**Week 6**

**Name: Joyal Joseph**

**Mobile No : 6238189424**

| **Data Structure Workouts** |
| --- |
| 1. Learn the concepts of Tree. Complete at least three sample workouts. 2. Learn the concepts of Binary Search Tree. Complete at least three sample workouts. Example: 3. Create a Binary Search Tree with insertion, contains, delete, three traversals ( postorder, preorder, in order). 4. Find the closest value to a given number in a Tree. 5. Validate whether a given tree is BST or not. 6. Learn the concepts of Heap. Complete at least three sample workouts.   Example:   1. Create a min heap & max heap with build, insert, remove. 2. Learn the concept of Heap sort. Complete at least three sample workouts 3. Learn the concepts of Trie. Complete at least 3 sample workouts. 4. Learn the concepts of Graph. Complete at least three sample workouts. 5. Learn the concepts of Graph traversals (BFS, DFS). 6. Do at least 3 problems each for every structure from any competitive coding websites 7. Learn about the applications of all structures you covered this week |
| *The concept of a tree in data structure is a fundamental concept in computer science and data structure.It is a hierarchical data structure that consists of nodes connected by edge, with a single node called the root. Each node in a tree can have zero or more child nodes.*  *Each element in a tree is called a node. It contains data and may have references or links to its child nodes. The topmost in a tree is called a node. It is the starting point of the tree. A node in a tree can have child nodes, and the node that points to its child node is called the parent node. Nodes that share the same parent are called siblings. A leaf node is a node that does not have any children. The depth of a node is the number of edges from the root to that node. Applications are representing file systems, search algorithms and more.*  [*https://drive.google.com/drive/folders/1xUfq8Q1IR3nsc2emD37vREX-UITSvV-v?usp=drive\_link*](https://drive.google.com/drive/folders/1xUfq8Q1IR3nsc2emD37vREX-UITSvV-v?usp=drive_link) |
| *A binary search tree is a type of tree where the left child of a node contains a value smaller than the node, and the right child contains a value greater than the node. The binary search tree property allows for efficient searching , insertion and deletion operations. In a binary search tree, for any node, all values in its left subtree are less than its value, and all values in its right subtree are greater than its value. Binary search trees support three main types of traversal: In-Order, Pre - Order and Post -Order. The main concepts of binary search trees are important for implementing efficient search and storage structures in many applications. It allows for efficient manipulation and retrieval of data, making it a widely used data structure in various domains.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/1xUfq8Q1IR3nsc2emD37vREX-UITSvV-v?usp=drive\_link*](https://drive.google.com/drive/folders/1xUfq8Q1IR3nsc2emD37vREX-UITSvV-v?usp=drive_link) |
| *A heap is a specialised tree based data structure that satisfies the heap property. The heap property depends on the type of heap,The two most common types of heaps are the binary heap and the binary min-max heap. The heap property defines the order or priority relationship between elements in a heap. In a binary min-heap, for example the value of each node is greater than or equal to the values of its children. In the binary max heap, the value of each node is greater than or equal to the values of its children. This property allows efficient retrieval of the minimum or maximum element from the heap.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/13a4L4XvkIcd1mk15Ik4kI\_9Z6iFtvaiQ?usp=drive\_link*](https://drive.google.com/drive/folders/13a4L4XvkIcd1mk15Ik4kI_9Z6iFtvaiQ?usp=drive_link) |
| *Heap sort is an efficient sorting algorithm that utilises the properties of a binary heap data stature. It works by transforming an input array into a binary heap, then repeatedly extracting the maximum element from the heap and placing it at the end of the array. The process is repeated until all elements are sorted in ascending or descending order. Here are the key concepts associated with heap sort. Heap sort has a time complexity of O(n log n), making it an efficient sorting algorithm. It is an in-place sorting algorithm.However, heap sort is not a stable sorting algorithm, as the relative order of equal elements may change during the sorting process.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/13a4L4XvkIcd1mk15Ik4kI\_9Z6iFtvaiQ?usp=drive\_link*](https://drive.google.com/drive/folders/13a4L4XvkIcd1mk15Ik4kI_9Z6iFtvaiQ?usp=drive_link) |
