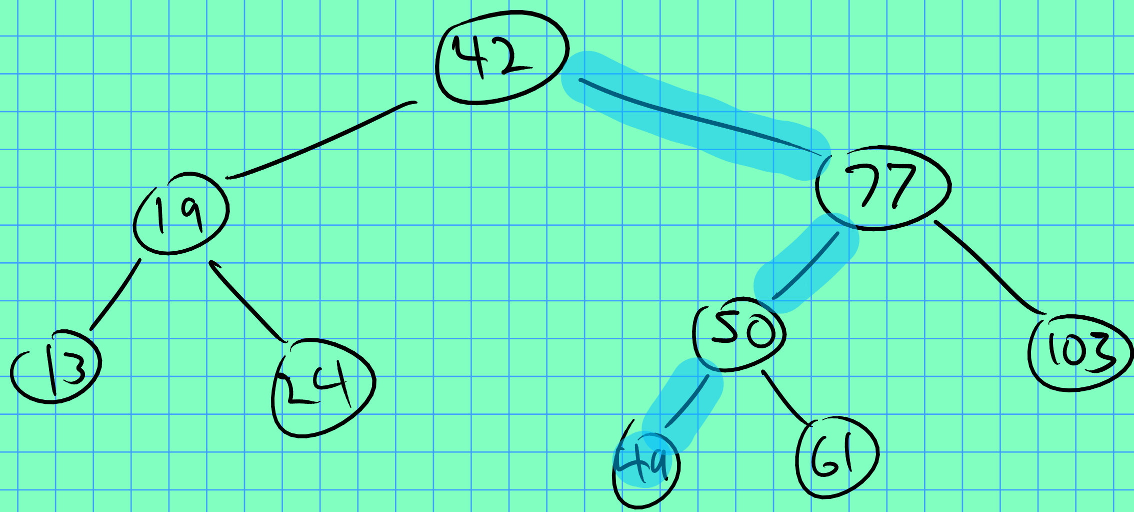
Sets & Maps

Behind the scenes, they are <u>binary search trees</u>.



Say total # of values is n.

For each comparison you perform how many locations are ruled out? $\approx$ half.

So, # of steps for search

$\approx$ # of times we can divide n by 2 before getting 1.

I.e., if k = # of steps,

$$\frac{n}{2^k} \approx 1 \Rightarrow n \approx 2^k$$

$$\Leftrightarrow k \approx \log_2 n.$$

Comparison with vectors

Pros:

Search is much faster
($\approx \log_2 n$ steps vs
$\approx n$ for a vector.)

Cons:

No "random access" (No $V[i]$ equivalent)

Shared features:

Easy to add elements. Container grows automatically.

$V.\text{push\_back}(x) \equiv S.\text{insert}(x)$

Other differences:

Sets represent mathematical sets, and hence they cannot store duplicates.

Aside: Sets as "characteristic functions".

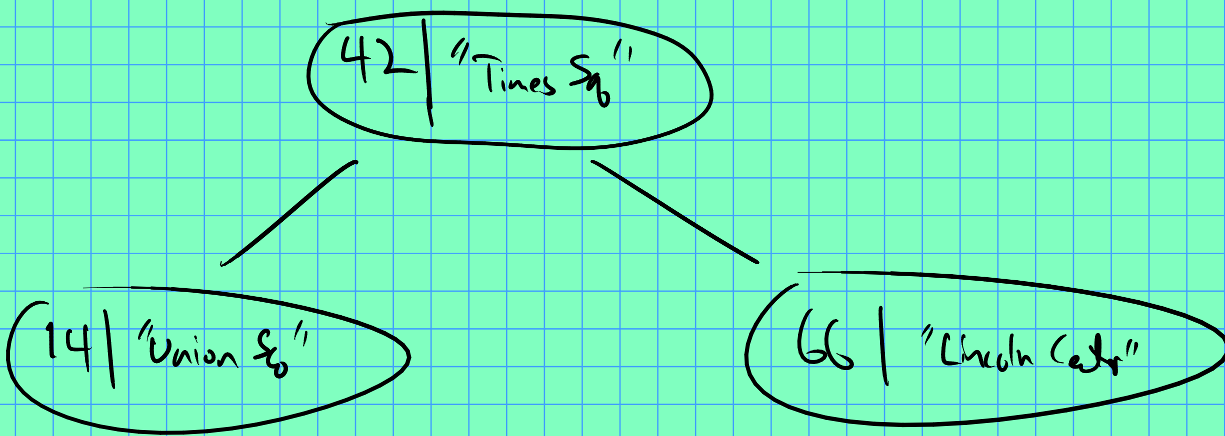
$$S \subseteq U \equiv f : U \longrightarrow \{0,1\}.$$

define $f_S : U \longrightarrow \{0,1\}$

by $f(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{else.} \end{cases}$

Then of course $S = f^{-1}(\{1\})$.

Maps: just like sets, but you can attach extra data:



$M[14]$ gives "union sq".