S.R.K.R ENGINEERING COLLEGE (A) :: BHIMAVARAM

DEPARTMENT OF INFORMATION TECHNOLOGY

JNTUK-R23 Regulation

ADVANCED DATA STRUCTURES SHORT ANSWER QUESTIONS

UNIT-1

1. **What is an AVL Tree?**
   Ans: An AVL tree is a self-balancing binary search tree where the balance factor of any node (the height of the left subtree minus the height of the right subtree) is either +1, -1, or 0. This balance factor ensures the time complexity of O(log n) for all its operations.

2. **What are the four types of rotations used in AVL trees?**
   Ans: The four types of rotations used to maintain the balance in an AVL tree are:
   I. Right Rotation (Single Rotation): Applied when the left subtree of the left child is heavier.
   II. Left Rotation (Single Rotation): Applied when the right subtree of the right child is heavier.
   III. Left-Right Rotation (Double Rotation): Applied when the left subtree of the right child is heavier.
   IV. Right-Left Rotation (Double Rotation): Applied when the right subtree of the left child is heavier.

3. **List out the applications of AVL Trees.**
   Ans: AVL trees, as self-balancing binary search trees, have several practical applications due to their efficient operations.
   - Efficient Searching
   - Priority Queues
   - Routing Tables
   - Algorithm Optimization

4. **What is a B-tree?**
   Ans: A B-tree is a self-balancing tree data structure that maintains sorted data and allows for efficient insertion, deletion, and search operations. B-trees are characterized by nodes that can have multiple children and are balanced, ensuring that all leaf nodes are at the same level.
   The order of a B-tree is a parameter m that defines the maximum number of children each node can have.

In a B-tree of order m:

- Each node (except the root) must have at least ⌈m/2⌉ children.
- Each node can have up to m children

The height of a B-tree with n keys and order m is $O(\log_m n)$.

5. **What are the applications of B trees?**
   Ans: B-trees are widely used in various applications due to their balanced nature and efficient performance in handling large datasets.
   - Database Indexing
   - File Systems
   - Operating Systems
   - External Storage Systems
   - Search Engines
   - Information Retrieval
   - Networking

6. **What is a heap tree?**
   Ans: A heap tree is an almost complete binary tree that satisfies the heap property. In a **max heap**, each node's value is greater than or equal to its children's values. In a **min heap**, each node's value is less than or equal to its children's values.

7. **How is a heap represented in an array?**
   Ans: In an array representation of a heap:
   a. The root node is at index 1.
   b. For any node at index *i*:
   c. The left child is at index *2i*.
   d. The right child is at index *2i+1*.
   e. The parent node is at index (*i/2*).

8. **List out some applications of heap trees.**
   Ans: Heap trees are widely used in many areas like:
   - Priority Queues
   - Heap Sort
   - Graph Algorithms
   - Memory Management
   - Load Balancing
   - Job Scheduling

UNIT-2

1. **What is an algorithm? List out the characteristics of an algorithm.**
   Ans: An algorithm is a step-by-step procedure or set of rules designed to perform a specific task or solve a particular problem.
   An algorithm should possess the following characteristics to be effective:
   - Input
   - Output
   - Effectiveness
   - Finiteness
   - Definiteness

2. **What is asymptotic notation?**
   Ans: Asymptotic notation is used to analyse and describe the efficiency of algorithms in terms of their time complexity and space complexity as the size of the input grows.
   Asymptotic notations commonly used are:
   - Big-Oh notation (O)
   - Big Theta notation (θ)
   - Big Omega notation (Ω)

3. **Define Space complexity and Time complexity.**
   Ans: Space complexity is an amount of memory required to run an algorithm. It is typically expressed as a function of the input size.
   Time complexity is the amount of time required to run an algorithm.

4. **What is a connected graph?**
   Ans: A connected graph is an undirected graph where there is a path between every pair of vertices. In other words, you can reach any vertex from any other vertex.

5. **What is the difference between BFS and DFS?**
   Ans: Breadth-First Search (BFS): Traverses the graph level by level, starting from a source vertex. It uses a queue to keep track of vertices to be explored next.
   Depth-First Search (DFS): Traverses the graph by exploring as far as possible along each branch before backtracking. It uses a stack (or recursion) to keep track of vertices to be explored.

