Problem 1a



Problem 1b

In-order traversal:

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

Pre-order traversal:

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

Post-order traversal:

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

Problem 1c

After deleting node 30:



After deleting node 20:



Problem 2a

struct Node

{

int data;

Node\* left;

Node\* right;

Node\* parent;

};

Problem 2b

Node\* insert(Node\* root, int toAdd)

{

if root is null,

return new Node with toAdd as data

check which child toAdd should be part of

recurse down appropriate child

set root’s child to appropriate child and set child’s parent to root

return root

}

Problem 3a



Problem 3b

As an array:



Problem 3c

Resulting array after one more h.remove(item);



Problem 4

1. O(C + S)
2. O(log C + S)
3. O(log C + log S)
4. O(log S)
5. O(1)
6. O(log C + S)
7. O(S \* log S)
8. O(C \*log S)

Problem 5b

The recursive version of listAll must have both the string path argument and the Class\* c argument. The path argument is needed so that the output can show the base classes of the current class given by c. The Class\* c argument is needed to print out information about the given class and its derived classes.