Exercise Sheet 4

Handout: Oct 10 — Deadline: Oct 17, 4pm

Question 4.1 (0.25 marks) Say whether the following array is a Max-Heap (justify your answer):

34	20	21	16	14	11	3	14	17	13

Question 4.2 (0.25 marks)

Consider the following input for HeapSort:

12	10	4	2	9	6	5	25	8

Create a heap from the given array and sort it by executing HeapSort. Draw the heap (the tree) after Build-Max-Heap and after every execution of Max-Heapify in line 5 of HeapSort. You don't need to draw elements extracted from the heap, but you can if you wish.

Question 4.3 (0.5 marks)

1. Provide the pseudo-code of a Max-Heapify(A, i) algorithm that uses a WHILE loop instead of the recursion used by the algorithm shown at lecture.
2. Prove correctness of the algorithm by loop invariant.

Question 4.4 (1.25 marks)

1. Show that each child of the root of an n-node heap is the root of a sub-tree of at most $(2/3)n$ nodes. (*HINT: consider that the maximum number of elements in a subtree happens when the left subtree has the last level full and the right tree has the last level empty. You might want to use the formula seen at lecture:* $\sum_{i=0}^{k-1} 2^i = 2^k - 1$).
2. As a consequence of (1) we can use the recurrence equation $T(n) \leq T(2n/3) + \Theta(1)$ to describe the runtime of Max-Heapify(A, n). Prove the runtime of Max-Heapify using the Master Theorem.

Question 4.5 (1 mark)

Argue that the runtime of HeapSort on an already sorted array of distinct numbers is $\Omega(n \log n)$.

Question 4.6 (0.25 marks)

Implement HeapSort(A, n).