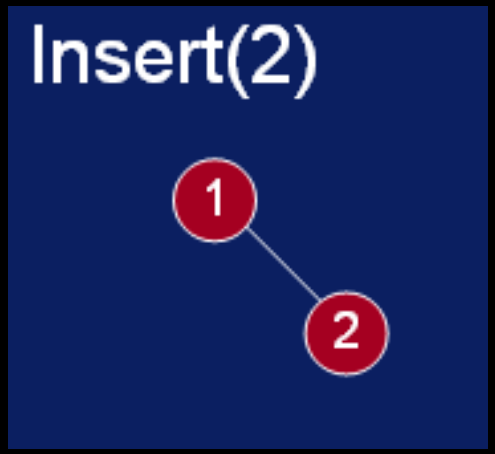
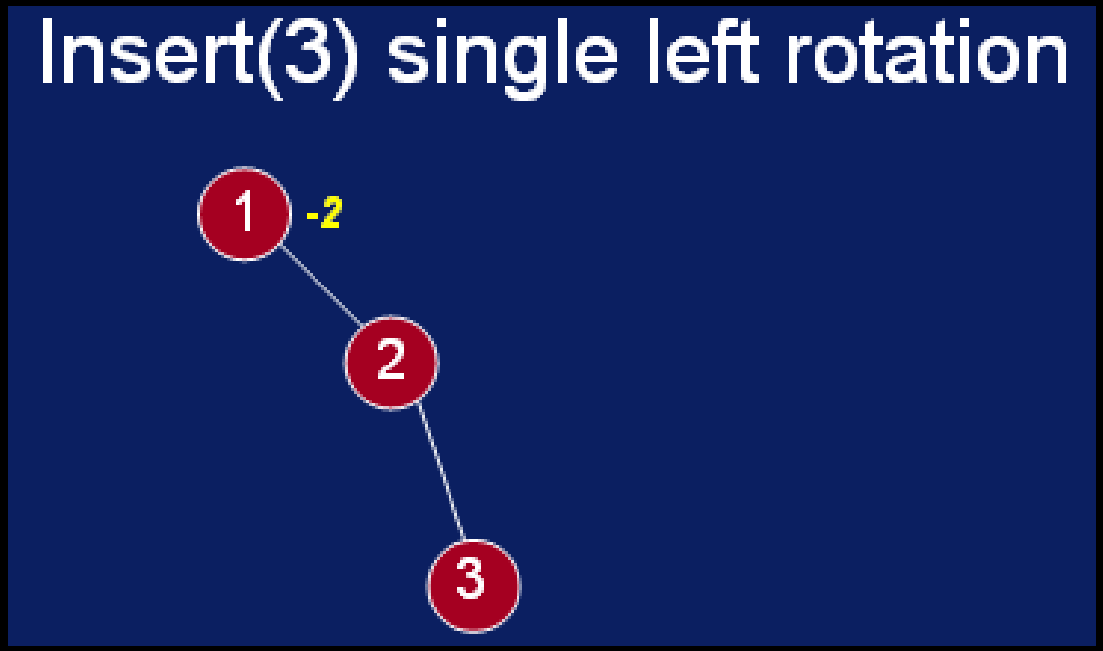
Lecture # 13

