(a) Write brief notes on top–down merge sort, contrasting it with insertion sort. State its worst-case and average-case complexity, with brief justification. (There is no need to present ML code.) [5 marks]

(b) Write brief notes on preorder, inorder and postorder tree traversal. Present efficient code for one of them and state, with justification, its worst-case complexity. [5 marks]

(c) The binary search tree $t_1$ is *superseded by* $t_2$ provided every (key, value) entry in $t_1$ is also present in $t_2$. Code an ML function to determine whether one binary search tree is superseded by another. Express its cost in terms of $n_1$ and $n_2$, the numbers of entries in $t_1$ and $t_2$, respectively. For full credit, the worst-case cost should be no worse than $O(n_1 + n_2)$. [10 marks]

All code must be explained clearly. You may assume that any necessary ML data structures or functions are available.