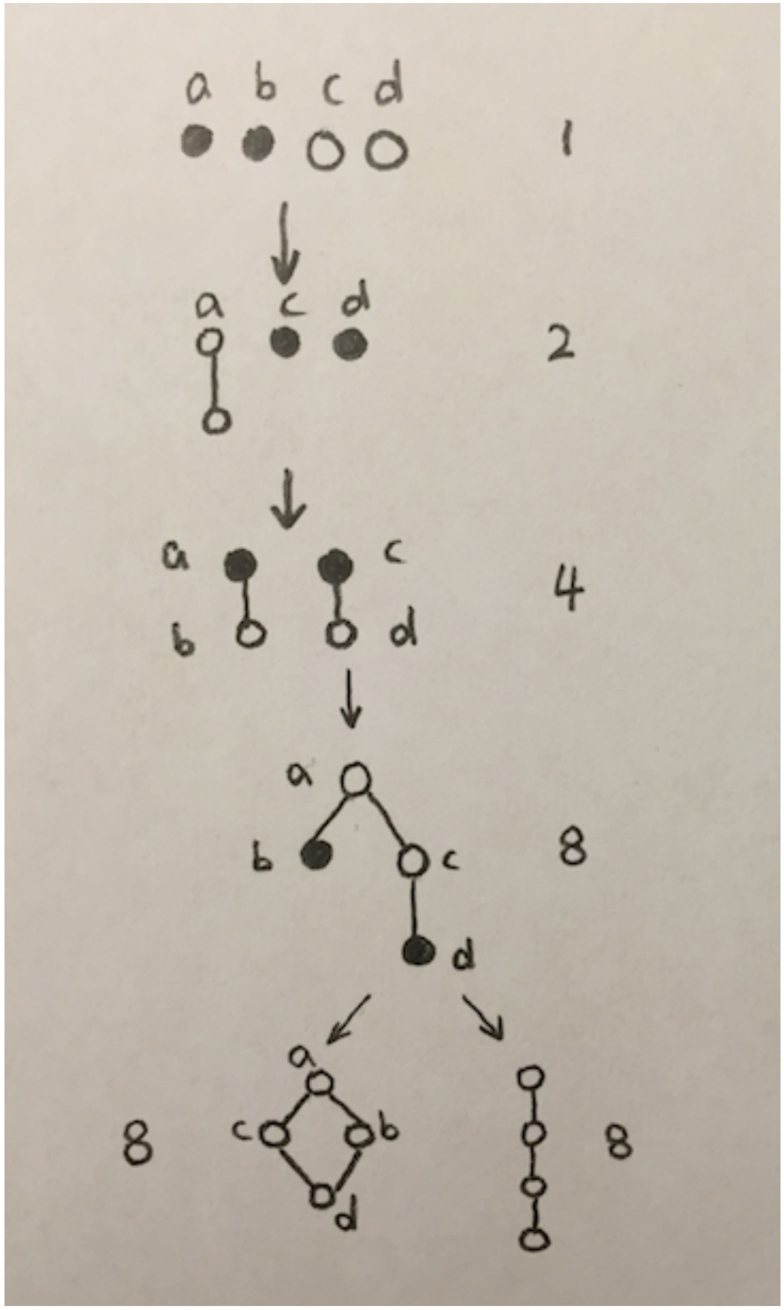
Written part:

1. In the style of the sorting hand-out, write out the tree of posets for the min-max problem when n = 4



2. Determine the smallest value of n for which mmA and mmB use a different number of comparisons.
   6 is the smallest value.

3. Write down a recurrence relation C(n) for the number of comparisons used by mmB. It can have several cases. Here's the recurrence relation for mmA:

$$C(n) = \begin{cases} 0 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \\ 2 + C(\lfloor n/2 \rfloor) + C(\lceil n/2 \rceil) & \text{if } n > 2 \end{cases}$$

$$\text{For mmB, } C(n) = \begin{cases} 0 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \\ 2 + C(\lfloor n/2 \rfloor + 1) + C(\lceil n/2 \rceil - 1) & \text{if } n\%4 = 2 \\ 2 + C(\lfloor n/2 \rfloor) + C(\lceil n/2 \rceil) & \text{if } n > 2\ \&\ n\%4 \neq 2 \end{cases}$$

Additional written part:
An inversion in a sequence is an out-of-order pair; i.e., $i < j$ but $a_i > a_j$, inversions are discussed briefly in the book on page 252. For example, the sequence (5,3,2,1,4) has 7 inversions.

1. What is the minimum number of inversions of a permutation of 1, 2…, n?
   The worst case is that there is no inversions in the sequence. Thus the result is 0.

2. What is the maximum number of inversions of a permutation of 1, 2…, n?
   The maximum number of inversions is [n(n-1)]/2 pairs in the sequence of inverted order.

3. Explain carefully how to use red-black trees to compute the number of inversions in a permutation time O(nlogn). Effectively, you may need to modify the code for Algorithm 3.4 on page 439. Explain in detail any changes that you would make to put.
   In order to get the number of inversion, we need to calculate the number of nodes in the right side. For the insertion process, every time when we insert a node, we need to check how many nodes in the right sides, then add the number on a global value (we can set it). That is because all the nodes' value on the right side is greater and earlier inserted than the node which is being inserted. It is the way that we can compute the number of inversions. The reason that a permutation time is O(nlogn): As every traversal is in the right side, which is logn, and will be n times traversal in the right side when there are n elements needed to be inserted. Thus, it is nlogn.