1. (20 points) In the AVL tree data structure, we need to add an extra field $\text{height}(x)$ to bookkeep the height of each node $x$ so that whenever a node becomes unbalanced, we can perform rotations to correct this. How much space is required for this field for each node? Can you improve the space complexity for each node to $O(1)$?

2. (20 points) Let $T$ be a binary search tree (BST) of height $h$. Design an algorithm which prints all the keys in $T$ whose values are between $v_1$ and $v_2$ ($v_1 < v_2$). If there are $m$ keys in this range, then your algorithm should run in $O(m + h)$ time.

3. (20 points) Suppose you are given an unsorted list of $n$ distinct numbers. However, the list is close to being sorted in the sense that each number is at most $k$ positions away from its original position in the sorted list. For example, suppose the sorted list is in ascending order, then the smallest element in the original list will lie between position 1 and position $k + 1$, as position 1 is its position in the original sorted list. Design an efficient algorithm that sorts such a list. The running time of your algorithm should depend on both $n$ and $k$ (a helpful sanity check whether your running time bound makes sense is that, when $k$ is close to $n$, then the input list is just a generic unsorted list).

4. (20 points) Given an array $A$ of $n$ integer and an integer $t$. Design an efficient algorithm that finds the number of total pairs $(i, j)$ ($i < j$) such that $|A[i] + A[j] - t|$ is minimal possible.

5. (20 points) Suppose there are $n$ types of coupons in a lottery and each lot contains one coupon (with equal probability $1/n$ to be each of the $n$ types). In expectation how many lots one has to buy until she has at least one coupon of each type?