AVL Tree Building Example

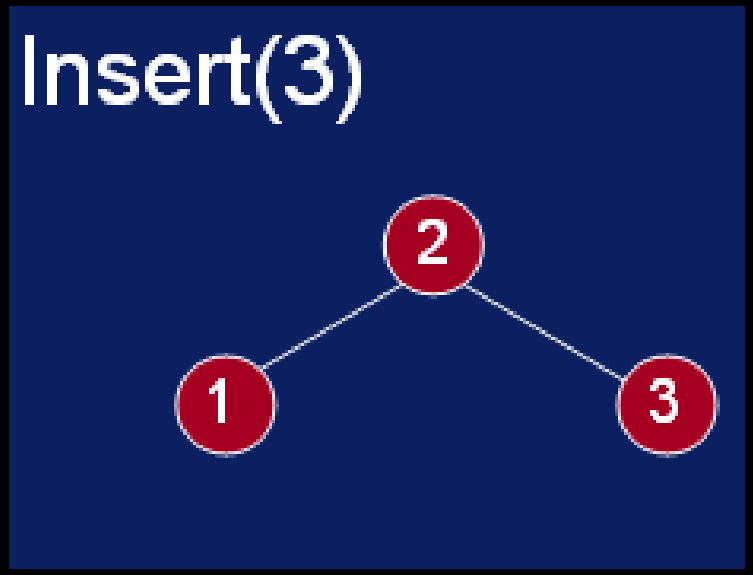
AVL Tree Building Example



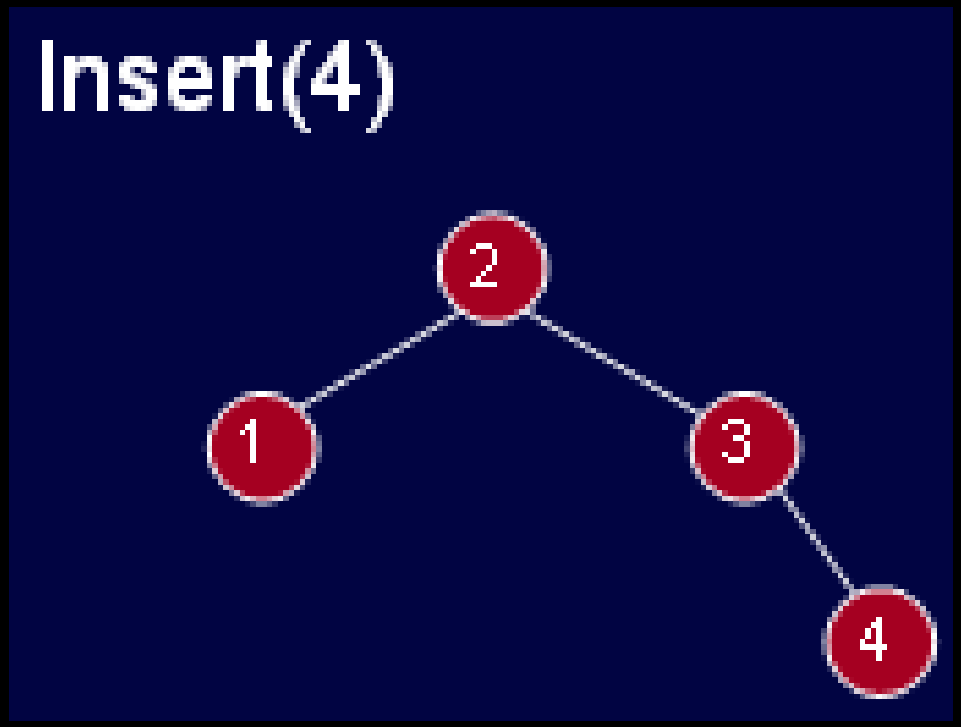
AVL Tree Building Example



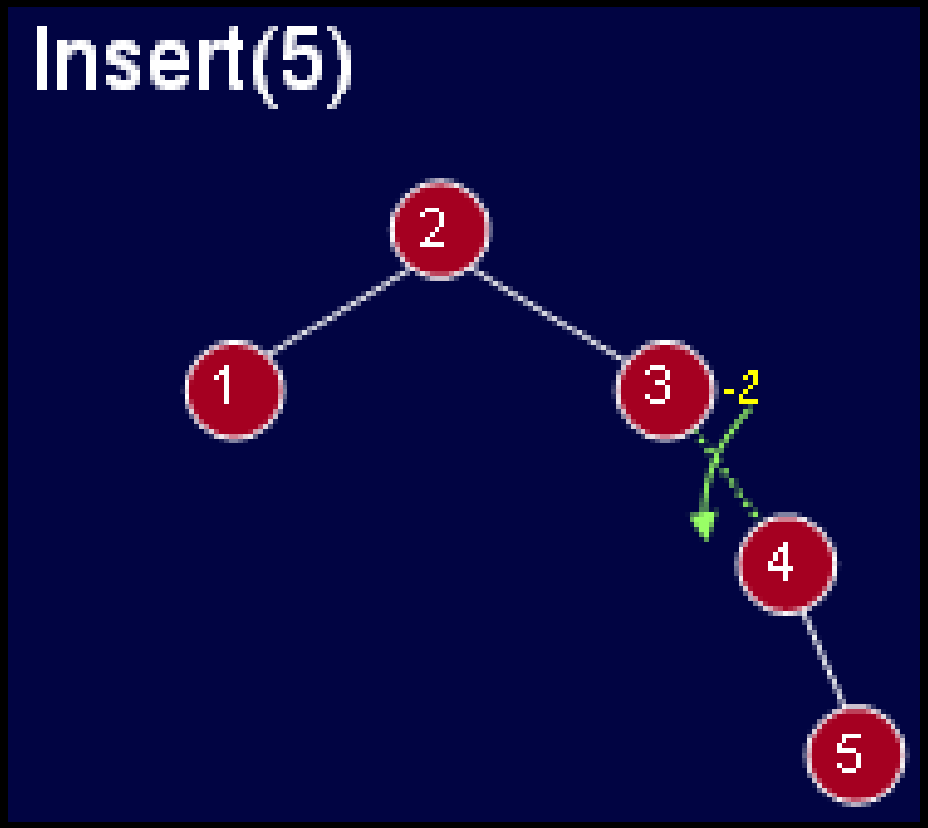
AVL Tree Building Example



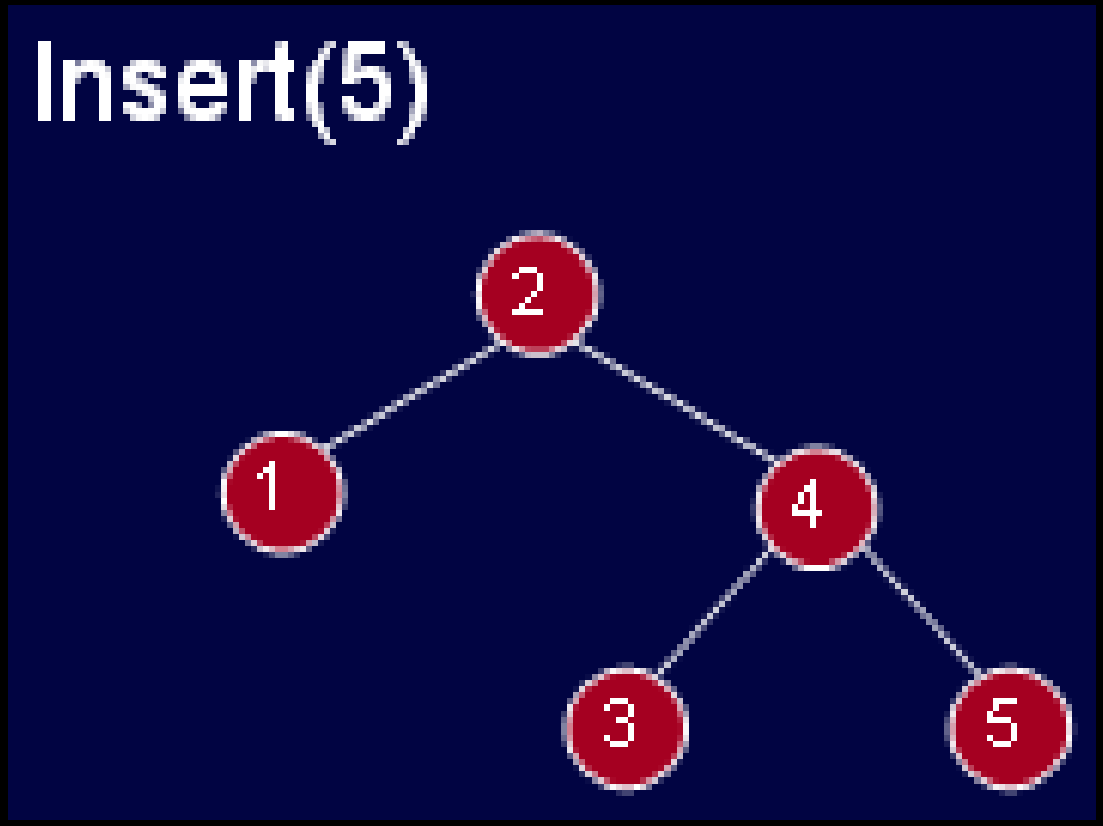
AVL Tree Building Example



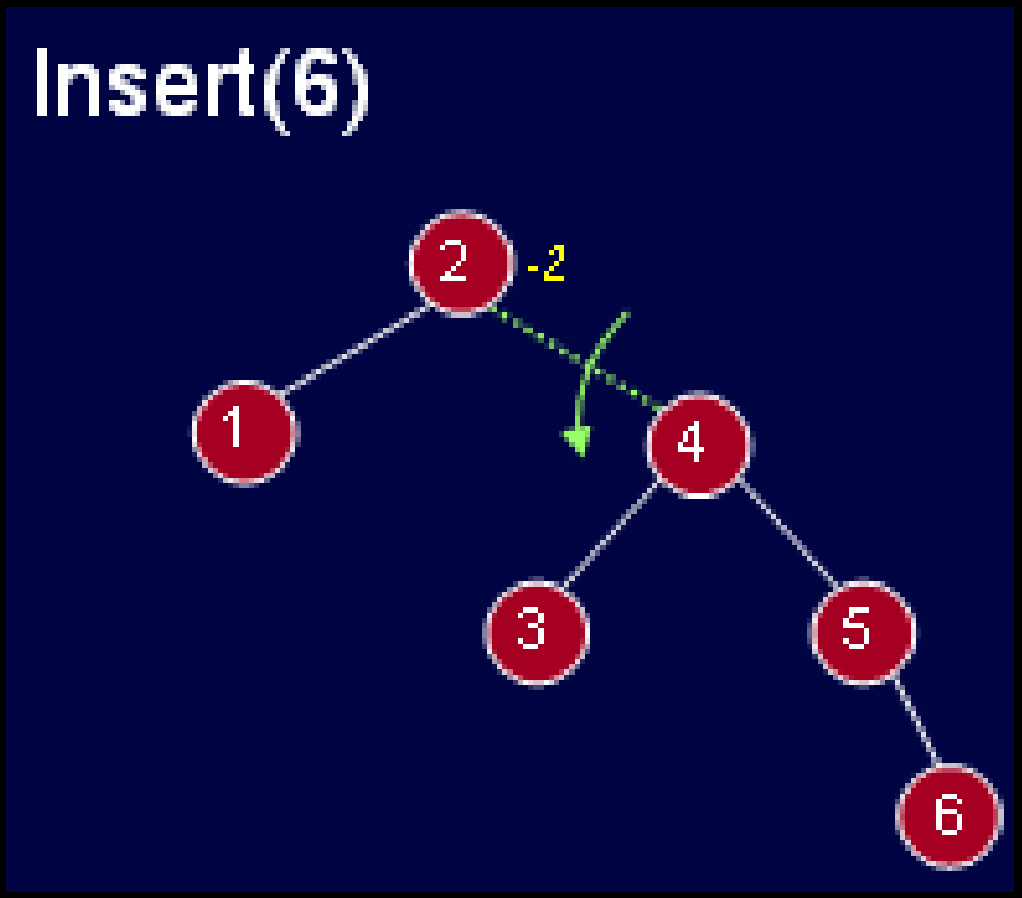
AVL Tree Building Example



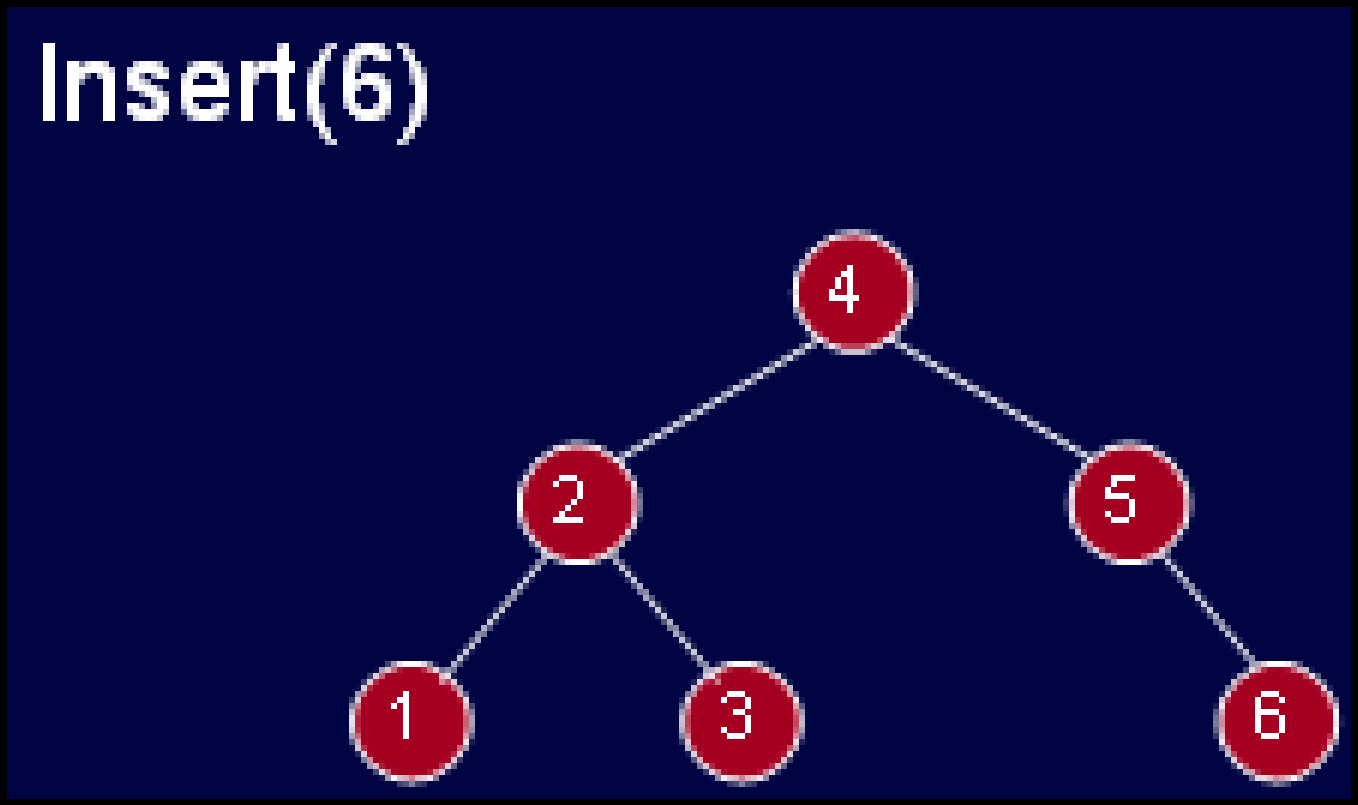
AVL Tree Building Example



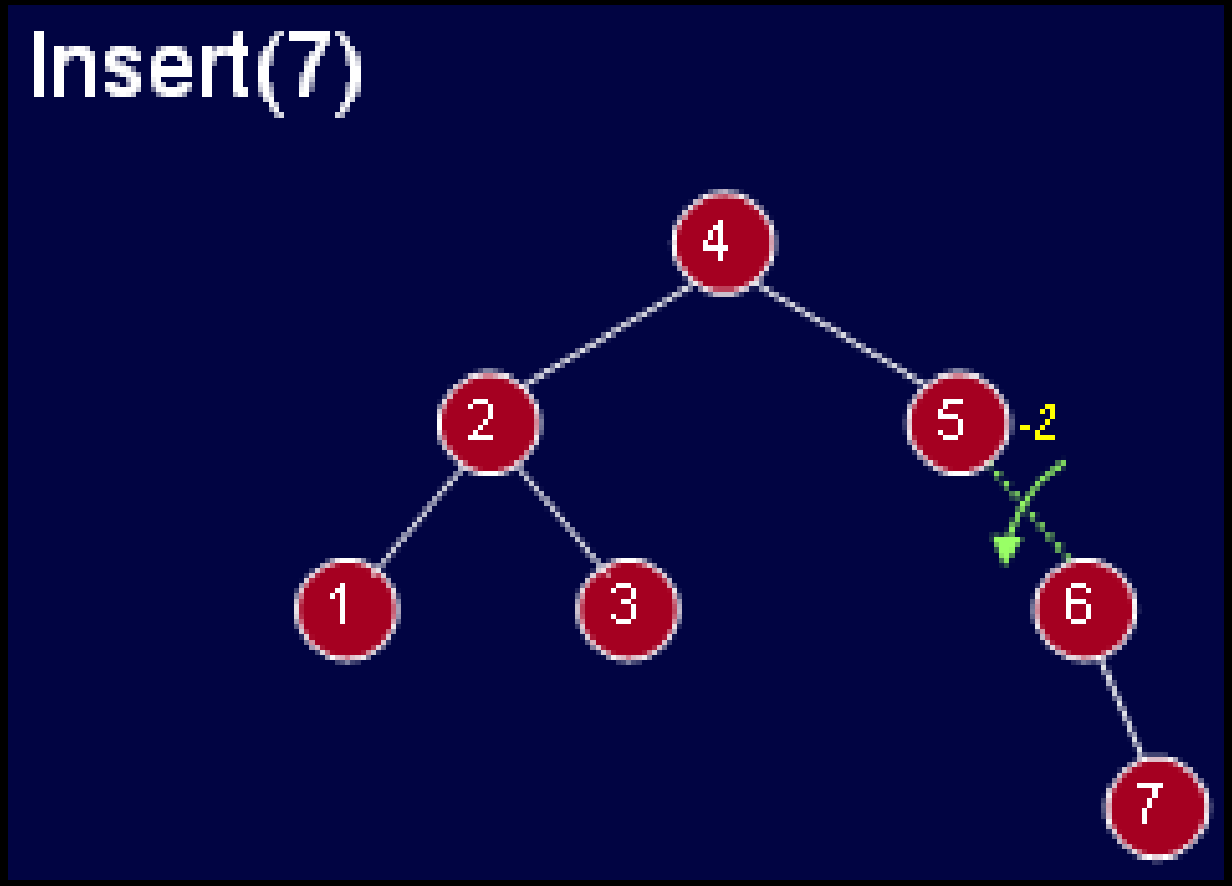
AVL Tree Building Example



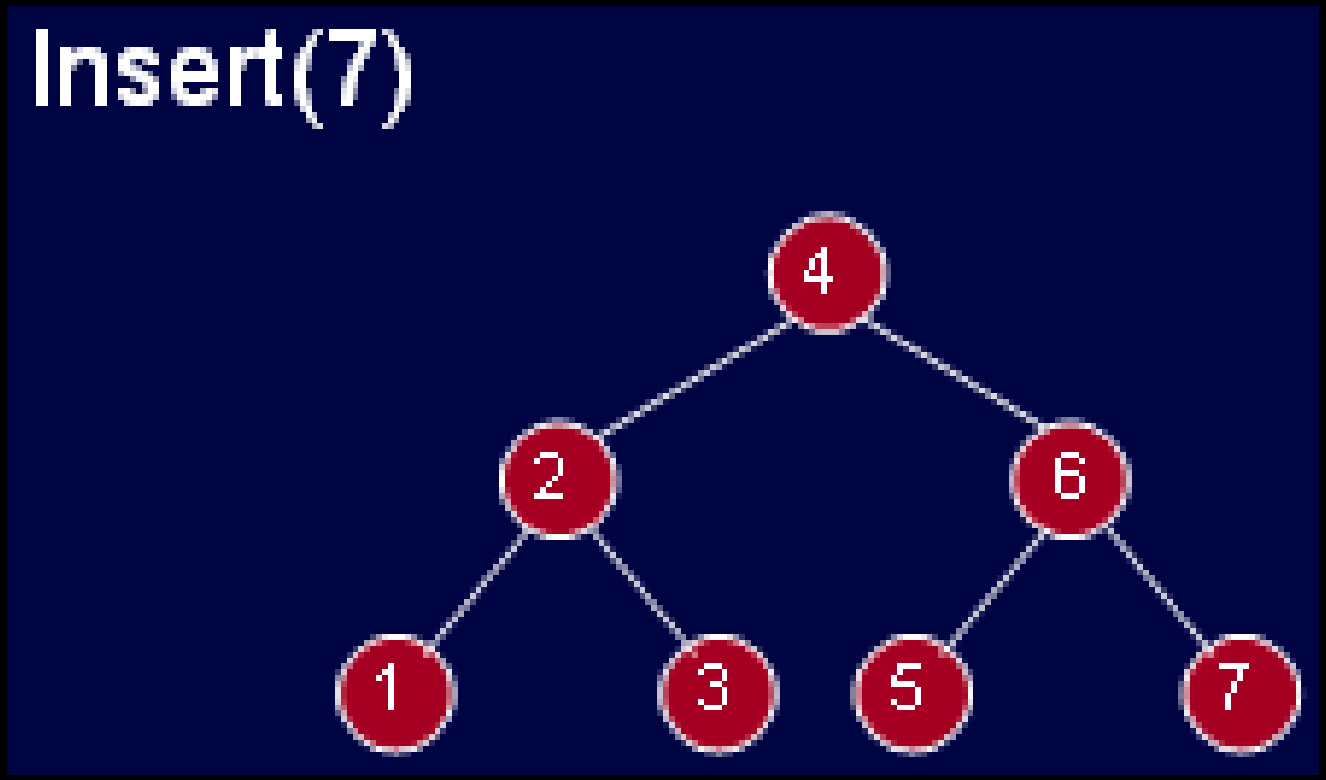
AVL Tree Building Example



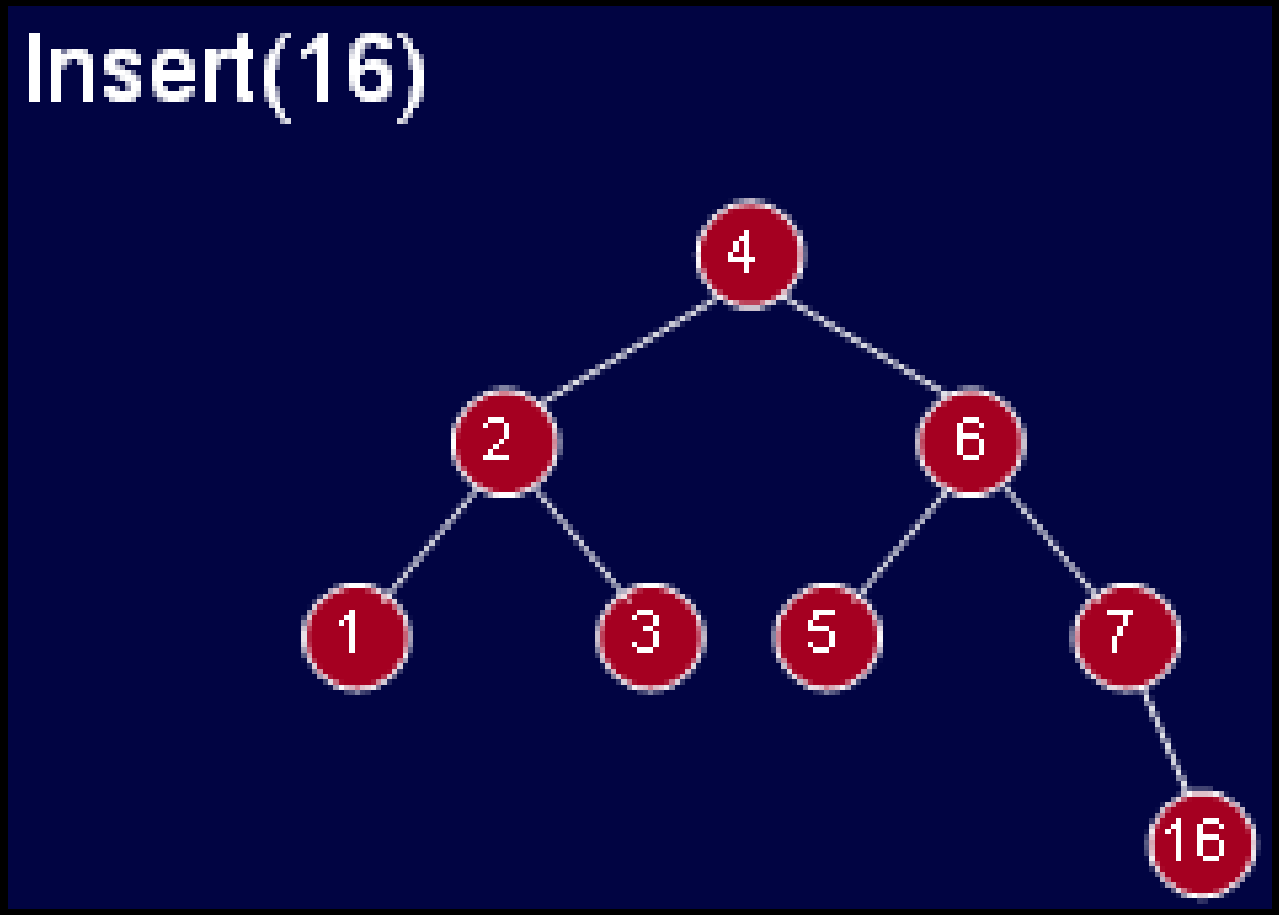
AVL Tree Building Example



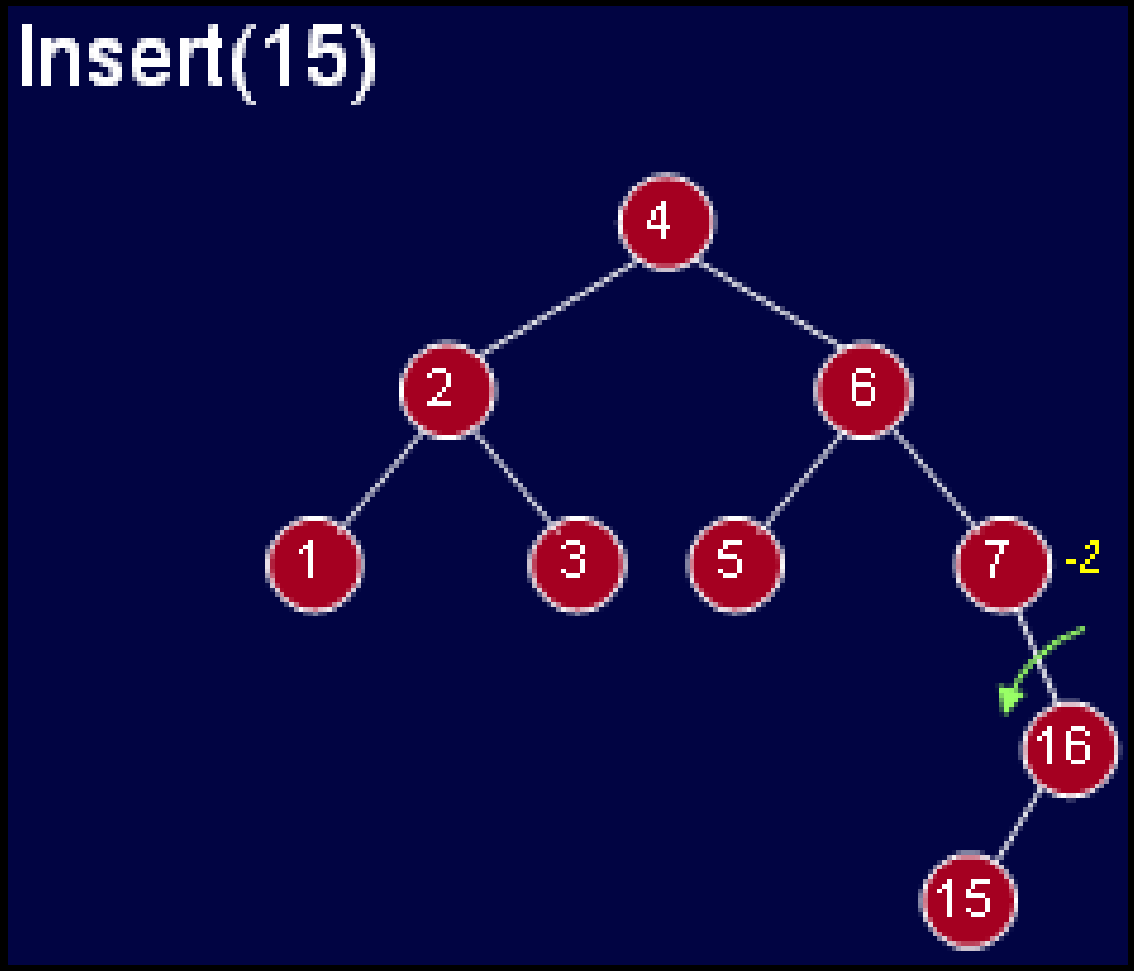
AVL Tree Building Example



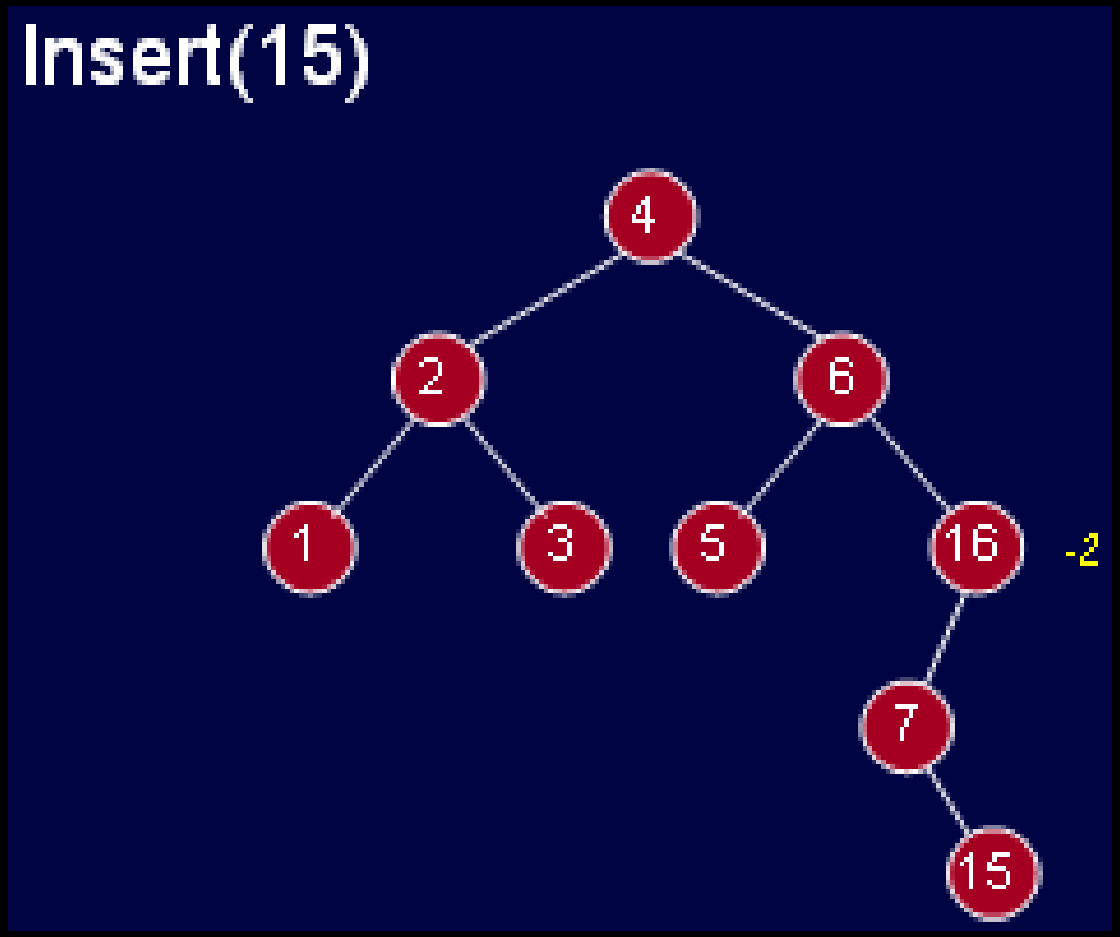
AVL Tree Building Example



AVL Tree Building Example



AVL Tree Building Example



Cases for Rotation

- Single rotation does not seem to restore the balance.
