21-3 Tarjan's off-line least-common-ancestors algorithm

The ***least common ancestor*** of two nodes u and v in a rooted tree T is the node w that is an ancestor of both u and v and that has the greatest depth in T. In the ***off-line least-common-ancestors problem***, we are given a rooted tree T and an arbitrary set $P = \{\{u, v\}\}$ of unordered pairs of nodes in T, and we wish to determine the least common ancestor of each pair in P.

To solve the off-line least-common-ancestors problem, the following procedure performs a tree walk of T with the initial call LCA(*root*[T]). Each node is assumed to be colored WHITE prior to the walk.

```
LCA(u)
 1  MAKE-SET(u)
 2  ancestor[FIND-SET(u)] ← u
 3  for each child v of u in T
 4       do LCA(v)
 5          UNION(u, v)
 6          ancestor[FIND-SET(u)] ← u
 7  color[u] ← BLACK
 8  for each node v such that {u, v} ∈ P
 9       do if color[v] = BLACK
10            then print "The least common ancestor of"
                        u "and" v "is" ancestor[FIND-SET(v)]
```

a. Argue that line 10 is executed exactly once for each pair $\{u, v\} \in P$.

b. Argue that at the time of the call LCA(u), the number of sets in the disjoint-set data structure is equal to the depth of u in T.

c. Prove that LCA correctly prints the least common ancestor of u and v for each pair $\{u, v\} \in P$.

d. Analyze the running time of LCA, assuming that we use the implementation of the disjoint-set data structure in Section 21.3.