1) (10 pts) DSN (Recursive Functions)

Write a ***recursive*** function that takes the root of a ternary tree (a tree where each node has at most three children) and determines whether all the nodes that have a middle child also have both a left child and a right child. If so, return 1. Otherwise, return 0. Note: If the function with a null input, the output should be 1.

The node struct and functional prototype for this question are:

```
typedef struct node
{
   char *data;
   struct node *left, *middle, *right;
} node;
```

```
int hasProperty(node *root)
{
  if (root == NULL)
     return 1;

  else if (root->middle == NULL)
     return hasProperty(root->left) && hasProperty(root->right);

  else if (root->left == NULL || root->right == NULL)
     return 0;

  else
     return hasProperty(root->left) &&
            hasProperty(root->right) &&
            hasProperty(root->middle);
}
```

Grading:

2 pts for root == NULL base case,
3 pts for handling root->middle == NULL case,
2 pts for handling case where exactly middle isn't NULL but left or right is
3 pts for handing case where none are NULL