1. The input is a set S with n real numbers. Design an $O(n)$ time algorithm to find a number that is not in the set. Prove that $\Omega(n)$ is a lower bound on the number of steps required to solve this problem.

2. Given two sets S_1 and S_2 and a real number x, find whether there exists an element from S_1 and an element from S_2 whose sum is exactly x. The algorithm should run in time $O(n \log n)$, where n is the total number elements in both sets.

3. The input is a sequence of n integers with many duplications, such that the number of distinct intergers in the sequence in $O(\log n)$.

   - Design a sorting algorithm to sort such sequences using at most $O(n \log \log n)$ comparisons in the worst case.
   - Why is the lower bound of $\Omega(n \log n)$ not satisfied in this case?