Name: Mohamed Abdelfatah Abdelnabi Ibrahim

1. Hash Table

It’s a data structure that implements an associative array, mapping keys to values using a hash function.

- Key Concepts
  - Hash Function: Converts a key into an index in the hash table.
  - Buckets : Storage slots for values; collisions can result in multiple keys mapping to the same bucket.
  - Collision Resolution :
  - Chaining : Storing multiple elements in the same bucket using a linked list.
  - Open Addressing : Finding alternative slots using methods like linear probing or quadratic probing.

- Operations
  - Insertion : Insert a key-value pair by computing the hash index.
  - Search : Retrieve the value associated with a given key.
  - Deletion : Remove a key-value pair by locating the key’s index.

- Advantages
  - Average O(1) time complexity for search, insert, and delete operations.
  - Efficient for large datasets.

- Disadvantages
  - Performance depends on a good hash function.
  - Requires resizing and rehashing when the load factor exceeds a threshold.

2. Graphs

It's a data structure used to represent relationships between pairs of objects. It consists of:

- Vertices (nodes) : Fundamental units representing entities.
- Edges (connections) : Relationships between pairs of vertices.

- Types of Graphs
  - Undirected Graph : Edges have no direction.
  - Directed Graph (Digraph) : Edges have a direction.
  - Weighted Graph : Edges have weights representing costs or distances.
  - Unweighted Graph : Edges have no weights.
  - Cyclic/acyclic Graphs : Graphs with or without cycles.

- Representations
  - Adjacency Matrix :
  - 2D array where rows and columns represent vertices.
  - Space complexity: $O(V^2)$, where V is the number of vertices.
  - Adjacency List :
  - Array of lists where each list contains adjacent vertices.
  - Space-efficient for sparse graphs.

- Algorithms
  - Traversal :
  - Breadth-First Search (BFS) : Explores all neighbors level by level.
  - Depth-First Search (DFS) : Explores as far as possible along one branch before backtracking.
  - Shortest Path :
  - Dijkstra's Algorithm : For graphs with non-negative weights.
  - Bellman-Ford Algorithm : Handles negative weights.
  - Minimum Spanning Tree :
  - Kruskal's Algorithm : Uses edges sorted by weight.
  - Prim's Algorithm : Builds the tree starting from a single vertex.

- Advantages
  - Models complex relationships effectively.
  - Flexible representation for varying applications.

- Disadvantages
  - Storage can be inefficient for dense graphs.
  - Traversal algorithms may be computationally expensive for large graphs.