Announcements

Binary Trees

A binary tree is a tree that has a left branch and a right branch.

Idea: Fill the place of a missing left branch with an empty tree.

Idea 2: An instance of BTree always has exactly two branches.

$$
\begin{array}{c}
3 \\
1 \\
0 \\
\end{array}
$$

E: An empty tree

```python
class BTree(Tree):
    empty = Tree(None)

def __init__(self, label, left=empty, right=empty):
    Tree.__init__(self, label, [left, right])

@property
def left(self):
    return self.branches[0]

@property
def right(self):
    return self.branches[1]
```

```python
t = BTree(3, BTree(1), BTree(7, BTree(5), BTree(9, BTree.empty, BTree(11))))
```

Demo

Binary Search

A strategy for finding a value in a sorted list: check the middle and eliminate half.

```python
20 in [1, 2, 4, 8, 16, 32, 64]  # True
4 in [1, 2, 4, 8, 16, 32]     # True
1 in [1, 2, 4, 8, 16, 32]     # False
```

For a sorted list of length n, what Theta expression describes the time required? $O(\log n)$

```
def largest(t):
    if t.right is BTree.empty:
        return t.label
    return largest(t.right)

def second(t):
    if t.is_leaf():
        return None
    elif t.right is BTree.empty:
        return largest(t.left)
    elif t.right.is_leaf():
        return t.label
    return second(t.right)
```

Discussion Questions

What's the largest element in a binary search tree?
```python
def largest(t):
    if t.right is BTree.empty:
        return t.label
    return largest(t.right)
```

What's the second largest element in a binary search tree?
```python
def second(t):
    if t.is_leaf():
        return None
    elif t.right is BTree.empty:
        return largest(t.left)
    elif t.right.is_leaf():
        return t.label
    return second(t.right)
```
Sets as Binary Search Trees

Membership in Binary Search Trees
• If the element is not at the root, it can only be in either the left or right branch
• By focusing on one branch, we reduce the set by the size of the other branch

def contains(s, v):
    if s is BTree.empty:
        return False
    elif s.label == v:
        return True
    elif s.label < v:
        return contains(s.right, v)
    elif s.label > v:
        return contains(s.left, v)

If 9 is in the set, it is in this branch

Order of growth? \( H(n) \) on average \( \log(n) \) on average for a balanced tree

Adjoining to a Tree Set

Right! Left! Right! Stop!

(Demo)