1. **Social network connectivity.** Given a social network containing nnn members and a log file containing mmm timestamps at which times pairs of members formed friendships, design an algorithm to determine the earliest time at which all members are connected (i.e., every member is a friend of a friend of a friend ... of a friend). Assume that the log file is sorted by timestamp and that friendship is an equivalence relation. The running time of your algorithm should be mlog⁡nm \log nmlogn or better and use extra space proportional to nnn.

Note: these interview questions are ungraded and purely for your own enrichment. To get a hint, submit a solution.

1. **Union-find with specific canonical element.** Add a method find() to the union-find data type so that find(i) returns the largest element in the connected component containing iii. The operations, union(), connected(), and find() should all take logarithmic time or better.

For example, if one of the connected components is {1,2,6,9}\{1, 2, 6, 9\}{1,2,6,9}, then the find() method should return 999 for each of the four elements in the connected components.

1. **Successor with delete**. Given a set of nnn integers S={0,1,...,n−1} S = \{ 0, 1, ... , n-1 \}S={0,1,...,n−1} and a sequence of requests of the following form:

* Remove xxx from SSS
* Find the *successor* of xxx: the smallest yyy in SSS such that y≥xy \ge xy≥x.

design a data type so that all operations (except construction) take logarithmic time or better in the worst case.