- The problem is the node 15 is in an inner subtree that is too deep.
- Let us revisit the rotations.

Cases for Rotation

- Let us call the node that must be rebalanced Ň.
- Since any node has at most two children, and a height imbalance requires that Ň's two subtrees differ by two (or –2), the violation will occur in four cases:

Cases for Rotation

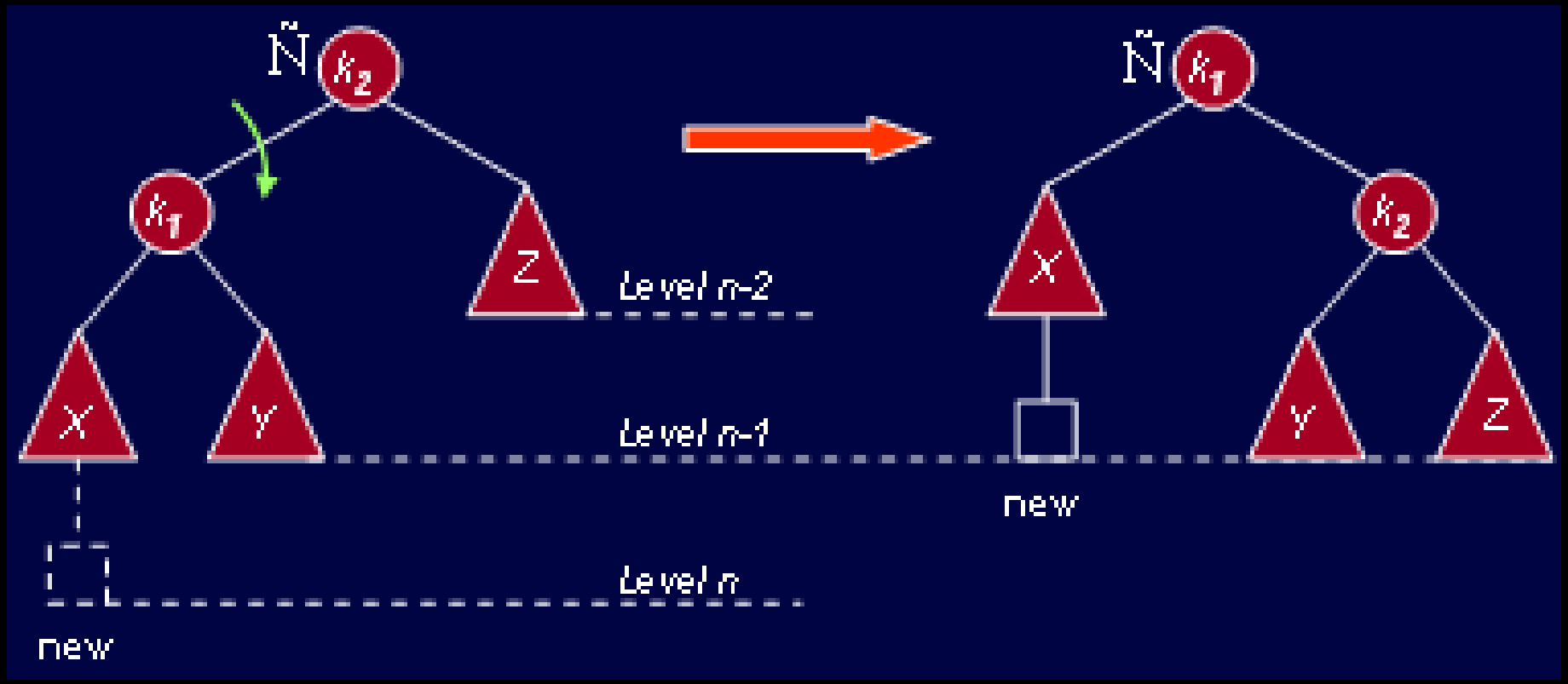
- An insertion into left subtree of the left child of Ň.
- An insertion into right subtree of the left child of Ň.
- An insertion into left subtree of the right child of Ň.
- An insertion into right subtree of the right child of Ň.

Cases for Rotation

- The insertion occurs on the "outside" (i.e., left-left or right-right) in cases 1 and 4
- Single rotation can fix the balance in cases 1 and 4.
- Insertion occurs on the "inside" in cases 2 and 3 which single rotation cannot fix.

