1. Give a procedure that manipulates an array $A[1..n]$ and whose time complexity is described by the recurrence

(a) $T(n) = 3T(\lceil n/2 \rceil) + O(n)$

(b) $T(n) = 3T(\lceil n/2 \rceil) + O(n^2)$

(The procedures (a) and (b) do not need to perform any reasonable task.) Analyze the complexity of the procedures.


(b) Which items of an array $A[1..2^{20}]$ are examined while binary-searching for key (i) $A[1]$, (ii) $A[2^{19}]$, (iii) that is larger than any key in the array?

3. Let’s consider the behavior of an unbalanced binary search tree.

(a) Build such by inserting the letters of the word OIVALLISTA, in left-to-right order, into an initially empty tree. How many nodes do the insertions manipulate in total? (The insert of 'O' manipulates the root node that’s created, the insertion of 'I' manipulates the root node and the child node created for it, etc.)

(b) What is the smallest and the largest number of nodes being manipulated while inserting 10 keys into an initially empty tree?

(c) The corresponding numbers while inserting 100 keys?

(d) Compare the numbers of (b) and (c) with the corresponding asymptotic complexity estimates discussed at the lecture.

4. The task is to build a balanced binary search tree from given keys $(a_1, \ldots, a_n)$. Give an algorithm and analyze its time complexity. Solve the problem without applying actual tree balancing operations. (Hint: sorting + divide-and-conquer.)

5. A sequence of items can be sorted by first inserting its keys into a 2–3–4 tree, and then picking them during an appropriate traversal.

(a) Sketch the algorithm; Assume that a sub-routine for inserting a key into a 2–3–4 tree is available.

(b) Demonstrate the working of the algorithm by using it to sort the keys 3, 1, 4, 5, 11, 2, 6, 10, 7, and 9.

(c) Analyze algorithm’s time complexity and need for space (of the tree). (Assume that records of a fixed size are used for the nodes of the tree, and that each key and pointer takes one memory word each.)