Announcements


TODAY: balanced BST (intro)


Running times:
insert
remove
find
traverse
Binary Search Tree - miscellaneous characteristics and analysis

```cpp
BST<int> myT;
myT.insert(2);
myT.insert(7);
myT.insert(15);
myT.insert(22);
myT.insert(28);
...
```

Give a sequence of inserts that result in a tree that looks like:

How many “bad” n-item trees are there?
Binary Tree -

The *algorithms* on BST depend on the height (h) of the tree.

The *analysis* should be in terms of the amount of data (n) the tree contains.

So we need a relationship between h and n.

\[ h \geq f(n) \]
\[ h \leq g(n) \]

Reminder:

height(T) is:

- ______ if T is empty
- \( 1 + \max\{\text{height}(T_L), \text{height}(T_R)\} \), otherwise
Binary Tree (theory moment #1)

what is maximum number of nodes in a tree of height $h$?

what is the least possible height ($h$) for a tree of $n$ nodes?
Binary Tree (theory moment #2)

what is minimum number of nodes (n) in a tree of height h?

what is the greatest possible height (h) for a tree of n nodes?

thus: lower bd on ht _______, upper bd on ht ________, good news or bad?
Binary Search Tree -

The height of a BST depends on the order in which the data is inserted into it.

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

How many different ways are there to insert n keys into a tree?

Avg height, over all arrangements of n keys is _____________.

<table>
<thead>
<tr>
<th>operation</th>
<th>avg case</th>
<th>worst case</th>
<th>sorted array</th>
<th>sorted list</th>
</tr>
</thead>
<tbody>
<tr>
<td>find</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insert</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delete</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>traverse</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
something new… which tree makes you happiest?

The “height balance” of a tree T is:

\[ b = \text{height}(T_L) - \text{height}(T_R) \]

A tree T is “height balanced” if:

- 
- 
-
Binary Search Tree - is this tree “height balanced”?