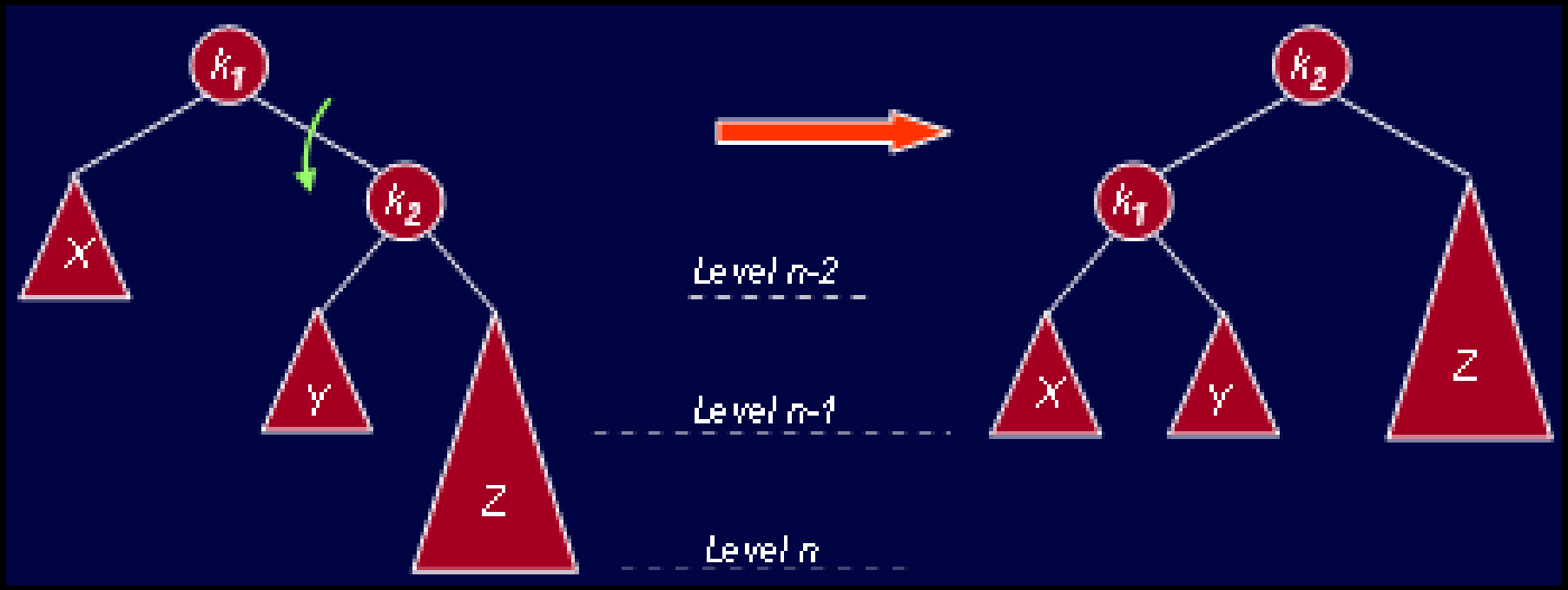
Cases for Rotation

- Single right rotation to fix case 1.



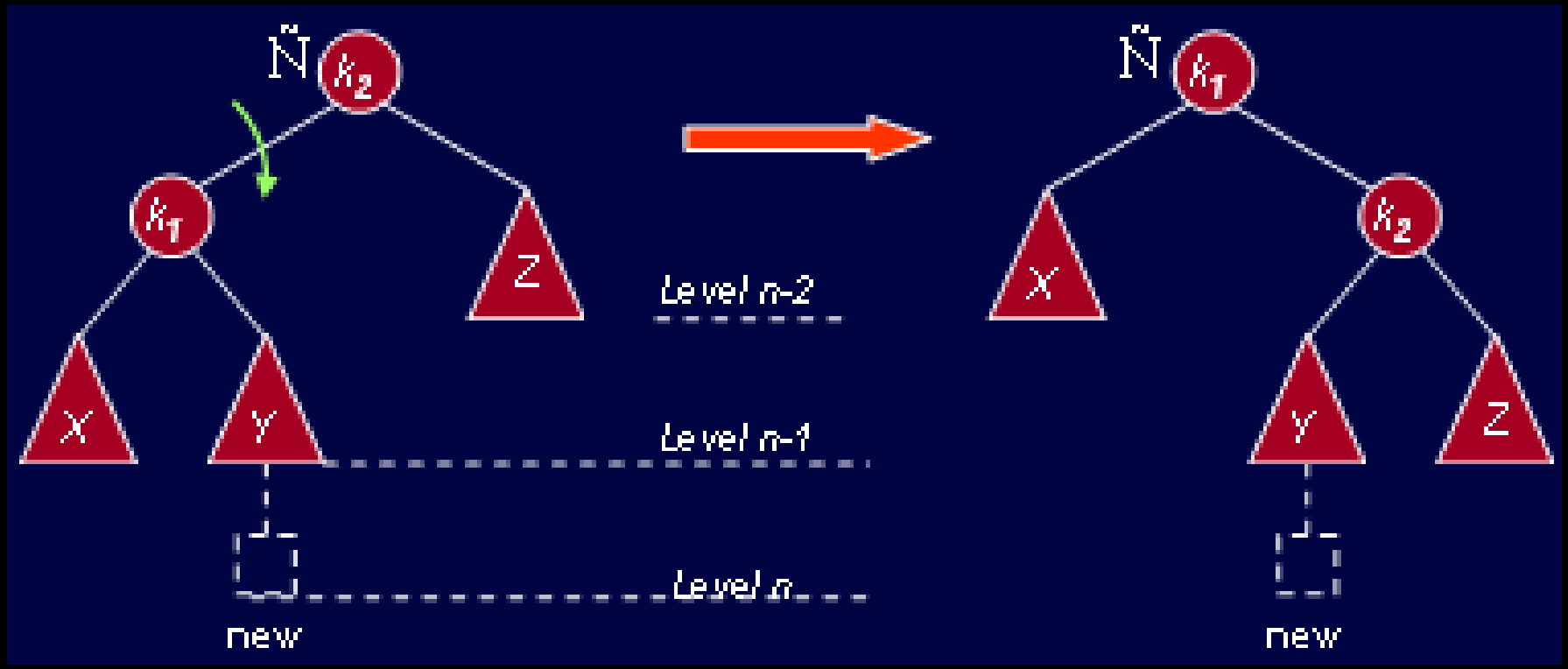
Cases for Rotation

- Single left rotation to fix case 4.

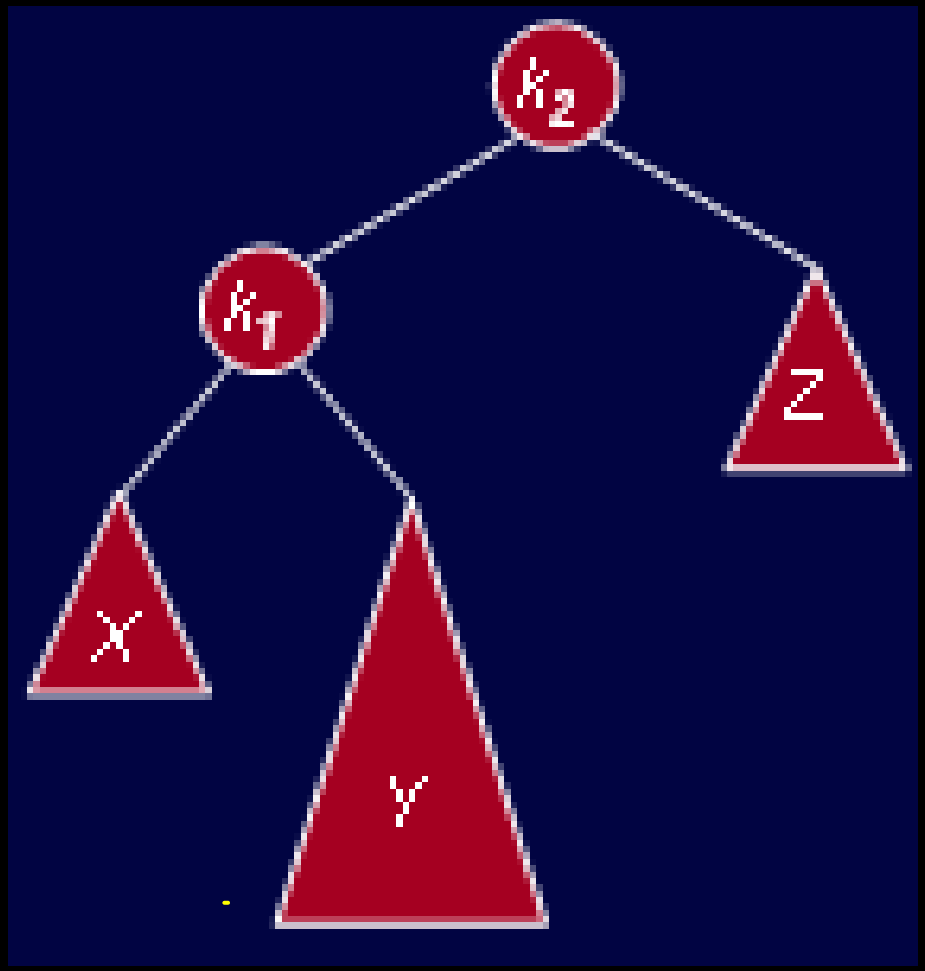


Cases for Rotation

- Single right rotation *fails* to fix case 2.

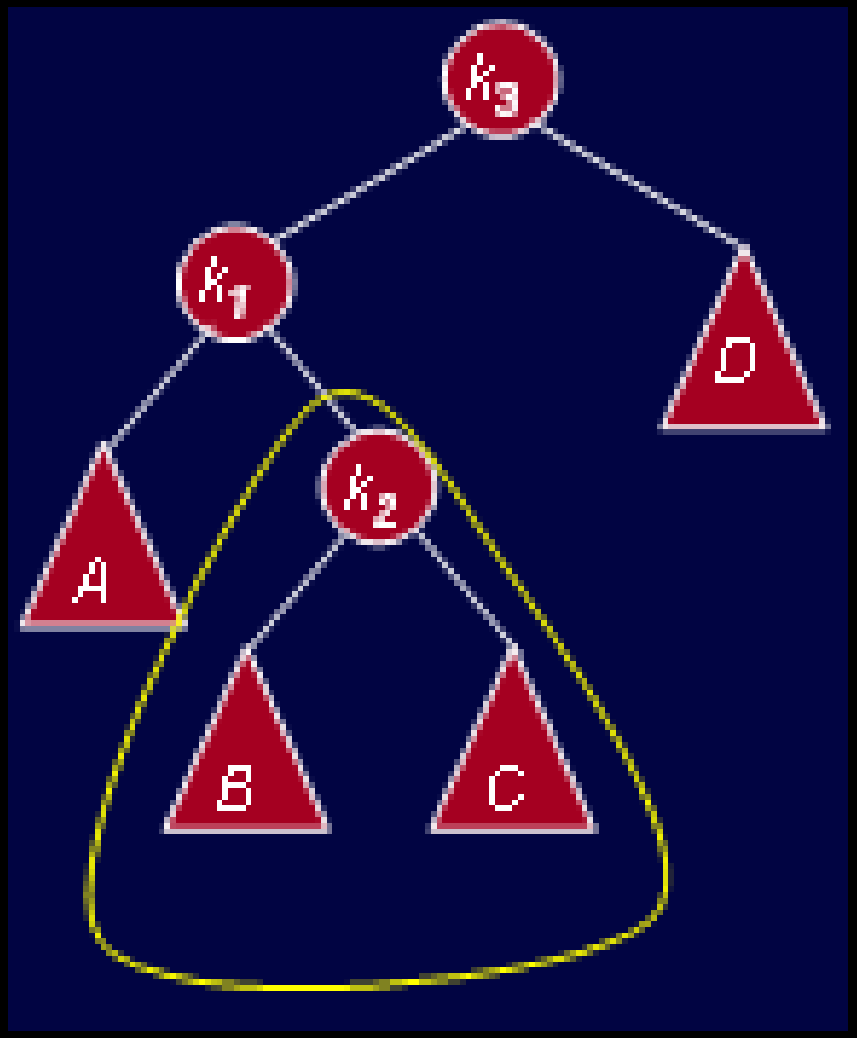


Cases for Rotation



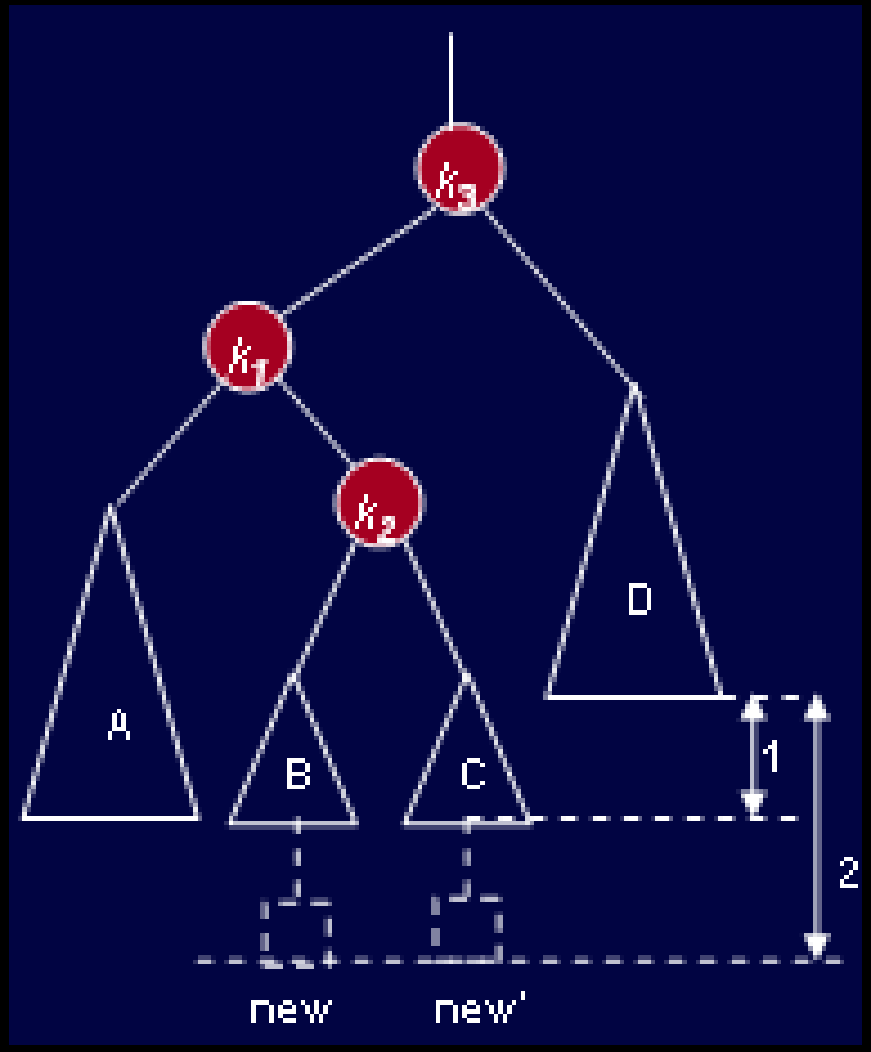
- Y is non-empty because the new node was inserted in Y.
- Thus Y has a root and two subtrees.
- View the entire tree with four subtrees connected with 3 nodes.

