1. Let $L_1, \ldots, L_k$ be nonempty lists of integers in the range 1 to $n$, and let $m := |L_1| + \cdots + |L_k|$. Show how to sort all of the $L_i$s (individually) in time $O(m + n)$.

2. Consider maintaining a database for a biologist storing results concerning plant experiments. Each experiment records a distinct ID, as well as a plant weight and a plant height. Experiments may be inserted and deleted. In addition, we would like to support the following queries:

   (i) for plants with heights in a given range $[h_1, h_2]$, what is the average weight?
   (ii) for plants with weights in a given range $[w_1, w_2]$, what is the average height?

   Show how to support all of these operations in time $O(\log n)$, where $n$ is the size of the database.

3. Consider maintaining a collection of lists of items on which the following operations can be performed:

   (i) Given two lists $L_1$ and $L_2$, form their concatenation $L$ (destroying $L_1$ and $L_2$ in the process).
   (ii) Given a list $L$ and a positive integer $k$, split $L$ into two lists $L_1$ and $L_2$, where $L_1$ consists of the first $k$ items of $L$, and $L_2$ the rest ($L$ is destroyed in the process).
   (iii) Report the first item in given list $L$.
   (iv) Create a new list with one item.

   (a) Describe data structures and algorithms supporting these operations so that operations (i)–(iii) can be performed in time $O(\log n)$ (where $n$ is the length of $L$), and operation (iv) takes constant time.

   (b) Suppose we also want to an operation that reverses a given list $L$. Show how this operation can be implemented in constant time, while the other operations can still be performed within the time bounds of part (a).

4. Suppose we have an abstract data type that represents sets of items. You are allowed to perform two types of operations: $\text{size}(S)$ returns the size $|S|$ of the set $S$, and $\text{union}(S_1, S_2)$ returns the union $S_1 \cup S_2$ of the sets $S_1$ and $S_2$. Each operation has a cost: the cost of $\text{size}(S)$ is zero, and the cost of $\text{union}(S_1, S_2)$ is $|S_1| + |S_2|$.

   You are given as input $n$ disjoint sets of items, $S_1, \ldots, S_n$. Design an algorithm that computes their union, $S_1 \cup \cdots \cup S_n$, using minimal cost. Not counting the time to perform the set operations, your algorithm should run in time $O(n \log n)$.

5. Suppose we have an abstract data type that represents lists of items. You are allowed to perform two types of operations: $\text{length}(L)$ returns the length $|L|$ of the list $L$, and $\text{concat}(L_1, L_2)$ returns the concatenation $L_1 \parallel L_2$ of the lists $L_1$ and $L_2$. Each operation has a cost: the cost of $\text{length}(L)$ is zero, and the cost of $\text{concat}(L_1, L_2)$ is $|L_1| + |L_2|$.

   You are given as input $n$ lists of items, $L_1, \ldots, L_n$. Design an algorithm that computes their concatenation, $L_1 \parallel \cdots \parallel L_n$, using minimal cost. Not counting the time to perform the list operations, your algorithm should run in time $O(n^3)$.

6. Design an efficient algorithm for the following problem. Given bit strings $u_1, \ldots, u_n$, $u'_1, \ldots, u'_n$, and $t$, determine if $t$ can be written

   $$ t = z_1 \parallel \cdots \parallel z_k \parallel z'_1 \parallel \cdots \parallel z'_k, $$

   where for each $i \in \{1, \ldots, k\}$, there exists a $j \in \{1, \ldots, n\}$ such that $z_i = u_j$ and $z'_i = u'_j$.

7. Problem 16-1 of CLRS (p. 402).