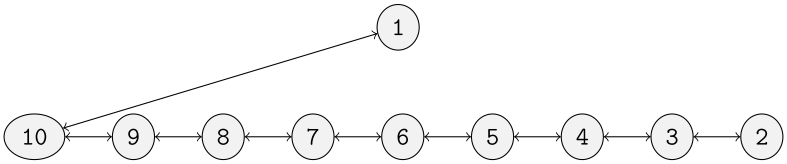
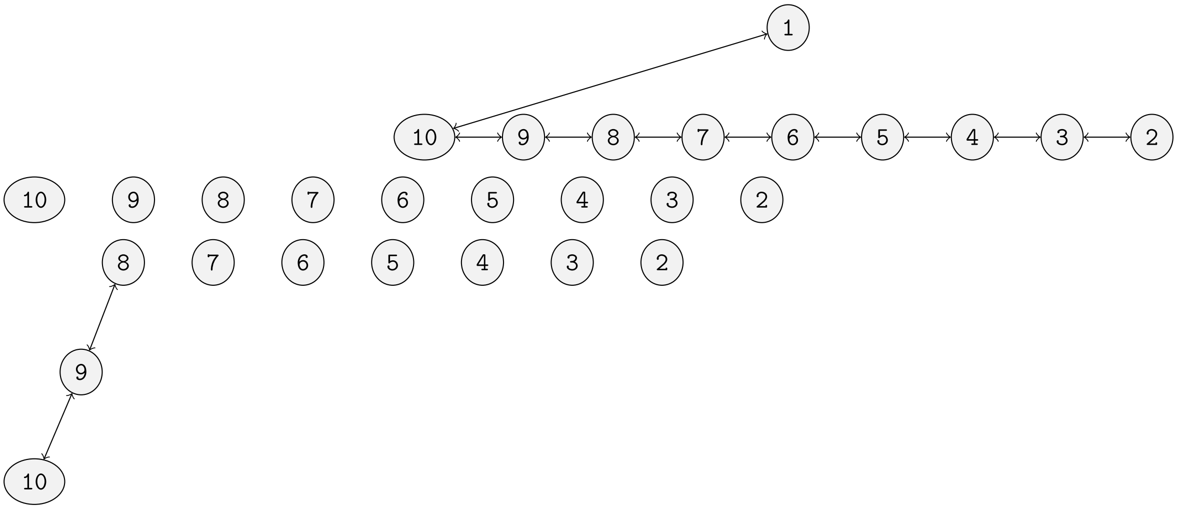
Today: Pairing heap. § 23.2.
Next class: Graphs and paths. §§ 14.{0,1,2,3}. *(Differs from syllabus schedule.)*
Reminders: Portfolio work; newsgroup; homework.

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

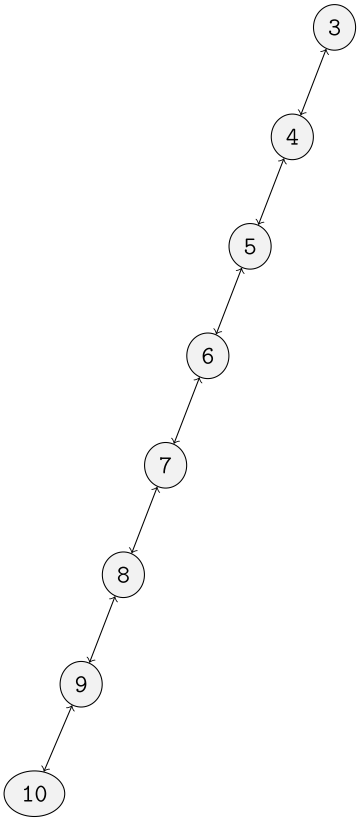
2. In the context of pairing heaps, consider the *concrete tree* depicted below.
   (a) Explicitly depict the null nodes.
   (b) Use dashed lines to depict the *abstract tree* corresponding to this concrete tree.
   (c) Outline an appropriate Java class for the nodes, paying special attention to data members.
   (d) Outline the Java object structure corresponding to the concrete tree and the above class, using arrows for references.

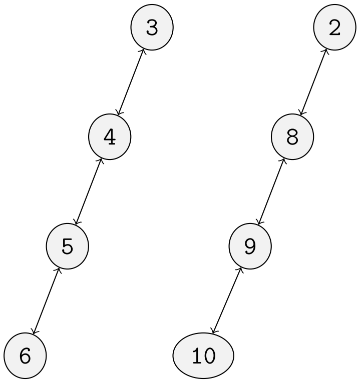


3. Consider an initially empty structure similar to a pairing heap, but that is maintained using a simple one-pass linking strategy in which subtrees are merged one at a time in left-to-right order.
    (a) Trace the insertion of the keys $1, 2, \ldots, 10$ into this heap, depicting the intermediate trees after 2 and 5 insertions.
    (b) Then trace two *deleteMin* operations.
    (c) Then trace one *decreaseKey* operation that changes the key 7 to 2.



[additional space for answering the earlier question]





4. Repeat Question 3 using a two-pass linking strategy that merges pairs of subtrees left to right in the first pass and then merges the merged pairs also in left-to-right order in the second pass. (In the second pass, we proceed left-to-right, merging the result of the previous merges in this pass with the next subtree.)

5. Repeat Question 4 using a right-to-left second pass. Explain any differences between this strategy and that of the textbook.