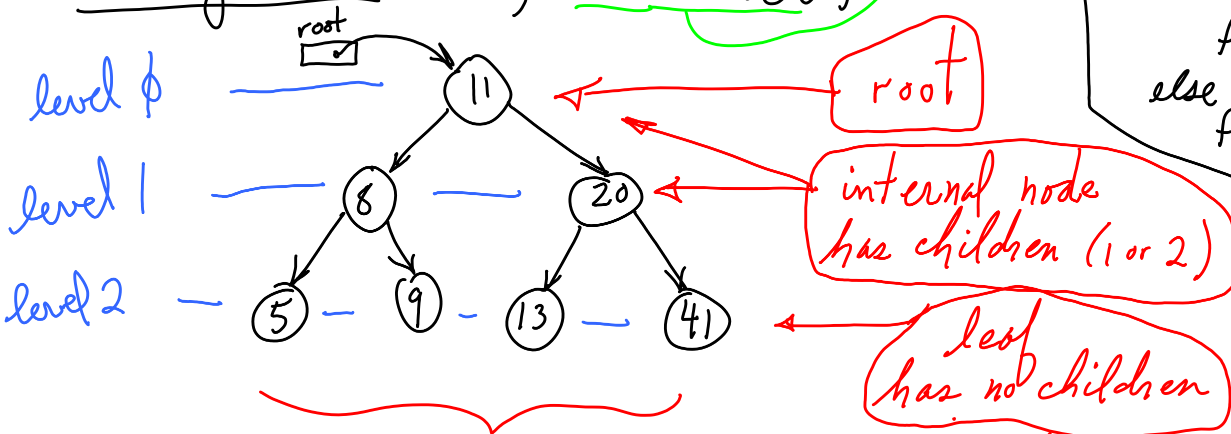
priority queue

- get item with least value of (?)
- insert item

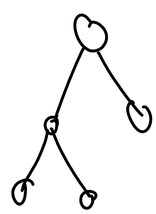
```
find (x, node)
   if (x == node->x)
        done
   else if (x < node->x)
        find(x, node->left)
   else
        find(x, node->right)
```

```
find (9, root)
```

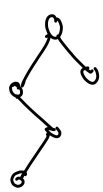
binary tree (sorted, search tree)



full: level i has 2^i nodes



complete, every node is either a leaf or has 2 children

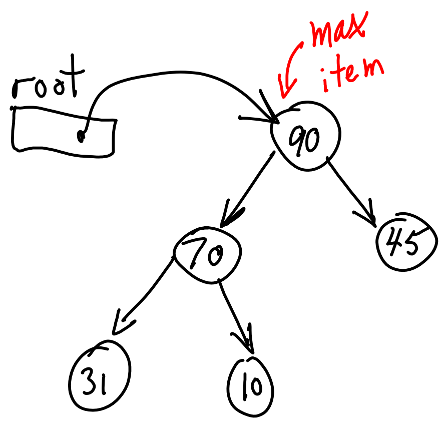


not complete + not full

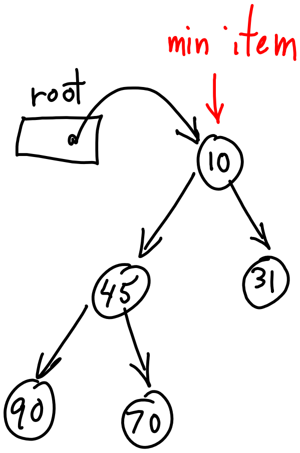
binary Heap

The (max) Heap Property
1. Tree is full, except lowest level
2. parent is GREATER than children
3. lowest level is complete to left

The (min) Heap Property
1. Tree is full, except lowest level
2. parent is LESS than children
3. lowest level is complete to left



max Heap



min Heap

Binary Search tree is completely ordered: more energy, less entropy

Q. output content in increasing order?

$\Longrightarrow$ suppose random placement of values in tree, what does it cost to order tree?

possible trees w/ n nodes $= n(n-1)\cdots(1) = n!$

Entropy $= \log(n!) \Rightarrow$ decreases to $\log(1) = 0$.

$H = -\sum^{N} p_i \log(1/p_i)$, suppose equally likely states, N

$\Rightarrow p_i = 1/N$

$$= -\sum^{N}\left(\tfrac{1}{N}\right)\log(1/N)$$

$$= -\left(\tfrac{1}{N}\right)\sum^{N}\log(1/N)$$

$$= -\left(\tfrac{1}{N}\right)N\log(1/N)$$

$$= -\log(1/N) = \log(N)$$

↑ number of states possible (lack of knowledge)

recall $\frac{1}{e^x} = e^{-x}$

$-\left(\ln\left(\frac{1}{e^x}\right) = \ln\left(e^{-x}\right) = -x\right)$

$= x$

$= \ln(e^x = N)$

Recall Stirling Approx.

