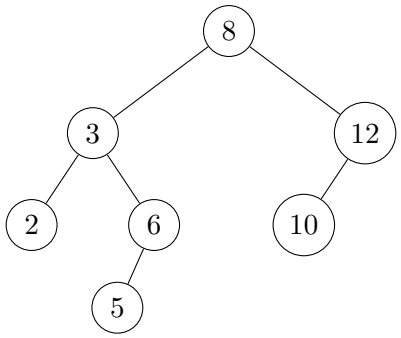
University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger, P. Schneider

Algorithms and Data Structures
Summer Term 2019
Sample Solution Exercise Sheet 8

Exercise 1: AVL Trees

Consider the following AVL tree



(a) Perform the operations `insert`(4), `insert`(7) and `insert`(1) and the necessary rotations to re-balance the AVL-tree. Draw the state of the tree after each operation.

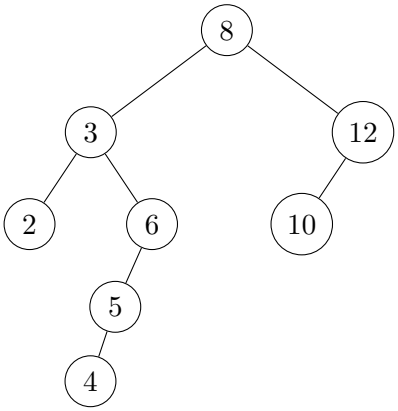
Remark: Inserting works the same as in binary search trees. Afterwards, for each ancestor of the inserted node (bottom up), repair the AVL condition (if violated) by performing an according rotation (left or right).

(b) In the resulting tree, perform the operations `delete`(5) and `delete`(7) and the necessary rotations to re-balance the AVL-tree. Draw the state of the tree after each operation.

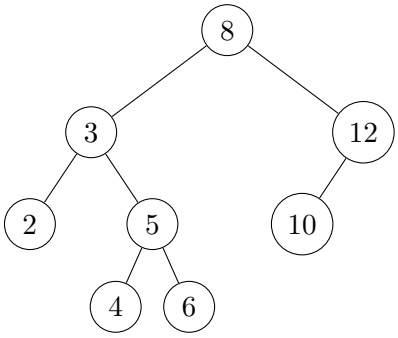
Remark: Deleting works the same as in binary search trees. Afterwards, starting at the position of the node that was used to replace the deleted key, for each ancestor (bottom up) repair the AVL condition (if violated) by performing an according rotation (left or right or double rotations).

Sample Solution

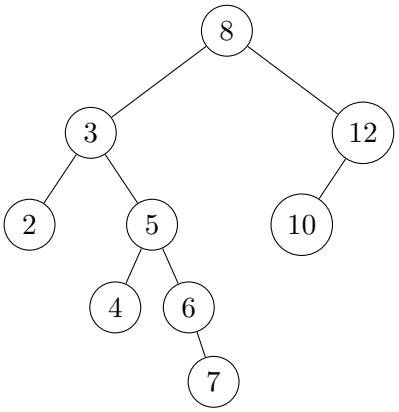
(a) `insert`(4), before balance:



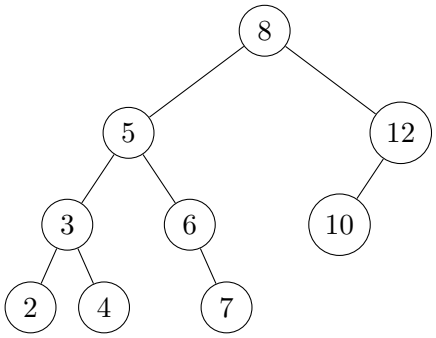
`insert`(4), after balance:



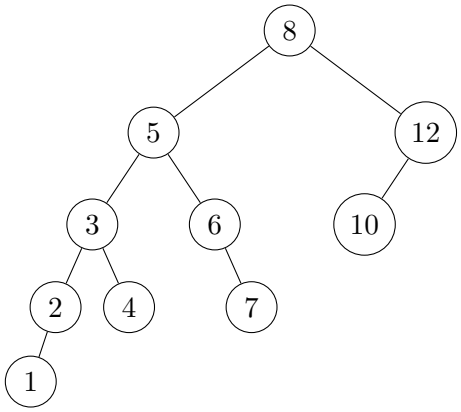
`insert`(7), before balance:



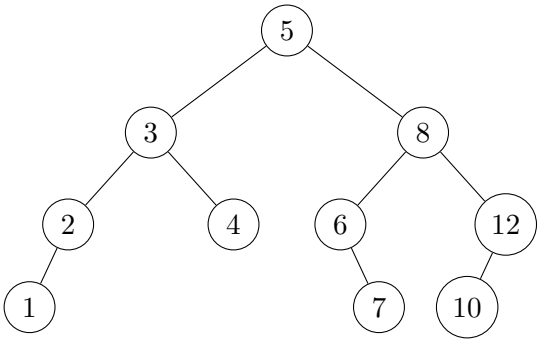
`insert`(7), after balance:



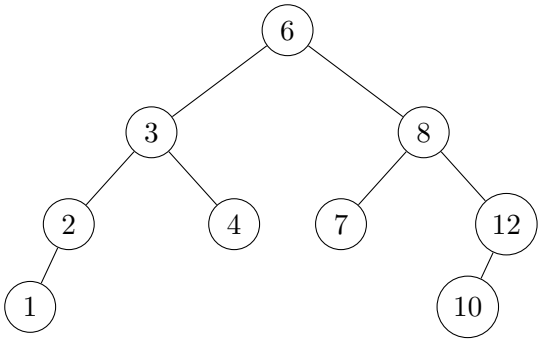
`insert`(1), before balance:



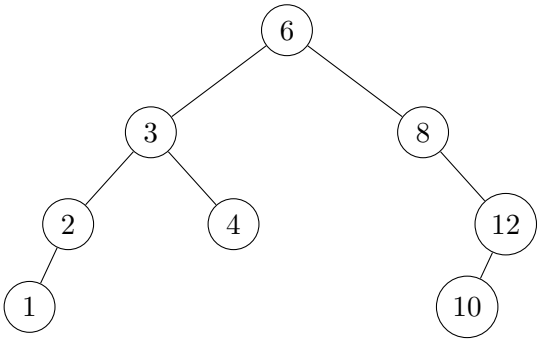
`insert`(1), after balance:



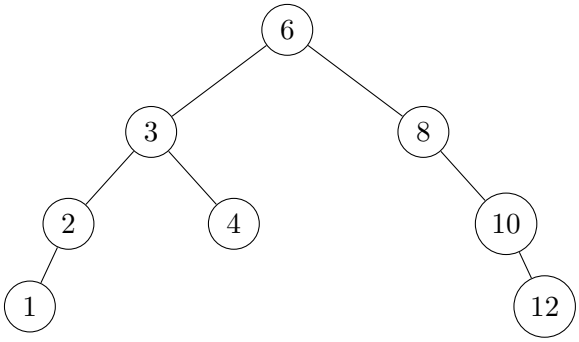
(b) `delete`(5), (no balance required):



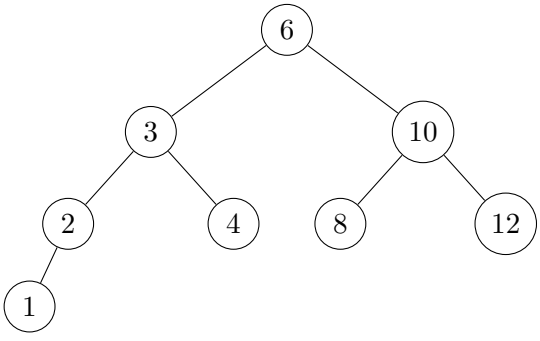
`delete`(7), before balance:



`delete`(7), double rotation part 1:

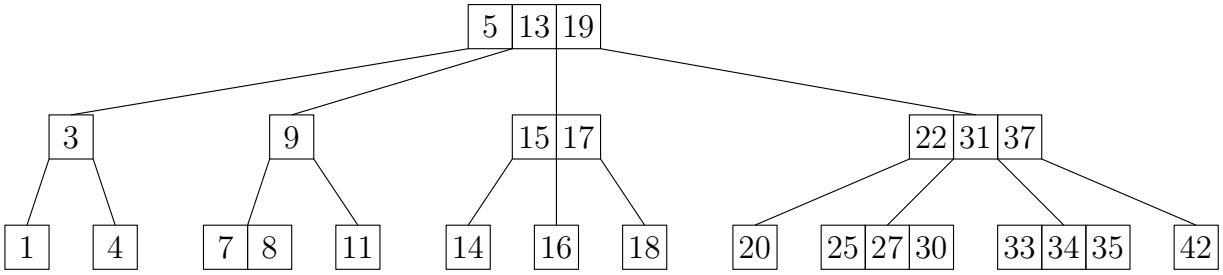


`delete`(7), double rotation part 2:



Exercise 2: (a, b)-Trees

Consider the following $(2, 4)$-tree



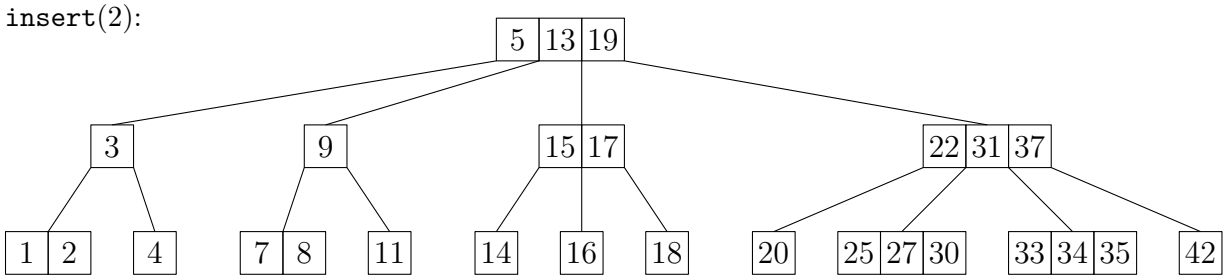
(a) Perform the operations `insert`(2), `insert`(26) and `insert`(36). Draw the state of the $(2, 4)$-tree after all operations.

(b) In the original tree, perform the operations `delete`(11), `delete`(3). Draw the state of the $(2, 4)$-tree after both operations.

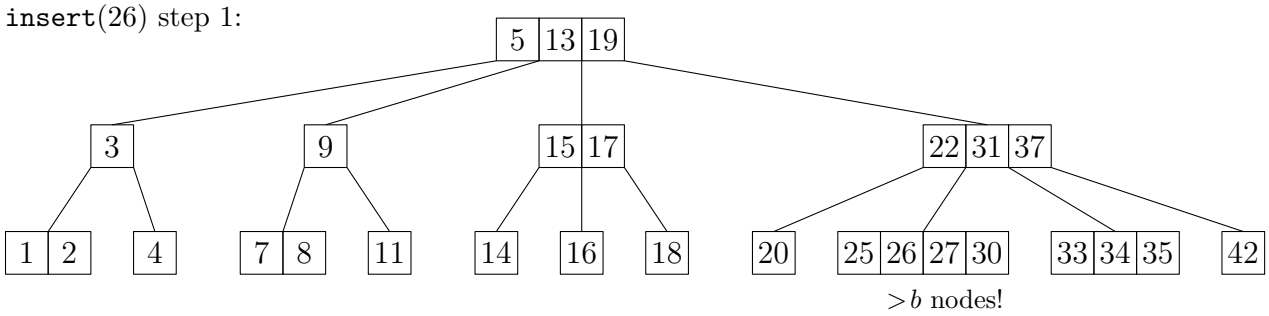
(c) For exercise lesson: also do a `delete`(13) operation.

