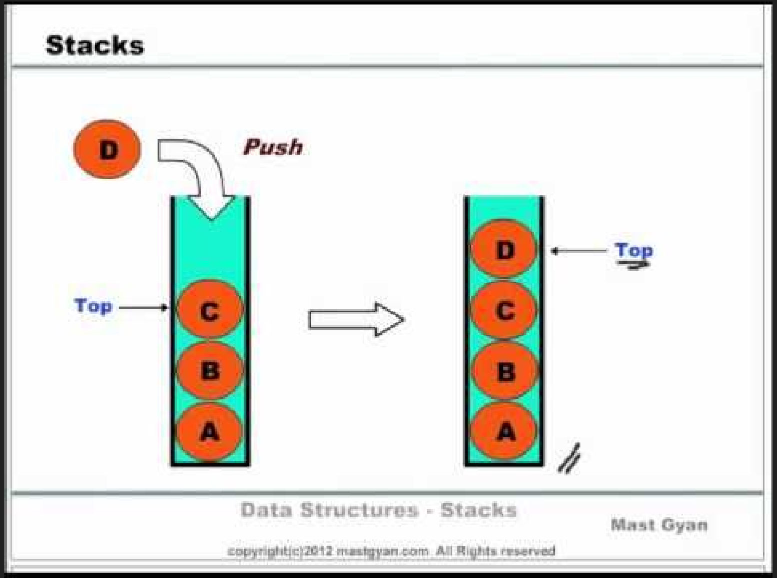
Competitive Programming Training

Graph Theory

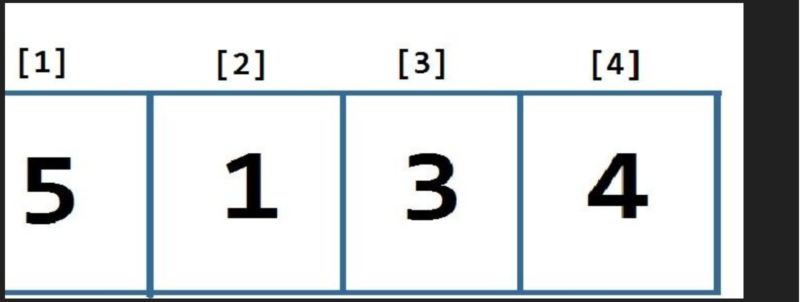
Conventional Data Structures

- ➔ Arrays
- ➔ Stacks
- ➔ Queues
- ➔ Linked Lists

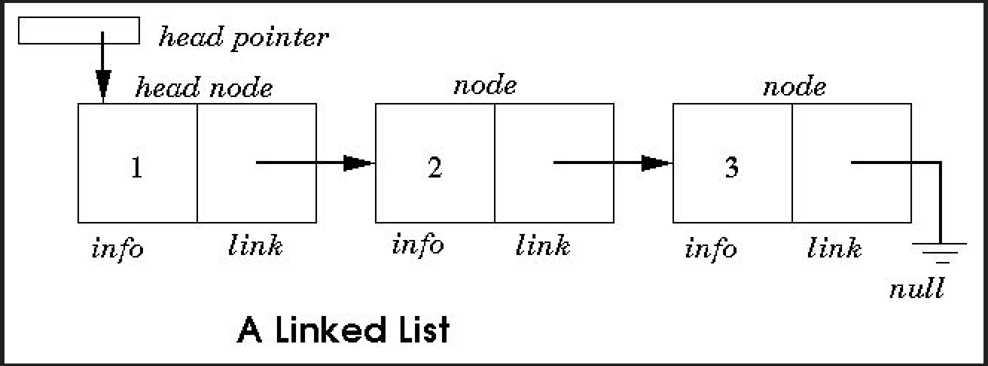


Data Structures - Stacks

Mast Gyan



[1]	[2]	[3]	[4]
5	1	3	4



A Linked List

A New Scenario

- Alice is a friend of Bob
- Alice is a friend of Carl
- David is friend with Fred
- Alice and Carl are friends with Fred

