Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

**Example:** *Treap-insert*(8) with randomly chosen priority 2
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

**Example:**  $\text{Treap-insert}(8)$ with randomly chosen priority 2
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

**Example:** *Treap-insert*(8) with randomly chosen priority 2
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

Example: *Treap-insert*(8) with randomly chosen priority 2
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

**Example:** \textit{Treap-insert}(8) with randomly chosen priority 2
Treap Insertion Example

(The lower numbers indicate the “priority” of the node.)

Example: *Treap-insert*(8) with randomly chosen priority 2
Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
AVL Insertion: Second example

Example: *AVL-insert*(45)
AVL Insertion: Second example

Example: \textit{AVL-insert}(45)
**Example:** $AVL\text{-}insert(45)$
Example: AVL-insert(45)
AVL Insertion: Second example

Example: AVL-insert(45)
Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
Scapegoat insertion

- First, insert KVP into $T$ with the usual BST insertion.
- We assume that this returns the path $P$ to the new leaf.
- Put a “token” on every node on $P$.
  (These are only needed for analysis, not for implementation.)
- If $|P| > \log_{1/\alpha}(n)$
  - Find highest node $v$ on $P$ such that $\text{size}(v) > \alpha \cdot \text{size}(\text{parent}(v))$
  - $p \leftarrow \text{parent}(v)$ (completely rebuilt sub-tree at $p$)
  - Extract descendants $D$ of $p$ in in-order (sorted).
  - Remove all tokens from $D$.
  - Re-organize $D$ into a perfectly balanced tree:
    - $|\text{size}(z.\text{left}) - \text{size}(z.\text{right})| \leq 1$ for all nodes $z$.
  - This takes $O(|D|) = O(\text{size}(p))$ time.
  - Can argue: This releases $\geq (2\alpha + 1)\text{size}(p)$ tokens.
Scapegoat Tree Insertion Example

The lower numbers indicates the subtree-size.

Example:

```
Example:

20
4

10
1

70
2

30
1
```
Scapegoat Tree Insertion Example

The lower numbers indicates the subtree-size. The stars indicate tokens.

**Example:**  \( \text{Scapegoat-insert}(60), \ |P| = 3 < \log_{3/2}(5) \approx 3.96 \)
Scapegoat Tree Insertion Example

The lower numbers indicates the subtree-size. The stars indicate tokens.

**Example:** \( Scapegoat-insert(50), |P| = 4 < \log_{3/2}(6) \approx 4.41 \)
Scapegoat Tree Insertion Example

The lower numbers indicate the subtree-size. The stars indicate tokens.

**Example:**  \( \text{Scapegoat-insert}(40), \ |P| = 5 > \log_{3/2}(7) \approx 4.79 \)
Scapegoat Tree Insertion Example

The lower numbers indicate the subtree-size. The stars indicate tokens.

**Example:** \( \textit{Scapegoat-insert}(40), |P| = 5 > \log_{3/2}(7) \approx 4.79 \)
Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
MTF-heuristic for binary search trees

Example: $BST$-search(60)
MTF-heuristic for binary search trees

Example: $BST$-search$(60)$
MTF-heuristic for binary search trees

Example: \textit{BST-search}(60)
MTF-heuristic for binary search trees

**Example**: $\text{BST-search}(60)$

This can get quite unbalanced!
Double Left Rotation = Zig-zag Rotation

First, a right rotation at \( p \). Second, a left rotation at \( g \).
Zig-zig Rotation

First, a left rotation at $g$. Second, a left rotation at $p$. 
Compare to doing two single rotations

Seemingly minor change, but allows for amortized analysis.
Outline

1. Dictionaries and Balanced Search Trees
   - Insertion in Treaps
   - More AVL insertions
   - Scapegoat Trees
   - Search in Self-Adjusting Trees
   - Splay Trees
Splay Tree Insertion

\textbf{SplayTree-insert}(r, k, v)

1. \( x \leftarrow \text{BST-insert}(r, k, v) \)
2. \textbf{while} (\( x \) is not the root)
3. \( p \leftarrow \text{parent of } x \)
4. \textbf{if} (\( x \) is the left child of \( p \))
5. \textbf{if} (\( p \) is the root)
6. \textit{rotate-right}(p)
7. \textbf{else} let \( g \) be the parent of \( p \)
8. \textbf{case}
9. \begin{align*}
\qquad & \begin{array}{c}
\quad g \\
\quad p
\end{array} : \quad /\!/ \text{Zig-zig rotation} \\
\quad & \text{rotate-right}(g) \\
\quad & \text{rotate-right}(p)
\end{align*}
10. \begin{align*}
\qquad & \begin{array}{c}
\quad g \\
\quad p
\end{array} : \quad /\!/ \text{Zig-zag rotation} \\
\quad & \text{rotate-right}(p) \\
\quad & \text{rotate-left}(g)
\end{align*}
11. \textbf{else} … \quad /\!/ \text{symmetric case, } x \text{ is right child}
**Splay Tree rotations**

**Example:** *Splay Tree-search(60)*

```
Example: Splay Tree-search(60)
```

```
Biedl, Petrick, Veksler (SCS, UW)
CS240 – Module 4
Winter 2019
```
Splay Tree rotations

Example: \textit{Splay Tree-search}(60)