Example: Red-Black Tree Insertion Algorithm

Idea:

- Although we could convert to a 2-3 tree to perform operations, a "direct" implementation of INSERT is a simple traversal back to the root, performing a rotation or two at each step.
procedure INSERT($d,T$):
    Binary search and add a new red leaf $v$ that contains $d$.

    while $v$ has a red sibling or a red parent do begin
        if $v$ has a red sibling then begin
            Color $v$ and its sibling black.
            $v := \text{PARENT}(v)$
            Color $v$ red.
        end

        else there are 4 cases for $v$ and PARENT($v$) being red:

        Case 1L: $v$ is a right child and PARENT($v$) is a left child:

        Case 1R: symmetric to case 1L.

        Case 2L: $v$ is a left child and PARENT($v$) is a left child:
            $v := \text{PARENT}(v)$
            Do the second half of Case 1L (RR($v$) and color LCHILD($v$) black).

        Case 2R: symmetric to case 2L.

        end

    if the root is red then color it black

end
Example: Inserting into a Red-Black Tree in Sorted Order

- Initial tree: Insert(1)
- Insert(2), Insert(3): Case 2R: RL(2) & color 3 black, color the root black
- Insert(4), Insert(5): Case 2R: RL(4) & color 5 black
- Insert(6), Insert(7): Case 2R: RL(6) & color 7 black
- Final tree: Case 2R: RL(4) & color 6 black, color the root black