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n = \Theta(n^n)$$
$$= \Omega(n^n)$$

$$\Longrightarrow \log(N) = \log(n!) \sim \log(n^n)$$

Let $n = 2^k$

$$= \log(2^{kn})$$

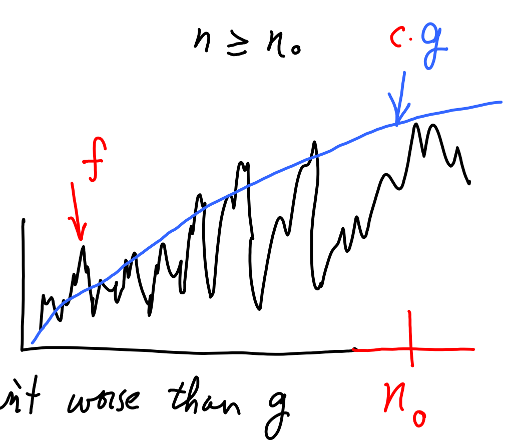
$$= kn$$

$$= (\log n)\cdot n \text{ bits}$$

$f(n) = \Theta(g(n))$

$\exists\, c, n_0 \quad f(n) \le c\, g(n)$

$n \ge n_0$



f isn't worse than g

Bits become known = energy

$\Rightarrow$ decrease Entropy $\Rightarrow$ $n \log n$

sort

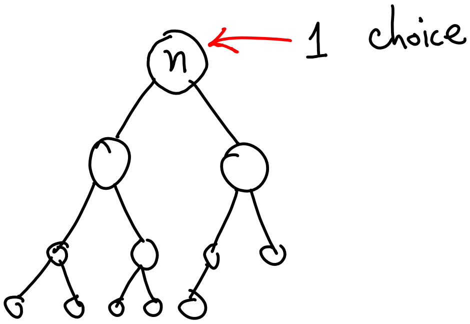
(min sorting Complexity)

$\Rightarrow \Omega(n \log n)$

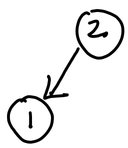
what about Heapify?

How many trees have heap property?

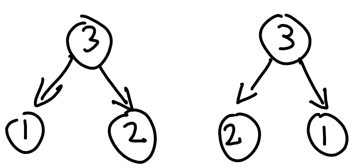
$\{1, 2, 3, \ldots, n\}$



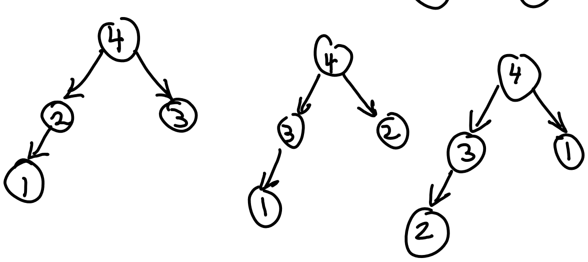
$\Rightarrow 1$



$\Rightarrow 1$



$\Rightarrow 2$



$\Rightarrow 3$

...

I don't know. $\Theta(\ )$? Anyway, it's much more than 1.

