For this assignment it really was a challenge especially when it came to removing nodes from our tree as it took probably 10-15 emails and hours of troubleshooting just to figure out that I had taken my public remove function and made it into my private removeNode. This assignment really was difficult for me but after completing it, I found that I do like how you code when using a binary search tree.

Psuedocode

**Creating the Bid**We start by giving it the string variables bidId, title, and fund and also the double variable for the amount of money in the bid.  
  
**Creating the Nodes**Under node we make bid, left and, right.  
after that we make left and right both equal to null pointer  
then we initialize the bid under our node  
  
**Defining the Binary Search Tree**Under private we make a new node for the root of the tree, we add functions for adding nodes, calling the tree in order, pre order and, post order as well as a way to remove a node.  
Under public we set up the Binary search tree, the virtual of the binary search tree, the post, pre and, in order, the insert, the remove and, the search function.  
  
**Setting up root**  
set the root to equal null pointer  
**Setting up the public versions of inOrder, postOrder, preOrder, insert and, remove**  
**In order**  
we call the private in order function and pass the root  
**Post Order**  
we call the private Post Order function  
**Pre Order**  
we call the private Pre Order Function  
**Insert**If the root is equal to null pointer we set the root to equal the new Node(bid)  
else we call the Private add node function and pass the root and bid to that function.  
**Remove**  
we call the remove Node function  
**Searching the Tree**  
we start by making the current equal to the root, next we create a while loop that will continue to iterate while current is not equal to null pointer.  
**inside the while loop**   
if the current bid id is the same as the searched bid Id we return the current bid  
if the searched bid is smaller than the current bid we traverse the current down the left path of the tree  
else we will traverse the tree in the right path of the tree as the searched bid will be greater than the one currently held.  
**Outside of the while loop**  
we have Bid bid then return the bid.

**Adding a Node**if the current bid is larger than the inputted one we add this to the left side of the tree, if there is no left node than the inputted bidId is created as the left node, if there is a left node we keep traversing until there is no left node.  
if the current bid is smaller than the inputted one we add it to the right side of the tree, if there is no right node than the inputted bidId is created as the right node, if there is a right node we continue until there is no right node.

**Removing Node**if the node is equal to null pointer we return the node.  
we compare the bid id and the node’s bid id if it is less than 0 we remove the node on the left, if the comparison is greater than 0 we remove the node on the right.  
if the left node and right node are both equal to null pointer we delete the given node and set the node equal to null pointer.  
if the left node is not equal to null pointer and right node is equal to null pointer we create the temp node and set it equal to the node we then set the node equal to the right node, we then delete the temp node.  
if left node is equal to null pointer and right node is not equal to null pointer, we create the node temp and set it equal to node, we then set node to node right then we delete the temp node.

Else we create a temp node equal to node right then we make a while loop that iterates as long as temp left is not equal to null pointer we traverse the tree till it reaches null pointer. After the loop we set the node bid equal to temp bid and node right we remove node (node right, temp bidId)

**In order**if node is not equal to null pointer we call in order with node left, we then output the bidId, title, amount and, fund. Then we call in order node right.  
  
**Post order**if node is not equal to null pointer, post order node left then post order node right, we then output the bidId, title, amount and, fund. Then we call in order node right.

**Pre Order**if node is not equal to null pointer, we output the bidId, title, amount and, fund. Then we call in order node right. Then call pre order node left then pre order node right.