Sample Solution

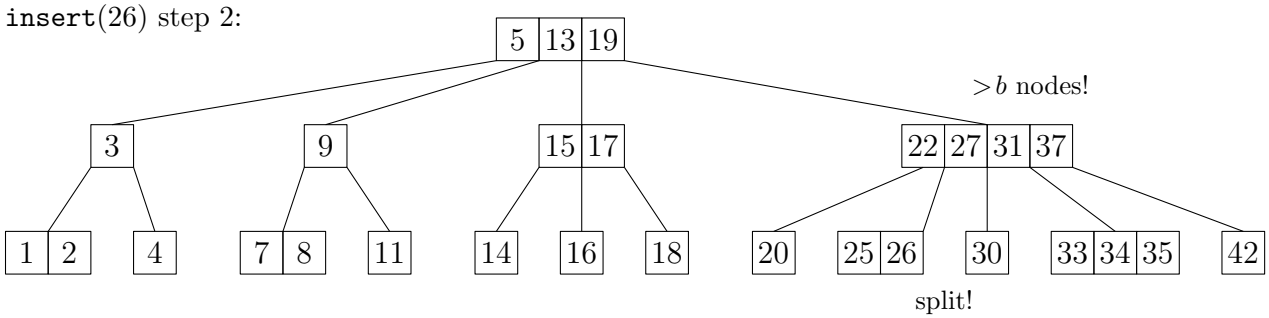
(a) `insert`(2):



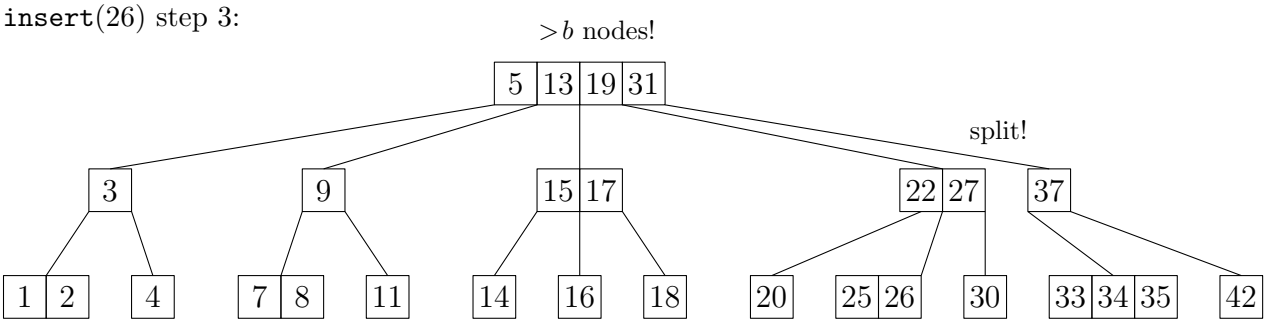
`insert`(26) step 1:



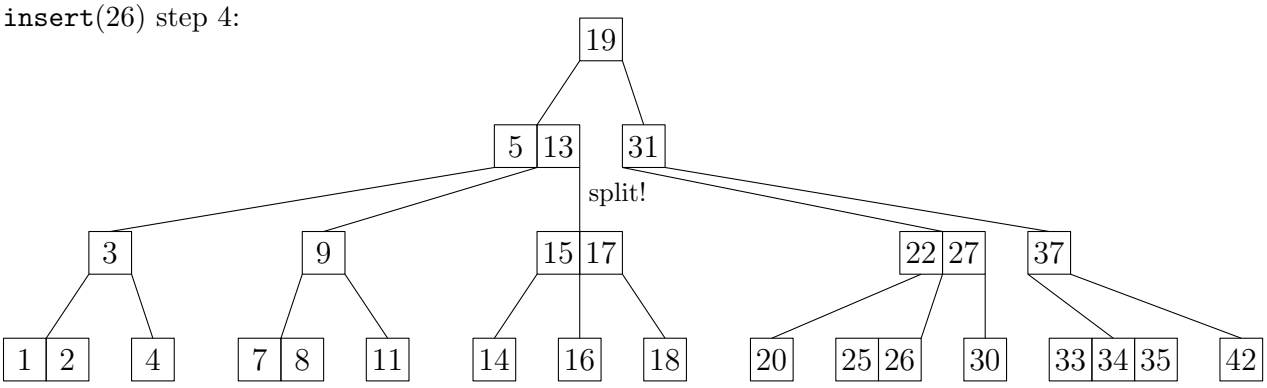
`insert`(26) step 2:



