1.State the advantages of B+ Tree Over & Tree

Range Queries: B+ trees excel at range queries because all leaf nodes are linked in a linked list, enabling efficient sequential traversal of data.

Disk-Based Storage: B+ trees are optimized for disk-based storage, minimizing I/O operations by storing the bulk of the data in separate nodes.

Predictable Performance: B+ trees offer consistent O(log N) time complexity for search, insert, and delete operations due to their balanced structure.

Database and File Systems: They are widely used in databases and file systems for efficient key-value storage.

2.Difference between AVL Tree & B Tree

AVL Tree	**B Tree**
It is a self-balancing binary search tree	It is a multi-way tree
Every node contains at most 2 child nodes	In this tree, nodes can have multiple child nodes
It has a balance factor whose value is either -1, 0, or 1. Balance factor = (height of left subtree)-(height of right subtree)	B-Tree is defined by the term minimum degree 't'. The value of 't' depends upon disk block size. Every node except the root must contain at least t-1 keys. The root may contain a minimum of 1 key.
AVL tree has a height of log(N) (Where N is the number of nodes)	B-tree has a height of log(M*N) (Where 'M' is the order of trees and N is the number of nodes).

3.Compare Binary Tree, Binary Search Tree, AVL Tree

Binary Tree:

i)A binary tree is a hierarchical data structure where each node has, at most, two children: a left child and a right child.

ii)It does not impose any specific ordering of elements or keys.

iii)Binary trees can have unbalanced structures, leading to potentially inefficient search and traversal operations.

Binary Search Tree (BST):

i)A BST is a type of binary tree where nodes are organized in a way that elements with smaller values are stored in the left subtree, and elements with larger values are stored in the right subtree.

ii)This ordering property makes searching for elements more efficient, with average time complexity for search, insertion, and deletion operations being O(log N), where N is the number of nodes.

iii)However, if the tree becomes unbalanced, it can degrade into a linked list, leading to worst-case time complexities of O(N) for search and other operations.

AVL Tree:

i)An AVL (Adelson-Velsky and Landis) Tree is a self-balancing binary search tree.

ii)In an AVL tree, the balance factor of each node (the difference in heights between its left and right subtrees) is maintained at -1, 0, or 1, ensuring that the tree remains balanced.

iii)This balance property guarantees that the height of the tree is logarithmic, leading to efficient O(log N) search, insertion, and deletion operations.

4.Write the possible constraints to remove a node from Binary Search Tree.

i)Node to be removed doesn't exist: Ensure that the node you want to remove exists in the tree.

ii)Node with no children (Leaf node):

iii)If the node to be removed is a leaf node, you can simply remove it without any issues.

iv)Node with one child:

If the node to be removed has one child, you can replace the node with its child.

5.Write the properties of properties of Heap Data Structure

In a max-heap, for any given node, the value of that node is greater than or equal to the values of its children. This property ensures that the maximum element is always at the root.

In a min-heap, for any given node, the value of that node is less than or equal to the values of its children. This property ensures that the minimum element is always at the root.

6. List the types of Heap Tree

Max-Heap:

In a max-heap, for any given node, the value of that node is greater than or equal to the values of its children. This means that the maximum element is always at the root of the heap.

Min-Heap:

In a min-heap, for any given node, the value of that node is less than or equal to the values of its children. This ensures that the minimum element is always at the root of the heap.

7.List out most frequency used terminology is graph data structure

Node,edge,Directed Graph,Undirected Graph,Adjacent,Degree,Path,Cycle,Connected Graph,Disconnected Graph.

8.How to find degree of a vertex

One way to find the degree is to count the number of edges which has that vertx as an endpoint.

9. What is the cardinality of the given graph

Vertex Cardinality: The number of vertices in the graph.

To find the vertex cardinality, simply count the number of nodes in the graph.

Edge Cardinality: The number of edges in the graph.

To find the edge cardinality, count the number of edges in the graph. This involves counting the connections between vertices.

10. How to find indegree and outdegree of vertex.

i)based on the direction of the vertex, we can classify the edges as incoming or outgoing.

ii)The in-degree of a vertex can be defined as the number of edges coming to the vertex.

iii)the out-degree of a vertex can be defined as the number of edges coming out from the vertex.

