Name: ____

1. (1 pt.)
   - **Read all material carefully.**
   - *If in doubt whether something is allowed, ask, don't assume.*
   - You may refer to your books, papers, and notes during this test.
   - E-books may be used subject to the restrictions noted in class.
   - No computer or network access of any kind is allowed (or needed).
   - Write, and draw, carefully. Ambiguous or cryptic answers receive zero credit.
   - Use class and textbook conventions for notation, algorithmic options, etc.
   - There is an extra-credit question (marked with ⋆). It is harder than the rest.

   Write your name and group ID (e.g., C3) in the space provided above. The group is for reference only; all work on this quiz is individual work.

2. (14 pts.) Trace the action of *insertion sort* on the following array: Depict the state of the array after each iteration of the outer loop, underlining the elements that moved in that iteration.

   ```
   73 53 87 95 71 75 83 35 91 45
   ```

3. (15 pts.) Trace the action of *quicksort* on the data of Question 3, using the first element of each sub-array as the pivot (N.B. for toy use only; not a good implementation choice).

   For each recursive invocation of quicksort, clearly indicate:

   (a) the sub-array,
   (b) the pivot, and
   (c) the result of partitioning on that pivot.

   ```
   73 53 87 95 71 75 83 35 91 45
   ```

[additional space for answering the earlier question]

4. (20 pts.) Consider an initially empty *pairing heap* (organized as a min-heap).
   (a) Trace the insertion of the following keys into this heap. Depict the intermediate heaps after the second and fifth insertions (at least).
   (b) Then trace two *deleteMin* operations. Depict the state of the heap after each left-to-right and right-to-left pass.
   (c) Then trace one *decreaseKey* operation that changes the key 95 to 25.

   Use the *abstract* form of pairing heaps for all depictions here (cf. textbook Fig. 23.4).

   ```
   73 53 87 95 71 75 83 35 91 45
   ```

[additional space for answering the earlier question]

5. (10 pts.) Depict the result of inserting the following keys, in the listed order, into an initially empty *bottom-up red-black tree.*

   Depict the state of the tree after each insertion.

   Use our usual convention: red nodes are round, black nodes are boxed.

   `73 53 87 95 71 75 83 35 91 45`

6. ⋆ (10 pts.) (Difficult; **do not attempt unless** done with earlier questions.)

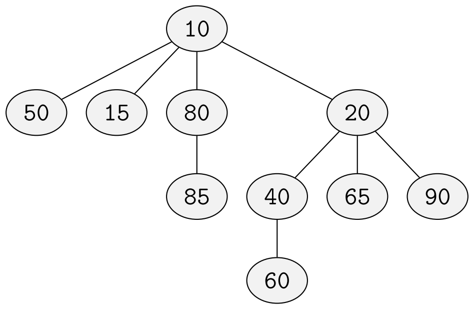
   Is there a sequence of operations (excluding explicit merges; see clarification below) that when applied to an empty pairing heap, generates the *pairing heap* (min-heap) depicted below in the *abstract* form (cf. textbook Fig. 23.4)?

   - If so, provide a shortest sequence that generates the heap. Justify your answer by
     (a) outlining the effect of the operations in sufficient detail and
     (b) explaining why no shorter sequence exists.
   - Otherwise, explain precisely why no such sequence exists.

   There is no credit for answers without proper explanations.

   Clarification: The sequence of operations is not permitted to explicitly merge the pairing heap being operated on with another one built separately. However, merges that occur as a side effect of other operations (insertion, delete-min, decrease-key) are permitted.

   Reminder: Follow the textbook's algorithms exactly. For instance, new children are added to the left, not right, of existing ones.



[additional space for answering the earlier question]