Heaps / Priority Queues:

Want Operations:
- Insert \( H, x \): insert elt with key \( x \) into \( H \)
- Extract-Min \( H \): Remove and return elt with smallest key
- Decrease-Key \( H, x, k \): Decrease key of \( x \) in \( H \) to \( k \)
- Meld \( H_1, H_2 \): Replace heaps \( H_1, H_2 \) with their union

Extra ops:
- Find-Min \( H \): Return elt with smallest key
- Delete \( H, x \): Delete elt \( x \) from \( H \)

<table>
<thead>
<tr>
<th></th>
<th>Insert</th>
<th>E-M</th>
<th>D-K</th>
<th>Meld</th>
</tr>
</thead>
<tbody>
<tr>
<td>LL</td>
<td>( O(1) )</td>
<td>( O(n) )</td>
<td>( O(1) )</td>
<td>( O(1) )</td>
</tr>
<tr>
<td>Sorted list/tom</td>
<td>( O(n) )</td>
<td>( O(1) )</td>
<td>( O(n) )</td>
<td></td>
</tr>
<tr>
<td>Balanced search tree</td>
<td>( O(\log n) )</td>
<td>( O(\log n) )</td>
<td>( O(\log n) )</td>
<td>( O(n) )</td>
</tr>
</tbody>
</table>
Amortized analysis of multiple operations

Types of ops $i$

$m_i$ ops of type $i$

"amortized cost $a_i$, for type $i$"

$\sum$ total time $\leq \sum a_i m_i$
Binary Heaps

Complete binary tree except maybe for bottom level (fill in L<R).

Heap order: every parent smaller than its children

Insert(C, t): Insert at next spot, "swim up": Time: $O(\log n)$
Extract-Min ($H$); swap root with last elt, return if "swim down" from root.

$\text{weed}(H_l, H_r) = O(n \log n)$

In reverse order ($R$-select): "swim down".

At height $h$ root node $t$ satisfies $t \leq n \leq 2^h$.

\[ t \leq \frac{n}{2^{h+1}} \leq \frac{1}{2} \begin{cases} \log_2 n & \text{if } h \geq 0 \\ \frac{n}{2^{h+1}} & \text{if } h < 0 \end{cases} \leq O(h) \]
Anonymized Extract MIN:
\[ w(x) = \text{depth} - x \]
\[ T(n) = \sum_{x} w(x) \]

Insert: \( O(\log n) + O(\log n) = O(\log n) \)

Extract-MIN: \( 1 + \log n + \Delta \log n \leq n \leq n \)
Lemma: $B_k$ has the following properties:

1) Height $k$
2) $2^k$ nodes
3) $\deg\ (\text{root}) = k$
4) delete root, get $B_{k-1}, B_{k-2}, \ldots, B_0$
Def: Binary Heap: collection of binary trees, s.t. either 0 or 1 child, and every tree is heap ordered.

Write n as binary b_k b_{k-1} \ldots b_0.

\[ \leq \text{less n trees in a heap} \]

\[ \geq \text{less trees in heap} \]
Find Min: $O(\log n) + O = O(\log n)$
\[ \text{Meld}(H_1, H_2) \]

**Special case:** \( H_1, H_2 \) both \( B_k \) (ex: \( B_3 \))

\[ H_1 \quad \quad \quad \quad H_2 \]

---

\[ \text{link of the } B_k \text{'s} \]
Time: worst-case $O(\log n)$
Analyzed $O(\log^2 n)$
Insert (H, ξ): Make B = F ξ. Meld

Time: it be K carries:

\[ k + 1 - (k - 1) = 2 \]

100

1

100