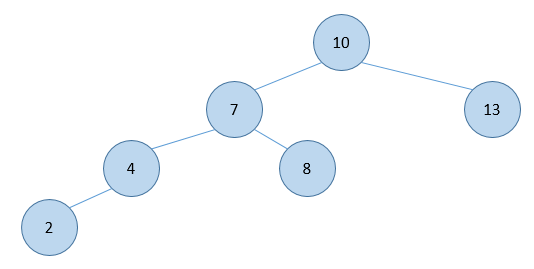
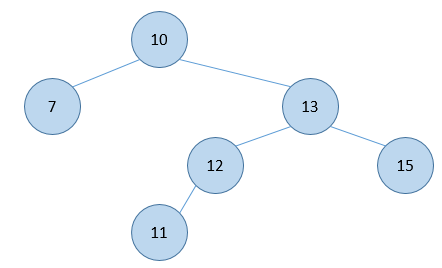
Exercises for lesson 4: Balanced binary search trees

*Exercises 4.1-4.3 concern AVL trees, while exercises 4.4-4.6 concern red-black trees.*

**Exercise 4.1**

Note the balance factor for each note in the following AVL trees and perform the necessary rotations on the trees below to correct the imbalance:

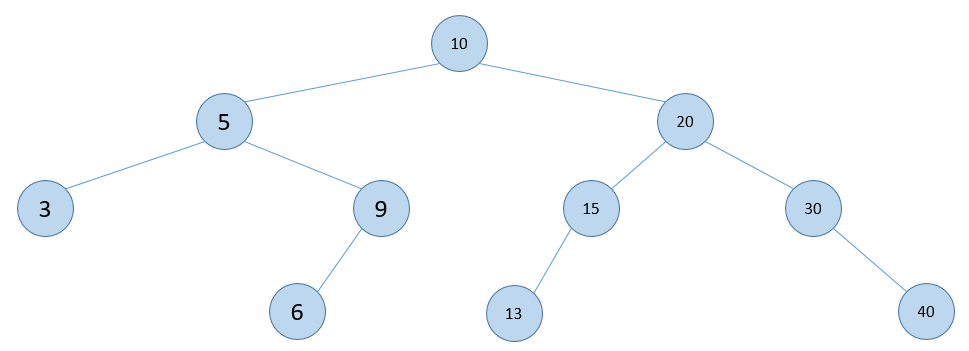
What type of rotations did you use?

**Exercise 4.2**

Draw the AVL tree which results from adding the following sequence of numbers to an empty tree:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10

**Exercise 4.3**



Draw the resulting AVL tree after performing the following operations:

* add(14)
* remove(40)
* add(7)
* add(8)
* remove(30)

Write the balance factor next to each node.

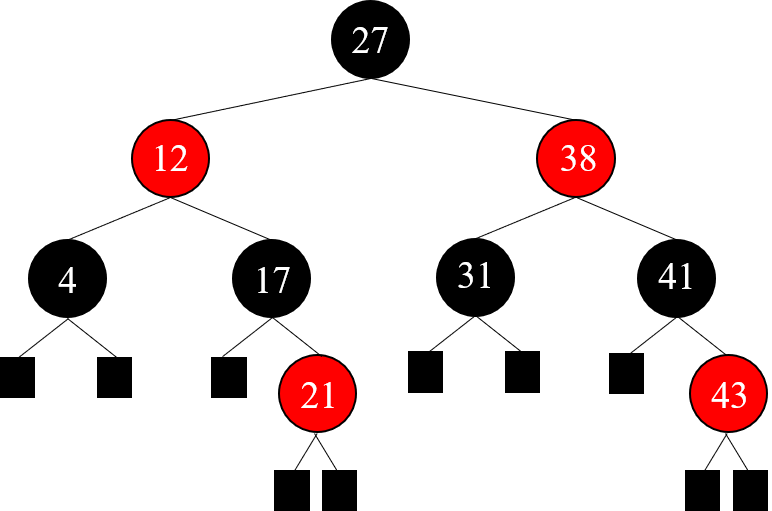
**Exercise 4.4**

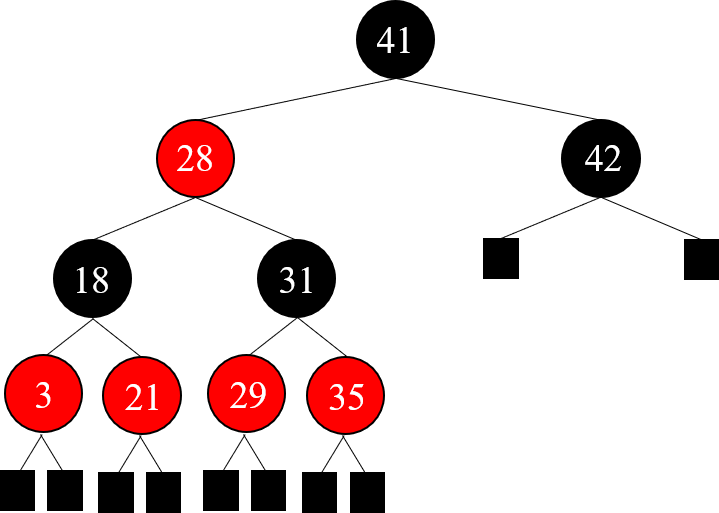
For the tree in exercise 4.3, assume that all nodes are initially black. State whether the tree becomes a valid red-black tree if the following nodes are colored red:

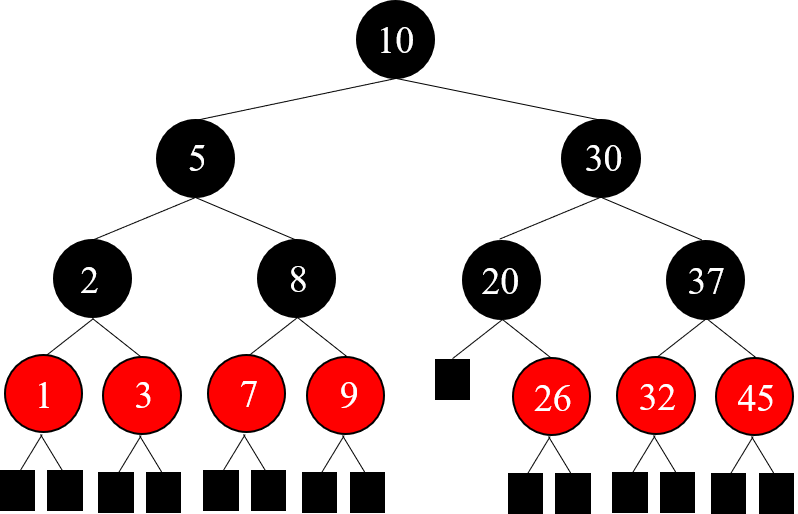
1. 5, 6, 20, 13, 40
2. 10, 6, 13, 40
3. 6, 15, 30
4. 10, 9, 15, 30
5. 10, 6, 13, 40
6. 6, 13, 40

**Exercise 4.5**

For each of the red-black trees below, what red-black trees results from inserting 25? The black rectangles represent null references.







**Exercise 4.6**

What red-black tree do you get if you insert the following elements into an initially empty tree in the following order: 7, 6, 45, 3, 31, 4, 31, 23, 1.