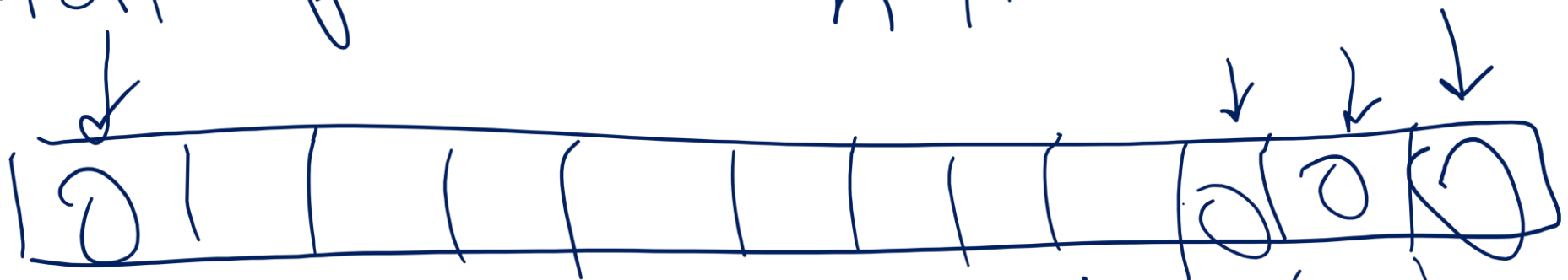
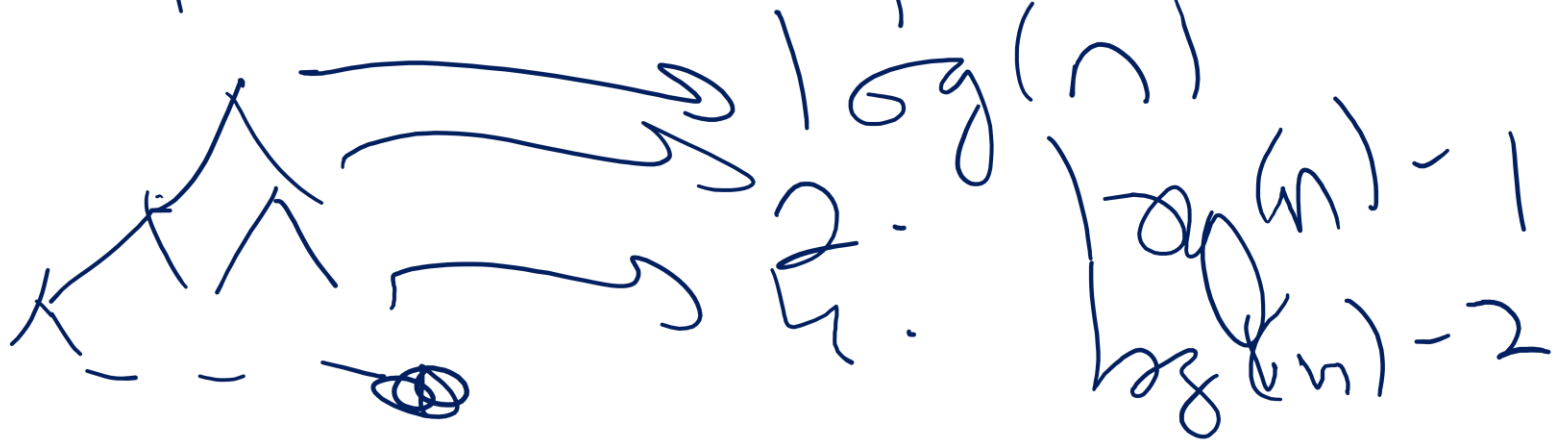
Floyd's BuildHeap

