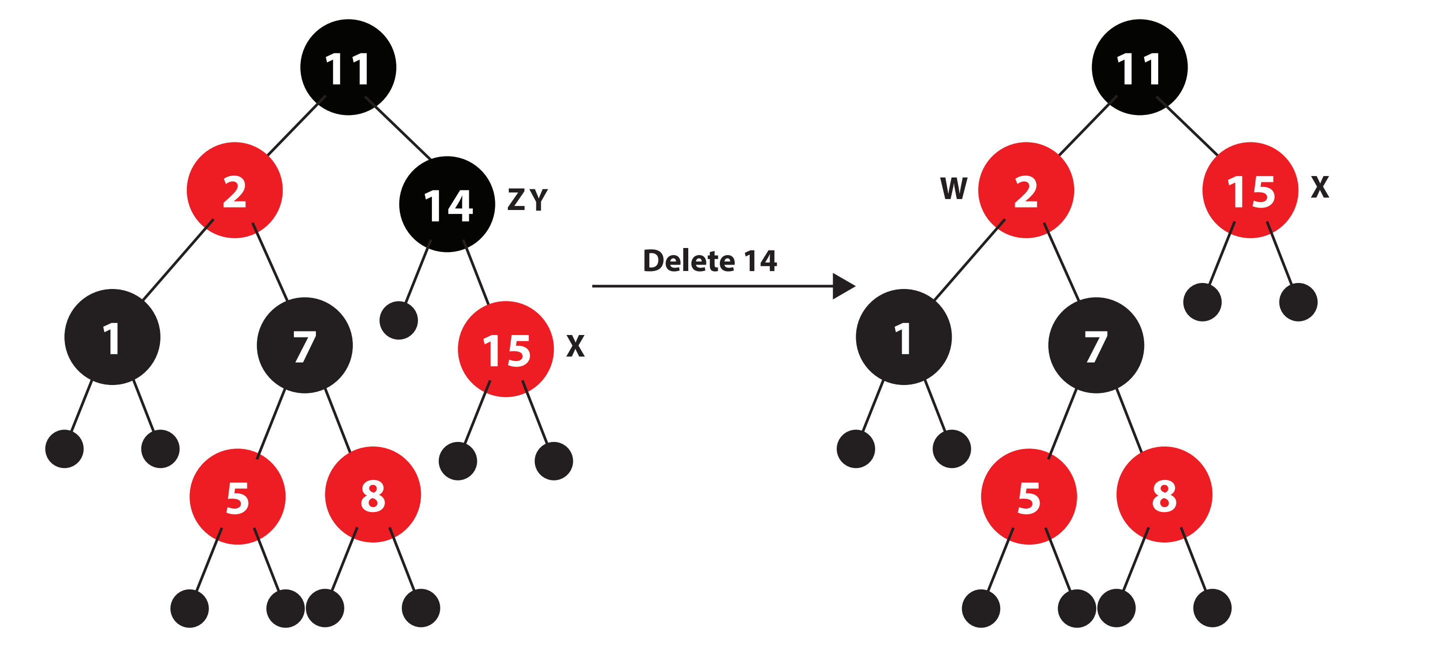
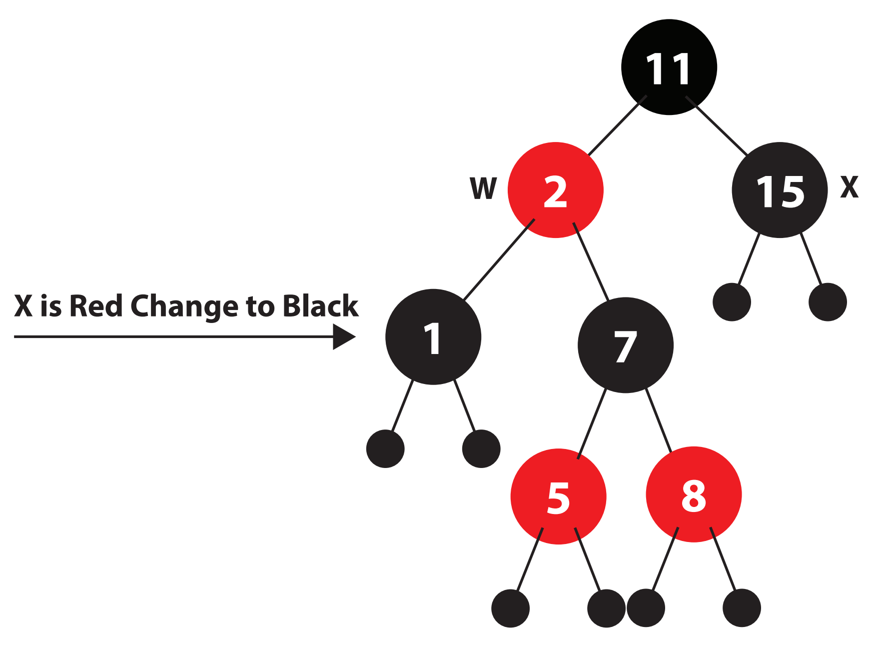
Red Black Tree Deletion :
Data Set: {14, 1, 5, 2, 11, 7, 15, 8}

