* Balanced Binary search trees
* Red-black trees
  + Height = log(n)
  + Well balanced
  + Rules:
    - Every node is either red or black
    - Root is black
    - leaf (NIL) is black
    - If a node is red, then both it’s children are black
    - All paths from a node x to a leaf have same number of black nodes. ( Black-Height(x))
  + Height Proof
    - Red-black tree w/ n keys has height at most of 2lg(n+1)
    - H’ >= H/2
    - Complete binary tree w/ h’ has 2^(h’)-1
* Red-black tree insertion
  + Rotations
    - Constant time
  + Insert x into tree
  + Color x red.
  + Insertion case 2
    - X’s uncle is black
    - x is the right child of x.p
      * Make a left rotation
      * Make a right rotation on x.p
  + Insertion case 3
    - X’s uncle is black
    - x is the left child of x.p
      * Make a right rotation
  + Example 1
  + Example 2
    - adding <41, 38, 31, 12, 19, 8>