1. Write a recursive, divide-and-conquer algorithm \( \text{Power}(a, n) \) that computes the number \( a^n \), where \( a, n \) are positive integers. Analyze your algorithm. Your algorithm must work in \( o(n) \) time.

2. Rewrite your algorithm \( \text{Power}(a, n) \) as a non-recursive one.

3. Consider the closest-pair algorithm. Suppose we do not sort \( A[i..j] \) by \( y \)-coordinate in \( \text{Closest-Pair}(A[i..j], (p, q), \text{ptr}) \), but instead we sort the whole set of \( n \) points (i.e., \( A[1..n] \)) by \( y \)-coordinate into a linked list in the beginning of the algorithm, immediately after sorting them by \( x \). (The procedure \( \text{Closest-Between-Two-Sets} \) remains intact, but the linked list pointed to by \( \text{ptr} \) now contains the entire set of \( n \) points.) Does the modified algorithm work correctly? Justify your answer. (If YES, explain why; if NO, give a counterexample.)

4. Whether the above algorithm is correct or not, what is its time complexity?