 4.5$

$\lfloor 4.5 \rfloor = 4$

**Pf**: $\hat{j}$ odd, already know it works

$\hat{j}$ even, let $\hat{j} = 2k + 2$

$$\frac{\hat{j} - 1}{2} = \frac{2k+1}{2}$$

$$= k + 0.5$$

$$\left\lfloor \frac{\hat{j} - 1}{2} \right\rfloor = k$$

---

So can use array!

$\text{Heap}[i]$ = value in node # $i$ . . .

To access $i$'s childrens' values

$\text{Heap}[2i+1]$, $\text{Heap}[2i+2]$

To access $i$'s parent's value

$\text{Heap}\left[\left\lfloor \frac{i-1}{2} \right\rfloor\right]$ . . . .

---

Build a heap with $n$ values

" heapify ([list of $n$ values])

↙

## Naive

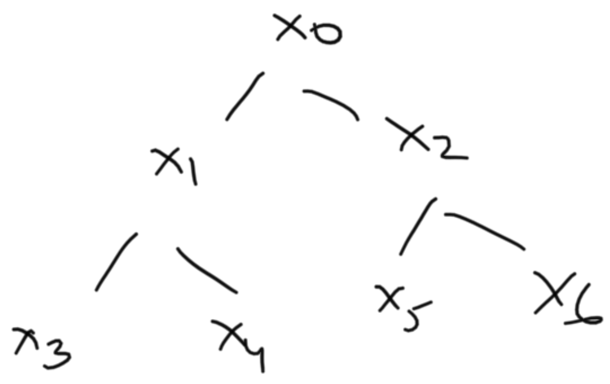
- empty heap
- insert one by one

⇝ $\log 1 + \log 2 + \dots + \log n$

$\leq n \log n$ time

## Better

List : $[x_0, x_1, \dots]$




⋮

"Damaged heap"

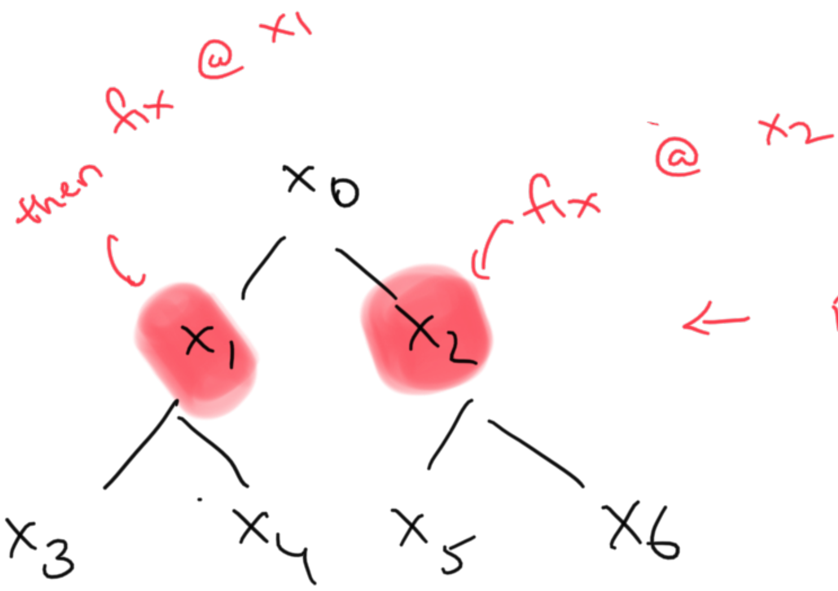
- shape OK
- but local value property ✗

Leaves : Last level : trivially local value property ✓

→ Repair previous level nodes . . .

⋮

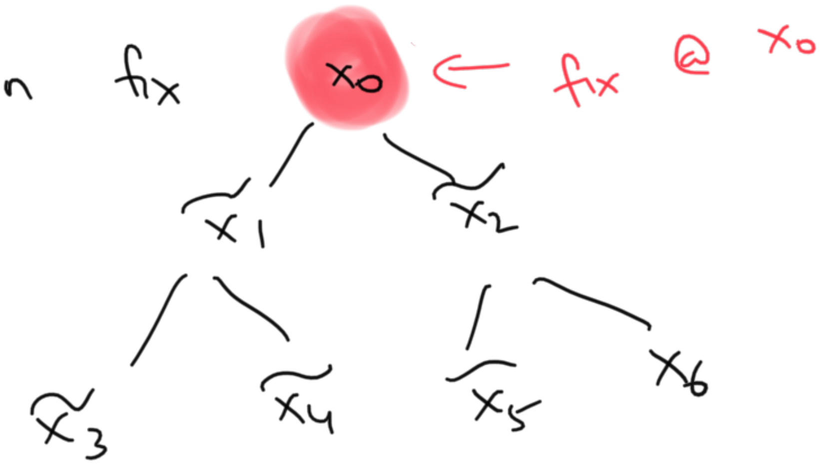
example




← Repair level 1

← level 2 ✓

↪ then fix




Fixing a node @ level $k-i$ :

- involves $i$ levels
- might have to walk down a path of length $i$

# of nodes @ level $k-i$ $= 2^{k-i}$

so over all

$$\sum_{i=1}^{k} 2^{k-i} (i)$$

← level $k-i$

$$= 2^{k-1} \cdot 1 + 2^{k-2} \cdot 2 + 2^{k-3} \cdot 3 + \dots + 2^{k-k} \cdot k$$

$$= 2^{k-1} \left( 1 \cdot 1 + \frac{1}{2} \cdot 2 + \frac{1}{2^2} \cdot 3 + \dots + \frac{1}{2^{k-1}} \cdot k \right)$$

$x = \frac{1}{2}$

$$= 2^{k-1}\left(1 + 2x + 3x^2 + \dots + kx^{k-1}\right)$$

$$= 2^{k-1}\left(\frac{d}{dx}\left(x + x^2 + \dots + x^k\right)\right)$$

$$= 2^{k-1}\left[\frac{d}{dx}\left(\frac{x^{k+1}-1}{x-1} - 1\right)\right]$$

$$= 2^{k-1}\left[\frac{d}{dx}\left(\frac{x^{k+1}-1}{x-1}\right)\right]$$

$$= 2^{k-1}\left[\frac{(k+1)x^k(x-1) - (x^{k+1}-1)\cdot 1}{(x-1)^2}\right]$$

$$= 2^{k-1}\left[\frac{(k \cdot x^k + x^k)(x-1) - x^{k+1} + 1}{(x-1)^2}\right]$$

$$= 2^{k-1}\frac{\left[kx^{k+1} + \cancel{x^{k+1}} - kx^k - x^k - \cancel{x^{k+1}} + 1\right]}{(x-1)^2}$$

$$= 2^{k-1}\left[\frac{x^k}{(x-1)^2}\right]\left[kx - k - 1 + \frac{1}{x^k}\right]$$

$$= \frac{2^{k-1}}{\left(\frac{1}{2}\right)^2} \left[ \frac{1}{2^k} \right] \left[ \frac{k}{2} - k - 1 + 2^k \right]$$

$$= 2 \left[ \frac{-k}{2} - 1 + 2^k \right]$$

$N$

$=$ no of nodes

$= 1 + 2^1 + \dots + 2^{k-1}$

$= \underline{2^k - 1}$

$$= O(N)$$