Incremental connectivity is the problem of maintaining an undirected graph under edge insertions and connectivity queries (no deletions). This is essentially equivalent to the union-find problem. This problem asks to maintain a forest of rooted trees, under the following operations:

**UNION**\((a, b)**: assume \(a\) is the root of a tree, and \(b\) lies in a different tree; this operation creates an edge from \(a\) to \(b\), so that \(a\) is no longer a root.

**FIND**\((a)**: return the root of the tree containing \(a\).

Incremental connectivity is easy to implement:

**CONNECTED**\((u, v)**: answer true iff **FIND**\((u) = \text{FIND}(v)**.

**INSERT**\((u, v)**: if **CONNECTED**\((u, v)**, ignore this edge. Otherwise, run **UNION**\((\text{FIND}(u), v)**.

You should know that union-find can be solved in a running time given by the inverse Ackerman function, which is very-very close to constant. However, this running time is only amortized.

**Prove the following**: For any given \(b\) satisfying \(b = \Omega(\log_b n)\), one can support **UNION** in \(O(b)\) time, and **FIND** in \(O(\log_b n)\) time, where both running times are worst-case.

It is known that this tradeoff is tight, so there is a very interesting separation between what is possible with and without amortization.