1. Prove the following recurrence by one of the three methods (substitution, Induction, recursion tree) and show that T(n)= n lg n

$$T(n) = \begin{cases} 2 & \text{if } n = 2, \\ 2T(n/2) + n & \text{if } n = 2^k, \text{ for } k > 1 \end{cases}$$

2. Describe the characteristics,properties and benefits of splay trees.

3. We can extend our notation of asymptotic analysis to the case of two parameters n and m that can go to infinity independently at different rates. For a given function $g(n, m)$, we denote by $O(g(n, m))$ the set of functions $O(g(n, m)) = \{f(n, m)$: there exist positive constants c, $n0$, and $m0$ such that $0 \leq f(n, m) \leq cg(n,m)$ for all $n \geq n0$ and $m \geq m0\}$. Give corresponding definitions for $\Omega(g(n, m))$ and $\Theta(g(n, m))$.

4. How a non stable sorting algorithm can be made stable. Give the idea taking into context any exiting non stable algorithm.

5. Show that if a node in a binary search tree has two children, then its successor has no left child and its predecessor has no right child.

6. Show and prove that the longest simple path from a node x in a red-black tree to a descendant leaf has length at most twice that of the shortest simple path from node x to a descendant leaf.

7. Describe a $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose sum is exactly x.

8. Explain the deleteminimum process in Fibonacci Heaps.

9. Explain path compression heuristic in disjoint set Union-Find data structure and the resulting improvement due to this.

10. Write the recurrence for optimal binary search tree and explain it with the help of an example.