HW 5 - Angela Kan

1. **a.**

50

20

10

40

30

60

70

80

65

78

15

32

25

**b.**

Preorder: 50, 20, 10, 15, 40, 30, 25, 32, 60, 70, 65, 80, 78

Inorder: 10, 15, 20, 25, 30, 32, 40, 50, 60, 65, 70, 78, 80

Postorder: 15, 10, 25, 32, 30, 40, 20, 65, 78, 80, 70, 60, 50

**c.**

50

25

10

40

32

60

70

80

65

78

15

1. **a.**

struct Node {

Node(int val) : value(val), parent(nullptr), leftChild(nullptr), rightChild(nullptr) { }

int value;

Node\* parent;

Node\* leftChild;

Node\* rightChild;

};

**b.** pseudocode to insert a new node:

*if tree is empty:*

*have root pointer point to newly allocated node with the value to be inserted*

*return*

*else have current node point to root node of tree*

*while we haven’t inserted the node yet:*

*if value is equal to current node*

*return*

*if value is less than current node*

*if current node has left child:*

*set current node to equal left child*

*else if current node doesn’t have left child:*

*have current node’s left pointer point to newly allocated node w/ value*

*have new node’s parent pointer point to current node*

*return*

*if value is greater than current node*

*if current node has right child:*

*set current node equal to right child*

*else if current node doesn’t have right child:*

*have current node’s right ptr point to newly allocated node w/ value*

*have new node’s parent pointer point to current node*

*return*

1. **a.**

7

5

6

4

0

2

**b.** {7, 5, 6, 4, 0, 2}

**c.** {6, 5, 2, 4, 0}

1. **a.** O(C + S)

**b.** O(log C + S)

**c.** O(log C + log S)

**d.** O(log S)

**e.** O(1)

**f.** O(log C + S)

**g.** O(S log S)

**h.** O(C log S)