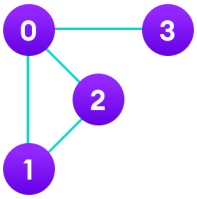
11. Difference between connected graph and complete graph.

A connected graph is defined as a graph in which a path of distinct edges connects every pair of vertices. Meanwhile, a complete graph depicts every vertex connected by a unique edge. Therefore, every complete graph is connected, but not every connected graph is complete.

12. Show the representation of graph data structure

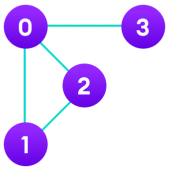
Adjacency matrix

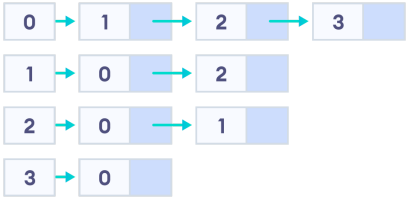
example:



i → / j ↓	0	1	2	3
0	0	1	1	1
1	1	0	1	0
2	1	1	0	0
3	1	0	0	0

Adjacency list





13.Define topological sort

Topological sorting is a linear ordering of the vertices of a directed acyclic graph (DAG) in such a way that for every directed edge (u, v), vertex u comes before vertex v in the ordering.it's an ordering of the vertices that respects the partial order defined by the directed edges, ensuring that there are no cycles in the graph.

14.Mention the types of graph traversals

Depth-First Search (DFS):

i)DFS explores as far as possible along each branch or path before backtracking. It starts at a source node and explores its neighbors before moving deeper into the graph.

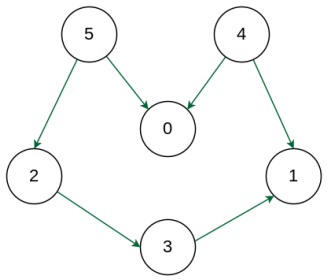
ii)It uses a stack (or recursion) to manage the nodes to visit.

Breadth-First Search (BFS):

i)BFS explores all the vertices at the current level before moving on to vertices at the next level. It systematically explores the graph in layers.

ii)It uses a queue to manage the nodes to visit.

15.Identify the possible topological ordering of given graph example



Output: 5 4 2 3 1 0

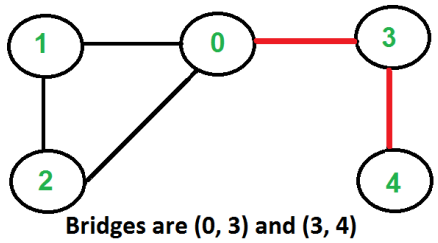
Explanation: The first vertex in topological sorting is always a vertex with an in-degree of 0 (a vertex with no incoming edges). A topological sorting of the following graph is “5 4 2 3 1 0”. There can be more than one topological sorting for a graph. Another topological sorting of the following graph is “4 5 2 3 1 0”.

16.What is articulation point and state the use

a vertex which, when removed along with associated edges, makes the graph disconnected (or more precisely, increases the number of connected components in the graph).

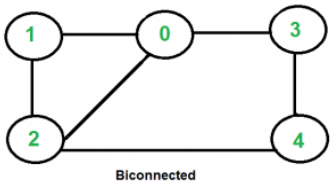
17.Define Bridges, biconnected graph with Suitable example

a bridge is an edge of a graph whose deletion increases the graph's number of connected components.



Bridges are (0, 3) and (3, 4)

A biconnected undirected graph is a connected graph that is not broken into disconnected pieces by deleting any single vertex (and its incident edges).

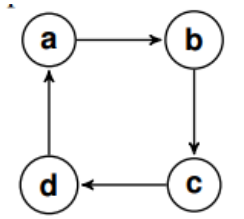


Biconnected

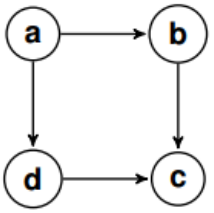
18. Differentiate Strongly connected and weakly connected component in graph

A directed graph is strongly connected if there is a path from a to b and from b to a whenever a and b are vertices in the graph.

A directed graph is weakly connected if there is a path between every two vertices in the underlying undirected graph.

