COP 3502 Study Group Sheet: Binary Trees Solutions

1) Show the result of inserting the following integers into an initially empty binary search in the order shown: 16, 8, 50, 60, 55, 90, 4, 5, 12, 9, and 45.

```
    16
   /    \
  8     50
 / \    / \
4   12 45  60
 \  /      / \
  5 9     55 90
```

2) Write function that returns the sum of all the values in the nodes in the binary tree with the root pointed to by root. If the pointer is NULL, return 0. Use the struct provided below.

```
typedef struct bintreenode {
    int data;
    struct bintreenode* left;
    struct bintreenode* right;
} bintreenode;

int sum(bintreenode* root) {
    if (root == NULL) return 0;
    return root->data + sum(root->left) + sum(root->right);
}
```

3) Draw a single **binary search tree** that meets all the following conditions:

- The tree contains 7 nodes.
- The tree's pre-order traversal is the same as its in-order traversal.
- The tree does not contain any duplicate values.

If it is not possible to draw such a tree, say so and explain why not.

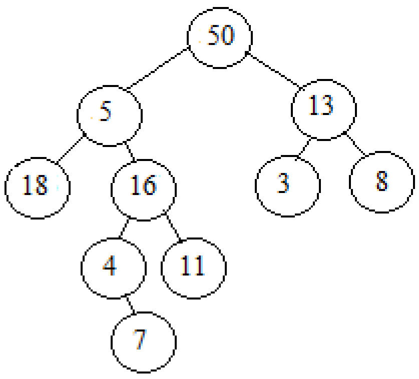
```
1
 \
  2
   \
    3
     \
      4
       \
        5
         \
          6
           \
            7
```

4) Complete writing the function shown below ***recursively***, so that it takes in a pointer to the root of a binary search tree of strings, *root*, and a string, *target*, and returns 1 if the string is contained in the binary search tree and false otherwise. You may assume all strings stored in the tree contain lowercase letters only. In order to receive full credit, your function must run in O(h) time, where h is the height of the binary search tree storing all of the words.

```
typedef struct bstNode {
   struct bstNode *left, *right;
   char str[100];
} bstNode;

int search(bstNode *root, char* target) {
    if (root == NULL) return 0;
    int res = strcmp(target, root->str);
    if (res < 0) return search(root->left, target);
    else if (res > 0) return search(root->right, target);
    return 1;
}
```

5) What does the function call `solve(root)` print out if root is pointing to the node storing 50 in the tree shown below? The necessary struct and function are provided below as well. Please fill in the blanks shown below. (Note: the left pointer of the node storing 50 points to the node storing 5, and all of the pointers shown correspond to the direction they are drawn in the picture below.)



```
typedef struct bstNode {
    int data;
    struct bstNode *left;
    struct bstNode *right;
} bstNode;

int solve(bstNode* root) {

    if (root == NULL) return 0;

    int res = root->data;
    int left = solve(root->left);
    int right = solve(root->right);

    if (left+right > res)
        res = left+right;

    printf("%d, ", res);
    return res;
}
```

18, 7, 7, 11, 18, 36, 3, 8, 13, 50