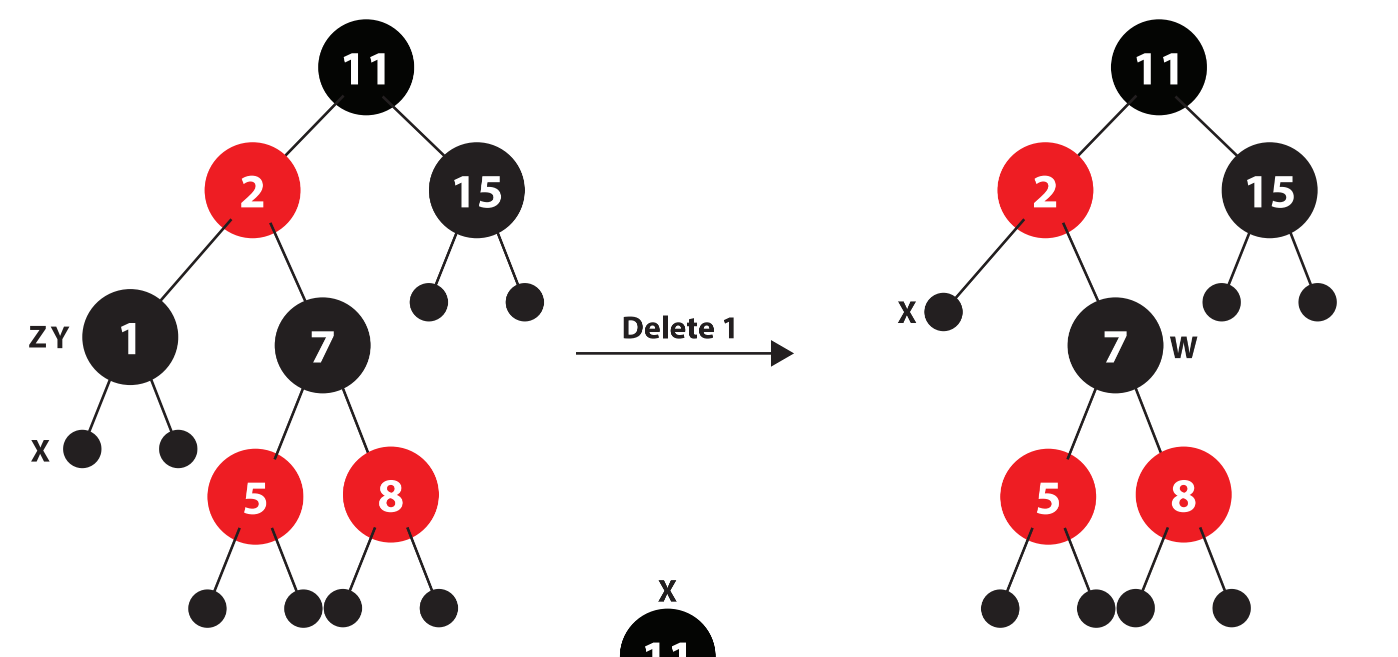
Step 1:
Insert {14, 1, 5, 2, 11, 7, 15, 8}

