Alex Chen

CS 32

HW 5

1)

a. 50



20 60



10 40 70



15 30 65 80



25 34 76

b.

i. Pre-order: 50, 20, 10, 15, 40, 30, 25, 34, 60, 70, 65, 80, 76

ii. In-order: 10, 15, 20, 25, 30, 34, 40, 50, 60, 65, 70, 76, 80

iii. Post-order: 15, 10, 25, 34, 30, 40, 20, 65, 76, 80, 70, 60, 50

c. 50



15 60



10 40 70



34 65 80



25 76

2)

a.

struct TreeNode

{

TreeNode(int val)

{

m\_value = val;

left = right = parent = nullptr;

}

int m\_value;

TreeNode\* left;

TreeNode\* right;

TreeNode\* parent;

};

b.

void insert(int val)

{

If root ptr is null

Create a new node with val and set root to node

Else, traverse through tree starting at root

If current node value is equal to val, return

if val is less than current node val

if left side has nullptr

create new node with val and link up nodes, including setting parent to current node and return

else go left

else

if right side has nullptr

create new node with val and link up nodes, including setting parent to current node and return

else go right

}

3)

a. 7



3 5



0 2 4

b. int arr = {7, 3, 5, 0, 2, 4};

c. arr = {5, 3, 4, 0, 2};

4)

a. O(C + log(S))

b. O(log(C) + S)

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

d. O(log(S))

e. Constant

f. O(log(C) + S)

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

h. O(S)

5)

b. A two-parameter overloaded function of listAll is required because in addition to the current class pointer being passed, a string must be passed to store the string representation of all of the previous levels of inheritance. Otherwise, a recursive solution would not have previous base classes to print out.