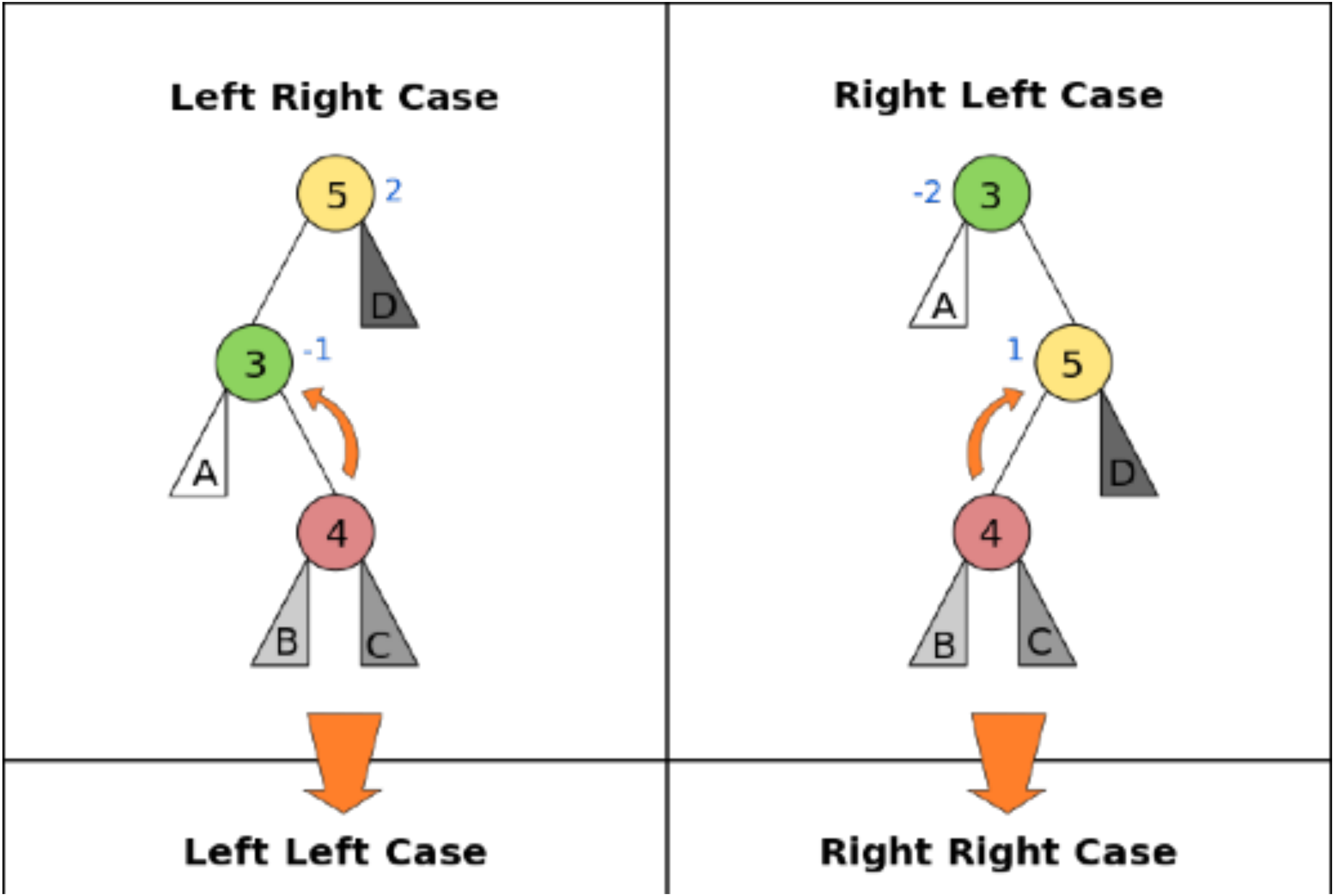
AVL tree balancing

```
balanceFactor = height(left subtree) − height(right subtree)
```


Left Right Case
5
2
D
3
-1
A
4
B
C
Left Left Case
Right Left Case
3
-2
A
5
1
D
4
B
C
Right Right Case

- if at any time they differ by more than one, rebalancing is done to restore this property

