Deletion in Red-Black Trees

As with ordinary BST deletion, if the node is interior, then its inorder predecessor and successor are both leaves or "elbows" (one child). Choosing either we move up its data and delete that node instead.

If the node to delete now is Red, we blip it as before. But if Black, after blipping it, we have a Subtree with One Black Too Few—a problem even if its 0th empty subtree:

- Note we just blipped—was black
- Sibling—exist since y was black
- "Away-Learning Niece" might not exist though

The double-line "=" means "This subtree Needs a Black."
The big question is: does 5's sibling's subtree have a Black to give? Answer is Yes when it has a Red "Away-Learning Niece."
Then 5 itself is Black, so the subtree X, and the subtrees rooted at M, Q, and R all have the same # of Blacks.

Now, rotate left at P:

1. [Diagram showing rotation]
2. Now suppose S has an In-Learning Red Nephew.
   Again 5 is Black and X, Q, R, N have all Black heights. Rotate right at 5:
   Then 5 itself becomes an Away-Learning Red Niece—prev case!
3. Now suppose 5 has no red children. If 5 is Black and 0 is red, we swap their colors and are done.
   If 5 is Black and 0 is Black, we rotate left at P:
   [Diagram showing rotation]
   Now we resolve with X as new root asking for 1 more Black. Since X went Red that equalsizes,
   If 5 now is the root of the whole tree this is when its Black height goes down.
4. So, we are left with the case where 5 is Red. Since 5 was originally Black, 5 must have two Black children

   Red
   [Diagram showing rotations]
   It seems like nothing has changed—the paths thru M and N still pick up 2 Blacks from the diagram while
   paths thru X or still missing 1 Black—what X took a
   skip backwards. However, X now has a Black sibling 0 and
   another Black sibling awaits at node N. Thus Cases 0–3 will
   be want push the recursion up to 5, so progress gets made. END