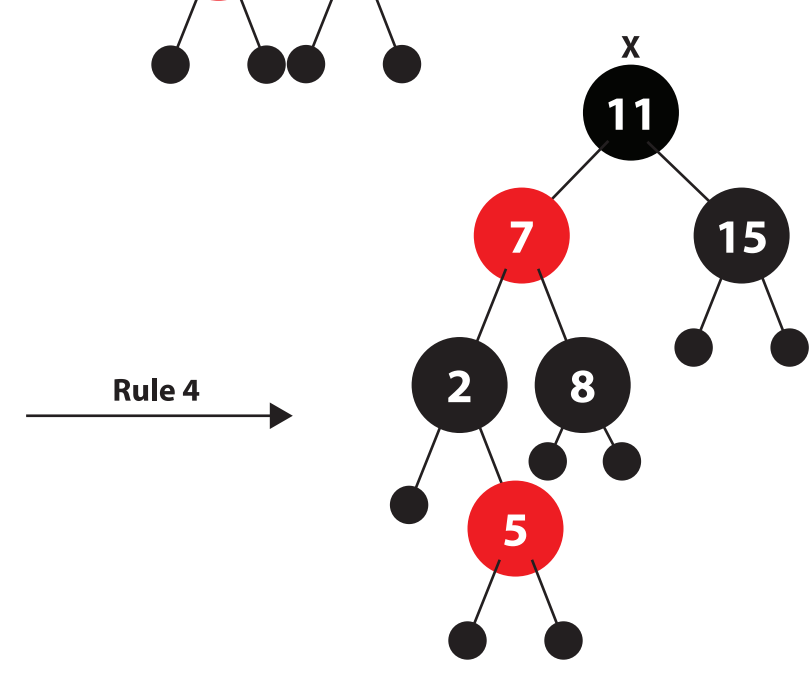




First Z does not have two non-nil children so we set Y also equal to Z. We set X to Z's child. Now we delete node Z normally. Next we notice X is red so we set to black and we are done. No further damage.

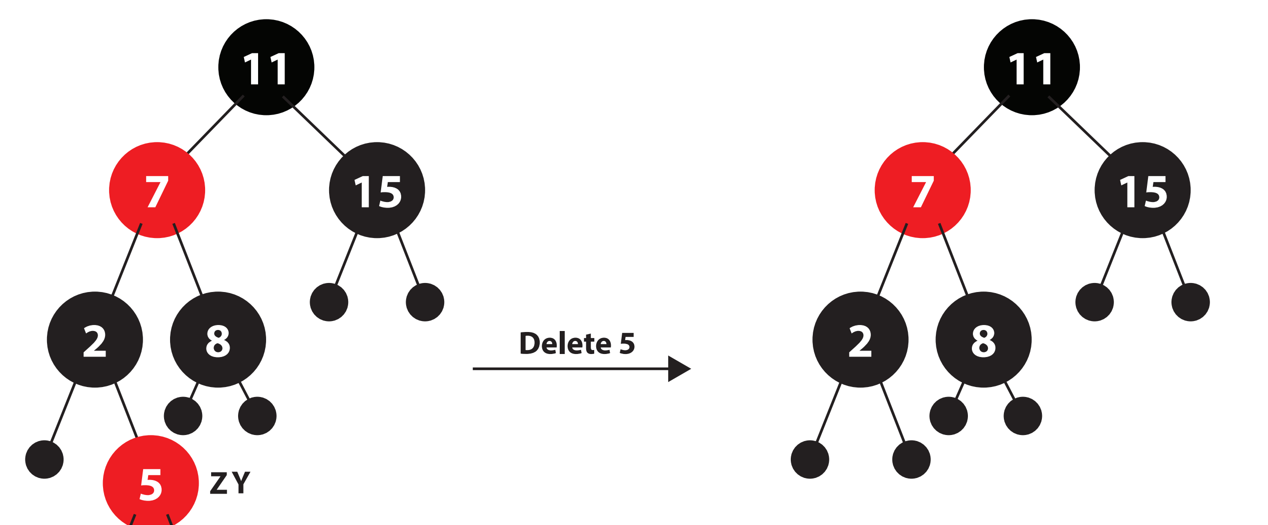
Step 2:
Insert {14, 1, 5, 2, 11, 7, 15, 8}





