$T(n) = 2T(\frac{n}{2}) + \Theta(n)$ → kn for some $k > 0$.

kn

we don't pick the k.

Claim $T(n) = O(n \lg n)$

$T(n) \le cn \lg n$ for some c (which I pick)

Induction: $T(n) = 2T(n/2) + kn$

$\le 2\left(c \frac{n}{2} \lg\left(\frac{n}{2}\right)\right) + kn$

$= cn(\lg n - 1) + kn$

$= cn \lg n + \underbrace{kn - cn}_{\le 0}$

want $\le cn \lg n$

$kn - cn \le 0$

$\iff (k - c)n \le 0$

$\iff k - c \le 0$

$k \le c$

Choose $c = k$

$= cn \lg n$ ⊗

Fantasy Land

Suppose I wanted $k + c \leq 0$

$c \leq -k$

choose $c \leq -k$

choose $c \leq 0$

Proof doesn't work.

Sep 17-4:21 PM

$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

Claim $T(n) \le cn^2$ for some c

Pf $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$\le 4\left(c\left(\frac{n}{2}\right)^2\right) + n$$

$$= cn^2 + n$$

want $n \le 0$

Proof by induction is a "program" that for any n generates a proof for that n

$S(n)$

$$\left(S(1) \wedge S(2) \wedge \dots \wedge S(n-1) \right) \Rightarrow S(n)$$

Sep 17-4:26 PM

$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

Claim $T(n) \le cn^2 - dn$ for some +ve constants c, d

Pf

$$\begin{aligned} T(n) &= 4T\left(\frac{n}{2}\right) + n \\ &= 4\left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n \\ &= cn^2 - 2dn + n \\ &= cn^2 - dn + n - dn \\ &= cn^2 - dn + (1-d)n \end{aligned}$$

want $(1-d)n \le 0$

$\Leftrightarrow (1-d) \le 0$

$\Leftrightarrow d \ge 1$

choose $c = 1$, $d = 1$

$T(n) \le cn^2 - dn$ ⊠

$T(n) \le n^2 - n = O(n^2)$

Sep 17-4:30 PM

Priority Queue

Insert(S,x)

Max(S)

ExtractMax(S)

IncreaseKey(S,x,k)
increases x's value to k

Solved by balanced binary tree O(lg n)

D.S.
- Arrays
- Linked Lists
- Trees (balanced)
- Hash Tables

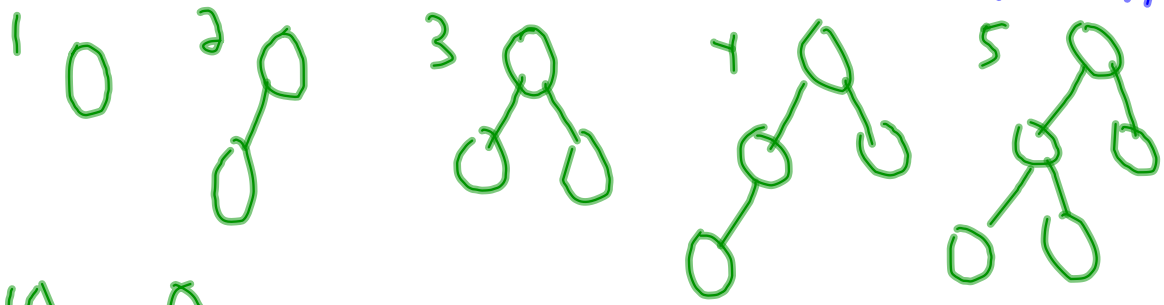
Dictionary
Insert Pred.
Delete Succ.
Member
Max
Min