Cases for Rotation



- Y is non-empty because the new node was inserted in Y.
- Thus Y has a root and two subtrees.
- View the entire tree with four subtrees connected with 3 nodes.

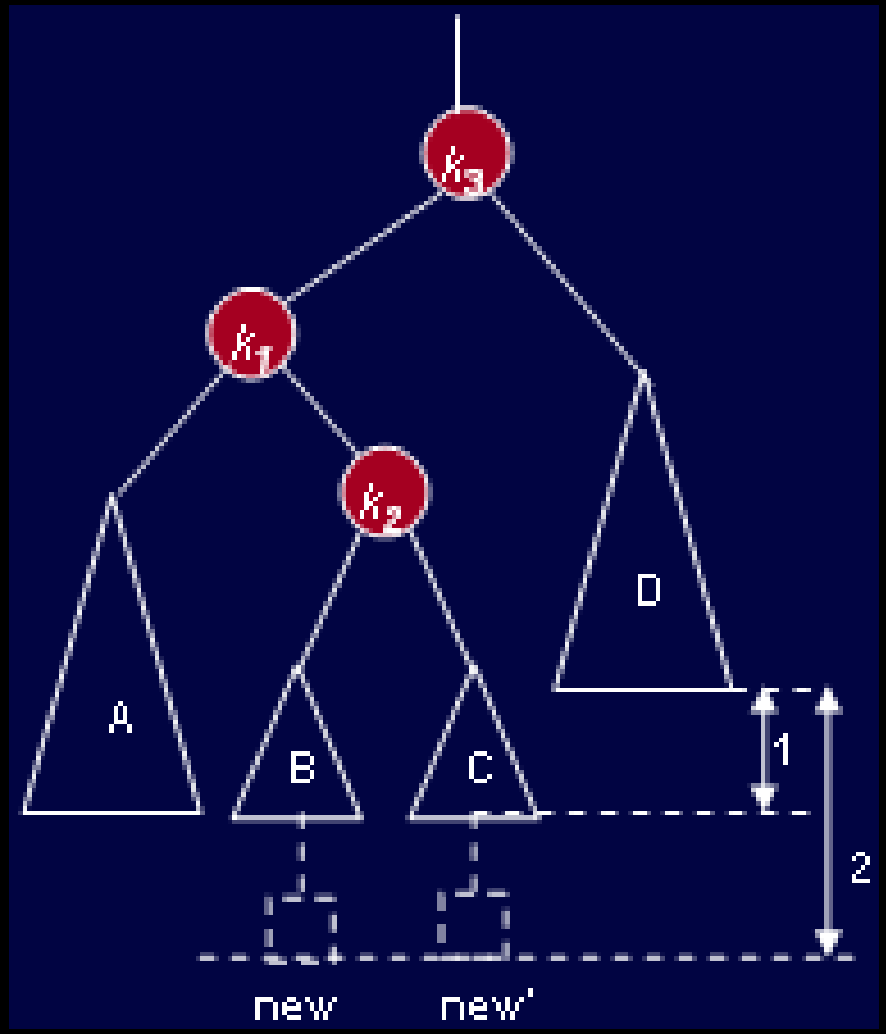
Cases for Rotation



- Exactly one of tree *B* or *C* is two levels deeper than *D*; we are not sure which one.
- Good thing: it does not matter where it is added.
- To rebalance, *k3* cannot be left as the root.

New node inserted at either of the two spots

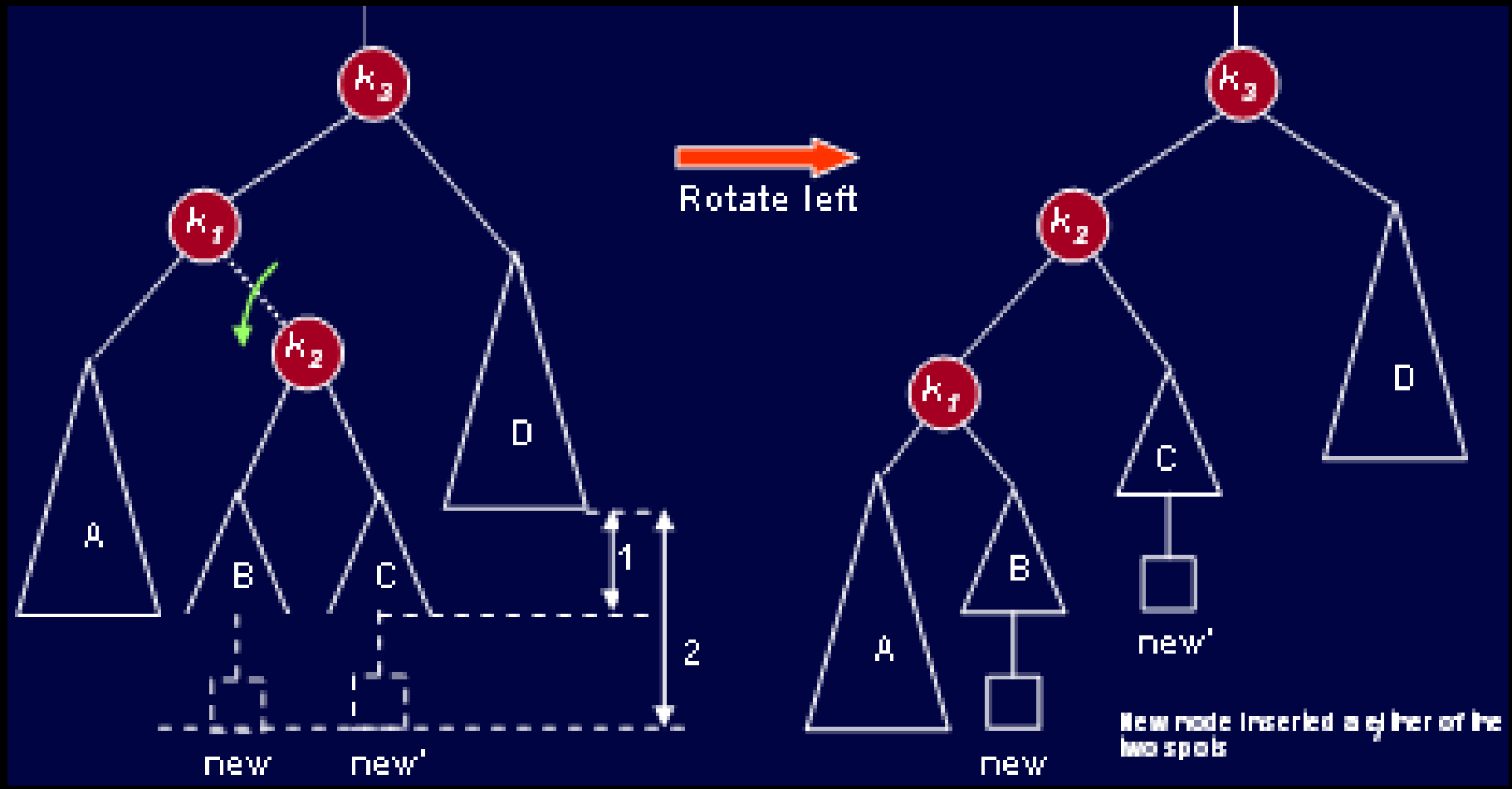
Cases for Rotation



- A rotation between *k3* and *k1* (*k3* was *k2* then) was shown to not work.
- The only alternative is to place *k2* as the new root.
- This forces *k1* to be *k2*'s left child and *k3* to be its right child.

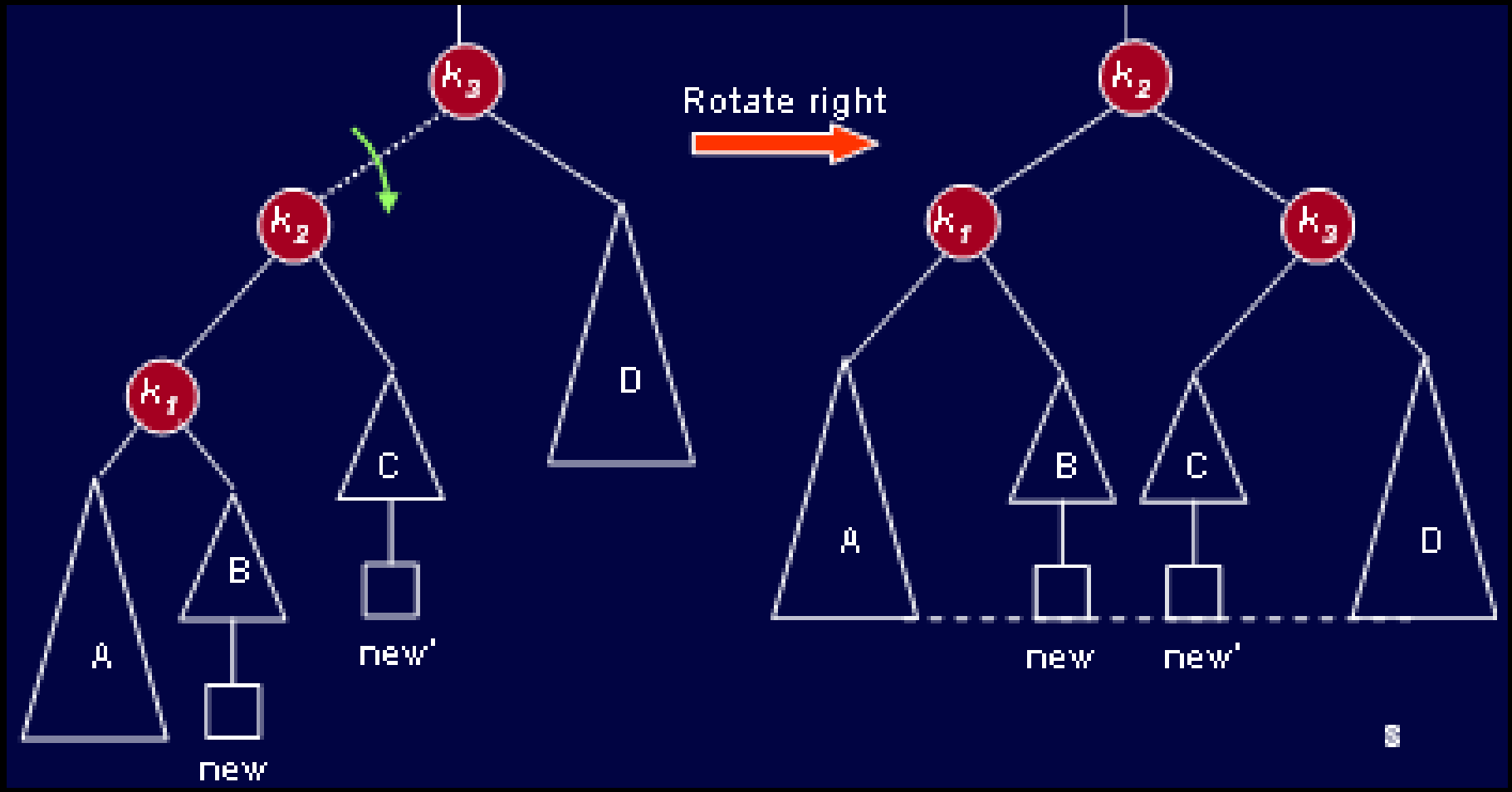
Cases for Rotation

- Left-right *double* rotation to fix case 2.



