Name: ____

1. (1 pt.)

   ☐ **Read all material carefully.**
   ☐ Budget your time: 60 minutes, 60 pts ⇒ 1 min./pt. avg.
   ☐ You may refer to your books, papers, and notes during this test.
   ☐ E-book use is permitted only under the specific conditions announced 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 one extra-credit question at the end, marked with a ⋆. It is harder, and graded more strictly, than the rest.

   Read the above carefully; check each box; then write your name in the space provided above.

Remaining questions begin overleaf.

2. (9 pts.) In the context of the textbook's implementation of the `add` operation for **binary heaps**, consider the following code from Figure 21.9 (p. 815):

```
/**
 * Adds an item to this PriorityQueue.
 * @param x any object.
 * @return true.
 */
public boolean add( AnyType x )
{
    if( currentSize + 1 == array.length )
        doubleArray( );

    // Percolate up
    int hole = ++currentSize;
    array[ 0 ] = x;

    for( ; compare( x, array[ hole / 2 ] ) < 0; hole /= 2 )
        array[ hole ] = array[ hole / 2 ];
    array[ hole ] = x;

    return true;
}
```

What, if any, changes must be made to the code to ensure correctness if the implementation is modified to **not use the sentinel element** at array-index 0 (so that n items are stored in array positions 0 through $n-1$ instead of 1 through n). *Explain your answer.*

3. (10 pts.)

   (a) Trace the insertion of the following keys, in given order, into an initially empty **skew heap**. Depict the state of the heap at least after all actions for each insertion have completed.

   (b) On the final heap above, trace a *decreaseKey* operation that modifies the key 23 to 2.

   (c) Trace two consecutive *deleteMin* operations applied to the final heap above.

23	92	82	3	60	52	47	24	88	43

[additional space for answering the earlier question]

4. (10 pts.) Repeat all parts of Question 3 for a **pairing heap** instead of a skew heap.

   `23 92 82 3 60 52 47 24 88 43`

[additional space for answering the earlier question]

5. (10 pts.) Depict the action of **heapsort** on the following array, depicting the states of *both the array the implicit tree* after the *buildHeap* operation and after each *deleteMax* operation.

23	92	82	3	60	52	47	24	88	43

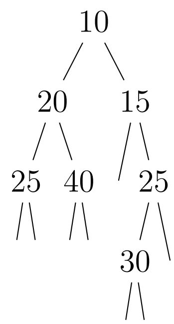
[additional space for answering the earlier question]

6. (10 pts.) Depict the insertion of the following keys, in given order, into an initially empty **bottom-up splay tree**. Depict the state of the tree at least after the completion of each insertion.

23	92	82	3	60	52	47	24	88	43

[additional space for answering the earlier question]

7. (10 pts.) Provide a **sequence of skew-heap** operations that yields the following tree when applied to an empty skew heap, and **trace** the action of the operations, or explain why no such sequence is possible.



[additional space for answering the earlier question]

8. $\boxed{\star}$ (10 $\star$ pts.) 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. *Explain your definitions briefly.*

(a) zig-zag

(b) skew-heap merge

[additional space for answering the earlier question]