Graph Representations and Comparisons

■ Comparing Adjacency Matrix vs Adjacency List

Feature	Adjacency Matrix	Adjacency List
Storage	$O(V^2)$	O(V + E)
Edge Lookup	O(1)	O(degree of vertex)
Add Edge	O(1)	O(1) (amortized)
Remove Edge	O(1)	O(degree of vertex)
Best for	Dense graphs (lots of edges)	Sparse graphs (few edges)
Memory Usage	High (even if few edges exist)	Efficient (only existing edges)

■ Rule of Thumb:
- Use **Adjacency Matrix** when the graph is **dense** or when you need **constant-time edge lookup**.
- Use **Adjacency List** when the graph is **sparse** or when memory efficiency matters.

■ Tree vs Graph

Tree

- Special type of graph.
- Connected and **acyclic** (no cycles).
- If there are V vertices → exactly V - 1 edges.
- Example: Family tree, file system hierarchy.

Graph

- More general structure.
- Can be connected or disconnected.
- May contain cycles.
- Edges can be directed/undirected, weighted/unweighted.

Visualization:

Tree (Acyclic):

```
       (0) / \ (1) (2) / \ (3) (4)
```

Graph (Can have cycles):

```
       (0) --- (1) | / \ | / \ (2) -------- (3)
```