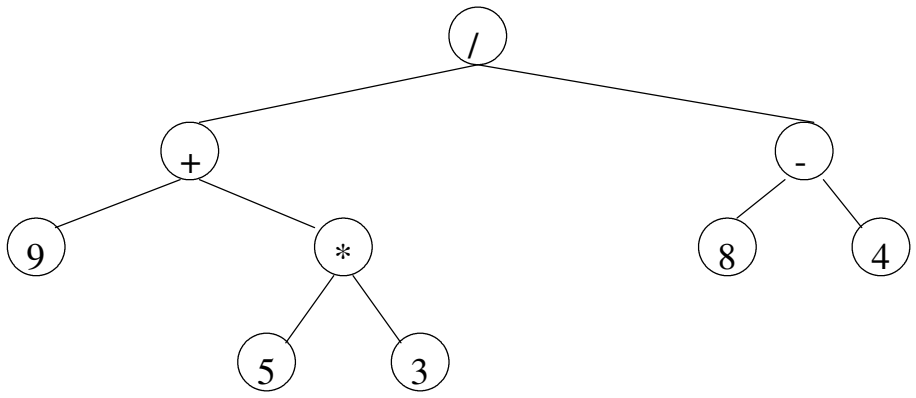
Team #:_____ Name:_____
Absent:

0. Consider the parse tree for (9 + (5 * 3)) / (8 - 4):



Indentify the following items in the above tree:

a. *node* containing "*"

b. *edge* from node containing "-" to node containing "8"

c. *root* node

d. *children* of the node containing "+"

e. *parent* of the node containing "3"

f. *siblings* of the node containing "*"

g. *leaf* nodes of the tree

h. *subtree* who's root is node contains "+"

i. *path* from node containing "+" to node containing "5"

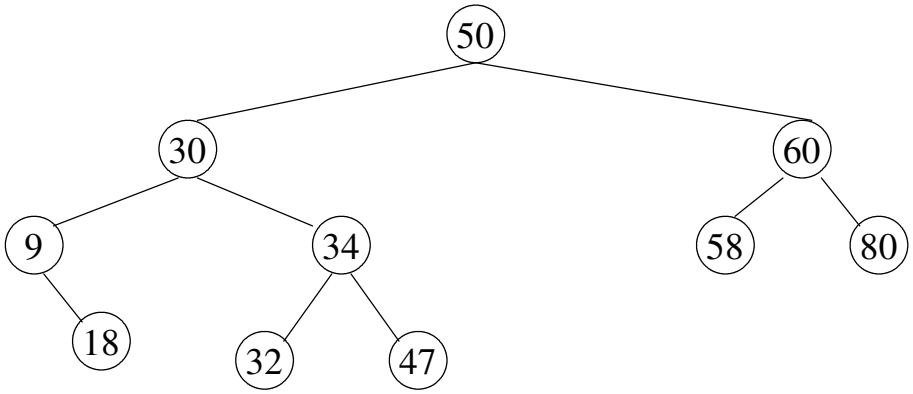
j. *branch* from root node to "3"

k. mark the *levels* of the tree (level is the number of edges on the path from the root)

l. What is the *height* (max. level) of the tree?

Team #:_____ Name:_____
Absent:

1. Consider the Binary Search Tree (BST):



a. What would be the result of an inorder traversal?

b. Starting at the root, how would you find the node containing "32"?

c. Starting at the root, when would you discover that "70" is not in the BST?

d. Starting at the root, where would be the "easiest" place to add "70"?

2. a) If a BST contains n nodes and we start searching at the root, what would be the worst-case theta Θ( ) notation for a successful search? (Draw the shape of the BST leading to the worst-case search)

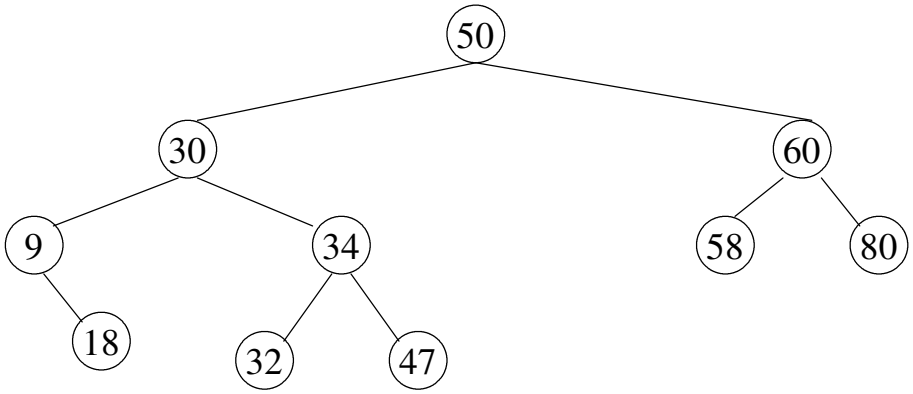
b) We could store a BST in an array like we did for a binary heap, what would be the worst-case storage needed for a BST with n nodes?

3. a) If a BST contains n nodes, draw the shape of the BST leading to best, successful search in the worst case.

b) What is the worst-case theta Θ( ) notation for a successful search in this "best" shape BST?

Team #:_____ Name:_____
Absent:

4. Consider the Binary Search Tree (BST):



a. What would be the result of deleting 58 from the BST?

b. What would be the result of deleting 9 from the BST?

c. What would be the result of deleting 50 from the BST? (Hint: One technique when programming is to convert a hard problem into a simpler problem. Deleting a BST node that contains two children is a hard problem. Since we know how to delete a BST node with none or one child, how could we convert"deleting a node with two children" problem into a simpler problem?)

Team #:_____ Name:_____
Absent:

```
// Specification file for the IntBinaryTree class
#ifndef INTBINARYTREE_H
#define INTBINARYTREE_H

class IntBinaryTree {
private:
   struct TreeNode    {
      int value;          // The value in the node
      TreeNode *left;     // Pointer to left child node
      TreeNode *right;    // Pointer to right child node
   };

   TreeNode *root;        // Pointer to the root node

   // Private member functions
   void insert(TreeNode *&, TreeNode *&);
   void destroySubTree(TreeNode *);
   void deleteNode(int, TreeNode *&);
   void makeDeletion(TreeNode *&);
   void displayInOrder(TreeNode *) const;
   void displayPreOrder(TreeNode *) const;
   void displayPostOrder(TreeNode *) const;

public:
   // Constructor
   IntBinaryTree()
      { root = NULL; }

   // Destructor
   ~IntBinaryTree()
      { destroySubTree(root); }

   // Binary tree operations
   void insertNode(int);
   bool searchNode(int);
   void remove(int);

   void displayInOrder() const
      {  displayInOrder(root); }

   void displayPreOrder() const
      {  displayPreOrder(root); }

   void displayPostOrder() const
      {  displayPostOrder(root); }
};
#endif
```