Pseudocode

Task 1: Define structures for tree node, housekeeping variables

Create a Bid container and call it bid

Create a pointer for the left side

Create a pointer for the right side

Create a constructor for the node with no parameters

Assign left pointer to null

Assign right pointer to null

Create a constructor for the node with a bid parameter

Assign bid to the bid parameter

Task 2: Implement inserting a bid into the tree

2a: Implement inserting a bid into the tree

Check if root is null

Assign root to a new node

If not

Call addNode passing the root and the bid

2b: Implement inserting a bid into the tree

Check if the passed node is greater than the root

Check if the passed node’s left pointer is null

Make the passed node’s left pointer to a new node

If not

Call addNode passing the passed node’s left pointer and the bid

If not

Check if the passed node’s right pointer is null

Make the passed node’s left pointer to a new node

If not

Call addNode passing the passed node’s right pointer and the bid

Task 3: Implement searching the tree for a bid

Make a new node pointer called current and assign it to the root node

Loop while current does not equal null

Check if the current node’s bid id is equal to the searched

Return the current’s bid

Check if the current node's bid id is less than the searched

Assign current to current’s left pointer

If not

Assign current to current’s right pointer

Make a new bid

Return the new bid

Task 4: Implement removing a bid from the tree

4a: Implement removing a bid from the tree

Call removeNode passing the root and the bid id

4b: Implement removing a bid from the tree

Check if the passed node is null

Return the passed node

Check if the passed node’s bid id is less than the searched

Assign passed node’s left to removeNode passing the passed node’s left and the bid id

If not check if the passed node’s bid id is larger than the searched

Assign passed node’s right to removeNode passing the passed node’s right and the bid id

If not

Check if the passed node’s left is null and the passed node’s right is null

Delete the passed node

Assign node to null

If not check if passed node’s left is not null and passed node’s right is null

Create a new node pointer called temp and assign it to passed node

Assign node to the node’s right

Delete temp

If not check if passed node’s right is not null and passed node’s left is null

Create a new node pointer called temp and assign it to passed node

Assign node to the node’s left

Delete temp

If not

Create a new node pointer called temp and assign it to passed node’s right

Loop while the temp’s left is not null

Assign temp to temp’s left

Assign node’s bid to the temp’s bid

Assign node’s right to removeNode passing the passed node’s right and the temp’s bid id

Return the passed node