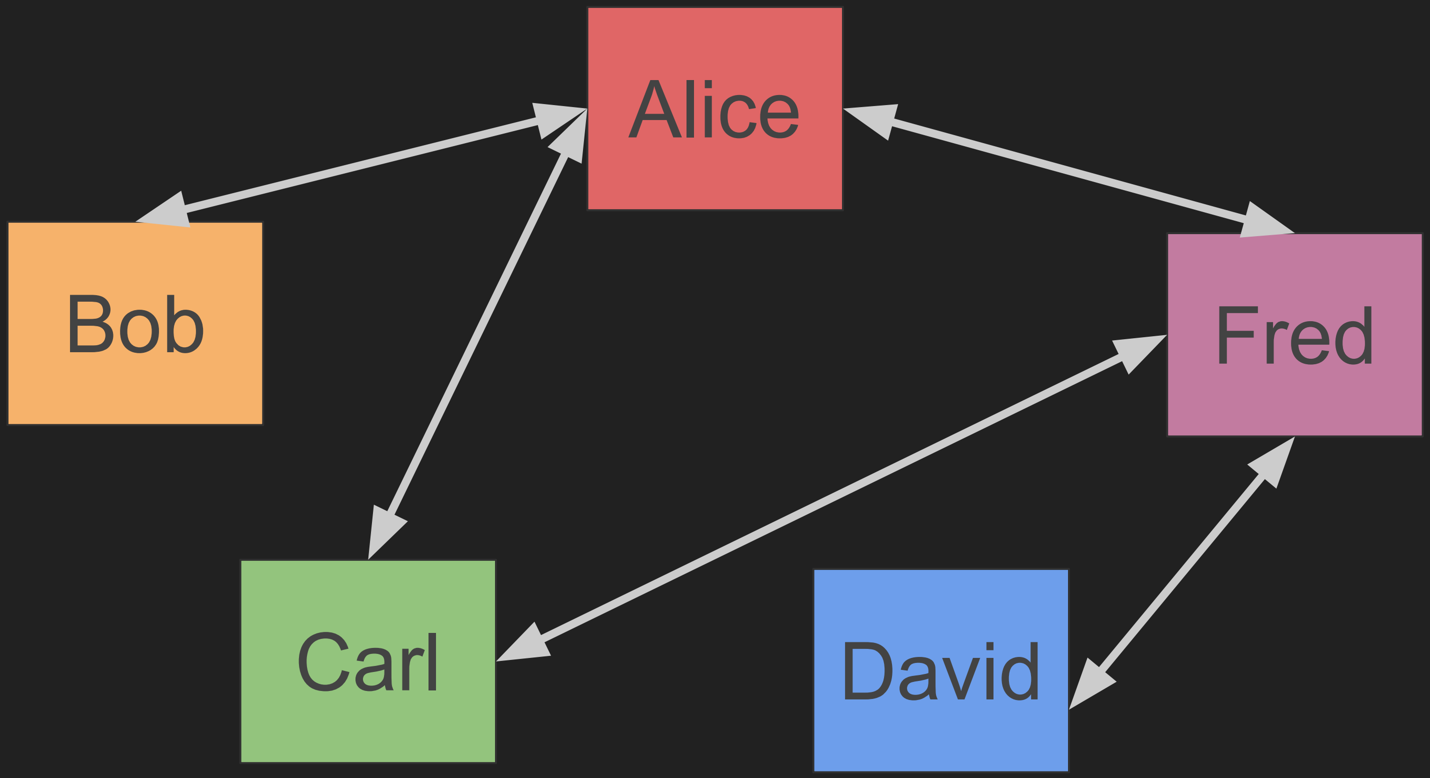
How should we represent this?