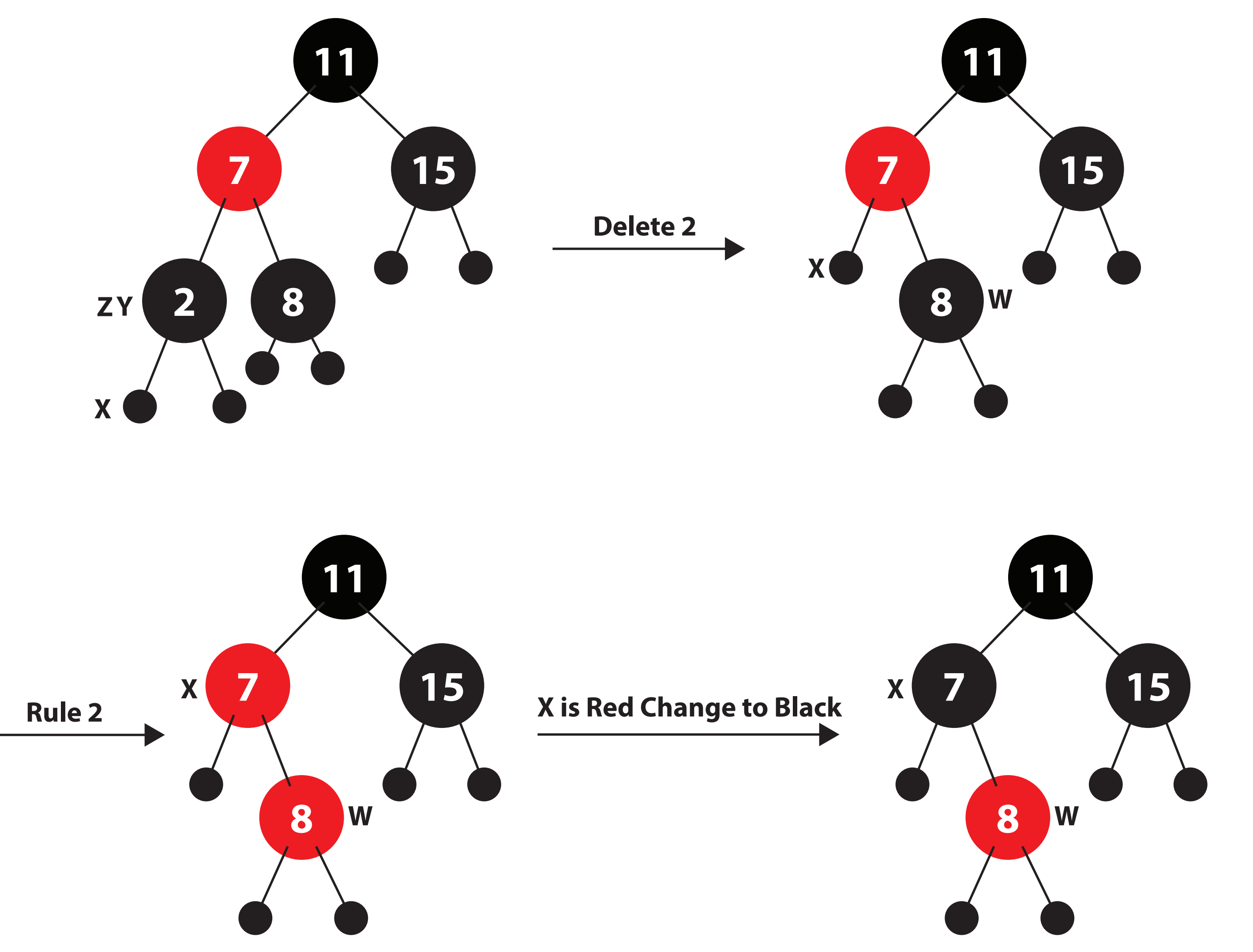
Z has two nill children so we set Y = Z and then X becomes left child of Z. We delete node Z like normal and then notice that X has a sibling node who is black and has 2 red children. The only Rule that will apply is Rule 4 so we perform a left rotation and then swap the color for the new subtrees root with its left child. We then set the subtrees roots right child to black and move X to the global root node (which ends the while loop). No further damage.

