Sample Paper

1. Write sequence in which nodes of the graph (Fig. 5) have been traversed using DFS and BFS, starting at vertex . To make a unique solution, assume that whenever you faced with a decision of which node to pick from a set of nodes, pick the node whose label occurs earliest in the alphabet.

Topic: GRAPHS Difficulty: 3

2. Let each node in a binary search tree keeps an attribute that points to inorder successor. Give a pseudocode for INSERT on a binary search tree using this representation.

Topic: BST Difficulty: 5

3. Find a pair with the given sum in an array

Topic: SORTING Difficulty: 5

4. WAP to Construct the longest palindrome by shuffling or deleting characters from a string

Topic: HASHING Difficulty: 9

5. You are given N identical eggs and you have access to a K-floored building from 1 to K. There exists a floor f where 0 <= f <= K such that any egg dropped at a floor higher than f will break, and any egg dropped at or below floor f will not break. There are few rules given below. An egg that survives a fall can be used again. A broken egg must be discarded. The effect of a fall is the same for all eggs. If the egg doesn't break at a certain floor, it will not break at any floor below. If the eggs breaks at a certain floor, it will break at any floor above. Return the minimum number of moves that you need to determine with certainty what the value of f is.

Topic: DP Difficulty: 9

6. Execute the algorithm given in Fig. 5 on an array A[1..15] = [0, 70, 74, 52, 86, 84, 62, 90, 56, 91, (10) 75, 94, 89, 58, 78, 88]. Write the final contents of an array A[ ] and stack S1. //Q1 and Q2 are empty Queues, S1 and S2 are empty Stacks, an Integer i is initialized to 1, j is an //Integer, and flag is a Boolean. Build_Min_Heap(A); 17. if (A.heap_size) 18. { Q1.enqueue(A[i]); 19. while (!Q1.empty() || !Q2.empty()) 20. { flag = !true; 21. while (!Q1.empty()) 22. S2.push(A[Right(i)]); } Q1.dequeue(); } flag = true; while (!Q2.empty()) { i++; 23. if(!flag) 24. { S1.push(Q1.front()); 25. if(!S2.empty()) 26. S1.push(S2.top()); 27. flag = !flag ? !flag : flag; } 28. if (Left(i) <= A.heap_size) 29. Q2.enqueue(A[Left(i)]); 30. if (Right(i) <= A.heap_size) 31. { Q2.enqueue(A[Right(i)]); 32. { i++; if(flag) { S1.push(Q2.front()); S1.push(S2.top()); flag = !flag ? !flag : !flag; } if (Left(i) <= A.heap_size) Q1.enqueue(A[Left(i)]); if (Right(i) <= A.heap_size) { Q1.enqueue(A[Right(i)]); S2.push(A[Right(i)]); } Q2.dequeue(); } } }

Topic: HEAP Difficulty: 9

7. N Queens Problem

Topic: BACKTRACKING Difficulty: 9

8. How to find a median of two sorts arrays?

Topic: ARRAY Difficulty: 9

9. How do you find the distance between two nodes in a binary tree?

Topic: TREE Difficulty: 9

10. Is this optimization problem on a bipartite graph NP-9?

Topic: P and NP Difficulty: 9