Exercise 22.22 – Binary Tree Delete (pg. 937)

Three Cases:

1. Item is contained in a leaf node (i.e. has no children)

*Node is deleted and the reference in the parent node is set to null*

1. Item is contained in a node with ONE child

*The reference in the parent node is set to reference the child node and the node containing the data item is deleted, causing the child node to take the place of the deleted node in the tree*

1. Item is contained in a node with TWO children

*Another node in the tree must take the deleted node’s place; which is:*

* The node containing the largest value in the tree less than the value in the node being deleted

-OR-

* The node containing the smallest value in the tree greater than the value in the node being deleted
  + Located by walking down the left subtree to the right until the reference to the right child of the current node is null. At which point the replacement node is being referenced.

If the replacement node is a leaf node

1. Store the reference to the node to be deleted in a temporary reference variable
2. Set the reference in the parent of the node being deleted to reference the replacement node
3. Set the reference in the parent of the replacement node to null
4. Set the reference to the right subtree in the replacement node to reference the right subtree of the node to be deleted
5. Set the reference to the left subtree in the replacement node to reference the left subtree of the node to be deleted

If the replacement node is a node with a left child

1. Store the reference to the node to be deleted in a temporary reference variable
2. Set the reference in the parent of the node being deleted to refer to the replacement node
3. Set the reference in the parent of the replacement node to reference the left child of the replacement node
4. Set the reference to the right subtree in the replacement node to reference the right subtree of the node to be deleted
5. Set the reference to the left subtree in the replacement node to reference the left subtree of the node to be deleted

Write method **deleteNode(String deleteValue)**

* Should locate the node in the tree containing the value to delete, and then delete said node
* If the value is not found in the tree, display a message
* After deleted an item/node, call methods inorderTraversal(), preorderTraversal(), and postorderTraversal() to confirm the delete operation was performed correctly