

Exercise Sheet 5

Handout: Oct 17th — Deadline: Oct 24th, 4pm

Question 5.1 (Marks: 0.25)

Illustrate the operation of QUICKSORT on the array

4	3	8	2	7	5	1	6
---	---	---	---	---	---	---	---

Write down the arguments for each recursive call to QUICKSORT (e. g. “QUICKSORT($A, 2, 5$)”) and the contents of the relevant subarray in each step of PARTITION (see Figure 7.1). Use vertical bars as in Figure 7.1 to indicate regions of values “ $\leq x$ ” and “ $> x$ ”. You may leave out elements outside the relevant subarray and calls to QUICKSORT on subarrays of size 0 or 1.

Question 5.2 (Marks:0.5)

Prove that deterministic QUICKSORT(A, p, r) is correct (you can use that PARTITION is correct since that was proved at lecture).

Question 5.3 (Marks: 0.25) What is the runtime of QUICKSORT when the array A contains distinct elements sorted in decreasing order? (Justify your answer)

Question 5.4 (Marks: 0.5)

What value of q does PARTITION return when all n elements have the same value?

What is the asymptotic runtime (Θ -notation) of QUICKSORT for such an input? (Justify your answer).

Question 5.5 (Marks: 0.5)

Modify PARTITION so it divides the subarray in three parts from left to right:

- $A[p \dots i]$ contains elements smaller than x
- $A[i + 1 \dots k]$ contains elements equal to x and
- $A[k + 1 \dots j - 1]$ contains elements larger than x .

Use pseudocode or your favourite programming language to write down your modified procedure PARTITION' and explain the idea(s) behind it. It should still run in $\Theta(n)$ time for every n -element subarray. Give a brief argument as to why that is the case. PARTITION' should return two variables q, t such that $A[q \dots t]$ contains all elements with the same value as the pivot (including the pivot itself).

Also write down a modified algorithm QUICKSORT' that uses PARTITION' and q, t in such a way that it recurses only on strictly smaller and strictly larger elements.

What is the asymptotic runtime of QUICKSORT' on the input from Question 5.4?

Question 5.6 (Marks:0.5)

Implement QUICKSORT, RANDOMIZED-QUICKSORT and QUICKSORT' from Question 5.4