4) (10 pts) DSN (Binary Trees)

For this problem you will write a modified inorder traversal of a binary tree. In a regular inorder traversal, you simply visit the nodes of the tree "in order". Consider the added task of adding up all of the numbers as you see them in the traversal, one by one, and printing out each intermediate sum. For example, if the input binary tree was:

```
         10
        /   \
      15     8
     /  \     \
    3   12     17
        /
      10
```

the corresponding inorder traversal would visit 3, 15, 10, 12, 10(root), 8 and 17, in that order. For the added task, the traversal should print out 3, 18, 28, 40, 50, 58 and 75, respectively, the running sums after visiting each value.

Complete the function below, ***recursively***, so that is performs the given task. One way to accomplish this is to have the function take in a second value, `prevSum`, representing the previous sum of values prior to visiting the given node, and also to have the function return the sum of the nodes in its subtree.

```
typedef struct treenode {
    int value;
    struct treenode *left;
    struct treenode *right;
} treenode;

int inorderSum(treenode* root, int prevSum) {

    if (root == NULL) return 0;

    int sum = 0;
    sum += inorderSum(root->left, prevSum);

    sum += root->value;
    printf("%d ", sum+prevSum);

    sum += inorderSum(root->right, prevSum+sum);

    return sum;
}
```

Grading: There are 10 blanks, give 1 pt for each blank. If two blanks are slightly wrong, you may award 1 pt for both of them together.