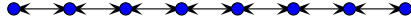


AVL Tree Deletion



Start out by swapping the value to be deleted to the appropriate leaf. Call this node x . Pass a pointer to x to *deleteFixUp*. After *deleteFixUp* returns, prune x from the tree.

```
function deleteFixup(x)
{
  set the height of x to zero //since it will be deleted
  loop
  {
    if (x is the root)
      exit the loop
    else if (parent favors x)           //case 1
    {
      set the balance of parent
      x = parent
      //continue looping
    }
    else if (parent has no favorite)    //case 2
    {
      set the balance of parent
      exit the loop
    }
    else
    {
      p = parent of x
      z = the sibling of x
      y = favorite of z
      if (y exists and y,z,p are not linear) //case 3
      {
        rotate y to z
        rotate y to p
        set the balance of p
        set the balance of z
        set the balance of y
        x = y
        //continue looping
      }
      else
      {
        rotate z to p           //case 4
        set the balance of p
        set the balance of z
        if (y does not exist)
          exit the loop
        x = z
        //continue looping
      }
    }
  }
}
```

Note that in this pseudocode, there are no references to leftness and rightness. This issue is deferred to the helper functions. For example, determining the linearity of a child, parent, and grandparent could be implemented as:

```
function linear(c,p,gp)
{
  return (gp.left == p && p.left == c)
  || (gp.right == p && p.right == c);
}
```

}