1. Questions?

2. Skew-heap implementation (SkewHeap).
   (a) Notion: a merge of two heaps h1 and h2.
      i. If one heap is empty, use the other as the result.
      ii. Otherwise, assume the h1 root is smallest:
          iii. Case 1: If h1 has no left child: make h2 its left.
              iv. Case 2: Otherwise, swap the children of h1, and merge h2 with h1’s new left (former right).
   (b) Notice how the leftmost branch appears to be the target of all merges. But: at each stage, children are swapped/twisted. Result:
   (c) Has amortized logarithmic cost even though the tree is not necessarily very balanced. Very cool analysis based on some clever bookkeeping/accounting tricks.
   (d) getFirst: return root.
   (e) remove: return root after merging children.
   (f) add: merge new value with existing heap.

   (a) An implementation of an OrderedStructure: add, remove, get, contains, iterator.
   (b) Comparable values are kept in an (internal) binary tree.
   (c) All values to the left of the root are smaller or equal.
   (d) All values to the right of the root are larger.
   (e) We write a method locate that determines the correct location for the value in the tree. Locate can be used to determine if the tree contains a value, or to find the best location to insert it.
   (f) We have an important notion of the predecessor and successor of a node in a tree. The predecessor is the rightmost descendent of the left child. The successor is the opposite. Adding a right child to the predecessor installs a new predecessor, and vise versa for the left child of the successor.
   (g) removeTop is an important method that allows you to remove the top node of a (sub)tree. It has to be done with care. Several cases: study these.

Notes.

(h) The iterator is an inorderIterator on the root of the underlying binary tree.

(i) Rotations.
   i. Tree is balanced if, at each node, children have heights within 1.
   ii. Left- and right- rotations fix problems of balance (See Figure 4.4). Rotations can be seen as bringing a node higher in the tree.