CS240: Data Structures
Winter Semester – Academic Year 2017-18
Panagiota Fatourou

4th Set of Theoretical Exercises

Submission deadline : Wednesday, 10/1/2018
How to submit: The assignments must be submitted in electronic format using the turnin program (see instructions on the website of the course). Assignments submitted after 16:00 of Wednesday, 10/1/2018 are considered out of date. Out of date assignments are accepted only in electronic format using the turnin program.

Exercise 1 [30 Points]

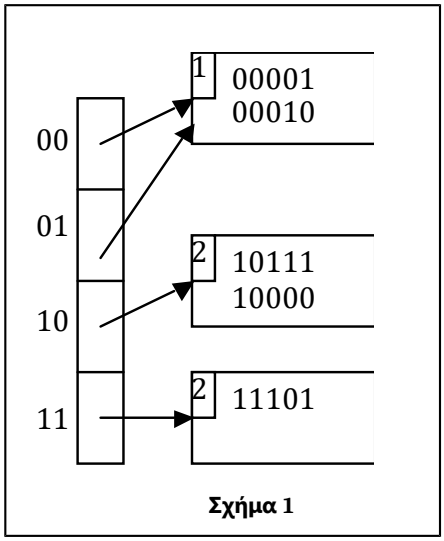
a. Suppose that H_1 and H_2 are two heaps of size n_1 and n_2, $n_1 \geq n_2$, whereas each element of H_2 is greater than each element of H_1. Explain how we can merge the two heaps into one at time O (n_1). [15P]

b. Prove that deleting any element of a heap can be done in time O (log n), if we have a pointer on the node (that is, you are asked for a O(log n) implementation of HeapDelete (j, h), which deletes the element with index j from the heap h. [15P]

Exercise 2 [40 Points]

a. Starting with 10 sets of S_i, $1 \leq i \leq 10$, each of which contains only one element, i, show the up-trees that occur after each of the following functions is performed, given that the strategy for height reduction is used: A = Union(S_3, S_2); B = Union(S_7, S_9); C = Union(A,B); D = Union(C,S_4); E = Union(S1, S5); F = Union(S6,S10); G = Union(F,E); I = Union(G, D);

   Present the up-trees after performing Find (9) and Find (10) if the path compression strategy is used (you should present two trees, that which results after Find (9) execution and that which results after Find (10) execution on the tree that resulted from Find (9) execution). [8P]



Σχήμα 1

b. Insert the key with a hash value 00100 in the expandable hash table of Figure 1. Delete the key with a hash value 11101 from the table of Figure 1. Consider that each sheet page of the table holds 2 records (that is, suppose that b = 2) and that a deletion strategy is used according to which the directory should be kept as small as possible. [6P]

c. Consider that the tree of Figure 2 is an AVL tree (ie ignore the color of the nodes). Delete the key 120. Insert the key 10 in the AVL tree of Figure 2. [6P]

d. A 2-3 tree consists of 7 2-nodes which have the keys 500, 300, 700, 600, 800, 200, 400. Paint the tree and delete the key 500. [8P]

e. A 2-3 tree consists of 13 3-nodes. The keys in the tree are as follows: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52. Paint the tree and insert the key 25 into it. [8P]

z. Insert the key 45 into the red-black tree of Figure 2. [4P]

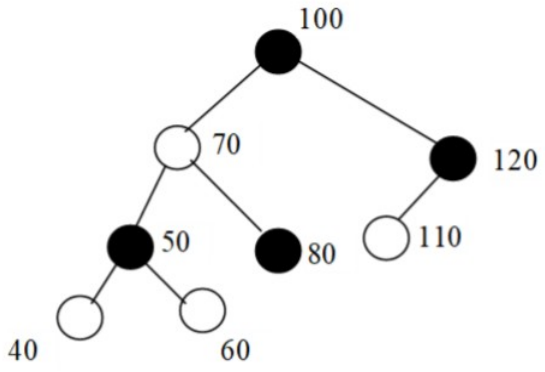


Figure 2

Note: In the above queries, **you must present the entire process** you follow in order to reach the final tree (the rotations where needed should also be described in detail).

Exercise 3 [30 Points]

Suppose that for every red black tree T we maintain a variable bh in which we store the black tree height. For example, the tree is represented as a struct with fields of an index r at the root of the tree and an integer bh in which the tree's black height is stored.

a. Prove that bh can be correctly computed when inserting or deleting a node in T without requiring extra memory and without affecting the time complexity of these algorithms (insertion and deletion). Explain how this can be achieved. [10P]

b. Prove that given the bh of the tree, as we descend from the root to any leaf, the black height of each node of the path traversed can be calculated in O(1) time (so that the black heights of all the nodes of the path can be calculated in logarithmic time). [10P]

c. Suppose the struct of the red-black tree node contains a bh field, which should store the black height of the node. Consider a tree in which the bh field of all nodes has the initial value of -1. Present a recursive algorithm that uses the pre-order tracing(and its knowledge of the black height of the entire tree) to store the black height of each node in the struct of the node. [10P]