Step 3:
Insert {14, 1, 5, 2, 11, 7, 15, 8}



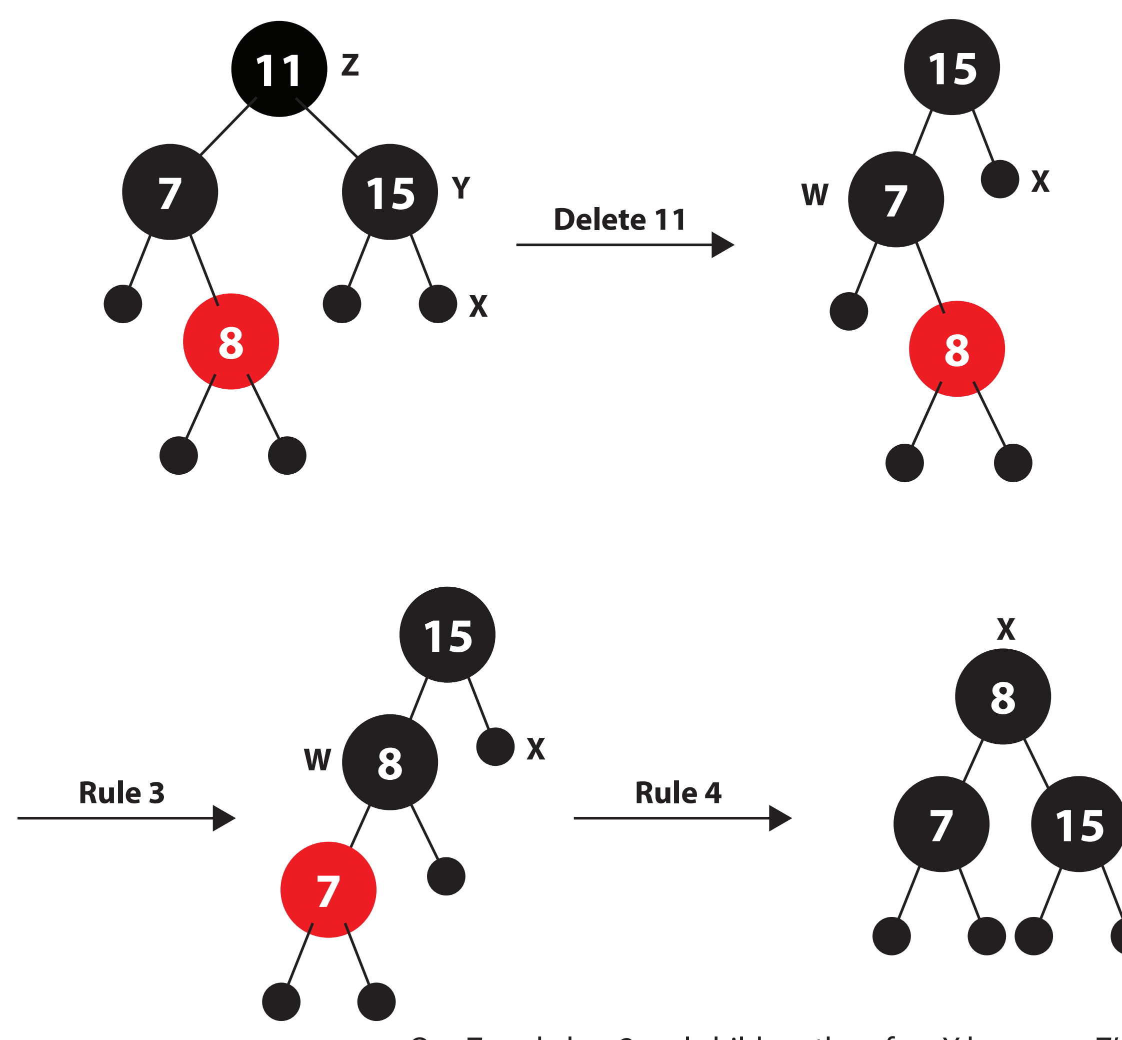
The node Z have two nill children nodes which menas Y = Z and X becomes the right child node. We then delete Z like normal. We got lucky however because Y was red before the deletion we are done and there is no damage.

