Exercise Sheet 3

Handout: Sep 26th — Deadline: Oct 10th - 4pm

Question 3.1 (0.25 marks)

Consider the following input for MergeSort:

12	10	4	2	9	6	5	25	8

Illustrate the operation of the algorithm (follow how it was done in the lecture notes).

Question 3.2 (0.5 marks) Prove using the substitution method the runtime of the MergeSort Algorithm on an input of length n, as follows. Let n be an exact power of 2, $n = 2^k$ to avoid using floors and ceilings. Use mathematical induction over k to show that the solution of the recurrence involving positive constants $c, d > 0$

$$T(n) = \begin{cases} d & \text{if } n = 2^0 = 1 \\ 2T(n/2) + cn & \text{if } n = 2^k \text{ and } k \geq 1 \end{cases}$$

is $T(n) = dn + cn \log n$ (we always use log to denote the logarithm of base 2, so $\log = \log_2$).

Hint: you may want to rewrite the above by replacing n with 2^k. Then the task is to prove that $T(2^k) = d2^k + c2^k \cdot k$ using the recurrence

$$T(2^k) = \begin{cases} d & \text{if } k = 0 \\ 2T(2^{k-1}) + c2^k & \text{if } k \geq 1 \end{cases}$$

Question 3.3 (0.5 marks) Use the Master Theorem to give asymptotic tight bounds for the following recurrences. Justify your answers.

1. $T(n) = 2T(n/4) + 1$
2. $T(n) = 2T(n/4) + \sqrt{n}$
3. $T(n) = 2T(n/4) + \sqrt{n} \log^2 n$
4. $T(n) = 2T(n/4) + n$

Question 3.4 (0.5 marks) Write the pseudo-code of the *recursive* BinarySearch$(A, x, \text{low}, \text{high})$ algorithm discussed during the lecture to find whether a number x is present in an increasingly sorted array of length n. Write down its recurrence equation and prove that its runtime is $\Theta(\log n)$ using the Master Theorem.

Question 3.5 (0.25 marks) Implement the MergeSort(A, p, r) algorithm and the BinarySearch$(A, x, \text{low}, \text{high})$ algorithm designed in the previous question.