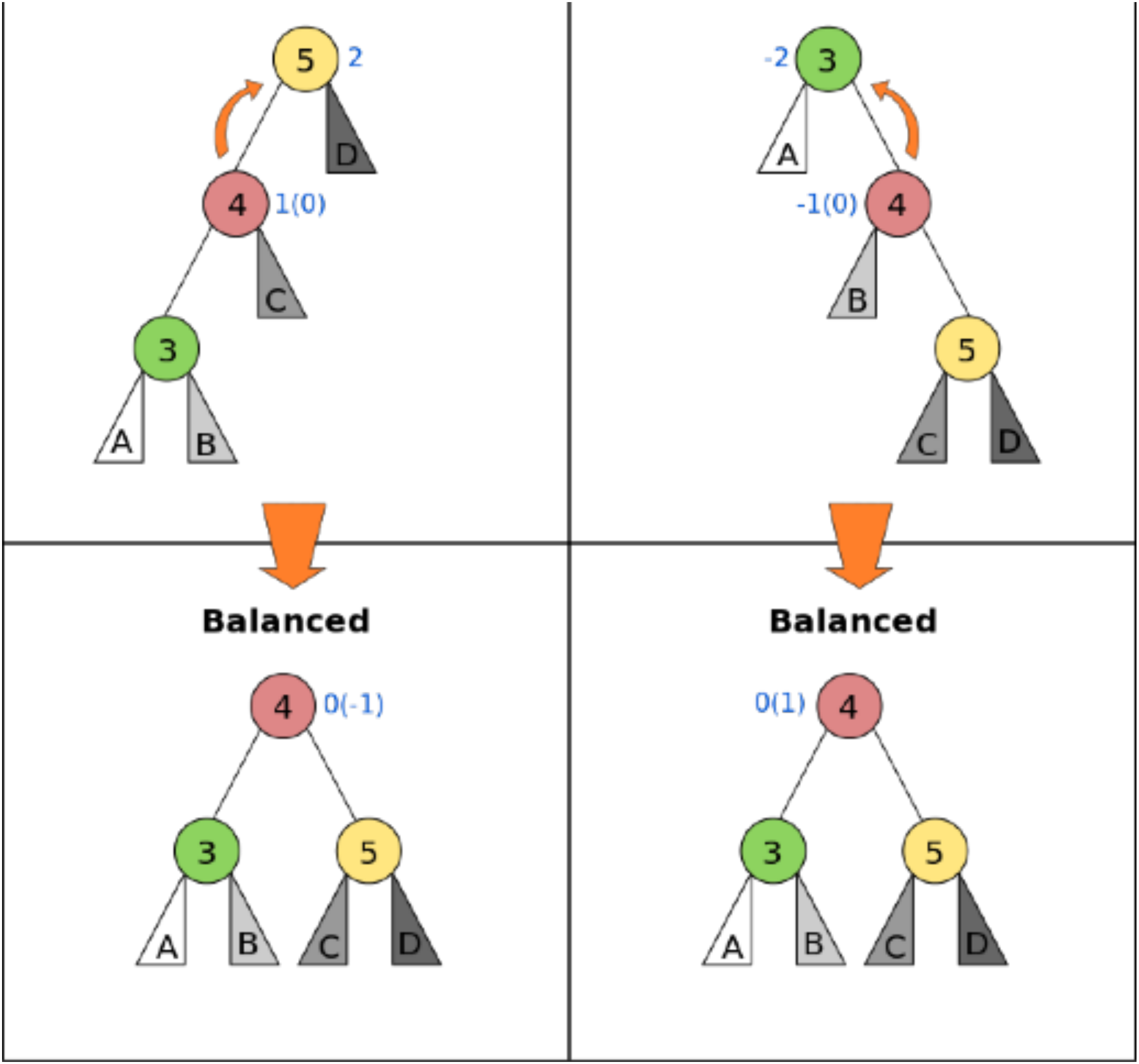
- To balance itself, an AVL tree may perform the following four kinds of rotations –

- Left rotation

- Right rotation

- Left-Right rotation

- Right-Left rotation


5
2
D
4
1(0)
C
3
A
B
-2
3
A
-1(0)
4
B
5
C
D
Balanced
4
0(-1)
3
5
A
B
C
D
Balanced
0(1)
4
3
5
A
B
C
D

- https://www.hackerrank.com/challenges/self-balancing-tree/problem
- https://www.tutorialspoint.com/data_structures_algorithms/avl_tree_algorithm.htm

- if a tree becomes unbalanced, when a node is inserted into the right subtree of the right subtree, then we perform a single left rotation

- if a node is inserted in the left subtree of the left subtree. The tree then needs a right rotation

- double rotations are also performed