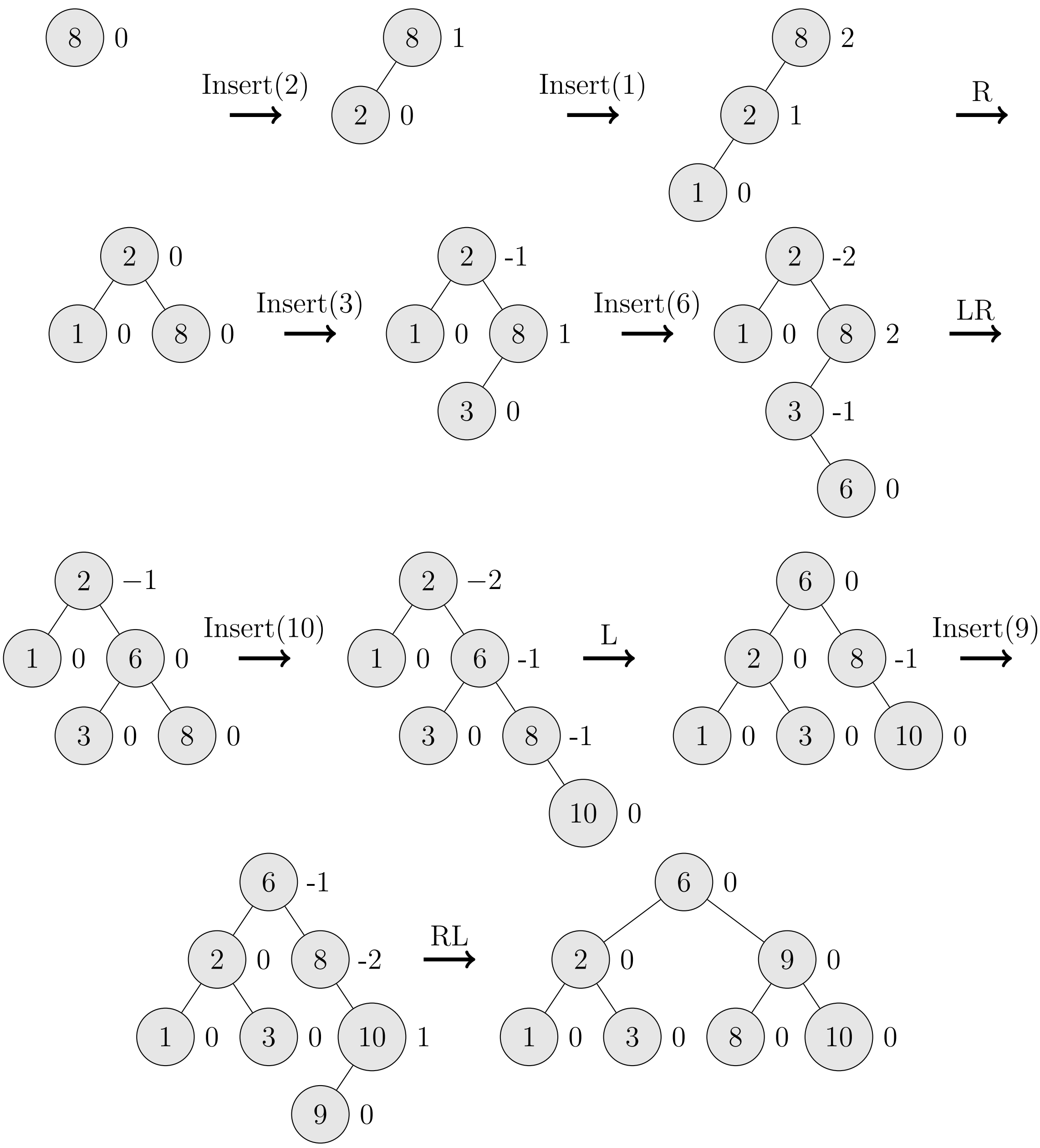
Solutions for Exercise Sheet 10

Handout: November 21st — Deadline: December 5th before 4pm

Question 10.1 (0.5 Marks)

Insert the keys $8, 2, 1, 3, 6, 10, 9$ in this order into an empty AVL tree. Draw the tree constructed after each insertion and after each (double-)rotation (cf. the example in the lecture notes). Write down the balance degree for each node next to the node as shown in the lecture notes.

Solutions: The following shows all trees constructed. Rotations are necessary whenever a balance degree falls outside $\{-1, 0, +1\}$. Since Insert moves up the search path, the rebalance procedure starts with the lowest unbalanced node on the search path. In particular, after inserting 6, the double rotation concerns node 8; node 2 becomes balanced during this double rotation, so we stop.



Question 10.2 (0.5 marks)

Say the minimum number of nodes that an AVL tree of height $h = 10$ must contain.

Solutions: Recall from the lecture that $A(h)$ is the minimum number of nodes in an AVL tree of height h. We claim that $A(10)$ is very large. To see this, recall that $A(0) = 1, A(1) = 2$ and $A(h) = 1 + A(h-1) + A(h-2)$ for $h \geq 2$. Applying this formula repeatedly, we get $A(2) = 1+2+1 = 4$, $A(3) = 1+4+2 = 7$, $A(4) = 1+7+4 = 12$, $A(5) = 1+12+7 = 20$, $A(6) = 1+20+12 = 33$, $A(7) = 1+33+20 = 54$, $A(8) = 1+54+33 = 88$, $A(9) = 1+88+54 = 143$, $A(10) = 1+143+88 = 232$. So every AVL tree of height 10 must contain at least 232 nodes.

Alternative solutions: An alternative argument would be to use that $A(10) = \text{Fib}(12) - 1 = 233 - 1 = 232$ where it's fine to look up the Fibonacci numbers somewhere.

Another solution is to recall that $h \leq 1.44 \log n$ from the lecture notes and to argue that this is equivalent to $h/1.44 \leq \log n$ and, raising both sides to the power of 2, $2^{h/1.44} \leq n$. Plugging in $h = 10$ gives $n \geq 2^{10/1.44} = 123.16\ldots$. Hence at least 124 nodes are needed (we can round up since n must be an integer). This statement is slightly weaker than the number 232 resulting from working out $A(10)$, but still a good answer.

Question 10.3 (2 marks) Implement an AVL tree using linked lists. You should implement an insert procedure which should run in $O(\log n)$ time while the balancing, including adjusting the balance factors, should run in constant time $\Theta(1)$. You don't need to implement a delete procedure (although if you do, you get a bonus of doubling your marks if it is correct).

In input you get a set of *distinct* integers that are the keys that need to be inserted in that order: eg., $8, 2, 1, 3, 6, 10, 9$.

Output: print the tree INORDER.