$\Delta H = H_{Tree}(n) - H_{Heap}(n) \ll H_{Tree}(n) - H_{Search\ Tree}(n)$

knowledge gained

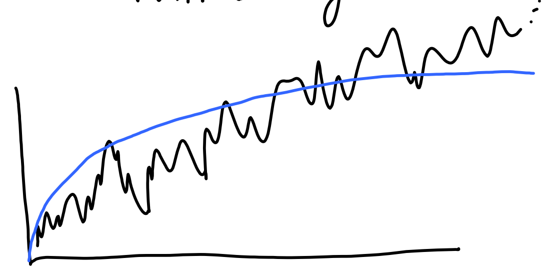
$= H_{Tree}(n) - 0$

$= n \log n$

Less knowledge $\Rightarrow$ less energy

$f(n) = \Omega(g(n))$

$f(n) \geq c\, g(n)$

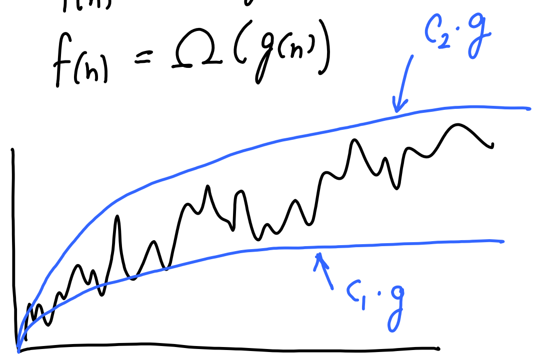


f is at least g

$f(n) = \Theta(g(n))$

$f(n) = O(g(n))$

$f(n) = \Omega(g(n))$

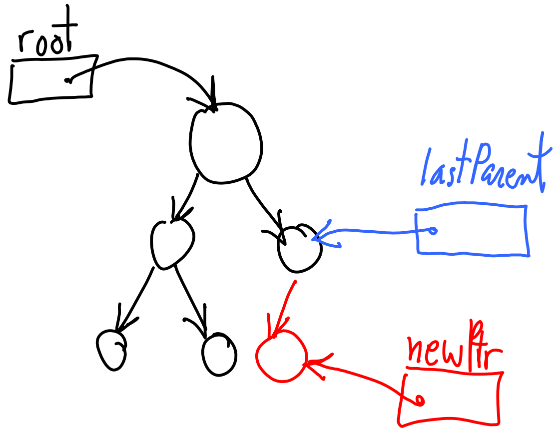


f is just like g

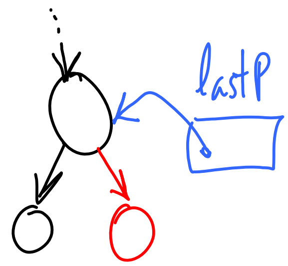
- log plot
- $n \to \infty$

Insert in Heap

1. **add new leaf**
   (left-to-right)

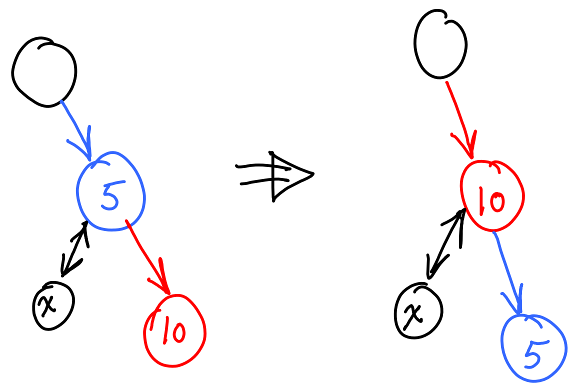



```
if (lastP->left == NULL)
    lastP->left = newPtr
    done
else
    lastP->right = newPtr
    curr = findLeftmostLeaf()
    lastP = curr->parent
    done
```

