Exercise Sheet 4
COMS10017 Algorithms 2022/2023

1 Algorithm Design

Describe an $O(n \log n)$ time algorithm that, given an array A of n integers and another integer x, determines whether or not there are two elements in A whose sum equals x (Hint: Sorting!).

2 Mergesort

The Mergesort algorithm uses the Merge operation, which assumes that the left and the right halves of an array A of length n are already sorted, and merges these two halves so that A is sorted afterwards. The runtime of this operation is $O(n)$.

Suppose that we replaced the Merge operation in our Mergesort algorithm with a less efficient implementation that runs in time $O(n^2)$ (instead of $O(n)$). What is the runtime of our modified Mergesort algorithm?

3 Bubblesort

Bubblesort is a popular, but inefficient, sorting algorithm. It works by repeatedly swapping adjacent elements that are out of order:

Algorithm 1 Bubblesort

```
Require: Array $A$ of $n$ integers
1: for $i = 0$ to $n - 2$ do
2:   for $j = n - 1$ downto $i + 1$ do
3:     if $A[j] < A[j - 1]$ then
4:       exchange $A[j]$ with $A[j - 1]$
5:     end if
6:   end for
7: end for
```

1. What is the worst-case runtime of Bubblesort?
2. Consider the loop in lines $2 - 6$. Prove that the following invariant holds at the beginning of the loop:

   $$A[j] \le A[k], \text{ for every } k \ge j \ .$$

   Give a suitable termination property of the loop.
3. Consider now the loop in lines $1 - 7$. Prove that the following invariant holds at the beginning of the loop:

   The subarray $A[0, i]$ is sorted and $A[0, i - 1]$ consists of the $i - 1$ smallest elements of A.

   Give a suitable termination property that shows that A is sorted upon termination.

4 Optional and Difficult Questions

Exercises in this section are intentionally more difficult and are there to challenge yourself.

4.1 Closest Pair of Points (hard!)

The input consists of two arrays of n real numbers X, Y and represent n points with coordinates $(X[0], Y[0]), (X[1], Y[1]), \ldots, (X[n-1], Y[n-1])$. Give a divide-and-conquer algorithm that finds the pair of points that are closest to each other, i.e., the output consists of a two indices i, j such that $(X[i], Y[i])$ and $(X[j], Y[j])$ are the two closest points.