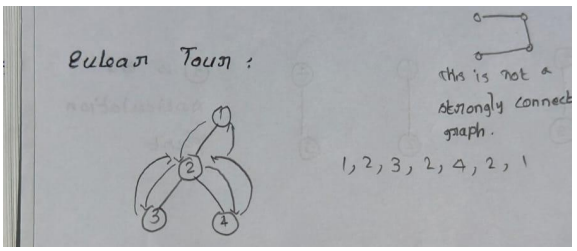


Strongly connected



Weakly connected

19.How to construct Eular graph / Steps.



20.Write the steps to construct Hamilton graph.

Create an empty path array and add vertex 0 to it. Add other vertices, starting from the vertex 1. Before adding a vertex, check for whether it is adjacent to the previously added vertex and not already added. If we find such a vertex, we add the vertex as part of the solution

21.construct AVL Tree For give example

22.Possible notations of AVL Tree
Left to left rotation,right to right rotation, either left to right and either right to left

23.Types of Minimum spanning Tree
Prim's Algorithm Minimum Spanning Tree
Kruskal's Algorithm Minimum Spanning Tree

24.What is Single Source shortest Path
The Single-Source Shortest Path (SSSP) problem consists of finding the shortest paths between a given vertex v and all other vertices in the graph. Algorithms such as Breadth-First-Search (BFS) for unweighted graphs or Dijkstra

25.Summarize the properties of B tree
In a B-Tree, each node has a maximum of m children. Except for the root and leaf nodes, each node in a B-Tree has at least m/2 children. There must be at least two root nodes. The level of all the leaf nodes should be the same.

26.Write the unique features / properties of B+ tree Over B tree
Every node in a B+ Tree contains at most “m” children.
Every node in a B+ Tree, except the root node and the leaf node contain at least “m/2” children.
Nodes can have a maximum of (m-1) keys.
Nodes can have a minimum of ⌈m/2⌉-1 keys.

27.Identify the way/method/formula to find child of B tree,Maximum child,minimum keys,maximum keys
1. **Finding Child of a B-tree:**
   - To find the child of a B-tree node, you typically perform a search or traversal through the tree by comparing the key you're looking for with the keys in the current node.
   - In a B-tree, each node contains a set of keys and a corresponding set of children (or pointers) that point to subtrees.
   - To find the child corresponding to a key in a node, you compare the key you're looking for with the keys in the current node and determine which child pointer to follow based on the comparison result.

2. **Maximum Child of a B-tree:**
   - The maximum number of children a node in a B-tree can have is often denoted by the symbol "M."
   - The value of M is defined when the B-tree is created and depends on the specific implementation and use case.

- The maximum child count usually depends on factors like the size of a disk page, memory constraints, and performance requirements. In typical B-tree implementations, M is a design parameter.

3. **Minimum Keys in a B-tree:**
   - In a B-tree, the minimum number of keys a non-root, non-leaf node can have is typically defined as "M/2."
   - This means that if a B-tree node has fewer than M/2 keys, it is considered underfilled, and balancing operations may be required to redistribute keys from siblings.
   - The root node may have fewer keys than M/2 if the root is a leaf node.

4. **Maximum Keys in a B-tree:**
   - The maximum number of keys a node in a B-tree can have is typically defined as "M-1."
   - This means that a node can have up to M-1 keys.
   - The restriction on the maximum number of keys in a node helps maintain balance and search efficiency in the B-tree.

28.what is walk in euler graph.how it differs from path
Walk in an Euler Graph:
i)It may visit vertices and edges more than once, and it can start and end at different vertices.
ii)The key property of a walk in an Eulerian graph is that it covers every edge in the graph at least once, but it does not necessarily traverse every vertex.
Path in a Graph:
i)it is a sequence of vertices and edges that connects two distinct vertices. A path does not have to visit every vertex or edge in the graph.
ii)Paths can occur in any graph, Eulerian or not. They are a fundamental concept in graph theory and are used to find routes, connections, and sequences in a graph.

29.differentiate tour with euler tour
Tour:
i)A "tour" in a graph refers to a sequence of vertices and edges that visit each vertex and edge exactly once and returns to the starting point.
ii)In other words, a tour is a closed path that covers all vertices and edges in the graph and returns to the initial vertex.
Euler Tour:
i)An "Euler tour" is a specific type of tour in a graph where the tour visits every edge exactly once and returns to the starting vertex.
ii)Unlike a general tour, an Euler tour focuses on covering edges rather than vertices. In an Euler tour, you traverse all edges in the graph exactly once.