1. Provide algorithms that are sensitive to the relative sizes of \( n \) and \( m \).
   
   (a) [30 points] Let \( R_1 \) be a red-black tree with \( n \) elements. Let \( R_2 \) be a red-black tree with \( m \) elements. How fast can you construct a red-black tree containing all elements from \( R_1 \) and \( R_2 \)?
   
   (b) [70 points] Same as (a), but you are also given that all the values in \( R_1 \) are smaller than all the values in \( R_2 \).

2. Let \( S = \{s_1, \ldots, s_n\} \) be a list of real numbers.
   
   (a) [20 points] Show how to find the smallest value \( |s_i - s_j| \) \( (i \neq j) \) in \( O(n \log n) \) time.
   
   (b) [60 points] Suppose that there will be a series of insertions (into \( S \)) and queries such as the one in part (a). Show how to maintain a simple data structure that can answer a query in constant time, and takes at most logarithmic time to update after each insertion.
   
   (c) [20 points] Can your structure handle deletions easily? If yes, explain. If not, can you handle deletions by further extending your data structure?