Step 4:
Insert {14, 1, 5, 2, 11, 7, 15, 8}



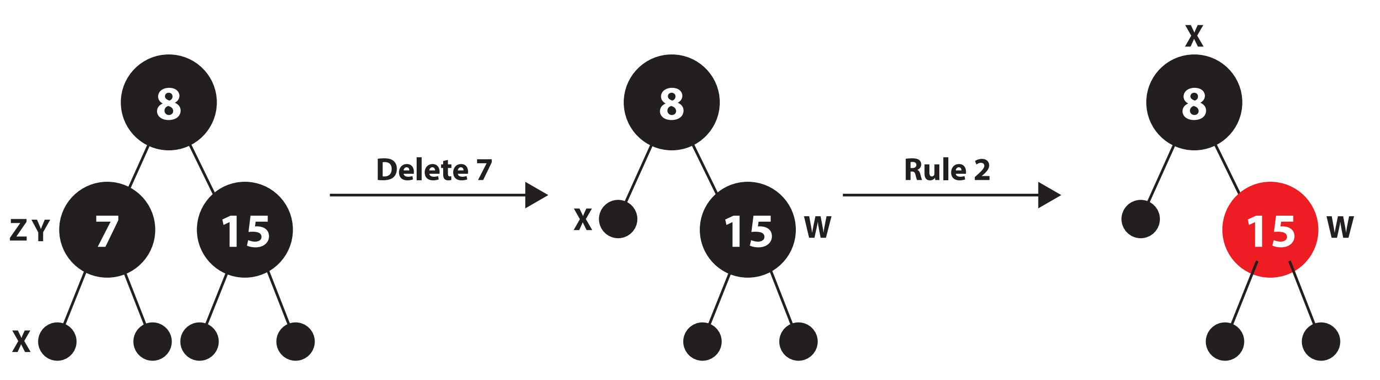
Node Z again has two nill children so Y = Z and then X becomes the left child of Z. Next we delete Z like normal and notice that X's sibling node is black and has two black children which means Rule 2 applies. We change the color of W and then set X = P(X). Now X is red so we change X to black and then the while loop ends and there is no further damage.

Step 5:
Insert {14, 1, 5, 2, 11, 7, 15, 8}



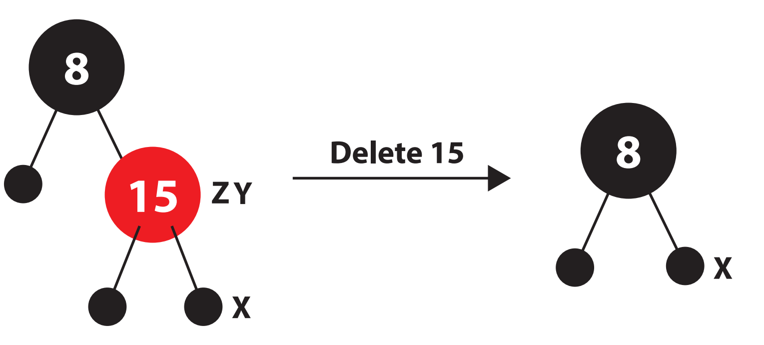
Our Z node has 2 real children therefore Y becomes Z's successor and X becomes the right child of Y. Next we delete our Z node like normal and notice that X has a sibling node whose child closest to X is red. This means Rule 3 applies and we perform a left roation and then reassign W to X's old siblings child. Next we notice that X has a black sibling whoes red child node is furthest away from X which means that Rule 4 applies we then perform a right rotation and swap the colors of our new subtree root and its right child. Then we set the left child of the subtree root (in this case the global root) to black. Then X is reassigned to the global root node and the while loop ends. No further damage.

Step 6:
Insert {14, 1, 5, 2, 11, 7, 15, 8}



Our Z node has 2 nill children so Y = Z and X becomes Z's left child node (I have been saying Z's child this whole time but really its Y's child which in alot of cases just happens to be Z as well). Next we delete Z as normal and realize that X's sibling is black with two black children nodes. This means Rule 2 applies and we change the color of W and then reassign X = P(X) which happens to be the global root node the while loop ends. No further damage.

Step 7:
Insert {14, 1, 5, 2, 11, 7, 15, 8}



Z has two nill children so Y = Z and X becomes Y's right child node. Next we delete Z like normal. Since Y was red prior to deletion we are done and there is no damage.

Step 8:
Insert {14, 1, 5, 2, 11, 7, 15, 8}

Z is the last node in the tree so all we have to do is delete it like normal. Now the tree is empty.