Algorithm Course
Assignment 2

Due Date: 27/03/2022

1. Consider an $n \times n$ matrix M where every row and column is sorted in increasing order. Given a number x, design an algorithm to find the position of x in M, which runs in:
   (a) $O(n \log n)$ time.
   (b) $O(n)$ time.
2. Given an array A of length n such that each element of A is in range $[0, n^3 - 1]$, design a $O(n)$ time sorting algorithm for A.
3. Write a pseudocode for *Heap sort* along with its proof of correctness.
4. Prove that the worst case (and also the average case) time complexity of binary search is optimal, that is, given a sorted array of integers A and another integer x, in order to test whether $x \in A$ via comparisons, $\Omega(\log n)$ comparisons are necessary.
5. Given an array-based min-heap and a number x, give an algorithm with $O(k)$ worst-case running time that determines whether the k-th smallest element in the heap is greater than x.
6. Prove that $\lceil \frac{3n}{2} \rceil - 2$ comparisons are necessary and sufficient to find both the maximum and the minimum elements of an array of n elements.