**BST Remove**

1. Is this a binary search tree? Y/N How do you know?
2. When you remove a node, you have to preserve the BST order. Suppose you remove O. What letter(s) should go in its place? \_\_\_\_\_\_

How do you find that letter?

1. Suppose you remove G. What letter(s) should go in its place?\_\_\_

How do you find that letter?

1. Suppose you remove D. What letter(s) should go in its place?\_\_\_

How do you find that letter?

1. Suppose you remove K. What letter(s) should go in its place?\_\_\_

How do you find that letter?

1. Suppose you remove U. What should go in its place?\_\_\_

How do you find that letter?

1. Suppose you remove S. What should go in its place?\_\_\_

How do you find that letter?

1. Suppose you remove the root, H. What should go in its place?\_\_\_

Is deleting the root a special case or not?

1. What case(s) have we missed?
2. Describe and organize all the different cases:

**Algorithm for Removing a Node from a BST**



Let's try it out. For each N in each tree, remove the N and re-draw the pointers:

