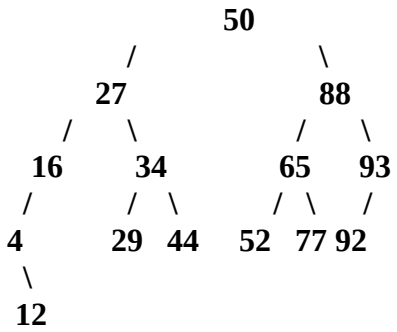
Computer Science I – Summer 2011
Recitation #7: Binary Search Trees (Solutions)

1. Draw the binary search tree that results from inserting the following values into an initially empty binary search tree in the following order: 50, 27, 16, 88, 34, 65, 52, 77, 93, 4, 12, 29, 44, 92



2. What are the outputs of a pre-order and post-order traversal of the final binary search tree drawn in question 1?

Pre-order: **50, 27, 16, 4, 12, 34, 29, 44, 88, 65, 52, 77, 93, 92**

Post-order: **12, 4, 16, 29, 44, 34, 27, 52, 77, 65, 92, 93, 88, 50**

3. If a search was conducted for the value 37 in the final binary search tree from question #1, which nodes would get visited? (List them in the order they get visited.)

50, 27, 34, 44

4. Write a function which returns the smallest value stored in a *non-empty* binary search tree. The prototype is below:

```
int minVal(struct treenode* root) {

  // Okay to look at left since root isn't NULL.
  if (root->left == NULL)
    return root->data;

  // Okay to call this since, root->left isn't NULL.
  return minVal(root->left);

}
```

5. Write a function which returns the number of leaf nodes in a binary search tree. The prototype is below:

```
int numLeafNodes(struct treenode* root) {

  if (root == NULL) return 0;

  if (root->left == NULL && root->right == NULL)
    return 1;

  return numLeafNodes(root->left) +
         numLeafNodes(root->right);

}
```

6. What does the following function do?

```
struct treenode* q6(struct treenode* root, int x) {

  if (root == NULL)
    return NULL;

  if (root->data > x) {

    struct treenode* tmp = q6(root->left, x);
    if (tmp == NULL)
      return root;
    else
      return tmp;
  }
  else
    return q6(root->right, x);
}
```

It returns a pointer to the node in the tree that stores the smallest value in the tree greater than x. If no such node exists (if all the values in the tree are less than or equal to x), then NULL is returned.