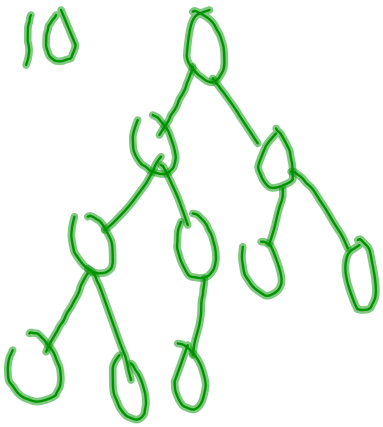
Sep 17-4:35 PM

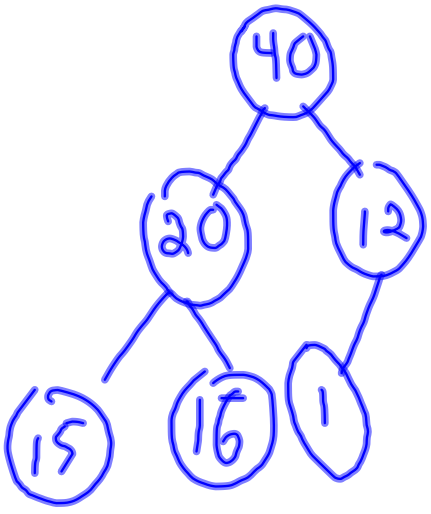
Heap

binary tree that is

1) filled in top down, left to right
2) value(parent) ≥ value(child)