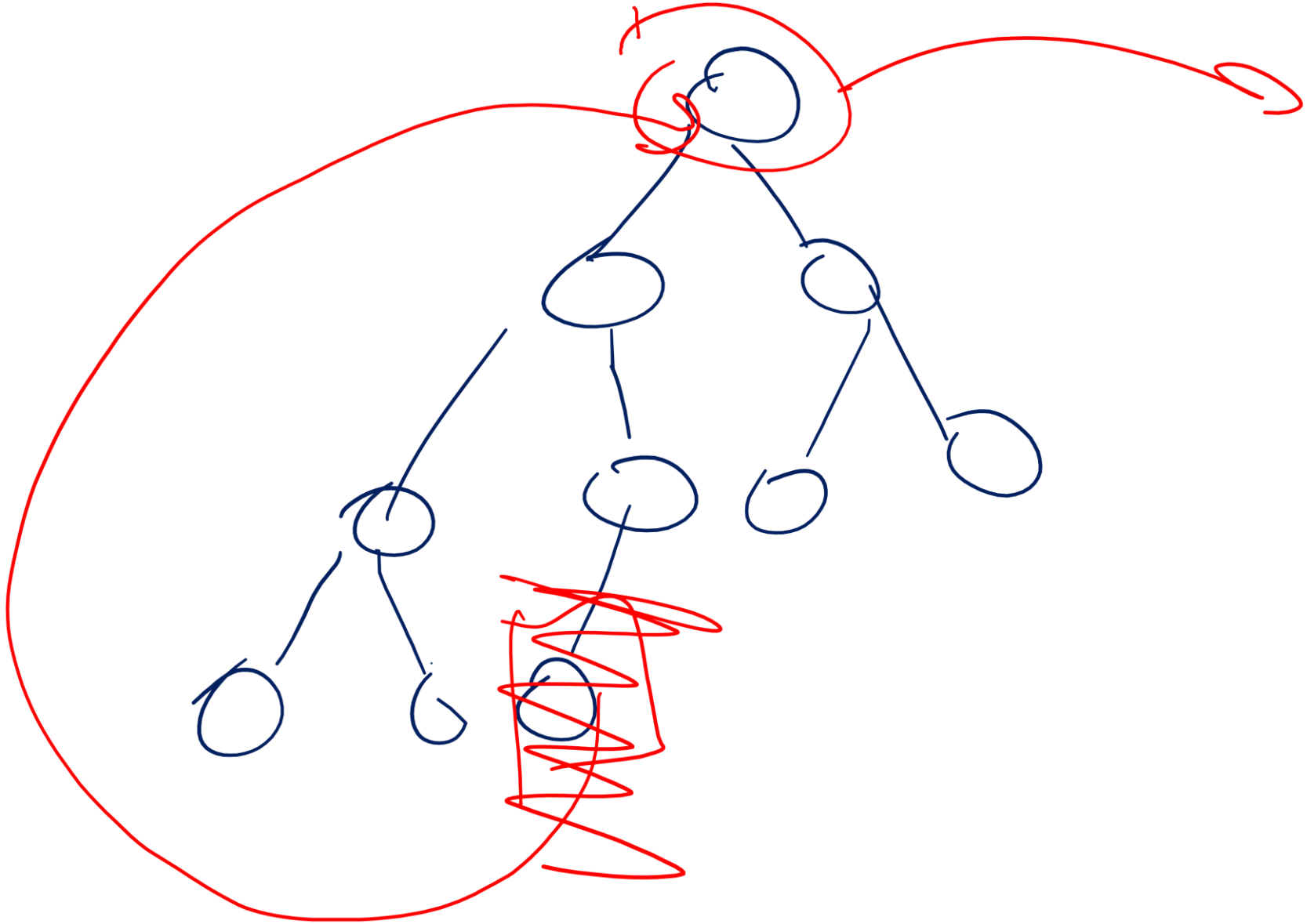
why is it O(n)