6. **What is an adjacency matrix and adjacency list?**

Ans: An adjacency matrix is a 2D array used to represent a graph. If the graph has n vertices, the matrix is $n \times n$ where the cell (i, j) contains a 1 (or the weight of the edge) if there is an edge from vertex i to vertex j, and 0 otherwise.

Adjacency list is a collection of lists or arrays where each list represents a vertex and contains the adjacent vertices (or edges) connected to it. It is an alternative way to represent a graph and is more space-efficient for sparse graphs.

7. **What is an articulation point and biconnected component in a graph?**

Ans: An articulation point is a vertex whose removal increases the number of connected components in the graph.
A biconnected component of an undirected graph is a maximal biconnected subgraph.

8. **What are the applications of graphs?**

Graphs are a fundamental data structure used in a wide range of applications across various fields.

- Computer Networks
- Social Networks
- Web Search Engines
- Route Planning and Navigation
- Transportation Networks
- Electrical Circuits

9. **What are the three main steps in a Divide and Conquer algorithm?**

Ans: **Divide**: Split the problem into smaller subproblems.
Conquer: Solve the subproblems recursively.
Combine: Merge the solutions of the subproblems to form the solution to the original problem (if necessary).

10. **What are the time complexities of the Merge Sort and Quick Sort algorithms?**

Ans:

Case	Quick Sort	Merge Sort
Best Case	O(n logn)	O(n logn)
Average Case	O(n logn)	O(n logn)

Worst Case	O(n^2)	O(n logn)

11. What is Strassen's Matrix Multiplication algorithm?

Ans: Strassen's Matrix Multiplication algorithm is an efficient method for multiplying two matrices that reduces the number of multiplications compared to the standard matrix multiplication approach. Strassen's algorithm has a time complexity of O($n^{\log_2 7}$) ≈ O($n^{2.81}$), where n is the dimension of the $n \times n$ matrices.

12. What is a convex and convex hull in computational geometry?

Ans:

Convex:

A polygon is defined to be convex if for any two points P1 and P2 inside the polygon, the direct line segment from P1 to P2 is fully contained in the polygon.

convex hull:

The convex hull of a set of points in a plane is the smallest convex polygon that contains all the points.

UNIT-3

1. **What is the greedy method?**

   Greedy is an algorithmic paradigm that builds up a solution stage by stage, always choosing the next stage that offers the most obvious and immediate benefit. Greedy algorithms are used for optimization problems.

   An optimization problem can be solved using Greedy if the problem has the following property:
   At every step, we can make a choice that looks best now, and we get the optimal solution to the complete problem.

2. **What is job sequencing with deadlines?**

- We are given a set of n objects, associated with job i is an integer deadline di ≥ 0 and a profit pi > 0. For any job i, the profit pi is earned if and only if the job i is completed by its deadline.
- To complete a job, one job to process on a machine for one unit of time and only one machine is available for processing jobs.
- A feasible solution for this problem is a subset J of jobs such that each job in this subset can be complete by its deadline. The value of a feasible solution is the sum of the profits of the jobs. An optimal solution is a feasible solution with maximum profit value.
- time complexity is O(n2).

3. **What is the knapsack problem?**

- We have 'n' objects and a knapsack or bag. Each object has weight 'Wi' and profit 'Pi' and knapsack has capacity 'm'.
- The fraction xi, 0 ≤ xi ≤ 1, for object i, wixi weight is placed into the knapsack and pixi profit is earned.
- Objective is filling of Knapsack that maximizes the total profit earned. So the problem can be stated as

Maximize $\sum_{1 \le i \le n} WiXi$

Subject to $\sum_{1 \le i \le n} WiXi \le m$ and $0 \le Xi \le 1$, $1 \le i \le n$

O(n logn), due to sorting the items by their value-to-weight ratio.

4. **What is a minimum cost spanning tree (MST) and How does Kruskal's and prim's Algorithms work?**

A. A subset of edges from a connected, weighted graph that connects all vertices with the minimum total edge weight.

Kruskal's algorithm sorts all edges by weight and adds the smallest edge to the MST, ensuring no cycles are formed, until all vertices are connected.