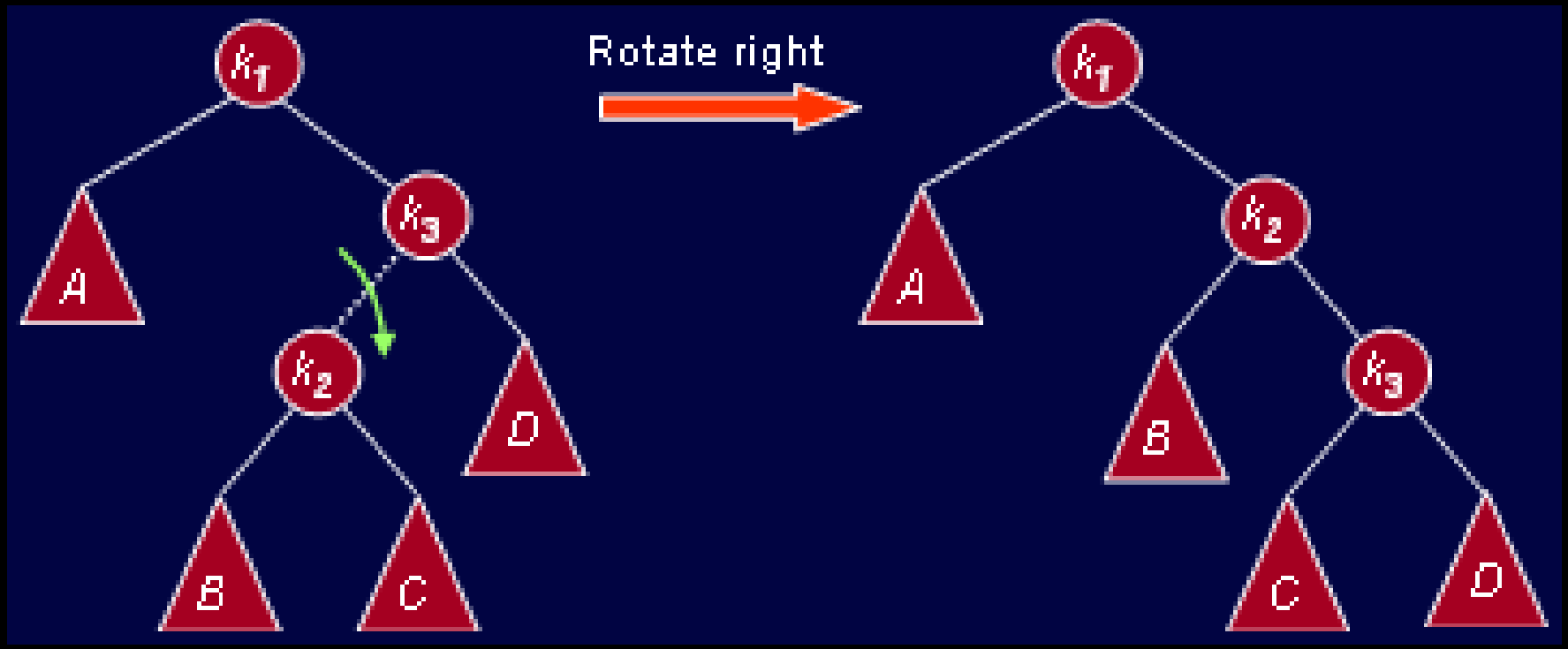
Cases for Rotation

- Left-right *double* rotation to fix case 2.



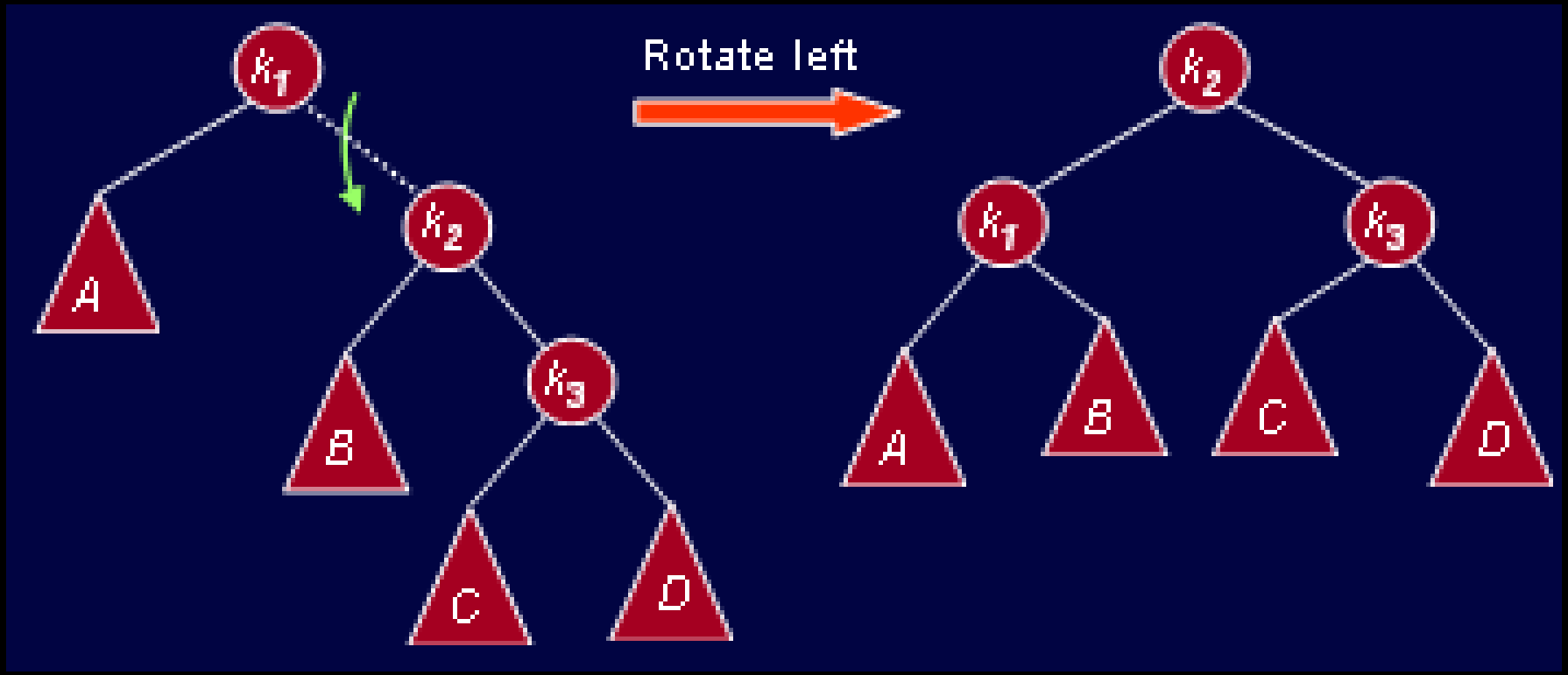
Cases for Rotation

- Right-left *double* rotation to fix case 3 .

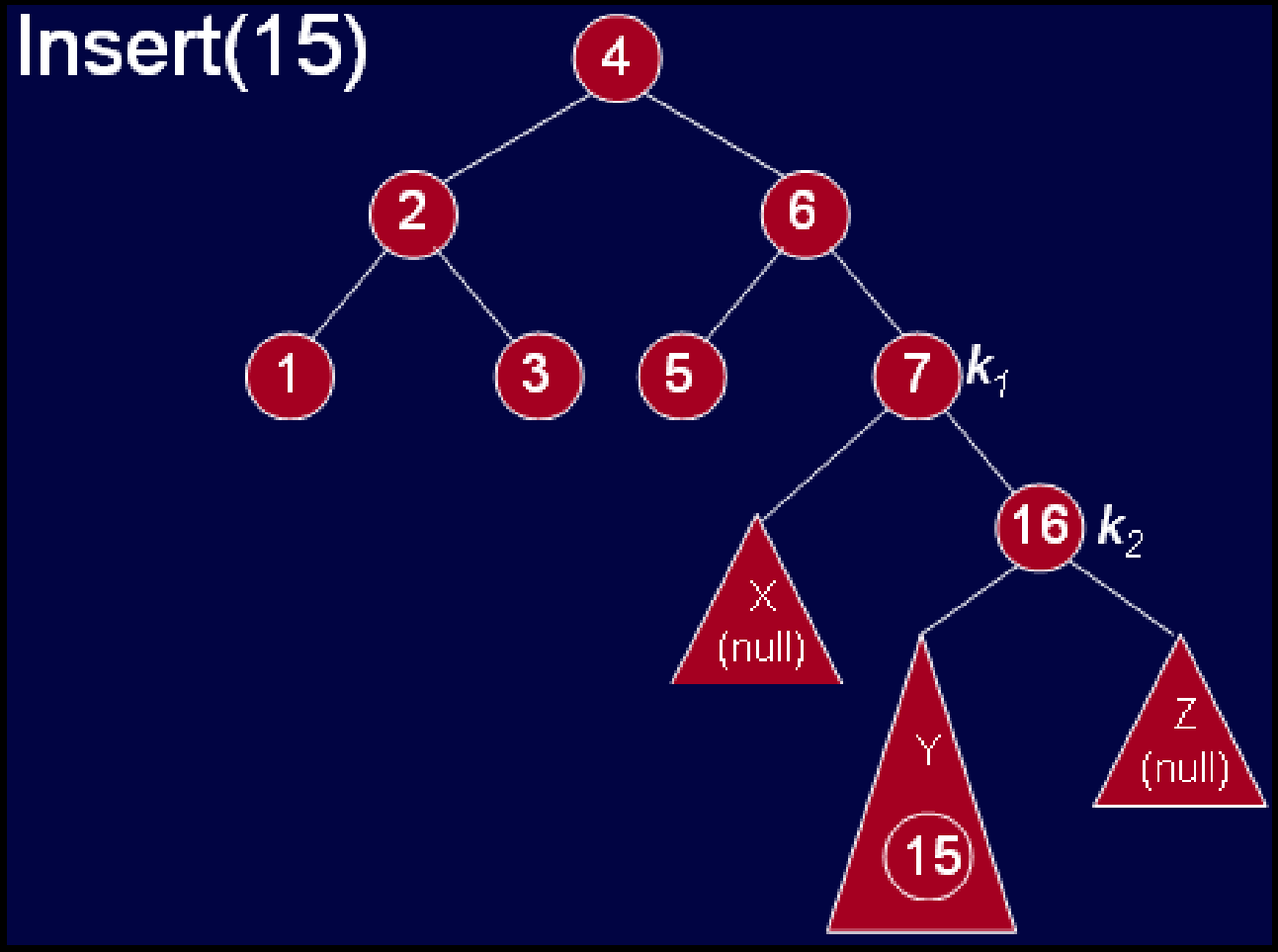


Cases for Rotation

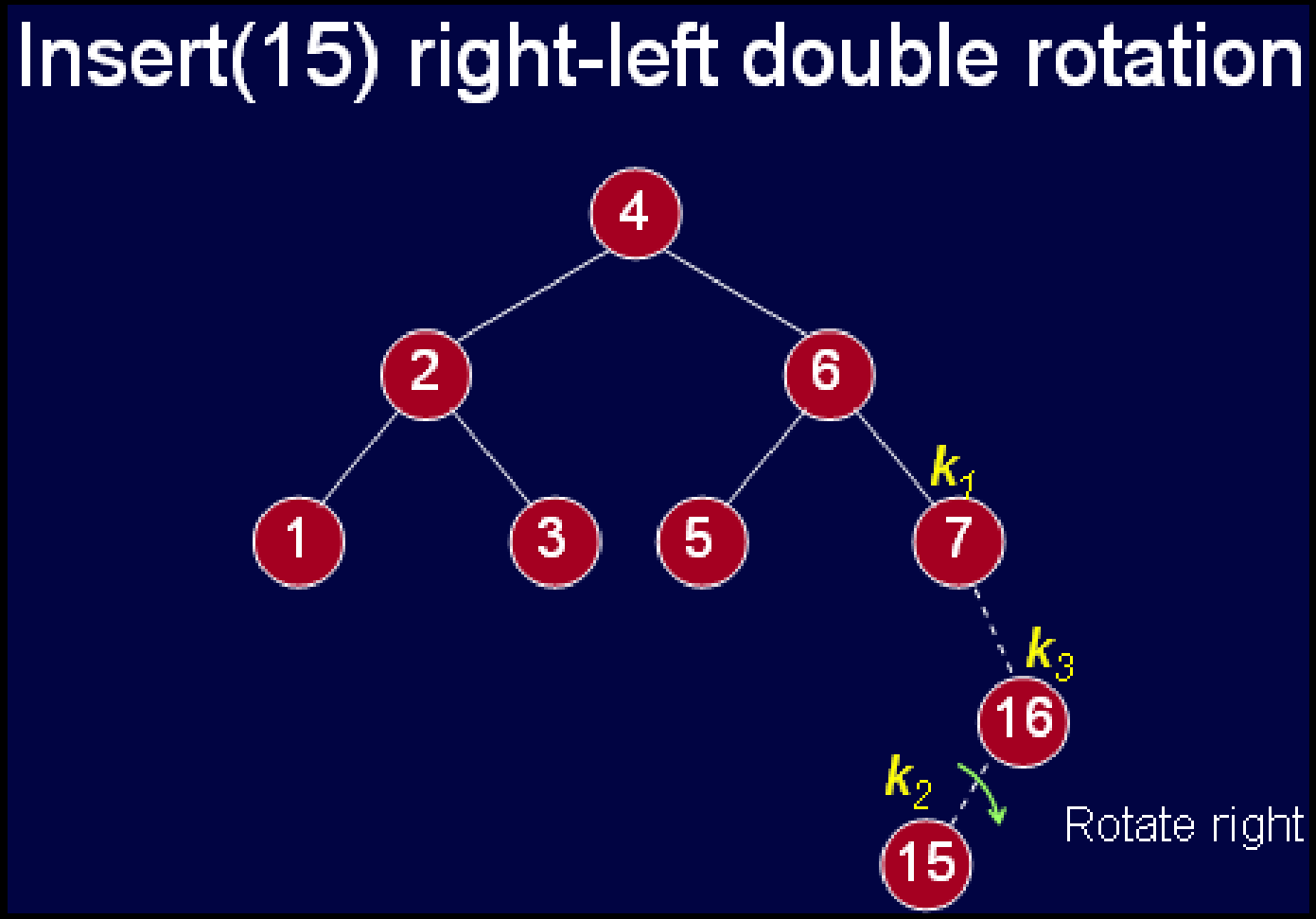
- Right-left *double* rotation to fix case 3 .