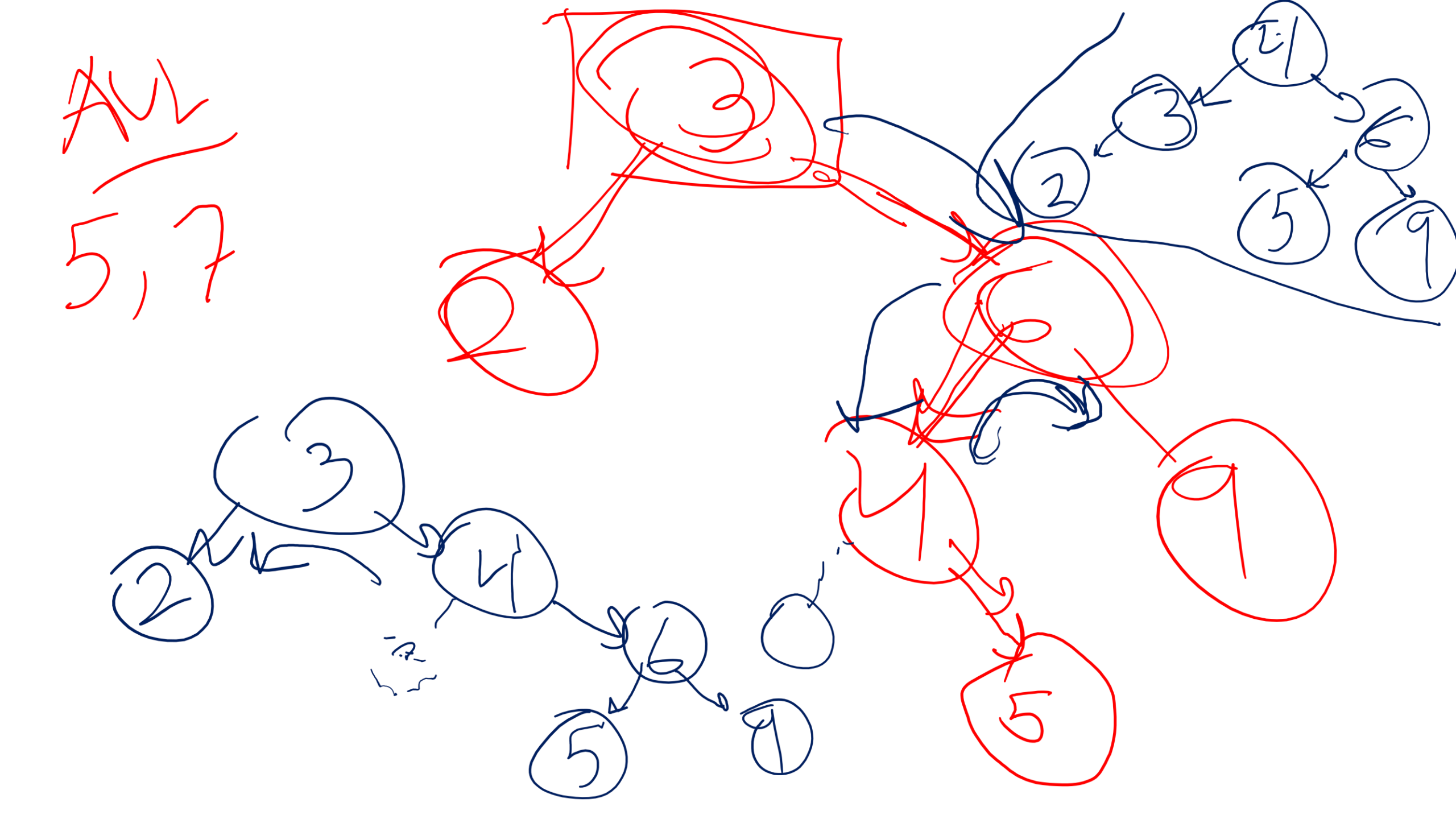
n items



