DESIGN AND ANALYSIS OF ALGORITHM

IMP

Based on the frequency and weightage of the questions in the provided past papers, here's a prioritized study plan for "Design and Analysis of Algorithms (BTCOC401)":

High Priority Topics (Guaranteed to Appear & High Marks)

These topics consistently appear across all provided papers and often carry significant marks. Master these first.

1. **Asymptotic Notations:**
    - Define and explain Big O, Big Omega, and Big Theta notations with expressions and neat graphs.
    - Proving relationships between functions using asymptotic notations (e.g., f(n)=sqrt(n) and g(n)=n, prove f(n) is O(g(n))).
    - This is fundamental and usually asked in Q.1.
2. **Recurrence Relations (Master Method):**
    - Solve recurrence relations using the Master Theorem (e.g., T(n)=4T(n/2)+n, T(n)=4T(n/2)+n2, T(n)=4T(n/2)+n3, T(n)=16T(n/4)+n, T(n)=T(2n/3)+1, T(n)=3T(n/4)+nlogn).
    - Verify solutions using the substitution method.
    - Another core topic for Q.1.
3. **Merge Sort:**
    - Algorithm description and finding its time complexity.
    - Applying Merge Sort to a given array (e.g., A=5 1 2 6 3 7 4, A={5 2 4 7 3 2 6}, A={13,4,22,1,16,9,0,2}).
    - Best case and worst case time complexity.
    - Very frequent in Q.2.
4. **Strassen's Matrix Multiplication:**
    - Algorithm description and its time complexity.
    - Apply the algorithm to multiply two 2×2 matrices (e.g., (6574)(1324), A=(1111), B=(2222), A=(1221), B=(5667), A=(2236), B=(6487)).
    - A common application-based question in Q.2.
5. **Huffman Coding:**

- Huffman Coding algorithm and elements of greedy strategy.
- Obtain Huffman tree and codes for given character frequencies (e.g., a:50, b:25, c:15, d:40, e:75; a:6, b:11, c:19, d:35, e:50; a:45, b:13, c:12, d:16, e:9, f:5).
- Frequently asked in Q.3.

6. **Longest Common Subsequence (LCS):**
   - Compute LCS using Dynamic Programming approach for given sequences (e.g., X={A,B,C,B,D,A,B} and Y={B,D,C,A,B,A}; X=<x,m,j,y,a,u,z> and Y=<m,z,j,a,w,x,u>).
   - State the length of the LCS.
   - A recurring question in Q.4.
7. **4-Queens Problem (Backtracking):**
   - Solve the 4-Queens problem using the backtracking approach and draw the state space tree.
   - Consistently asked in Q.3 or Q.5.
8. **P, NP, NP-Complete Problems:**
   - Explain the concepts of P, NP, and NP-Complete classes and show the relationship between them.
   - A theoretical question often found towards the end (Q.4 or Q.5).
9. **Fractional Knapsack Problem:**
   - Solve the fractional knapsack problem given objects' profits/weights and knapsack capacity.
   - This is a common application of the greedy approach. (Q.4)
10. **Floyd Warshall Algorithm:**
    - Analyze or apply Floyd's Warshall Algorithm for all-pairs shortest path.
    - Important dynamic programming application (Q.5 or Q.4).

Medium Priority Topics (Likely to Appear, Important)

These topics are also important but might appear slightly less frequently or as part of a broader question.

1. **Algorithm Basics:**
   - Define Algorithm and state its main characteristics/criteria.

   - Need for algorithm analysis and factors affecting runtime of an algorithm.
   - Often combined with other basic questions in Q.1.

2. **Quick Sort:**
   - Algorithm and its best/worst case analysis with examples.
   - Time complexity. (Q.2)
3. **Dynamic Programming vs. Greedy Approach vs. Divide and Conquer:**
   - Differentiate between these algorithm design paradigms.
   - Often asked as a comparative question (Q.2 or Q.5).
4. **Job Sequencing with Deadlines:**
   - Solve problems using this greedy approach.
   - A specific application of greedy strategy. (Q.4)
5. **Travelling Salesman Problem (TSP) using Branch and Bound:**
   - Apply Branch and Bound technique to solve TSP for a given matrix/graph.
   - A key application of Branch and Bound (Q.3 or Q.5).
6. **Graph Coloring Problem:**
   - Describe with a suitable example. (Q.3)

Lower Priority Topics (May Appear, Good to Know)

These are less frequent but could still be part of a question or asked as a standalone if other high-priority topics are covered differently.

1. **Binary Search:**
   - Algorithm and its time complexity. (Q.2)
2. **Max and Min Heap:**
   - Define and write algorithm to insert into a heap. (Q.1)
3. **Bellman Ford Algorithm:**
   - Find the shortest path for a given graph. (Q.4)
4. **State Space Tree:**
   - General concept and its use (e.g., for sum of subsets or 15-puzzle). (Q.5)
5. **Red-Black Tree:**

- Properties and potential operations (insertion/deletion justification). (Q.6 in older papers)

6. **B-Tree:**
   - Insertion of keys and configurations. (Q.6 in older papers)
7. **Minimum Cost Spanning Tree:**
   - Explanation with a suitable example. (Q.4)
8. **Polynomial Time Reduction:**
   - Explain with an example. (Q.6 in older papers)

CHAPTER 2

A) Write algorithm for Binary Search and calculate its time complexity.

B) Explain Quick Sort algorithm with its performance analysis.

C) Explain Strassen's Matrix Multiplication with example.

D)Describe an algorithm for Merge Sort and find its time complexity.

E) Distinguish between Divide and Conquer and Dynamic programming approach.