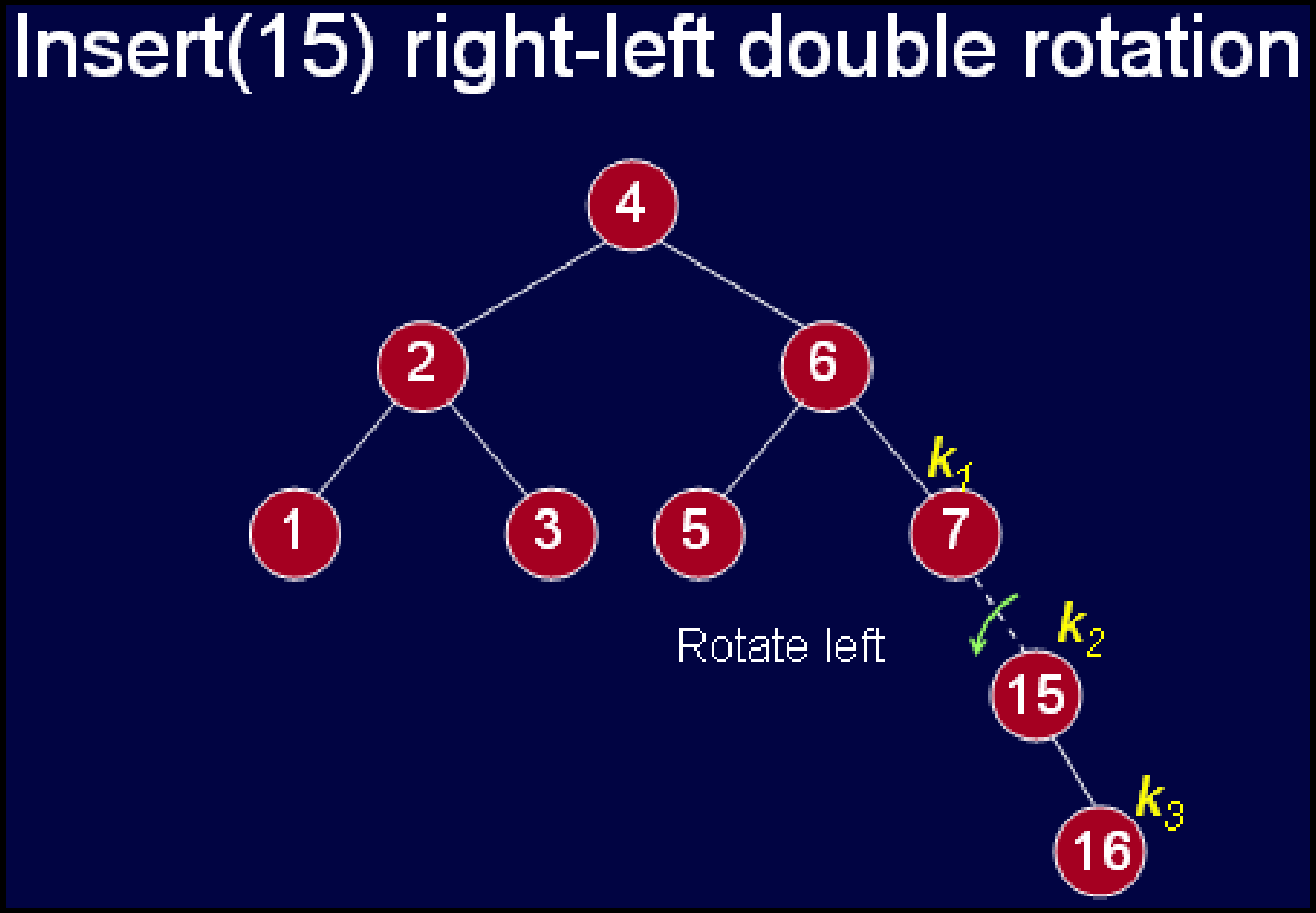
AVL Tree Building Example



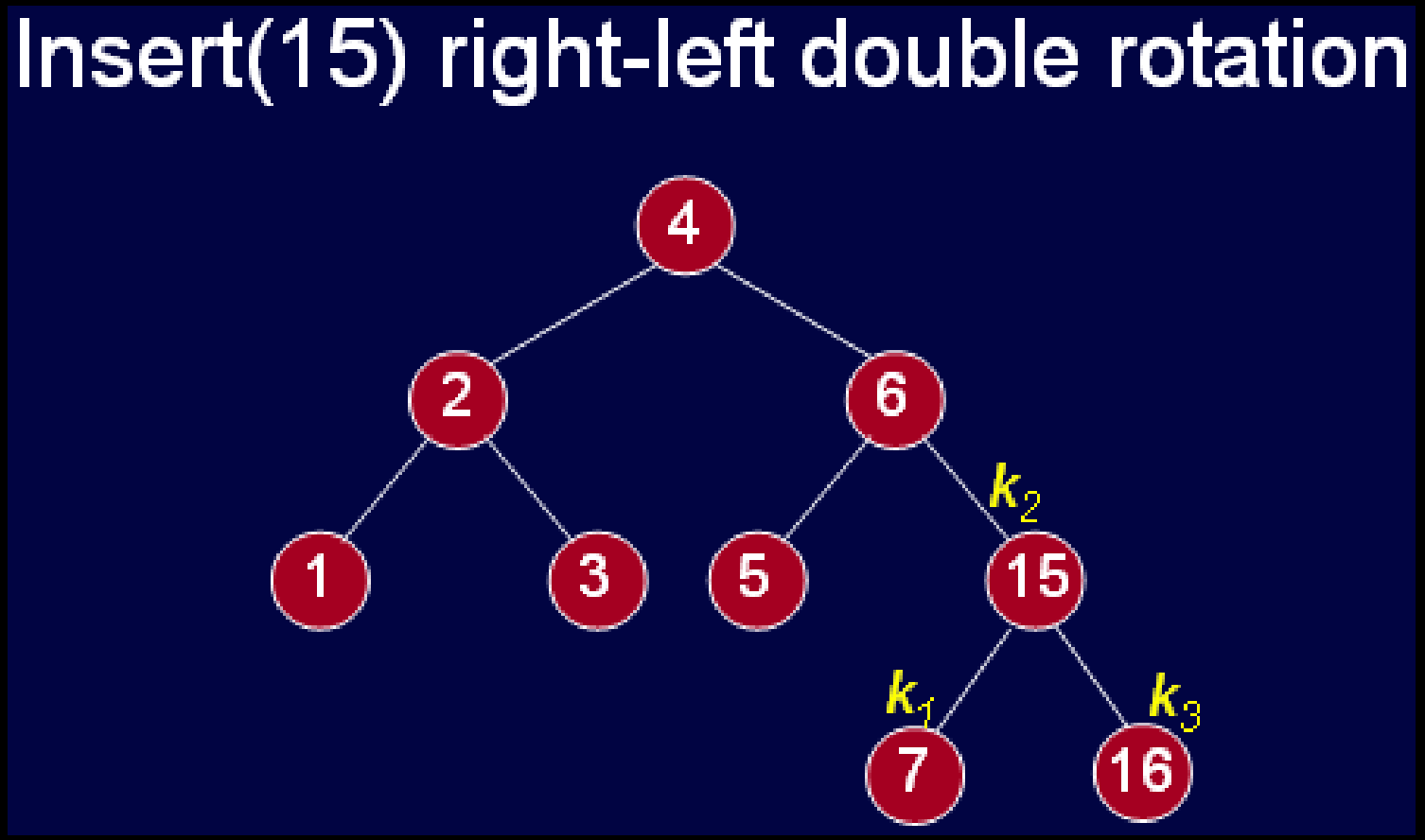
AVL Tree Building Example



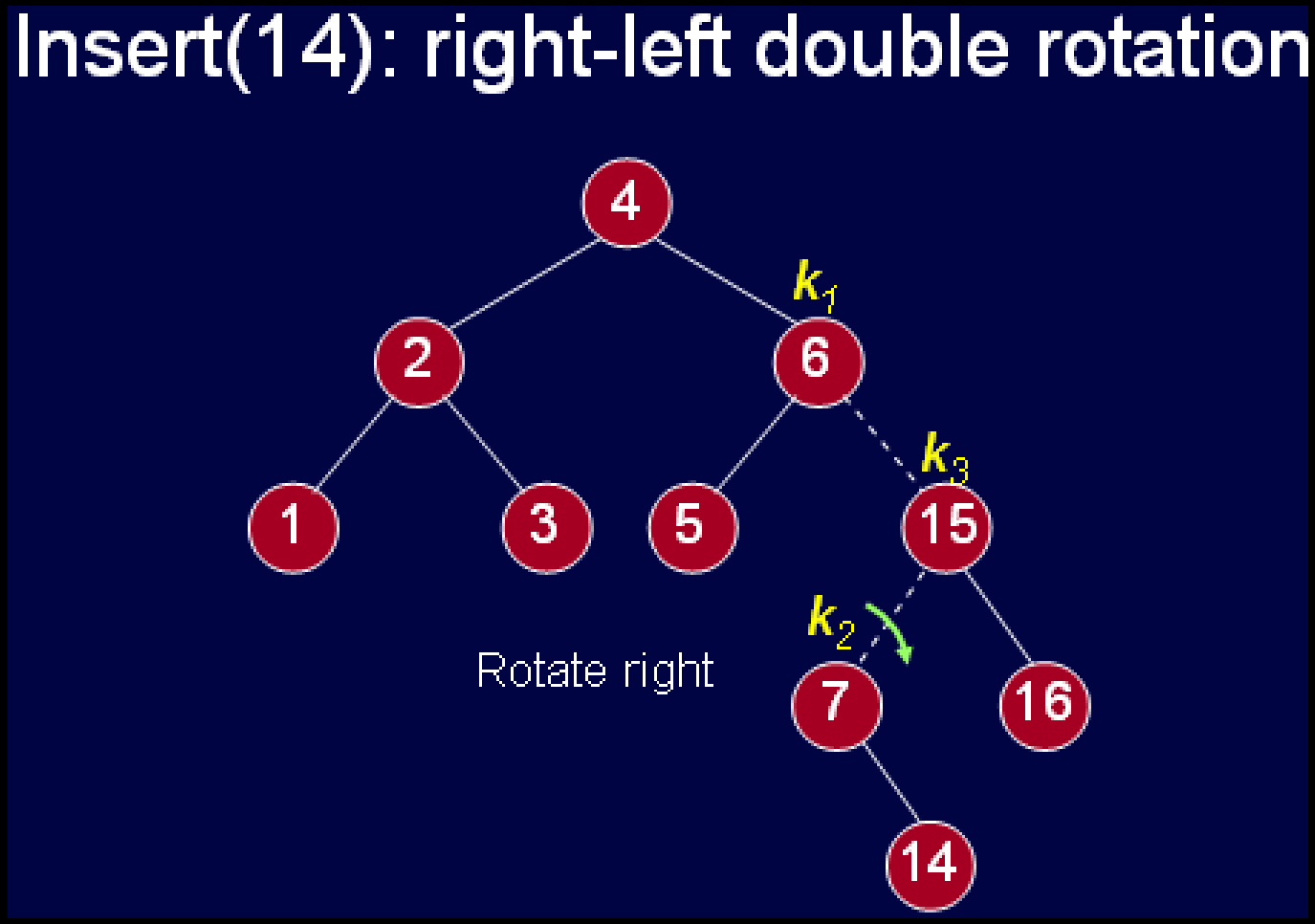
AVL Tree Building Example



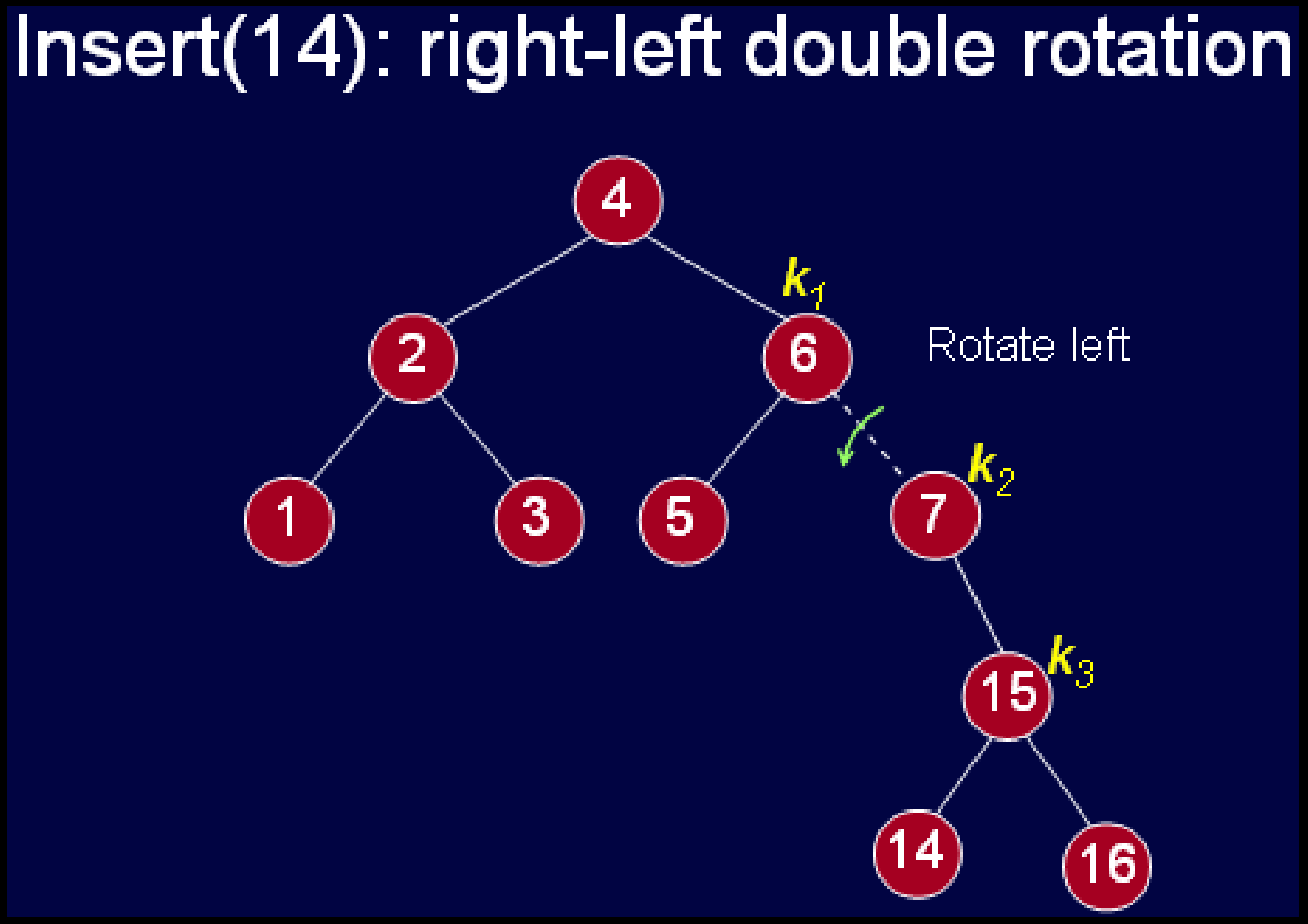
AVL Tree Building Example