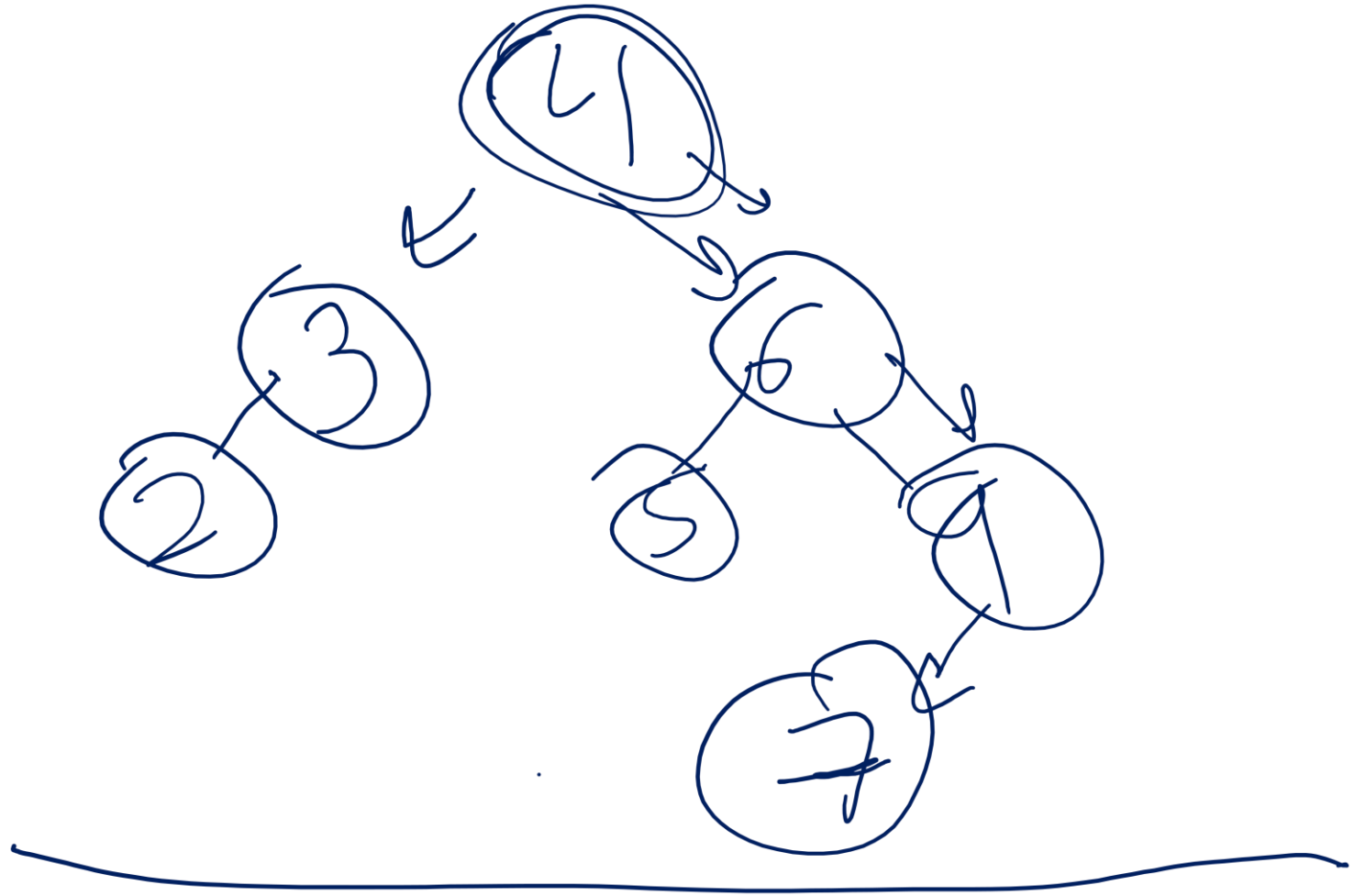


$\sim cn$




AVL
5,7

7



No handouts/slides.

