You are encouraged to discuss amongst yourselves, but please write your answers yourselves, based on your own understanding.

1. Prove the correctness of the standard Merge Sort algorithm. You may assume that you are given a $Merge()$ function that takes as input two sorted arrays, say A and B, and outputs a single array that contains all the elements of A and B in sorted order.
2. Suppose that $T(n) = T\left(\frac{c_2}{c_1}n\right) + T\left(\frac{c_3}{c_1}n\right) + \Theta(n)$, where c_1, c_2, c_3 are constants.
   (a) If $c_1 = c_2 + c_3$, show that $T(n) = \Theta(n \log n)$,
   (b) If $c_1 > c_2 + c_3$, show that $T(n) = \Theta(n)$.

   For each of these parts, draw the first few levels of the recursion tree and generalize it for the proof.
3. With reference to the algorithm for the "Closest Pair of Points" problem, answer the following. Proper justifications should be given, where applicable.
   (a) Was it necessary to filter out the vertices that were over d distance away from the partition line?
   (b) Show that it is possible that no point gets filtered out during the conquer part. In this case, can we conclude that the divide step was not needed?
   (c) Why was it *safe* to consider the horizontal shaded rectangular region instead of the vertical shaded rectangular region?
   (d) For the algorithm to run in $\mathcal{O}(n \log n)$ time, what extra information should be passed to each recursive call? Also, specify when this information would be used.
4. Based on the discussion in the lecture video on the Convex Hull problem, give an algorithm for the problem using the divide-and-conquer approach. Give all the details of the algorithm, prove its correctness and analyze its complexity.

   You may assume, without proof, that the elementary geometric operations can be performed in constant time.