Tyler Stovsky

705512370

1 a. 50

/ \

20 60

/ \ \

10 40 70

\ / / \

15 30 65 80

/ \ /

25 31 79

1 b. Inorder: 10 15 20 25 30 31 40 50 60 65 70 79 80

Preorder: 50 20 10 15 40 30 25 31 60 70 65 80 79

Postorder: 15 10 25 31 30 40 20 65 79 80 70 60 50

1 c.

After removing the 30:

50

/ \

20 60

/ \ \

10 40 70

\ / / \

15 25 65 80

\ /

31 79

After removing the 20:

50

/ \

15 60

/ \ \

10 40 70

/ / \

25 65 80

\ /

31 79

2. a.

struct Node {

int data;

Node\* left;

Node\* right;

Node\* parent;

}

2 b.

If the tree is empty, create a new node with value and point children and parent to null

If the value to insert is less than the current value

If the current left child is empty

Create a new node with the correct value and children as null

Set the current node’s left pointer to the new node

Set the new node’s parent pointer to the current node

Otherwise, recursively call the function again

Else if the value to insert is greater than the current value

If the current right child is empty

Create a new node with the correct value and children as null

Set the current node’s right pointer to the new node

Set the new node’s parent pointer to the current node

Otherwise, recursively call the function again

Otherwise, we have a duplicate value so do nothing

3 a. 7

/ \

5 6

/ \ /

4 1 2

3 b. 7 5 6 4 1 2

3 c. 6 5 2 4 1

4 a. O(C + S)

4 b. O(log C + S)

4 c. O(log C + log S)

4 d. O(log S)

4 e. O(1)

4 f. O(log C + S)

4 g. O(s log S)

4 h. O(c logS)