Dictionaries ~> Insert, find, delete

AVL trees

Hash tables

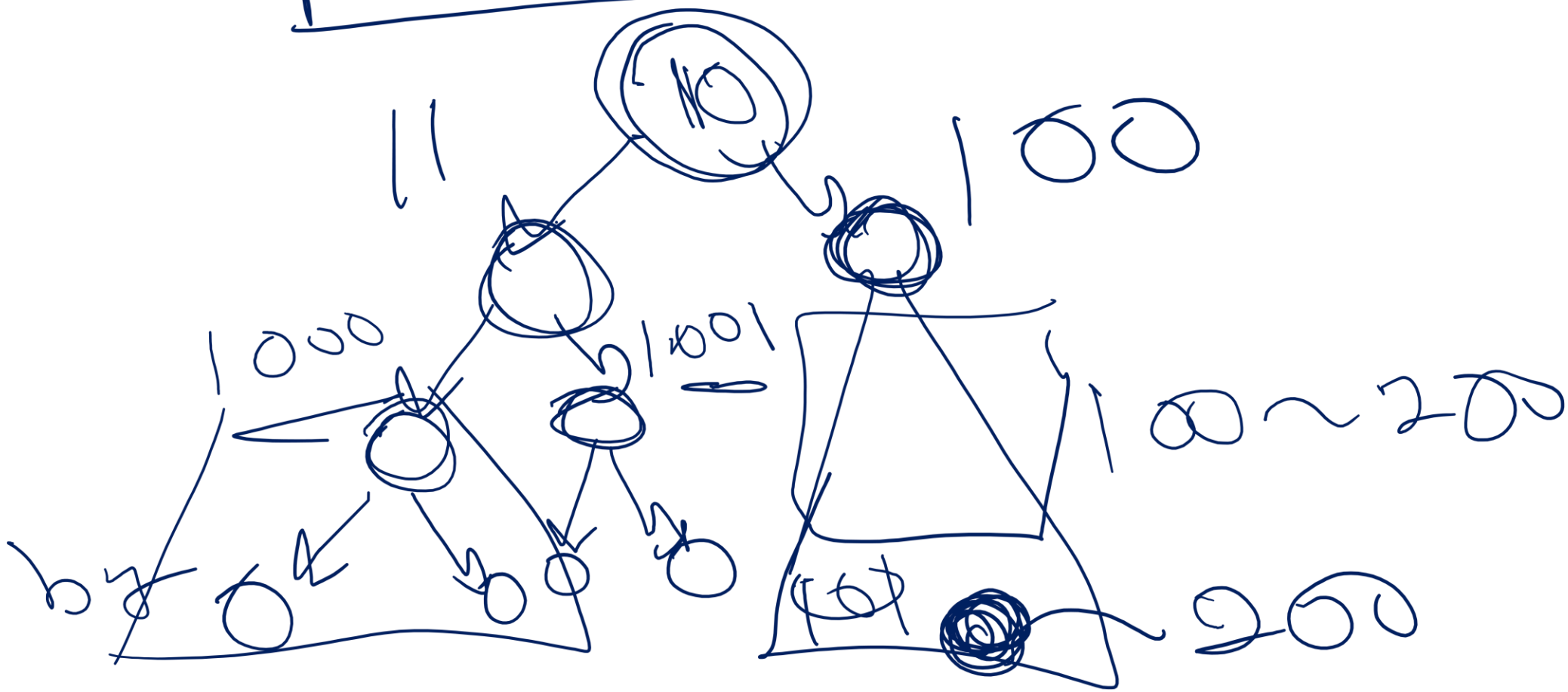
$2^{\log_2(n)+1} = 2 \cdot 2^{\log_2(n)}$

$= 2 \cdot n$

```
{for(i=0; i<n; i++
  for(j=0; j<i; j++
```

RemoveMin heap → best case?