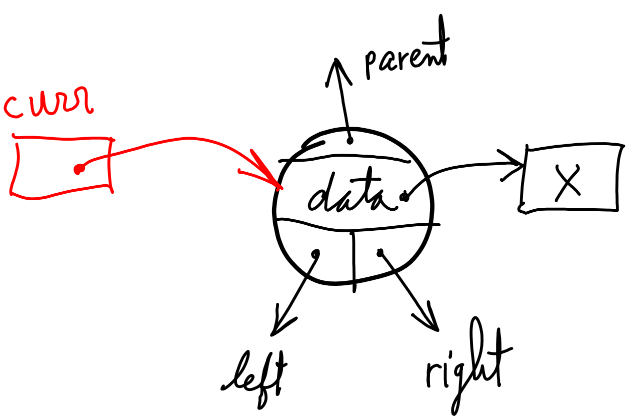


2. **Fix heap**

```
if (parent->x < newPtr->x)
    swap
```



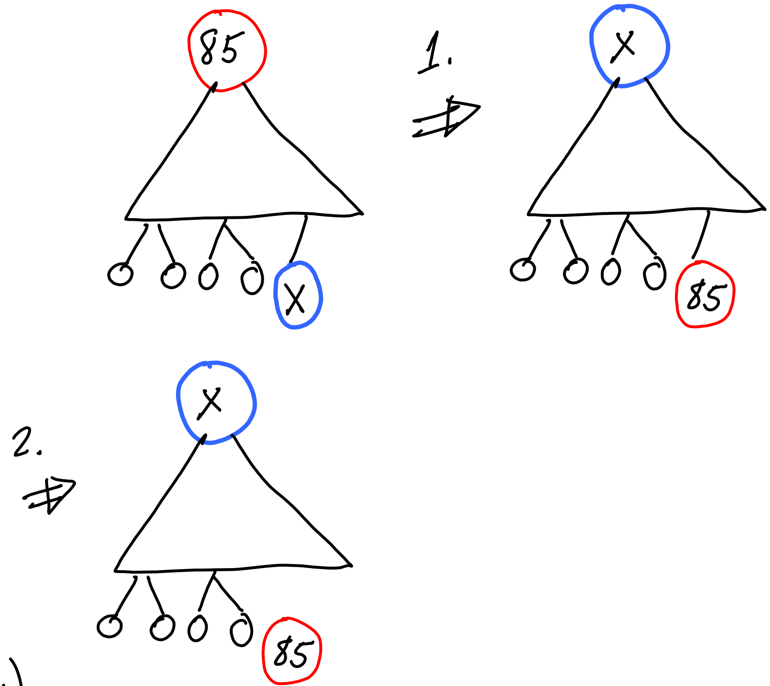
heap property?
Since $5 > x$, then $10 > x$
OK



loop until (curr->parent->data->x ≥ curr->data->x
OR
curr->parent == NULL) ← we are root

getRoot

1. swap(root, lastLeaf)
2. Delete last leaf
3. fixheapDown



if curr->data->x < curr->left->data->x)
    swap
else if curr->data->x < curr->right->x
    swap

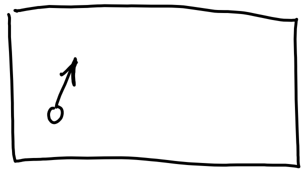
recurse until both tests fail
OR
at leaf

insert
$O(\log n)$

delete
$O(\log n)$

NOTES
1. Array implementations are not hard, take less space, and are a bit faster.
2. See projects2/CourseDocuments/Readings:
   Wayne-BinaryAndBinomialHeaps-2002.pdf
   Carrano-HeapImplementation-chp18-.pdf

ball in box system

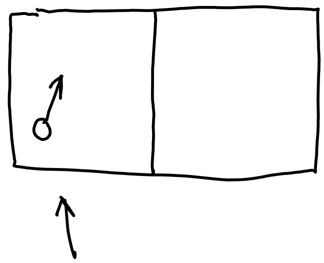


ball can be anywhere, N states*

entropy $= H = \log(N)$

* volume of phase space

H is unknown information



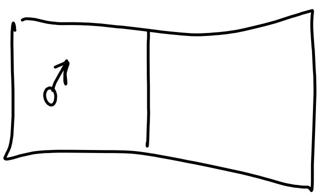
ball trapped in ½ space

$H' = \log\left(\frac{N}{2}\right)$ ($\frac{1}{2}$ as many states)

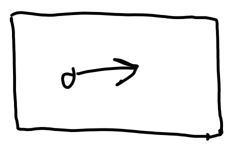
$= \log(N) + \log\left(\frac{1}{2}\right) = H - 1$

ball is in Left half $= 1$
" " " right half $= 0$
} 1 bit of information

Suppose ball is in left ½.

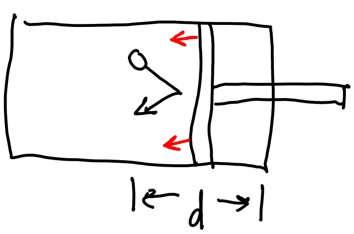


Remove barrier $\longrightarrow$



$\underbrace{H \to H+1}_{\text{lose 1 bit of info}}$

How to get more info? Get knowledge?



$\Leftarrow$ Work $=$ Pressure $\cdot\, d$

$= \frac{1}{2} k_B T$ (or 1 bit of energy)

$H \to H - 1$

Change in Entropy is info bits. Tree Problem, again.

$\{d_1, d_2, \ldots, d_n\}$ n values chosen from $\{1, 2, \ldots. W\}$

Number of states = (# ways to choose n values)$\cdot$(# ways to put n values in tree)

$N = W^n \cdot n!$

choose 1 tree, e.g., sorted $\Rightarrow$ $N' = W^n \cdot (1)$

Entropy difference is $\log(N) - \log(N')$

$$\sim \left(\log(W^n) + \log(n^n)\right) - \left(\log(W^n) + \log(1)\right)$$

$$\sim \log(n^n) - \log(1)$$

$$\sim n\log(n) - 0$$

$$= n\log(n)$$

= Change (loss) of Entropy

= bits of info gained by sorting

= energy lost in doing sorting