Problem Set 4
(due Monday, November 9)

**Problem 1** Design a data structure to support the following operations on a set of items:

1. Insert(x): Insert the item x.
2. Delete: Delete the item that has been in the set the longest.
3. Find-Min: Return the smallest item.

The Find-Min operation should run in worst-case time \(O(1)\). The Insert and Delete operations should run in worst-case time \(O(\log n)\), where \(n\) is the number of items in the set.

**Problem 2** Design a data structure to support the following operations on a set of items:

1. Insert(x): Insert the item x.
2. Delete(k): Delete the \(k\)th smallest item.
3. Member(x): Test whether \(x\) is a member.

Each operation should run in worst-case time \(O(\log n)\).

**Problem 3** Design a data structure to support the following operations on a set of items:

1. Insert-Max(x): Insert the item \(x\), and delete all items bigger than \(x\).
2. Extract-Min: Return the smallest item, and delete it.

The Extract-Min operation should run in worst-case time \(O(1)\). The Insert-Max operation should run in amortized time \(O(1)\). Amortized time \(O(1)\) means that the first \(k\) Insert-Max operations use at most \(O(k)\) time in total.

**Problem 4** Suppose you are given an array \(A[1..n]\) that is constant (its values will not change). Design a data structure to support the following operation:

1. Find-Min(i, j): Return the smallest item in \(A[i..j]\).

The Find-Min operation should run in worst-case time \(O(\log n)\). Your data structure should use worst-case space \(O(n)\).

*Hint:* First aim for space \(O(n \log n)\).