Introduction to Graphs

A **Graph** is a collection of **nodes** (vertices) and **edges** that connect pairs of nodes. Graphs are used in computer science for modeling relationships such as social networks, road maps, and the internet.

Basic Terminology
- Vertex (V): A node in the graph.
- Edge (E): A connection between two vertices.
- Undirected Graph: Edges have no direction.
- Directed Graph (Digraph): Edges have a direction.
- Weighted Graph: Each edge has a weight (cost, distance, etc.).
- Unweighted Graph: Edges only represent connections, no weights.

Undirected Unweighted Graph
(0) ----- (1)
| |
| |
(2) ----- (3)

Directed Unweighted Graph
(0) → (1) → (2)
↑ |
■----------■

Weighted Graph (Undirected)
(0) ---5--- (1)
| |
2| |3
| |
(2) ---4--- (3)

Graph Representations
1. Adjacency Matrix
- 2D array (or vector of vectors).
- Good for dense graphs.
- Edge lookup: O(1).
- Uses $O(V^2)$ memory.

2. Adjacency List
- Each vertex stores a list of neighbors.
- Good for sparse graphs.
- Space-efficient O(V+E).
- Edge lookup: O(degree of vertex).

Next Steps
- Implement adjacency matrix and adjacency list.
- Add support for weighted and directed graphs.
- Explore graph algorithms:
- BFS (Breadth-First Search)
- DFS (Depth-First Search)
- Dijkstra's Algorithm
- Minimum Spanning Tree (Prim/Kruskal)