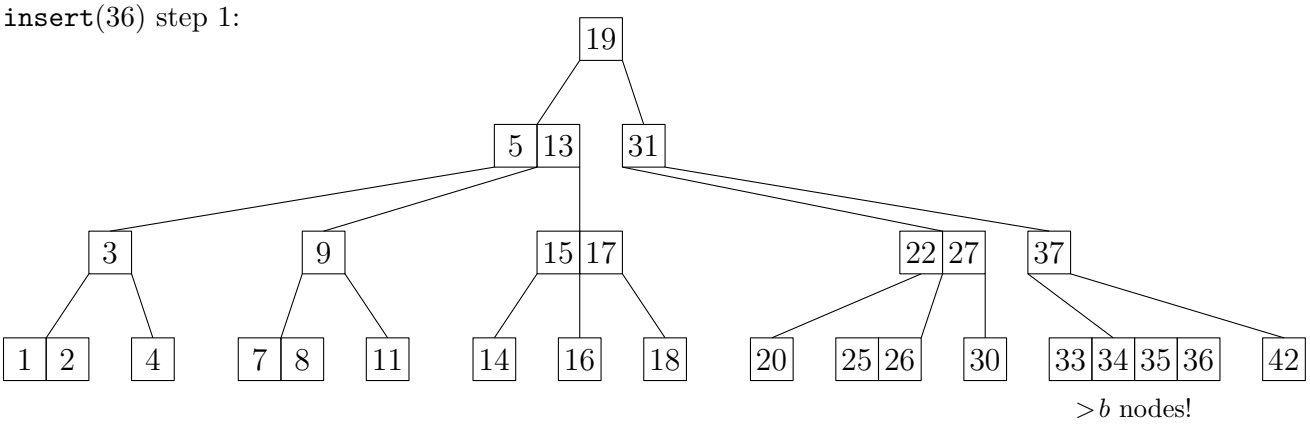
`insert`(26) step 3:



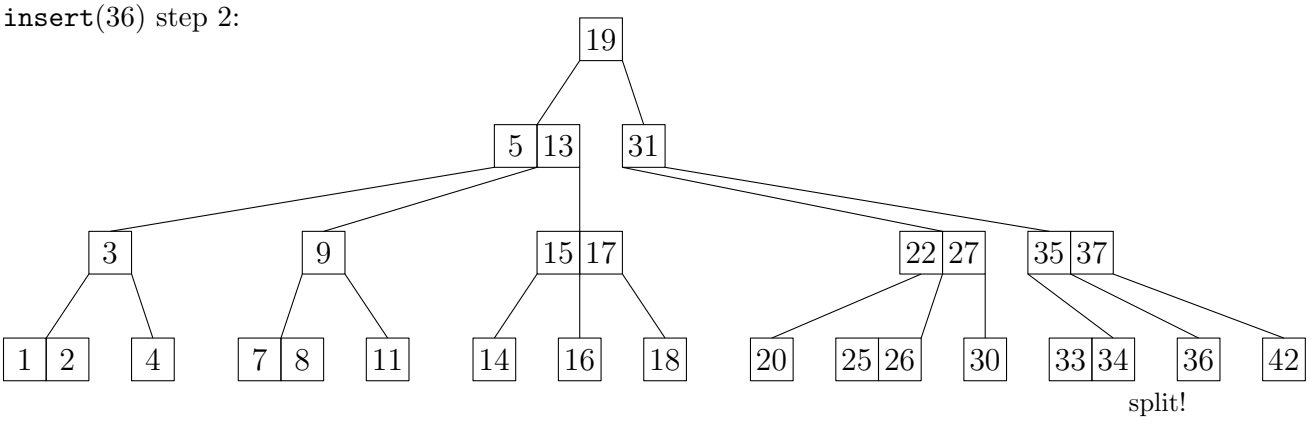
`insert`(26) step 4:



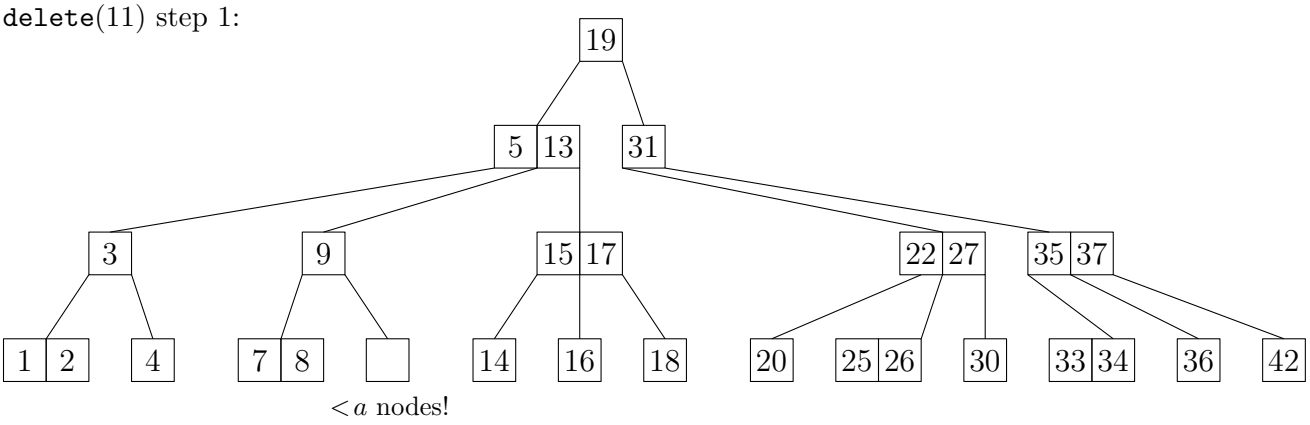
`insert`(36) step 1:



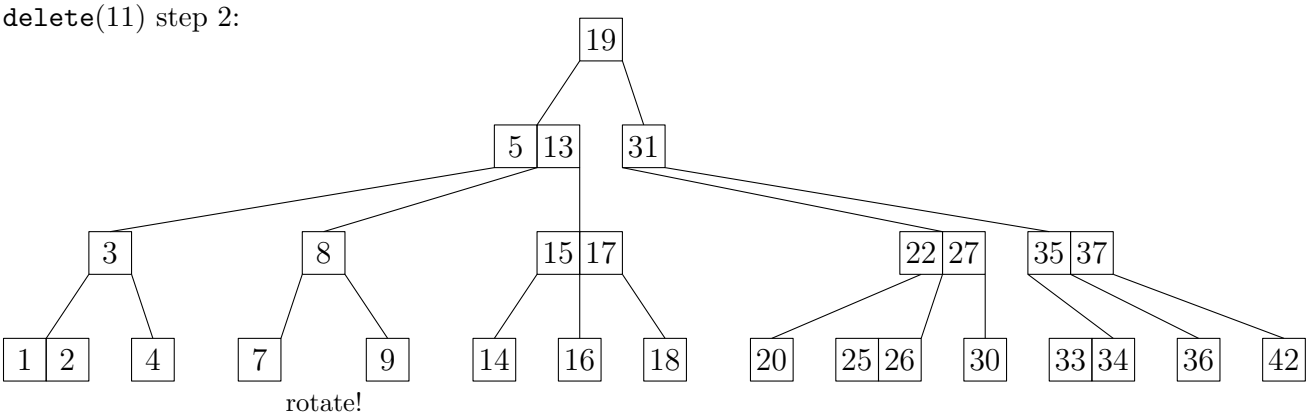
`insert`(36) step 2:



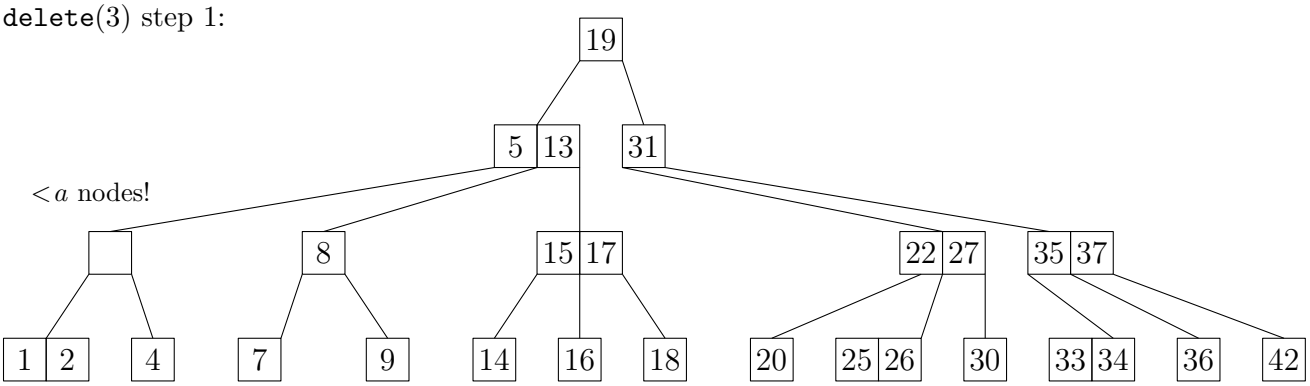
(b) `delete`(11) step 1:



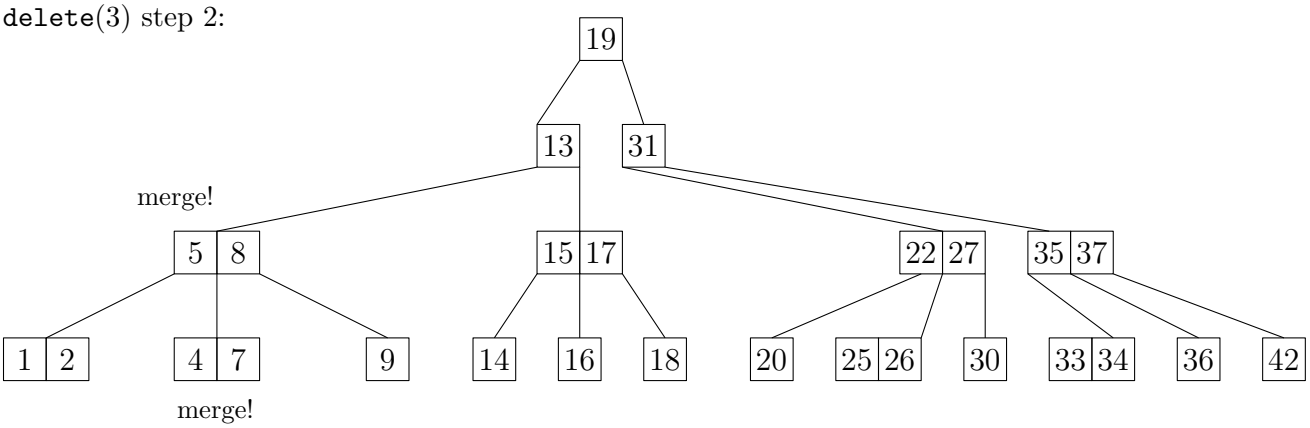
`delete`(11) step 2:



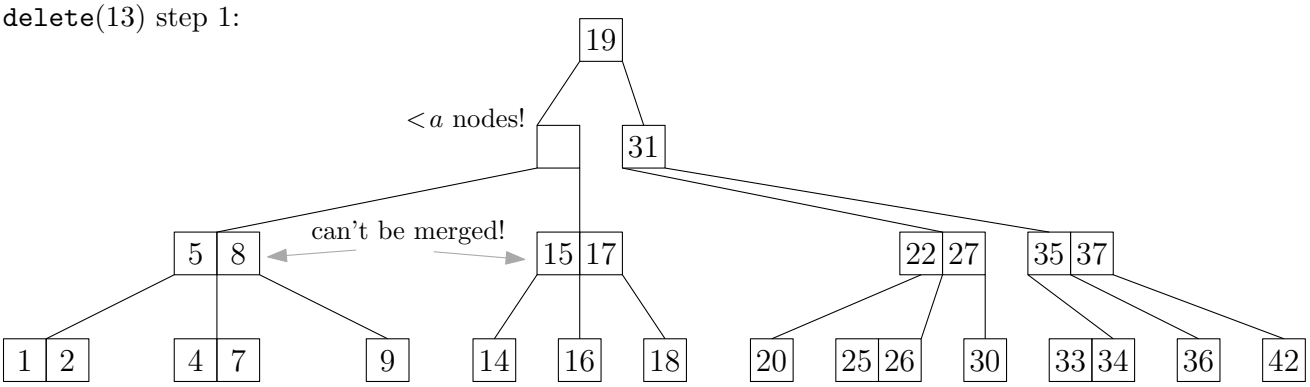
`delete`(3) step 1:



