Ethan Wong

Smallberg

CS32

Homework 5

1a)

1b)

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

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

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

1c) (Deleting the 30, then 20 from tree in 1a)

First part - deleting the 30

Second part – deleting the 20

2a)

struct Node

{

Node\* leftChild;

Node\* rightChild;

Node\* parent;

int value;

};

2b)

(inserting a new node into a binary search tree with parent pointers)

void insert(int v)

{

if(root == nullptr)

root = new Node with value = v;

point the root pointer at the new Node;

Node\* current = root;

while (we are not done yet/we have not inserted yet)

{

if (current’s value == v)

return; //the value is already stored within the tree, no need to insert it again

if (v < current’s value)

if (current’s left Child exists and is not nullptr)

current now points to current’s left Child;

else

create a new Node with value = v;

set current node’s left pointer to the new node;

return;

else if(v > current’s value)

if (current’s right Child exists and is not nullptr)

current now points to current’s right Child;

else

create a new Node with value = v;

set current node’s right pointer to the new node;

return;

}

}

3a)

3b) The array is as follows: [7, 3, 5, 0, 2, 4]

3c) The array is as follows: [5, 3, 4, 0, 2]

4a) O(C +S)

b) O(S + logC)

c) O(logC + logS)

d) O(logS)

e) O(1)

f) O(logC + S)

g) O(SlogS)

h) O(ClogS)

5b)

The two-parameter overload was necessary because it took in a string as one of its arguments. In the recursive call to this function, this string is needed because it takes on the name of the const Class\* c’s name (see line 7 of list.cpp). This string is then printed out. The recursive call for this function requires the string because It has to be printed along with the name of the object that the iterator is pointing to.