1.4 DATA TYPES & STRUCTURES · 1.4.2

Graphs

Nodes & edges, types, and how to represent them. Spec 1.4.2(b)(c).

01 What & types

Graph **Nodes** (vertices) joined by **edges** (arcs). No hierarchy; can have cycles.

Undirected Edges two-way (e.g. friendship).

Directed Edges one-way, with **arrows** (e.g. follows).

Weighted Edges carry a **value/cost** (e.g. distance).

02 Adjacency matrix

What 2-D grid (node × node); 1/weight = edge, 0 = none.

Undirected **Symmetrical** (A–B = B–A).

+ Fast check if two nodes linked (one cell).

– Stores every possible edge → wasteful for **sparse**.

03 Adjacency list

What Each node lists only its **neighbours**.

+ Stores only real edges → **memory-efficient** (sparse).

– Must **search a list** to check a connection.

Form Array/dictionary of nodes → list of neighbours.

04 Choose

Dense Many edges → **matrix**.

Sparse Few edges → **list**.

Uses Social networks, maps/routes, web links.

vs tree Tree = graph with hierarchy & no cycles.

FINAL PASS BEFORE THE EXAM

Rapid exam tips

Six slips on graph questions.

01
Undirected matrix is **symmetrical**; directed is not.

02
Weighted? Store the **weight** in the cell, not just a 1.

03
Matrix = quick edge check, good for **dense** graphs.

04
List = memory-efficient, good for **sparse** graphs.

05
Directed edges **must have arrows**.

06
A graph can have **cycles**; a tree can't.