(b) (10 points) Show how to implement `MAXAREA(D,p)`. What is its worst-case running time? Explain.

cont'd...

(c) (4 points) Show how to maintain the new information during a rotation. Why does the rotation itself take time $O(1)$?

(d) (4 points) Show how to maintain the new information during one of the following operations: `INSERT` or `DELETE` (without fix-up). Why does this operation take time $O(\log n)$?

cont'd...

(5) A *treap* is a binary tree where each node x has a priority $priority(x)$ and a key $key(x)$. The tree satisfies the BST property with respect to the keys, and the heap property with respect to the priorities. The tree does not have to be balanced or complete.

(a) (3 points) Draw a treap containing the following nodes (nodes are given by $(key, priority)$ pairs): $(1, 99)$, $(2, 70)$, $(3, 75)$, $(4, 10)$, $(5, 25)$, and $(6, 20)$.

cont'd...

(b) (7 points) Give the pseudocode for an algorithm to make a treap out of a set of nodes $x_1, \ldots, x_n$. Assume each node has fields $key(x_i)$, $priority(x_i)$, $left(x_i)$, $right(x_i)$, $parent(x_i)$. The nodes are given in no particular order.

(c) (3 points) In the worst case, what is the height of a treap on n nodes? Explain.

cont'd...

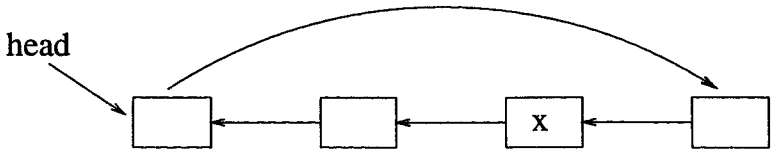
(6) Consider a linked-list with weighted union implementation of a Union-Find ADT. Assume we have an unlimited set of items $x_1, x_2, \ldots$, but no sets to begin with. The operations have the following costs (running times): `MAKE-SET(x)`: 1, `FIND-SET(x)`: 1, `UNION(x,y)`: $\min\{length(list_x), length(list_y)\}$.

(a) (4 points) Show a sequence of m operations that has cost $\Omega(m \log m)$. Explain.

(b) (6 points) Show that the worst-case sequence complexity of m operations is $O(m \log m)$.

cont'd...

(c) (6 points) We will now change the implementation of the lists. In general, an element x in a list will now point to the element directly in front of it instead of to the head of the list (see picture). The head of the list will still serve as the representative of that set.



For each of the operations, describe briefly how to implement it under the new system and state its running time (in Θ-notation) in terms of the number of pointers followed or assigned:

* `MAKE-SET(x)`:

* `FIND-SET(x)`:

* `UNION(x,y)`:

cont'd...

(d) (4 points) What is the worst-case sequence complexity of m operations now? Explain.

This page for scratch work.

This page for scratch work.

This page for scratch work.