8 Algorithms (fms27)

(a) Consider a Binary Search Tree. Imagine inserting the keys 0, 1, 2, \ldots, \(n\) (in that order) into the data structure, assumed initially empty.

(i) Draw a picture of the data structure after the insertion of keys up to \(n = 9\) included. \[2\text{ marks}\]

(ii) Clearly explain, with a picture if helpful, how the data structure will evolve for arbitrary \(n\), and derive the worst-case time complexity for the whole operation of inserting the \(n + 1\) keys. \[2\text{ marks}\]

(b) Repeat (a)(i) and (a)(ii) for a 2-3-4 tree, with some scratch work showing the crucial intermediate stages. \[2+2\text{ marks}\]

(c) \ldots and for a B-tree with \(t = 3\), again showing the crucial intermediate stages. \[2+2\text{ marks}\]

(d) \ldots and for a hash table of size 7 that resolves collisions by chaining. \[2+2\text{ marks}\]

(e) \ldots and for a binary min-heap. \[2+2\text{ marks}\]