Problem (Dynamic Predecessor Search)

build data structure for n numbers \( S = \{a_1, \ldots, a_n\} \)
to support:

- pred search: given \( q \), find largest \( a_i \in S \) less than \( q \)

query

- (succ similar)

update

- insert an elem to \( S \)
- delete \( \cdots \) from \( S \)

Method 0

sorted array

query time \( \mathcal{O}(\log n) \) by binary search

Space \( \mathcal{O}(n) \)

preproc time \( \mathcal{O}(n \log n) \)

but insert/delete \( \mathcal{O}(n) \) \( \leftarrow \) bad

Method 1

binary search tree (BST)

query: can still use binary search

insert: straightforward

in order: replace with pred/succ
Inert: straightforward
delete: replace with pred/succ

\[ \text{time } O(\text{tree height}) \]
but height may be \( \Omega(n) \) in worst case!

\[ 1, 2, 3, 4, \ldots \Rightarrow \text{need balanced search trees!} \]

**Method 2: AVL Tree** (Adelson-Velsky, Landis '62)
(sketchn only)

**Invariant:**

\[ \text{at every node } v, \]
\[ |\text{height}(L_v) - \text{height}(R_v)| \leq 1. \]

\[ \Rightarrow \text{height } \leq \log_\phi n \]
\[ \phi = \frac{1 + \sqrt{5}}{2}. \]

\[ F_h = F_{h_2} \quad \uparrow h \]

basic primitive: rotation

\[ \text{right-rotate} \]

\[ x \quad \downarrow \]

\[ y \]
insert: 4 cases ... (messy)
delete: 4 cases ...

all ops $O(\log n)$ time

Other balanced BSTs:
- red-black tree (Guibas-Sedgewick '78)
- AA tree (Andersson '??)
- treap (shortest code)
  - simple but randomized

Method 3: 2-3 Trees or 2-3-4 Trees (Hopcroft '70)
(easiest to understand conceptually)
be more flexible with degree

Invariants:
1. At node $v$, $a \leq \deg(v) \leq b$
2. all leaves at same depth

for params $a, b$ with $a = \lceil b/2 \rceil$
e.g. $b=3$  
$a=2$

```
3  7  10  11  13
2  3  5  7  10  11  13
```

elems stored at leaves

$n \geq a^h$

$\Rightarrow$ height $\leq \log a n$

query: still $O(\log n)$

insert: whenever $\deg(v) = b+1$ for some $v$:

split $v$ into 2 nodes of $\deg \sim \frac{b+1}{2}$

$\Rightarrow O(\log n)$ splits (up a path)

$\Rightarrow O(\log n)$ time.

delete: whenever $\deg(v) = a-1$ for some $v$:

fuse $v$ with sibling & split if $\deg \geq b$

$\Rightarrow O(\log n)$ fuses

$\Rightarrow O(\log n)$ time