1. Finding a redundant edge

Input: A connected, undirected graph G.
Output: An edge which can be removed from $G = (V, E)$ while still leaving G connected, or `null` if no such edge exists.

Find an $O(n)$ algorithm to solve this problem, where $n = |V|$. Recall that $|E|$ can be $\Omega(n^2)$.

2. Problem 17.2: Dynamic binary search

.

3. Disjoint sets

a) Prove by induction that a node of rank k in a disjoint-set data structure must be the root of a tree containing at least 2^k nodes.

b) Consider maintaining the disjoint-set data structure without path compression; that is, we do not change all the parent pointers along the search path to point to the root during a FIND operation. Prove that any m operations including at most n MAKE-SET operations requires time $O(m \lg n)$.

4. Recurrence relation

Get the best asymptotic upper bound you can for the recurrence relation:

$$T(n) = T(\lg n) + 1$$