Runtime Analysis on a Binary Tree:

A Searchable Binary Tree?

Binary Search Tree Property:

Finding an element in a BST:

```
BST.hpp

template <typename K, typename V>
void BST<K, V>::_insert(TreeNode *& root, K key, V value) {

Running time? ____________    Bound by? ___________

What if we did **not** pass a pointer by reference?
```
Removing an element from a BST:

_remove(40)
_remove(25)
_remove(10)
_remove(13)

<table>
<thead>
<tr>
<th>One-child Remove</th>
<th>Two-child remove</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BST – Simple Ideas

Q: Given a height \( h \), what is the maximum number of nodes \( n \) in a valid BST of height \( h \)? Provide an outline of a proof.

Q: Given a height \( h \), what is the minimum number of nodes \( n \) in a valid BST of height \( h \)? Provide an outline of a proof.

Final BST Analysis

For every height-based algorithm on a BST:

Lower Bound:

Upper Bound:

Why use a BST over a linked list?

Q: How does our data determine the height?

1 3 2 4 5 7 6 vs. 4 2 3 6 7 1 5

Running time? Bound by?

BST Analysis:

Every operation we have studied on a BST depends on:

...what is this in terms of the amount of data, \( n \)?

---

CS 225 – Things To Be Doing:

1. mp_lists due Monday
2. lab_trees due Sunday
3. Daily POTDs

---