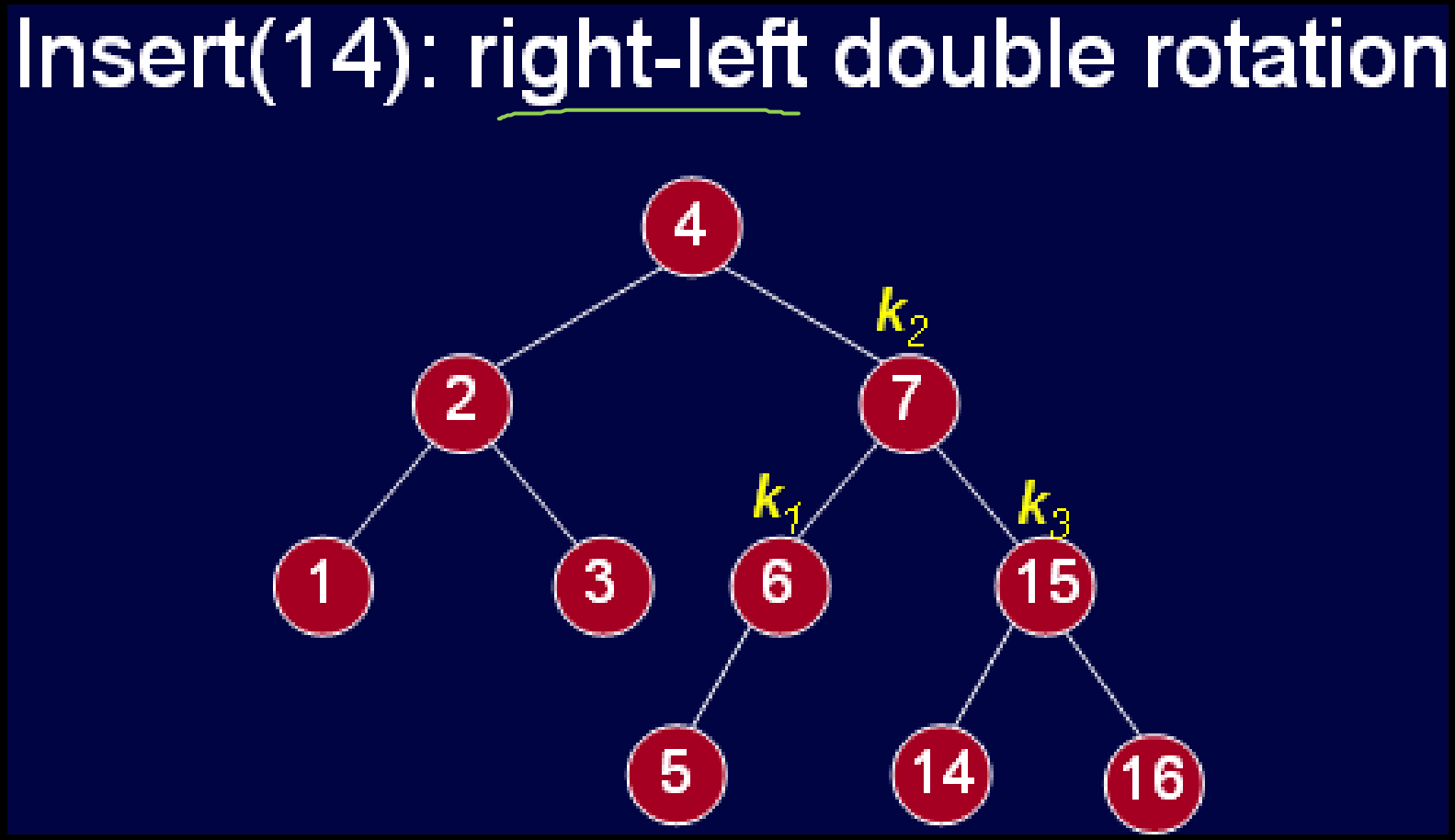
AVL Tree Building Example



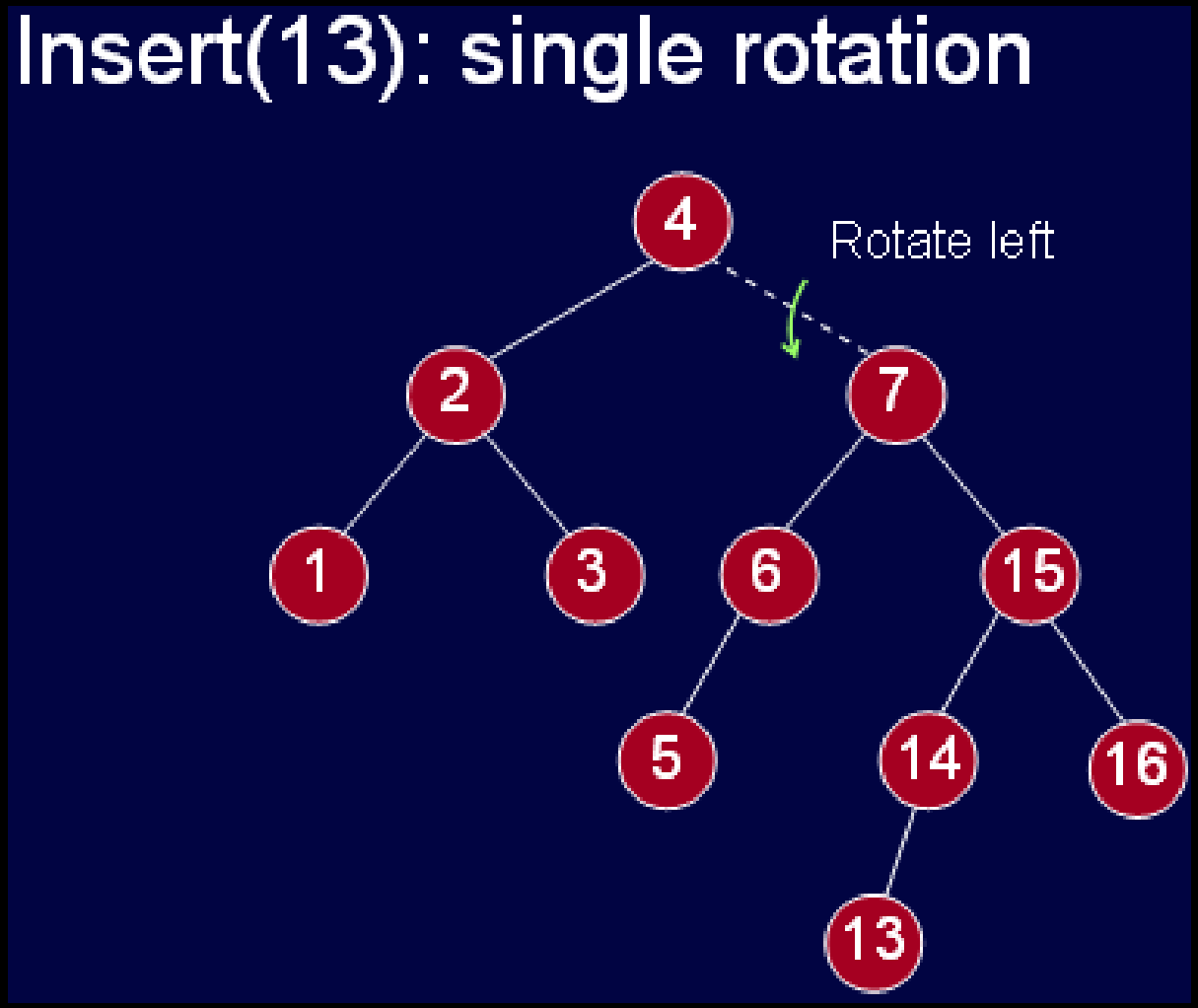
AVL Tree Building Example



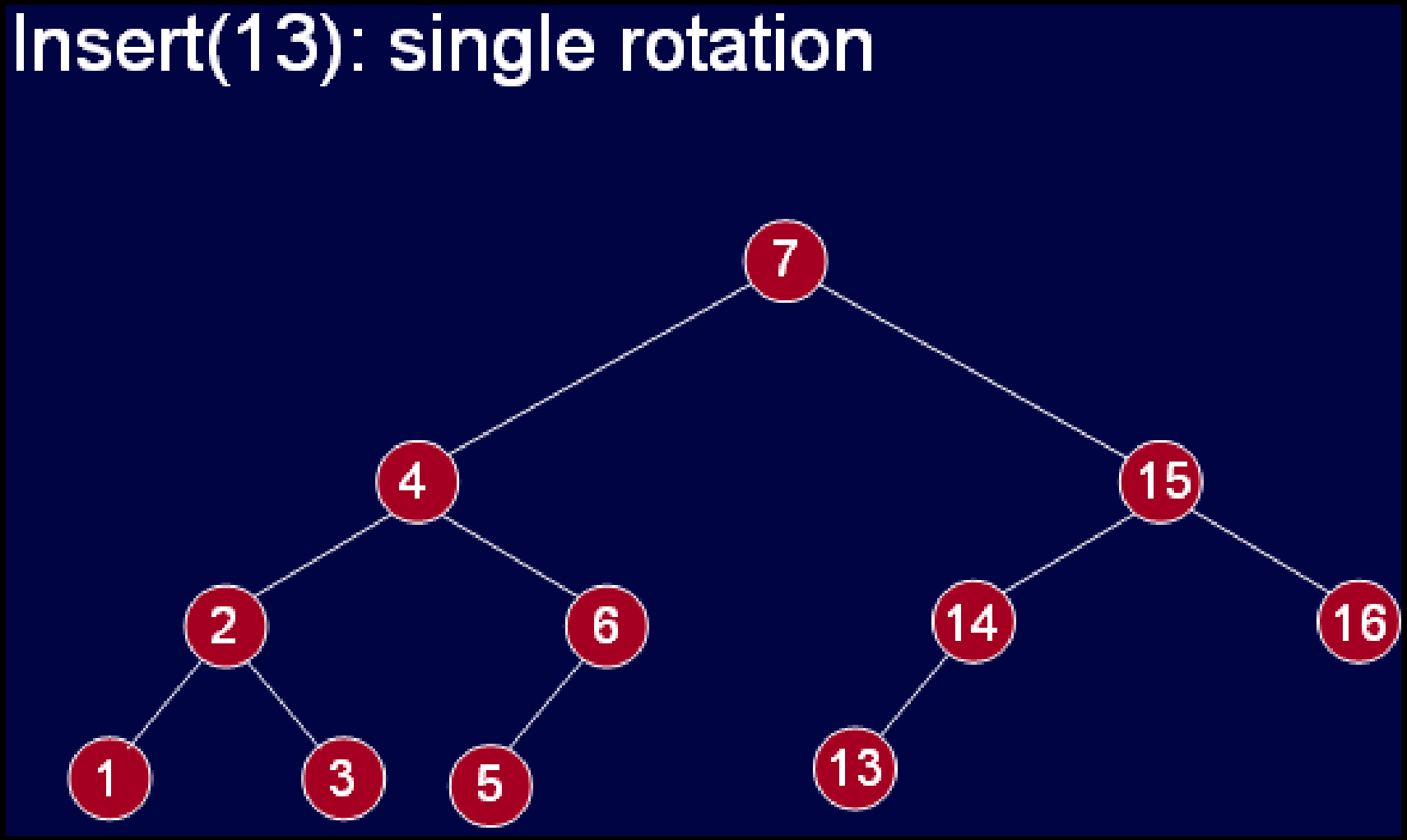
AVL Tree Building Example



AVL Tree Building Example



AVL Tree Building Example



Thanks …