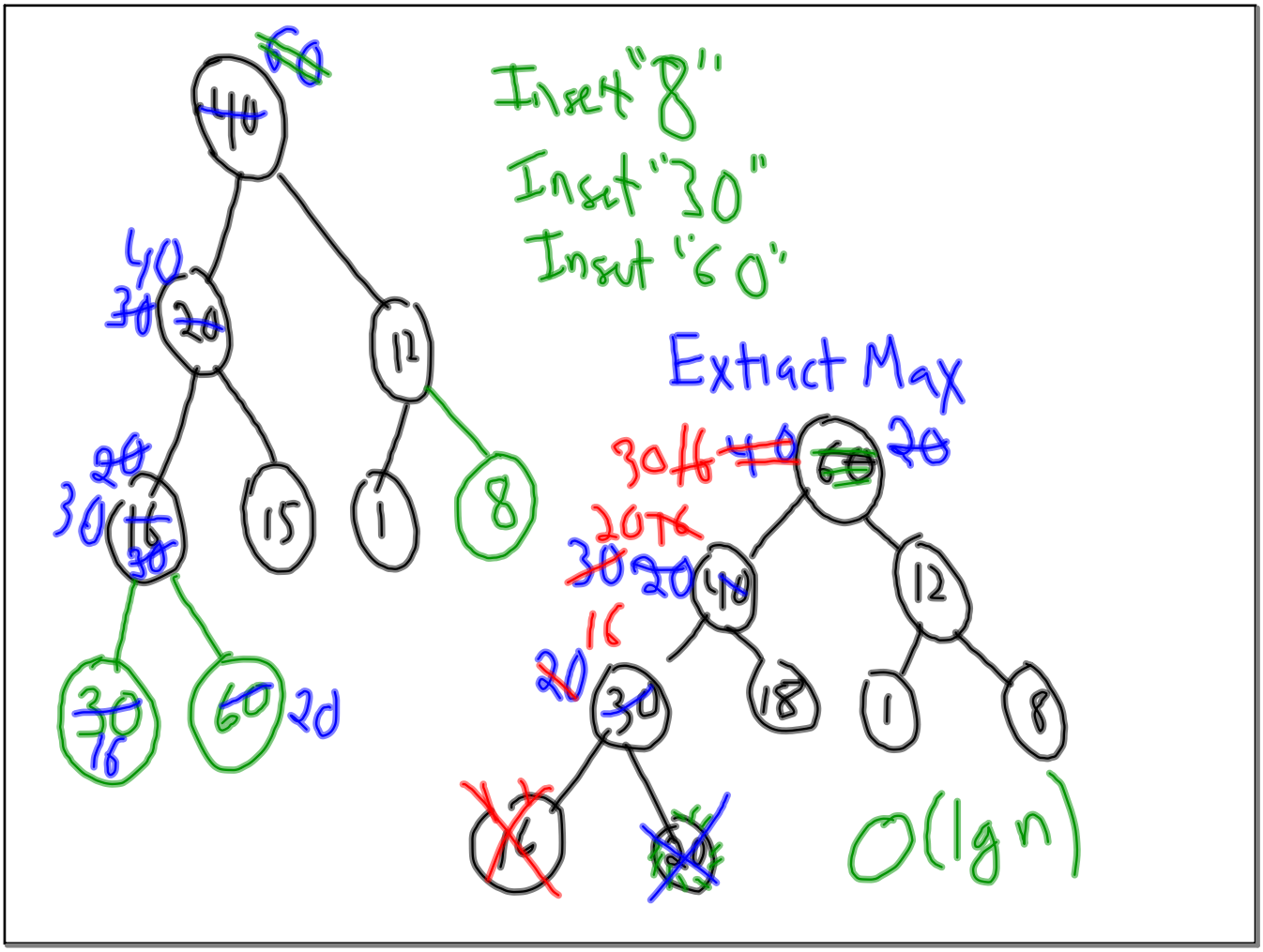


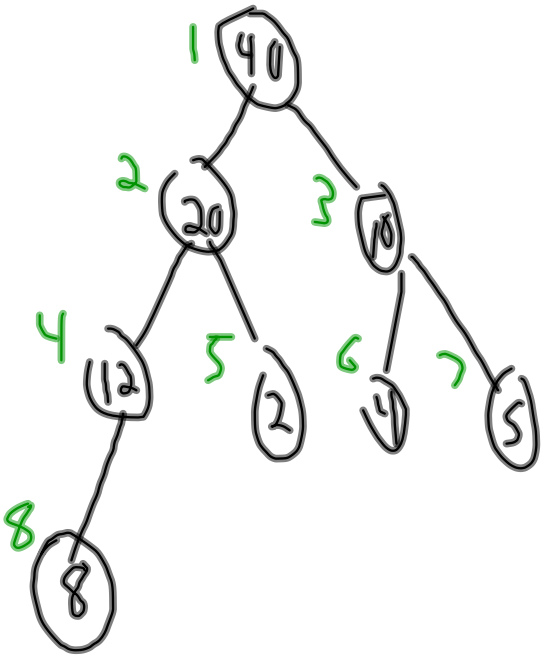




not good for number tests.

"Don't lose your key in a heap"


Insert "8"
Insert "30"
Insert "60"
Extract Max
O(lg n)



number nodes 1...

