1. Describe an algorithm that, given \( n \) integers in the range 1 to \( k \), preprocesses its input in \( O(n+k) \) time and then answers any query about how many of the \( n \) integers fall into a range \([a, b]\) in \( O(1) \) time.

2. Suppose that we have an array of \( n \) data records to sort and that the key of each record has the value 0 or 1. Give a simple, linear-time algorithm for sorting the \( n \) data records in place. Use no storage of more than constant size in addition to the storage provided by the array. Can your sort be used to radix sort \( n \) records with \( b \)-bit keys in \( O(bn) \) time?

3. Show that there is no comparison sort whose running time is linear for at least half of the \( n! \) inputs of length \( n \). What about a fraction of \( 1/n \) of the inputs of length \( n \)? What about a fraction \( 1/2^n \)?

4. Show a lower bound on the running time of an algorithm merging two sorted lists containing \( n \) elements each.
   (a) Show that there are \((2n)^n\) ways to divide \( 2n \) numbers into two sorted lists of length \( n \).
   (b) Using a decision tree show that any algorithm merging two sorted lists makes \( 2n - o(n) \) comparisons.
   (c) Show that if two elements consecutive in the merged list come from two different input lists, then the algorithm had to compare them.
   (d) Show that any merge algorithm has to make at least \( 2n - 1 \) comparisons in the worst case.

5. (Exam 2019) Consider the following sorting algorithm for an array \( T \) of size \( n \):
   — If \( n = 2 \) and \( T[0] > T[1] \), swap \( T[0] \) and \( T[1] \)
   — If \( n > 3 \):
     — Recursively sort the first \( \lceil \frac{2}{3} n \rceil \) elements of \( T \)
     — Recursively sort the last \( \lceil \frac{2}{3} n \rceil \) elements of \( T \)
     — Recursively sort the first \( \lceil \frac{2}{3} n \rceil \) elements of \( T \)
   (a) Show that the algorithm is a correct sorting algorithm.
   (b) Estimate the worst-case complexity of the algorithm.

6. In an ordered array of odd length, the median is the middle value. If the length of the array is even, the median is the mean of the two middle values. Design a data structure that supports the following two operations: \( \text{add}(x) \) (add \( x \) to the data structure) and median (returns the median of all elements so far).

Extra questions

7. (Exam 2019) Given an integer array \( A \) of length \( n \), where each entry has a color, and an integer \( 1 \leq k \leq n \), develop a \( O(n) \)-time algorithm to compute the maximum of the minimum values in all intervals of \( A \) containing cells of at least \( k \) different colors.

8. Show that \( \lceil 3n/2 \rceil - 2 \) comparisons are necessary in the worst case to find both the maximum and minimum of \( n \) numbers.
9. Let \( m \) be an integer, and \( 2 = p_1 < p_2 < \ldots < p_m \) be the first \( m \) prime numbers. As a reminder, \( p_m \approx m \ln m \). We say that an integer \( y \) is \( m \)-\textit{friable}, if any its prime divisor belongs to \( \{p_1, p_2, \ldots, p_m\} \), in other words if there exists a tuple \((e_1, e_2, \ldots, e_m)\) such that \( e_i \geq 0 \) are integers and \( y = p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m} \). Your task is to study an algorithm for generating friable numbers which constitutes the basis of one of the most efficient methods of integer factorization.

(a) Let \( m \) and \( r \) be two integers. Show that the number of different integer tuples \((e_1, e_2, \ldots, e_m)\) such that \( e_1 + e_2 + \ldots + e_m \leq r \) is bounded by \( \binom{m+r}{r} \).

(b) Suppose that \( r \) is even. Let \((e_1, e_2, \ldots, e_m)\) be an integer tuple such that \( e_1 + e_2 + \ldots + e_m \leq r \). Show that the number of ways to represent this tuple as a pointwise sum of two tuples \((e'_1, e'_2, \ldots, e'_m)\) and \((e''_1, e''_2, \ldots, e''_m)\) where both \( e'_1 + e'_2 + \ldots + e'_m \leq r/2 \) and \( e''_1 + e''_2 + \ldots + e''_m \leq r/2 \) is at most \( \binom{r}{r/2} \).

(c) Suppose that \( N > 2p_m \) is odd, and let \( r = \lfloor \ln N / \ln p_m \rfloor \). Show that the probability that a number chosen from \([1, N]\) uniformly at random is \( m \)-friable is at least \( (m^r/N!r!) \).

(d) Let \( N = q_1^{e_1} q_2^{e_2} \cdots q_d^{e_d} \) be its prime number decomposition and \( s = 2 \lfloor \ln N / \ln p_m \rfloor \). Show that \( Y \in [0, N] \) is a square modulo \( N \) (that is, \( Y = Z^2 \mod N \)) if all Legendre symbols \((Y|q_j)\) are equal to 1 for \( j = 1, 2, \ldots, d \). Recall that the Legendre symbol is a multiplicative function defined as follows:

\[
(Y|q_j) = \begin{cases} 
0 & \text{if } Y = 0 \mod q_j, \\
1 & \text{if } Y \text{ is a quadratic residue modulo } q_j, \\
-1 & \text{otherwise.} 
\end{cases}
\]

Show that such a number has exactly \( 2^d \) quadratic roots.

(e) We associate with every integer tuple \((e_1, e_2, \ldots, e_m)\) such that \( e_1 + e_2 + \ldots + e_m \leq s/2 \), an integer \( U(e) = p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m} \) and a tuple \( \lambda(e) = ((U(e)|q_1), \ldots, (U(e)|q_d)) \). Show that if \( e' \) and \( e'' \) are two integer tuples such that \( \lambda(e') = \lambda(e'') \), then \( U(e') \cdot U(e'') \) is a square modulo \( N \).

(f) For a sequence \( a \in \{-1, +1\}^d \), let \( n_a \) be the number of exponents \((e_1, e_2, \ldots, e_m)\) such that \( e_1 + e_2 + \ldots + e_m \leq s/2 \) and \( \lambda(e) = a \). Show that the number of integers \( X \in [0, N] \) such that \( X^2 \mod N \) is \( m \)-friable is at least

\[
\frac{2^d}{(s/2)} \sum_{a \in \{-1, +1\}^d} (n_a)^2
\]

(g) Hence show that the probability that an integer \( X \) chosen uniformly at random gives a square \( X^2 \mod N \) which is \( m \)-friable is at least \( m^s / N!s! \).

(h) Explain how to test whether \( X^2 \mod N \) is \( m \)-friable in \( O(m(\ln N)^2) \) time.

(i) Choose \( m \) so that \( \ln m = \sqrt{\ln N \ln \ln N} \). Show that \( N!s^s / m^s \) and \( m(\ln N)^2 \) are bounded by a function in \( \exp(O(\sqrt{\ln N \ln \ln N})) \). Hence show an upper bound on the worst-case expected time complexity of an algorithm that chooses \( X \in [0, N] \) uniformly at random, tests whether \( X^2 \mod N \) is \( m \)-friable, and stops when it finds such a number.