Q.1 a. Show that the time complexity of linear search is O(n).

Answer: O(n) Solve the recurrence equation $a_n = a_{n-1} + 1$.

b. How does a Eulerian path differ from a Hamiltonian path? Give a suitable example.

Answer: Eulerian path consists of all edges in the graph whereas in Hamiltonian path all nodes are involved. Give a suitable example. (Refer to the book by Cormen)

c. Explain in any application of Minimum Spanning Tree in real life. If a new vertex and corresponding edges are added to a graph whose Minimum Spanning Tree is already computed, how fast one can compute the revised Minimum Spanning Tree?

Answer: It is extensively used in networking/routing and optimization to find minimum cost path and minimum cost connectivity. Give definition of spanning tree.

f. Write the basic algorithm to compare any two strings s and t. Your algorithm should take care of cases when two strings are of unequal length.

Answer: Refer to the naïve string matching algorithm mentioned in the book by Cormen.

Q.4 a. Why rotations are required while constructing Red-Black Trees? Taking an example of your choice, explain RR, LR and RL rotations in a Red Black tree.

Answer: See Balanced tree (Red-Black Tree) topic in the book by Cormen.

Q.6 a. Define articulation point of a graph. Write an algorithm to determine whether an undirected graph is strongly connected or not.

Answer: Strongly connected graph topic in connectivity of graph algorithm. Refer to it in the book by either Cormen or “Johnsonbaugh and Schaefer”

b. What is collision in hash function? Write at least 3 methods to resolve the issue related to collision.

Answer: Refer to hash search algorithm in the book by either Cormen or “Johnsonbaugh and Schaefer”

Q.7 Write a short notes on any TWO of the following:

(i) Backtracking
(ii) Priority Queue
(iii) Max Flow Min cut Theorem

Answer:
(i) Backtracking is a general algorithmic technique that consider searching every possible combination in order to solve an optimization problem. Backtracking is also knows as **depth-first search** or **branch and bound**. By inserting more knowledge of the problem, the search tree can be pruned to avoid considering cases that don't look promising. While Backtracking is useful for hard problems to which we do not know more efficient solutions, it is a poor solution for the everyday problems that other techniques are much better at solving.

However, dynamic programming and greedy algorithms can be thought of as optimizations to backtracking, so the general technique behind backtracking is useful for understanding these more advanced concepts. Learning and understanding backtracking techniques first provides a good stepping stone to these more advanced techniques because you won't have to learn several new concepts all at once.

For more details refer to book by Cormen.

(ii) Refer to Heap/queue topics in the book by Cormen.

(iii)Refer to graph connectivity topic in the book by Cormen.

Text Book

Introduction to algorithms – T.M. Cormen, C.E. Leiserson, R.L. Stein, MIT Press, 3rd Edition, 2009