Prim's algorithm starts from an arbitrary vertex and grows the MST by adding the smallest edge connecting a vertex in the MST to a vertex outside it.

5. **What is the Single Source Shortest Paths (SSSP) problem?**

A. It is the problem of finding the shortest paths from a single source vertex to all other vertices in a weighted graph.

Time complexity isO((V+E)logV), where V is the number of vertices and E is the number of edges.

6. **Difference between Divide and Conquer approach and the Greedy approach?**

A.

S.No	Divide-and-Conquer	Greedy Method
1	Divide and conquer is used to obtain a solution to the given problem, it does not aim for the optimal solution.	The greedy method is used to obtain an optimal solution to the given problem.
2	In this technique, the problem is divided into small subproblems. These subproblems are solved independently. Finally, all the solutions to subproblems are collected together to get the solution to the given problem.	In Greedy Method, a set of feasible solutions are generated and pick up one feasible solution is the optimal solution.
3	Divide and conquer is less efficient and slower because it is recursive in nature.	A greedy method is comparatively efficient and faster as it is iterative in nature.

4	Divide and conquer may generate duplicate solutions.	In the Greedy method, the optimal solution is generated without revisiting previously generated solutions, thus it avoids the re-computation
5	Divide and conquer algorithms mostly run in polynomial time.	Greedy algorithms also run in polynomial time but take less time than Divide and conquer
6	Examples: Merge sort, Quick sort, Binary Search.	Examples: Fractional Knapsack problem, Minimum cost spanning tree, Job sequencing problem.

7. What is dynamic programming?

Dynamic Programming is an algorithmic technique with the following properties.

A. It is mainly an optimization over plain recursion. Wherever we see a recursive solution that has repeated calls for the same inputs, we can optimize it using Dynamic Programming.
B. The idea is to simply store the results of subproblems so that we do not have to re-compute them when needed later.

8. What is the Principle of Optimality?

A. The Principle of Optimality states that an optimal solution to a problem contains optimal solutions to its subproblems. In other words, any optimal solution to a problem can be constructed from optimal solutions of its subproblems.

9. What is an Optimal Binary Search Tree (BST)?

A. A binary search tree where the total cost of accessing all keys is minimized, considering the frequency of access for each key.

It involves calculating the minimum cost of a BST for each possible subtree and combining these results to find the overall minimum cost.

10. What is the Traveling Salesperson Problem (TSP) using dynamic programming?

A. It is a problem of finding the shortest possible route that visits each city exactly once and returns to the origin city, using dynamic programming to efficiently solve it.

11. Differentiate between Dynamic Programming and Greedy Method

S.No	Dynamic Programming	Greedy Method
1	Dynamic Programming is used to obtain the optimal solution.	Greedy Method is also used to get the optimal solution.
2	In Dynamic Programming, we choose at each step, but the choice may depend on the solution to sub-problems.	In a greedy Algorithm, we make whatever choice seems best at the moment and then solve the sub-problems arising after the choice is made.
3	Less efficient as compared to a greedy approach	More efficient as compared to a greedy approach
4	It is guaranteed that Dynamic Programming will generate an optimal solution using Principle of Optimality.	In Greedy Method, there is no such guarantee of getting Optimal Solution.
5	Example: 0/1 Knapsack, Optimal Binary Search Tree, Reliability Design	Examples: Fractional Knapsack problem, Minimum cost spanning tree, Job sequencing problem.

12. What is the difference between fractional knapsack and 0/1 knapsack?

Feature	**Fractional Knapsack**	**0/1 Knapsack**
Item Divisibility	Items can be divided into fractions.	Items must be taken whole; no fractions allowed.

Solution Approach	Greedy algorithm (based on value-to-weight ratio).	Dynamic programming.
Optimal Solution	Greedy algorithm always yields an optimal solution.	Greedy algorithms do not guarantee an optimal solution. Dynamic programming is required for optimal solutions.
Example	Taking portions of gold dust.	Taking whole peices of art, or whole electronic items.

