These must be completed and shown to your lab TA either by the end of this lab, or by the start of your next lab.

1. Download the binary search tree code from the course web page under Lab 4.
2. Implement the following functions in `bst.cpp`:

```
/**
 * Returns the number of nodes in the tree rooted at root.
 */
int numNodes( Node* root );

/**
 * Returns the number of leaves in the tree rooted at root.
 */
int numLeaves( Node* root );

/**
 * Returns the height of node x.
 */
int height( Node* x );

/**
 * Returns the depth of node x in the tree rooted at root.
 */
int depth( Node* x , Node* root );

/**
 * Traverse a tree rooted at rootNode in-order and use 'v' to visit each node.
 */
void in_order( Node*& rootNode, int level, Visitor& v );

/**
 * Traverse a tree rooted at rootNode pre-order and use 'v' to visit each node.
 */
void pre_order( Node*& rootNode, int level, Visitor& v );

/**
 * Traverse a tree rooted at rootNode post-order and use 'v' to visit each node.
 */
void post_order( Node*& rootNode, int level, Visitor& v );
```

3. Complete the missing portion of `delete_node` in `bst.cpp`:

```
/**
 * Deletes a node containing 'key' in the tree rooted at 'root'.
 */
bool delete_node(Node*& root, KType key);
```

4. Which functions would change if the Nodes were part of a binary tree that didn't have the search tree property (the invariant that requires `left < parent < right`)?
5. Be sure to show your work to your TA before you leave, or at the start of the next lab, or you will not receive credit for the lab!