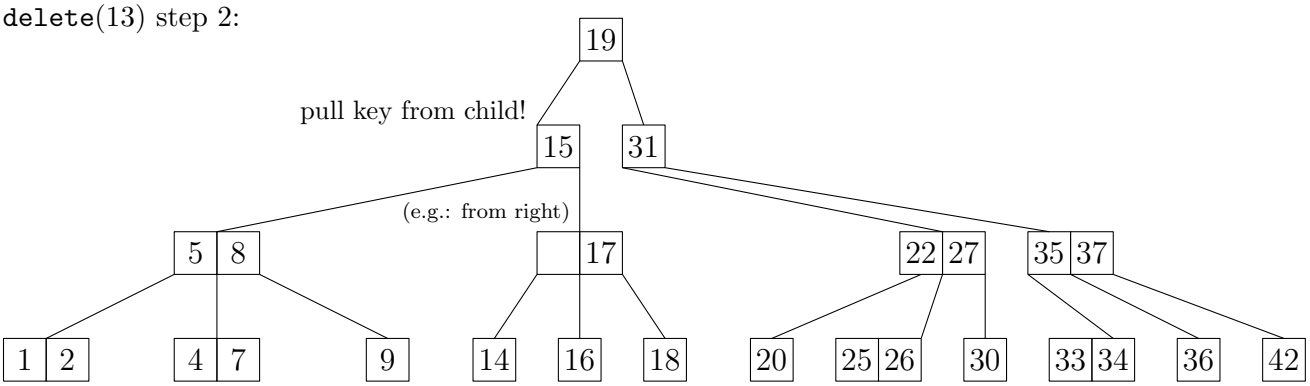
`delete`(3) step 2:



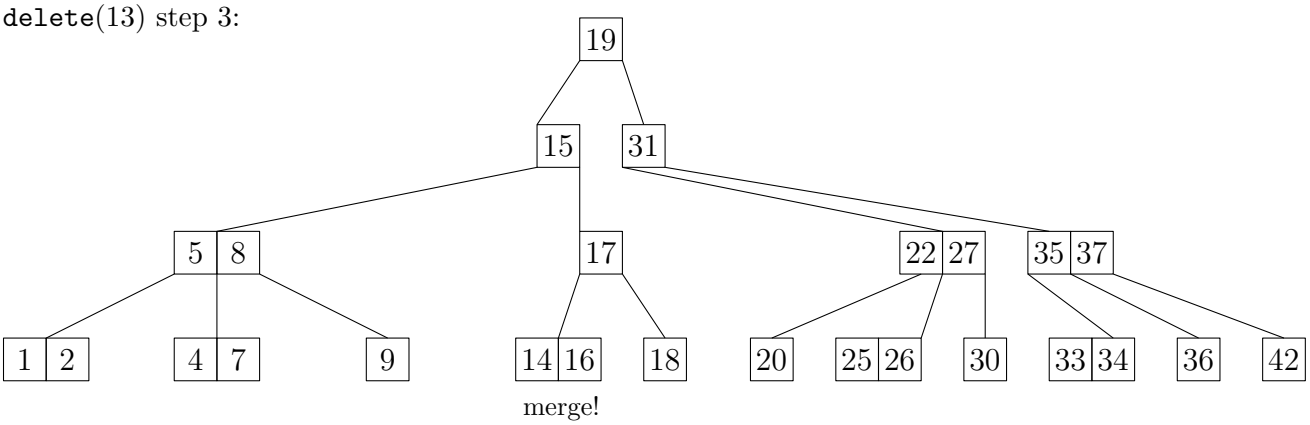
(c) `delete`(13) step 1:



`delete`(13) step 2:



`delete`(13) step 3:



Remark: For more details on all cases of the **`delete`** *operation consider e.g. "Introduction to Algorithms" by Knuth, Leierson, Rivest and Stein.*