Traversal vs. Search:
- **Traversal** visits every node in the tree exactly once.
- **Search** finds one element in the tree.

Breadth First Traversal + Search:

Depth First Traversal + Search:

Runtime Analysis on a Binary Tree:
- Find an element: Best case? Worst case?
- Insertion of a sorted list of elements?
  - Best case? Worst case?
- Running time bound by?

Dictionary ADT

```
Dictionary.h

3
4
class Dictionary {
5     public:
6             
7             
8             
9             
10
11
12
13
14
15
16
17
};
```

```
BST.cpp

find

(TreeNode *root, const K &key) const {

    }
}
```
Inserting an element into a BST:

![BST Diagram]

BST.cpp

```cpp
template <class K, class V>
void BST::_insert(TreeNode *& root, K key, V value) {
    // Insertion code here
}
```

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)

![BST Diagram]

BinaryTree.cpp

```cpp
template <class K, class V>
void BST::_remove(TreeNode *& root, const K & key) {
    // Removal code here
}
```

Running time? ____________    Bound by? ___________

CS 225 – Things To Be Doing:

1. Theory Exam 2 Topics List Posted (exam next week)
2. MP3 extra credit on-going; due Monday, Feb. 26
3. Upcoming Lab: lab_trees
4. Daily POTDs