1. All of the following questions are regarding the closest points in 2D algorithm.
   (a) What if, in the combine step, we looked at a region within $2\delta$ of the midline. Could the algorithm still work? Would you have to change anything else to compensate?
   (b) What if we looked at a region within $\delta/2$ of the midline. Could the algorithm still work? Would you have to change anything else to compensate?
   (c) Why do we need to maintain separate arrays sorted by $X$ and $Y$ coordinates?

2. (a) Prove the following algorithm is correct:

   Algorithm 1: Maximum($A$)
   
   Input : Array $A$ of unique integers of size $n$.
   Output: Maximum value in array.

   1 if $n$ equals 1 then
   2 return $A[1]$;
   3 end
   4 mid = $n/2$;
   5 $m_1 =$Maximum($A[1 : mid]$);
   6 $m_2 =$Maximum($A[mid + 1, n]$);
   7 return max{$m_1, m_2$};

   (b) What is the runtime of the algorithm?

3. Suppose you have a graph $T$ that is a binary tree, with weights on each vertex. Let $T_v$ be the subtree with root at vertex $v$. Let $S(T_v)$ be the max-weight-independent set of $T_v$ and let $W(T_v)$ be the weight of the max-weight independent set on $T_v$. We'll design a dynamic programming algorithm for this problem.

   (a) What are the options for $S(T_v)$ in terms of the vertex $v$?
   (b) For each option, write a recurrence relation for $S(T_v)$ in terms of max-weight-independent sets of subtrees of $T_v$.
   (c) Use this analysis describe in words (or write pseudocode) how to create a function that fills an array with the values $W(T_v)$ for each $v$.

4. Explain the idea of an exchange argument proof (the simple, single exchange version) in your own words.