UNIT-4

1. **What is backtracking in the context of algorithms?**

A. Backtracking is a technique for solving problems incrementally by trying partial solutions and then abandoning them if they are not viable. It involves exploring all potential solutions and "backtracking" when a partial solution is found to be invalid.

2. **What is the 8 Queens problem and give one solution?**

A. The 8 Queens problem is a classic puzzle where the goal is to place 8 queens on an 8×8 chessboard such that no two queens can be in the same row, column, or diagonal.

One solution is (4, 6, 8, 2, 7, 1, 3, 5).

3. **What is the Sum of Subsets problem and its time complexity?**

A. The Sum of Subsets problem is a combinatorial problem where given a set of integers, the goal is to find a subset whose elements sum up to a given target value.

The time complexity is $O(2^n)$, where n is the number of elements in the set. This is due to the need to explore all possible subsets.

4. **What is the Graph colouring problem and chromatic number?**

A. The Graph colouring problem involves assigning colours to the vertices of a graph such that no two adjacent vertices share the same colour, using the minimum number of colours possible and that number is called chromatic number.

5. **What is the Branch and Bound method?**

A. Branch and bound is one of the techniques used for problem solving. It is similar to the backtracking since it also uses the state space tree. It is used for solving the optimization problems and minimization problems. If we have given a maximization problem, then we can convert it using the Branch and bound technique by simply converting the problem into a maximization problem.

6. **What is a FIFOBB and bounding function in Branch and Bound?**

A. FIFO Branch and Bound is a search algorithm used to solve complex optimization problems. The FIFO Branch and Bound algorithm combines

the principles of Breadth-First Search (BFS) and Depth-First Search (DFS) to solve optimization problems.
The bounding function is a crucial component of the FIFO Branch and Bound algorithm. It provides an estimate of the optimal solution to the sub-problem and helps in pruning the search space. The bounding function is used to calculate an upper bound on the optimal solution, which helps in selecting the most promising sub-problems.

7. Write control abstraction for LCBB?

```
Algorithm LCSearch(t)
//Search t for an answer node
{
    if (*t is an answer node) then output *t and return;
    E = t; //E-node
    Initialize the list of live nodes to be empty;
    repeat
    {
        for (each child x of E) do
        {
            if (x is an answer node) then output the path from x to
            t and return;
            add(x);                // x is a new live node.
            (x → parent) = E;      // pointer for path to root
        }
        if (there are no more live nodes) then
        {
            write(“No answer node”);
            return;
        }
        E = Least();
        }until(false);
    }
```

8. Difference between Backtracking and Branch-and-Bound.

S.No	Backtracking	Branch-and-Bound
1	Backtracking is used to find all possible solutions available to a problem. When it realizes that it has made a bad choice, it undoes the last choice by backing it up. It searches the state space tree until it has found a solution for the problem	Branch-and-Bound is used to solve optimization problems. When it realizes that it already has a better optimal solution that the pre-solution leads to, it abandons that pre-solution. It completely searches the state space tree to get optimal solution
2	Backtracking traverses the state space tree by DFS (Depth First Search) manner.	Branch-and-Bound traverse the tree in any manner, DFS or BFS.
3	Backtracking involves feasibility function.	Branch-and-Bound involves a bounding function.
4	Backtracking is used for solving Decision Problem.	Branch-and-Bound is used for solving Optimisation Problem.
5	In backtracking, the state space tree is searched until the solution is obtained.	In Branch-and-Bound as the optimum solution may be present anywhere in the state space tree, so the tree needs to be searched completely.
6	Backtracking is more efficient.	Branch-and-Bound is less efficient.
7	Backtracking can solve almost any problem. (chess, sudoku, etc).	Branch-and-Bound cannot solve almost any problem.
8	Typically backtracking is used to solve decision problems.	Branch and bound is used to solve optimization problems.
9	Nodes in state space tree are explored in depth first tree.	Nodes in tree may be explored in depth-first or breadth-first order.
10	Next move from current state can lead to bad choice.	Next move is always towards better solution.
11	On successful search of solution in state space tree, search stops.	Entire state space tree is search in order to find optimal solution.
12	Example: N-Queen Problem, Sum of subset, Hamilton cycle problem, graph coloring problem	Example: Knapsack Problem, Travelling Salesman Problem.

