Let $S_1, S_2, ..., S_l$ be $l$ sets of integers such that $n = \sum_{i=1}^{l} |S_i|$. We want to support the following query:

- $q(i, j)$: report all the numbers in $S_i \cap S_j$.

For example, suppose that we are given $l = 4$ sets $S_1 = \{1, 4, 7, 11\}$, $S_2 = \{2, 4, 6, 8, 10\}$, $S_3 = \{3, 6, 9, 12\}$ and $S_4 = \{1, 4, 6, 8\}$. For query $q(2, 4)$, the answer should be $\{4, 6, 8\}$.

**Problem 1** (20%). Design a data structure with the following guarantees:

- The space consumption is $O(n)$.
- The preprocessing time is $O(n \log n)$.
- The query time for each $q(i, j)$ must be $O(\min\{|S_i|, |S_j|\} \cdot \log n)$.

**Answer.** In the preprocessing phase, for each $S_i$ ($i \in [1, n]$), store the numbers in $S_i$ in an array $A_i$ and sort the array in $O(|S_i| \cdot \log |S_i|)$ time. The total space is $O(\sum_{i=1}^{l} |S_i|) = O(n)$ and preprocessing time is $O(\sum_{i=1}^{l} |S_i| \log |S_i|) = O(n \log n)$.

Given a query $q(i, j)$, let us assume (without loss of generality) that $|S_i| \leq |S_j|$. Enumerate each number in $x \in S_i$ and use binary search to check if $x \in S_j$. The query time is therefore $O(|S_i| \cdot \log |S_j|) = O(\min\{|S_i|, |S_j|\} \cdot \log n)$.

**Problem 2** (20%). Design a data structure with the following guarantees:

- The space consumption and preprocessing time are both $O(n)$.
- The query time for each $q(i, j)$ must be $O(\min\{|S_i|, |S_j|\})$ in expectation.

**Answer.** In the preprocessing phase, for each $S_i$ ($i \in [1, n]$), create a hash table $H_i$ on $S_i$. The total space and preprocessing time are both $O(\sum_{i=1}^{l} |S_i|) = O(n)$.

Given a query $q(i, j)$, assume (without loss of generality) that $|S_i| \leq |S_j|$. Enumerate each number in $x \in S_i$ and use the hash table on $S_j$ to check if $x \in S_j$. The query time is therefore $O(|S_i|) = O(\min\{|S_i|, |S_j|\})$ in expectation.