leftchild(i) = 2i

rightchild(i) = 2i+1

parent(i) = ⌊i/2⌋

1	2	3	4	5	6	7	8
40	20	10	12	2	4	5	8

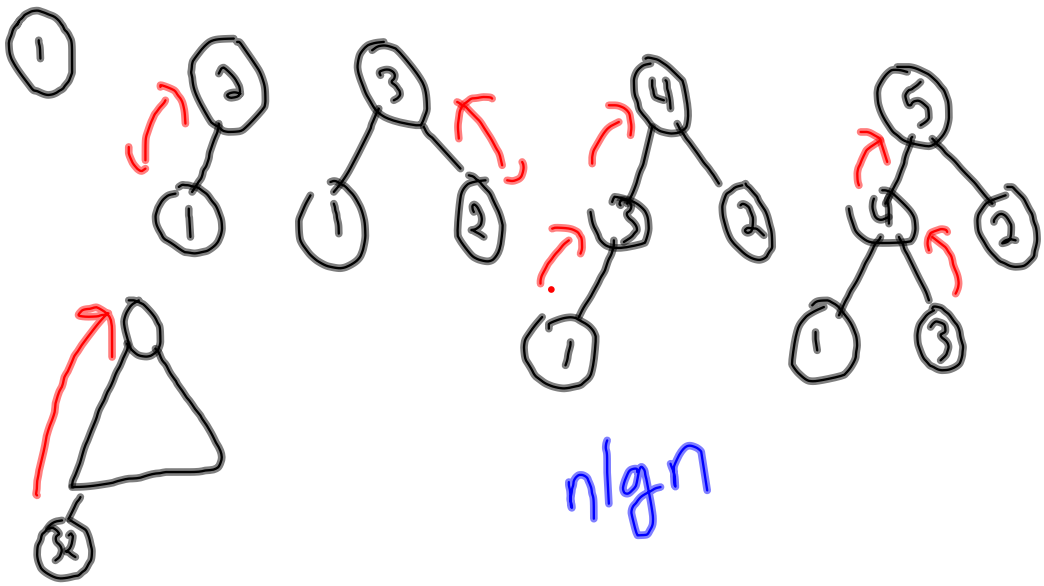
HeapSort

A[ ] data
H heap

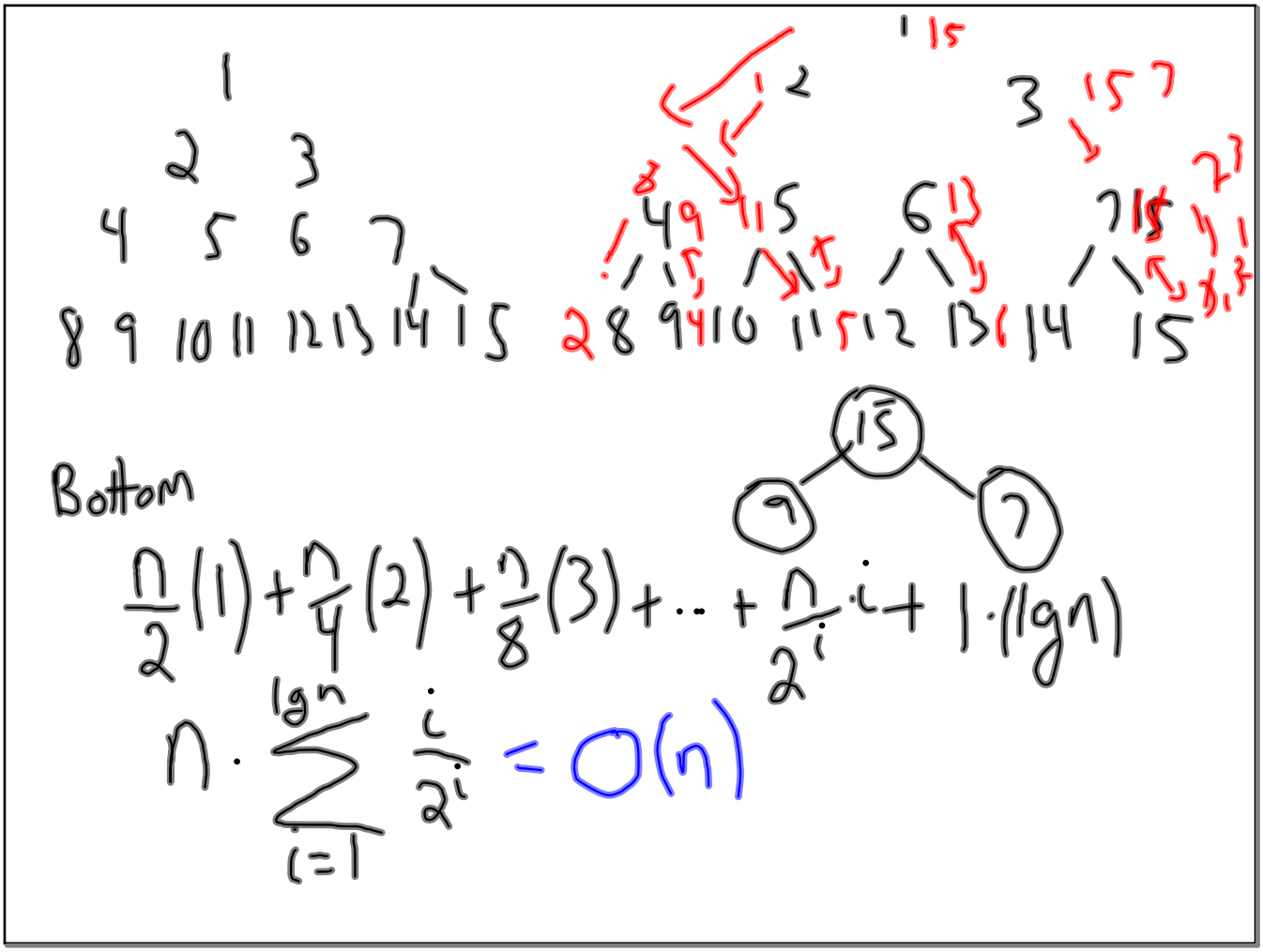
```
nlgn  [ for i = 1 to n
          H.Insert(A[i])

nlgn  [ for n down to 2
          A[n] = H.ExtractMax()
```

O(nlgn) time

n Inserts followed by n Extract Max



nlgn


Bottom
$\frac{n}{2}(1)+\frac{n}{4}(2)+\frac{n}{8}(3)+\ldots+\frac{n}{2^i}\cdot i+1\cdot(\lg n)$
$n\cdot\sum_{i=1}^{\lg n}\frac{i}{2^i}=O(n)$
15
9
7

Sep 17-5:20 PM