UNIT-5

1. **What are the functions used in non-deterministic algorithm?**

   We introduce three functions in nondeterministic algorithms,

   A. Choice (): arbitrarily choose one of the elements of set S.

   B. Failure (): signals an unsuccessful completion.

   C. Success (): signals a successful completion.

2. **Write non-deterministic algorithm for sorting?**

   A. Algorithm Nsort(A, n)

```
{
    for i = 1 to n do B[i] = 0;  // initialize B
    for i = 1 to n do
    {
            j = Choice (1, n);
            If (B[j] ≠ 0) then Failure ();
            B[j] = A[i] ;
    }
    for i = 1 to n-1 do //verify order
            If(B[i] > B[i + 1]) then Failure();
    write (B[1…n]);
    Success ();
}
```

   The time complexity of non-deterministic sorting algorithm is O(n)

3. **What is class P and class NP?**

   Class P:

   P is the set of all deterministic polynomial time algorithms.

   Examples- sorting, searching, MST, Huffman coding etc.

   Class NP:

   NP is the set of all nondeterministic polynomial time algorithms.

4. **What is Satisfiability problem?**

A. The satisfiability problem is to determine whether a formula is true for some assignment of truth values to the variables. CNF-satisfiability is the satisfiability problem for CNF formula.

5. **What is NP-hard and NP-Complete problems?**

NP-hard:

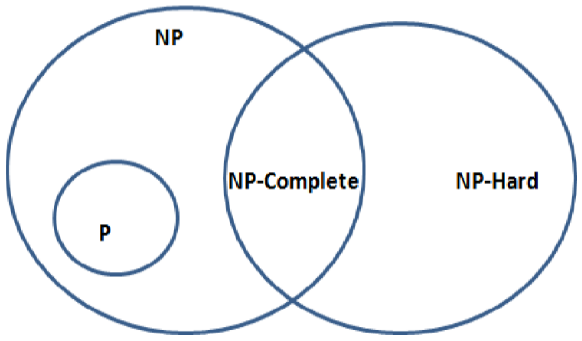
A problem L is NP-hard if and only if satisfiability reduces to L

(satisfiability α L). Ex: Chromatic Number Decision Problem.

NP-complete:

A problem L is NP-complete if and only if L is NP-hard and L is in NP.

Ex: SAT problem.



6. **What is clique and clique decision problem?**

A. Clique:

Let G = (V, E) be a graph. A clique is a complete subgraph of a given graph.

Clique decision problem:

check whether the given graph has a clique of size k or not.

7. **What is chromatic number decision problem?**

A. Let ‘G’ be a graph and ‘k’ be a given positive integer. If the nodes of ‘G’ can be colored in such a way that no two adjacent nodes have the same color.

The Chromatic Number Decision Problem is to determine whether G has a coloring for given k.

8. **What is traveling salesman decision problem?**

A complete graph G=(V, E) with a set of vertices V, edge weights (distances) between each pair of vertices, and an integer k.

Is there a Hamiltonian cycle (a cycle that visits each vertex exactly once and returns to the starting vertex) in G with a total weight less than or equal to k?

9. What is Partition?

A. To decide whether a multi-set A = {a1, a2, …., an} of n positive integers has a partition P such that $\sum_{i \in P} ai = \sum_{i \notin P} ai$

10. What is Scheduling Identical Processors?

A. Let Pi, 1 ≤ i ≤ m, be m identical processors or machines (example printers). Let Ji, 1 ≤ i ≤ n, be n jobs. Job Ji requires ti processing time. A Schedule S is an assignment of jobs to processors. For each job Ji, S specifies the time intervals and the processors on which this job is to be processed. A job cannot be processed by more than one processor at any given time.

11. What is Job Shop Scheduling?

A. Multiple jobs are processed on several machines. Each job consists of a sequence of tasks, which must be performed in a given order, and each task must be processed on a specific machine. The problem is to schedule the tasks on the machines to minimize the length of the schedule—the time it takes for all the jobs to be completed.