

# For Monday

- Read

<http://www.acm.org/serving/se/code.htm>

- No homework

# Program 9

# Uses for Traversals

# Dictionary ADT

- AbstractDataType *Dictionary*{

instances:

collection of elements with distinct keys

operations:

*Create()*

create an empty dictionary

*Search(k)*

return element with key *k*

*Insert(x)*

insert element *x* into dictionary

*Delete(k)*

delete element with key *k* from  
dictionary

}

# Dictionary Implementations

# Binary Search Trees

- An addition to our possible dictionary implementations
- We're interested in operations:
  - Search
  - Insert
  - Delete
  - Output in ascending order of keys
  - FindMin
  - FindMax

# Definition

- Binary tree
- Each element has an associated key
- Keys are not duplicated
- For any node, all nodes in its left subtree (if any) have keys smaller than its key and all nodes in its right subtree (if any) have keys larger than its key

# Binary Search Tree Operations

- Search
  - Time Complexity
- Insert
  - Time Complexity
- Deletion
  - Of a leaf
  - Of a node with one non-empty subtree
  - Of a node with two non-empty subtrees
  - Time Complexity
- Output Ordered List
- FindMin
- Find Max