| *It is a prefix tree or digital tree. It is used for efficient retrieval and searching of strings or sequences of characters. The trie is designed to provide fast prefix based operations, making it suitable for tasks such as autocomplete, spell checking and dictionary lookups. A trie is composed of nodes, where each node represents a character in a string or a sequence. Nodes typically contain a reference to the character they represent, pointers to child nodes and additional information if needed. Tries are widely used in various applications such as spell checkers, autocomplete systems, IP routing tables, text processing and more.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/1KMmnMfhn3kW8yNMvbNaZ\_Ki21O0lz-FO?usp=drive\_link*](https://drive.google.com/drive/folders/1KMmnMfhn3kW8yNMvbNaZ_Ki21O0lz-FO?usp=drive_link) |
| *A graph is a data structure that represents a collection of interconnected nodes, also known as vertices,and the relationships between them, known as edges. Graphs are used to model and analyse relationships between objects or entities in various fields, including computer science, mathematics and more. Nodes or vertices are the fundamental of a graph. Each node typically represents an entity or an object and can be labelled with an identifier or contain additional information. Understanding the concept of graphs is essential for modelling relationships, analysing networks, solving graph-related problems, and designing efficient algorithms for graph traversal, shortest path finding, and connectivity analysis.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/1jSVBHlVWM7nrDK6BEFSMcikorDg3aRFF?usp=drive\_link*](https://drive.google.com/drive/folders/1jSVBHlVWM7nrDK6BEFSMcikorDg3aRFF?usp=drive_link) |
| *Graph traversal algorithms are used to visit or explore all the nodes in a graph systematically. The two commonly used graph traversal algorithms are Breadth First search and Depth First search. BFS starts at a given node (often the root) and explores all its neighbouring nodes at the current depth level before moving to the next depth level. It uses a queue data structure to keep track of the nodes to visit. DFS starts at a given node and explores as far as possible along each branch before backtracking. It uses a stack data structure or recursion to keep track of nodes to visit. Understanding the concepts of BFS and DFS is valuable for efficiently exploring and analysing graphs, as they provide different approaches to traverse nodes and uncover relationships within a graph structure.*  *Link to the folder containing code and screenshot of the output* [*https://drive.google.com/drive/folders/1jSVBHlVWM7nrDK6BEFSMcikorDg3aRFF?usp=drive\_link*](https://drive.google.com/drive/folders/1jSVBHlVWM7nrDK6BEFSMcikorDg3aRFF?usp=drive_link) |
| *1. Tree: Applications: File systems, organisation charts, family trees, decision trees, hierarchical data structures. Trees are used to represent hierarchical relationships between elements. They provide efficient searching, insertion, deletion, and hierarchical organisation of data.*  *2. Binary Search: Applications: Searching algorithms, database systems, spell checking, range queries. Binary search is a divide-and-conquer search algorithm used to efficiently find a specific value in a sorted collection or determine its absence. It significantly reduces the search space by repeatedly dividing it in half.*  *3. Heap: Applications: Priority queues, efficient sorting algorithms, job scheduling, graph algorithms. Heaps are binary trees that satisfy the heap property. They are used for efficient insertion, deletion, and retrieval of the maximum (or minimum) element. Heap data structure supports operations like push, pop, and heapify.*  *4. Trie: Applications: Autocomplete systems, spell checking, dictionary lookups, IP routing tables, prefix matching.Tries, or prefix trees, are specialised tree structures used for efficient retrieval of strings. They provide fast prefix-based operations and are used to store and search for strings with common prefixes.*  *5. Graph:Applications: Network analysis, social networks, routing algorithms, recommendation systems, computer networks.Graphs are used to model relationships between objects or entities. They represent connections (edges) between nodes (vertices). Graphs enable analysis of connectivity, shortest paths, network structures, and optimization problems.*  *These data structures and concepts provide powerful tools for organising, searching, and analysing data in various domains. Understanding their applications helps in choosing the appropriate data structure for specific problems and developing efficient algorithms.* |
|  |
|  |

# 