→ repeated priorities



$$T(n) = \begin{cases} 2T\left(\frac{n}{2}\right) + 16 & o/w \\ 50 & \text{if } n < 10 \end{cases}$$

$16c \quad 8c \quad 4c \quad 2c \quad c$

$\vdots$

$2^{(\log_2(n))} \cdot c = n \cdot c$

```
for( i=0; i<n; i=(i*2)
    for(j=0; j<i; j++)
```

$1 + 2 + 4 + 8 + 16 + \cdots + n$

$$\sum_{i=0}^{\log_2(n)} 2^i = \frac{2^{\log_2(n)+1} - 1}{2 - 1} \quad (2n-1) \quad 2n-1$$

- BSTs

- Balance condition

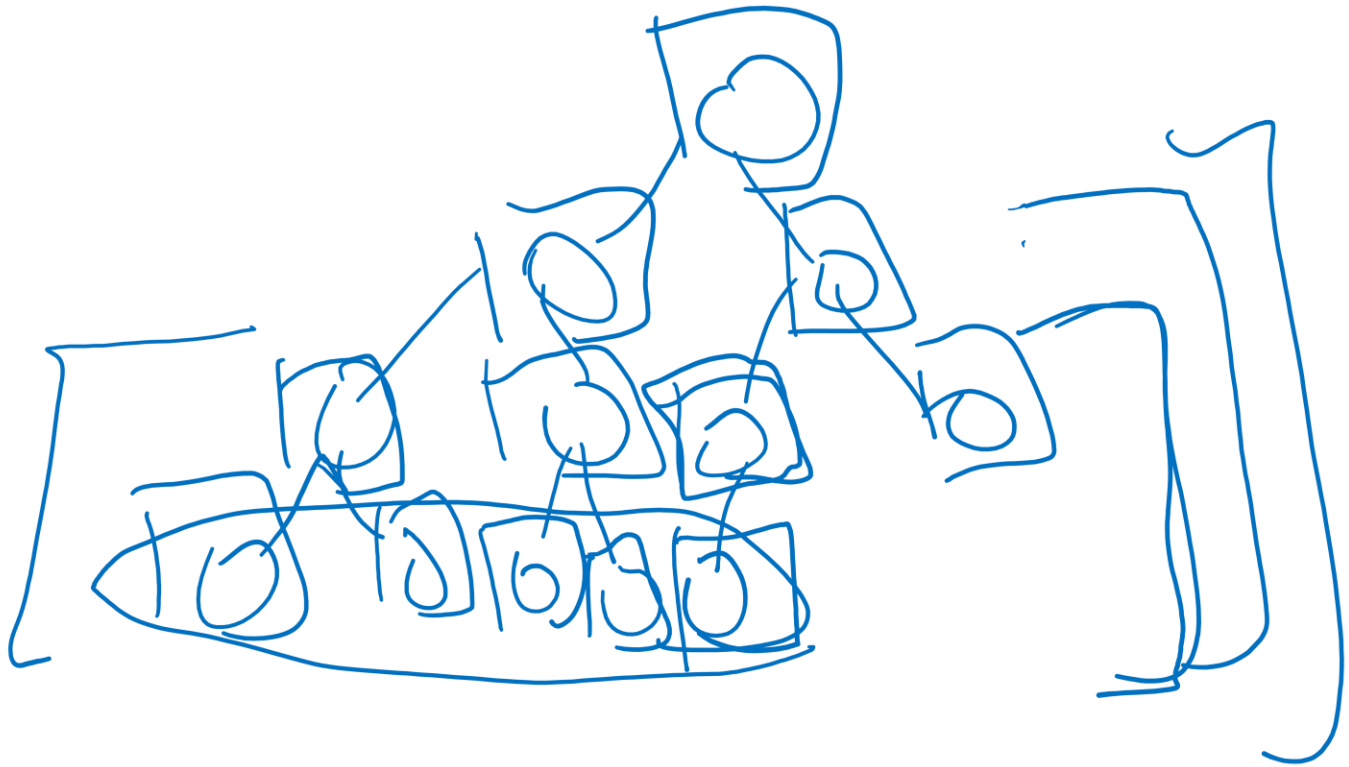
min $N(h) = \begin{cases} 1 & h = 0 \\ 2 & h = 1 \\ N(h-1) + N(h-2) + 1 & \text{otw} \end{cases}$

max $P(h) = 2^{h+1} - 1$

build heap

input array int

output heap w/ those same priorities



$\sim c \cdot n$