Traversal vs. Search:
- **Traversal** visits every node in the tree exactly once.
- **Search** finds one (or more) element(s) in the tree.

Breadth First Traversal + Search:

Depth First Traversal + Search:

Dictionary ADT

```cpp
class Dictionary {
public:

private:

};
```

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?

A Searchable Binary Tree?

Binary Search Tree Property:

Finding an element in a BST:

```cpp
BST.hpp

template <typename K, typename V> find(const K & key) const {
    
} const

template <typename K, typename V> find
    (TreeNode * & root, const K & key) const {
    
}
```
Inserting an element into a BST:

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

What if we did not pass a pointer by reference?

Running time? ____________    Bound by? ____________

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>

```
BinaryTree.hpp
template <class K, class V>
void BST<K,V>::_remove(TreeNode *& root, const K & key) {
}
```

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; MP3 due Monday, Feb. 25
3. Upcoming Lab: lab_trees
4. Daily POTDs