Today: Review.
Next class: Midterm Exam 2.
Reminders: Use newsgroup regularly. Homework.

By design, there are more questions here than can be answered in class. **Choose** the ones you find most beneficial to your learning for in-class work and answer the **rest later** as practice. Use the newsgroup for clarifications.

1. Write your group members' names below. Underline your name.

2. In the context of the code in Figure 23.7 (p. 880) of the textbook:

   (a) Explain, as precisely as you can, the effect of changing line 14 (which begins the class definition):

```
1  public class PairingHeap<AnyType extends Comparable<? super
       AnyType>>
```

   with the following:

```
1  public class PairingHeap<AnyType extends Comparable<AnyType>>
```

   (b) Explain the purpose of the interface `Position`. Suppose we wish to avoid using that interface. What are the smallest (fewest, least significant) changes to the code we can make for that purpose? Explain your answer.

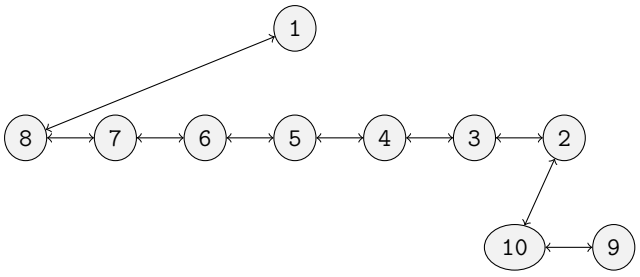
3. Recall the triple-based representation of binary trees: We represent the empty binary tree by $\perp$ and a nonempty binary tree with root label n, left subtree l, and right subtree r by the triple (n, l, r). Using this notation, define functions on binary trees that correspond to each of the following:

   (a) zig-zig

   (b) zig-zag

   (c) skew-heap merge

4. Is the following a concrete representation of a valid pairing heap (tree)? If not, make the fewest changes that yield a valid one. In either case, depict the abstract tree corresponding to this concrete one.



5. Provide a sequence of pairing-heap operations that, starting from the empty heap, yields the heap of Question 4 (or prove that no such sequence exists). Justify your answer.

6. Trace the result of the following operations, in given order, on the pairing heap of Question 4: `insert(11), deleteMin(), deleteMin(), insert(1), deleteMin`.

[additional space for answering the earlier question]

7. Trace the result of inserting the keys $1, 2, 3, \ldots, 7$ into an initially empty bottom-up splay tree.

8. Repeat Question 7 for a top-down splay tree.

9. Depict the action of heapsort on the following array, depicting both the states of both the array the implicit tree after the *buildHeap* operation and after each *deleteMax* operation.

   `54 19 61 91 53 15 47 29 48 60`

10. Repeat Question 9 using a 2-way merge sort.

    `54 19 61 91 53 15 47 29 48 60`