Assignment 4 CSC263
Due: Nov 18, 2008

1. Consider an implementation of the disjoint-sets ADT using rooted trees, where each node contains one member and each tree represents one set. As discussed in class, each node points only to its parent. Moreover, the root of a tree is the set representative and is its own parent.

   MAKE-SET(x) simply creates a tree with one node for x. FIND-SET(x) follows parent pointers until the root of the tree containing x is found. UNION(x, y) first finds the roots r_x and r_y of x's and y's trees, respectively, and then sets $\texttt{parent}(r_y) \leftarrow r_x$.

   (a) Assume we have performed n MAKE-SET operations and no other operations. Give a sequence of $n-1$ UNION operations that cumulatively take time $\Omega(n^2)$. Justify your answer.

   (b) Assume we have performed n MAKE-SET operations and no other operations. Give a sequence consisting first of $n-1$ UNION operations and then n *unique* FIND-SET operations (i.e., each element is searched for exactly once) such that (a) the UNION operations cumulatively take time $O(n)$ and (b) the FIND-SET operations cumulatively take time $\Omega(n^2)$. Justify your answer.

   Now modify our data structure to keep track of each tree's height. In addition, modify UNION(x, y) so that when linking we make the shorter tree a child of the taller tree's root. **Note:** (a) a singleton has height 0; (b) if trees of different heights are linked, the resulting tree's height is equal to the taller tree's height; and (c) if trees of the same height are linked, the resulting tree has height one larger.

   (c) Prove that in any sequence of m MAKE-SET, FIND-SET and UNION operations (of which n are MAKE-SET operations) no tree ever has height more than $\log_2 n + 1$. Conclude that any such sequence can be performed in cumulative time $O(m \log n)$.

   (d) Prove that the worst-case cumulative time complexity of a sequence of m MAKE-SET, FIND-SET and UNION operations (of which n are MAKE-SETs) is $\Omega(m \log n)$.

2. Consider the abstract data type DEPR that consists of a set S of positive integers upon which the following two operations can be performed:

   DELETE(S, i): Delete integer i from the set S. If $i \notin S$, there is no effect.

   PREDECESSOR(S, i): Return the predecessor in S of integer i, i.e., $\max\{j \in S \mid j < i\}$.
   If i has no predecessor in S, i.e., if $i \leq \min S$, then return 0. Note that it is not necessary for i to be in S.

   Initially, S is a set of n consecutive integers. Describe a data structure with $O(m \cdot \alpha(m, n))$ cumulative time complexity, where m is the number of operations that are performed. Justify the correctness and time complexity of your data structure.

   How do you initialize your data structure and how much time does the initialization take?

3. Recall that a *simple cycle* in an undirected graph $G = (V, E)$ is a sequence of distinct vertices $v_0, v_1, \ldots, v_{k-1}$, where $k \geq 3$ and $\{v_0, v_1\} \in E$, $\{v_1, v_2\} \in E$, $\ldots$, $\{v_{k-1}, v_0\} \in E$. An undirected graph G is *acyclic* (i.e., it contains no simple cycle) if and only if G is a *forest* (i.e., G consists of one or more tree(s)).

   (a) Explain how to use depth first search to determine in $O(|V|)$ worst case time whether a given undirected graph $G = (V, E)$ contains a cycle.

   (b) Prove that your algorithm is correct and runs in the required time.

   (c) For (a) and (b), does it matter how the input graph is represented? Explain.