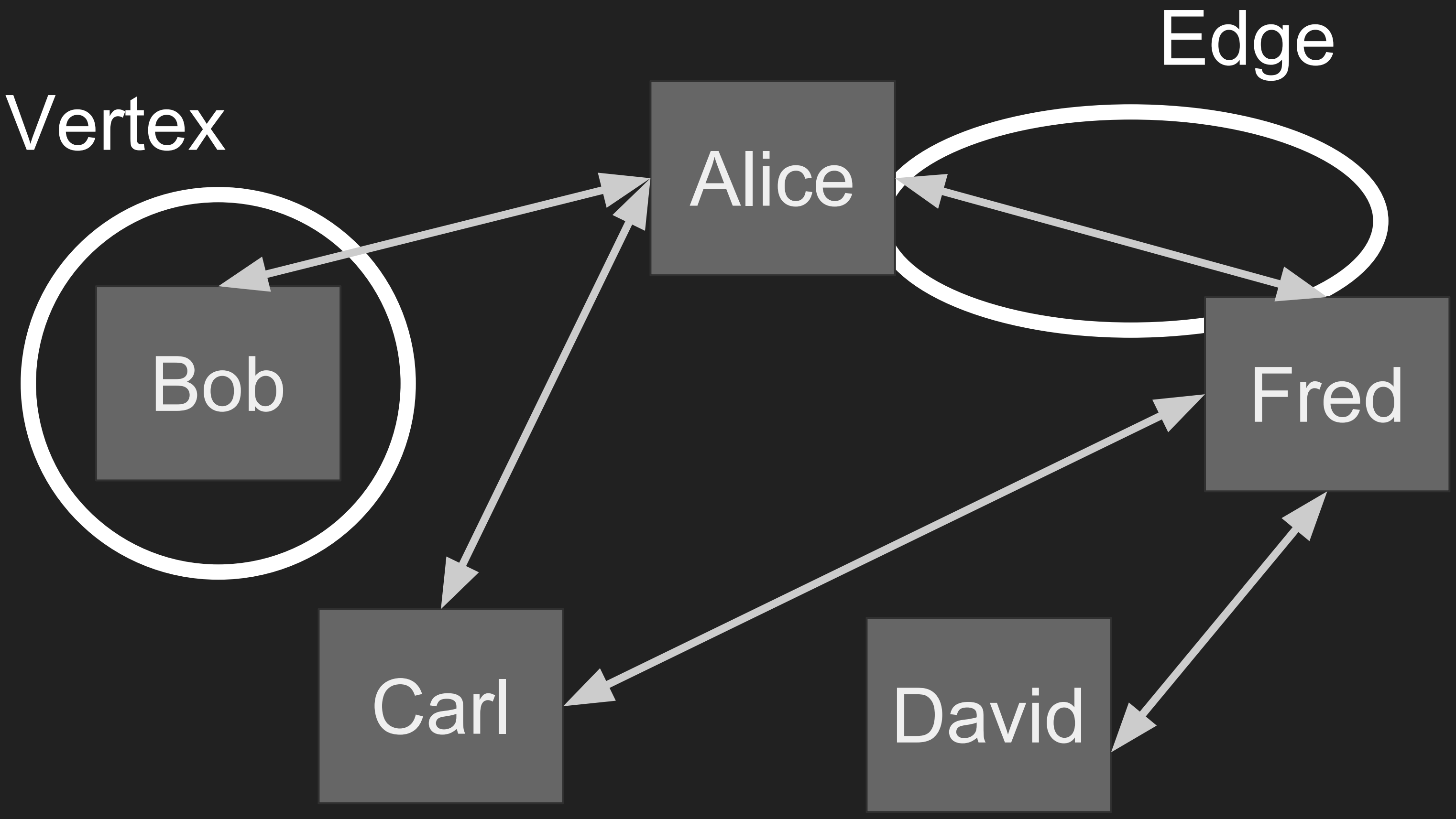
→ Alice is a friend of Bob

→ Alice is a friend of Carl

→ David is friend with Fred

→ Alice and Carl are friends with Fred

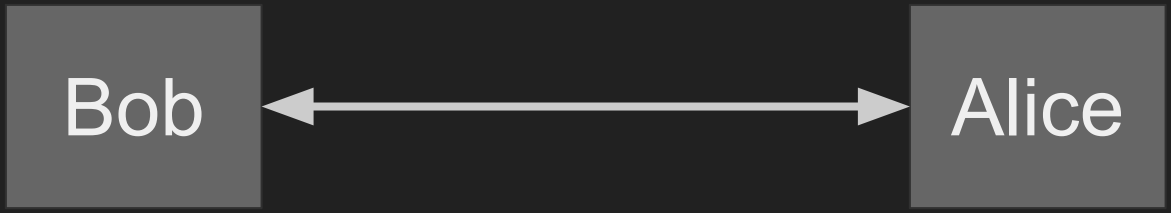




How many edges / vertices are present in the graph above?

How are they described?

Describing A Graph



A **vertex** is called by its name (e.g. Alice, Bob)

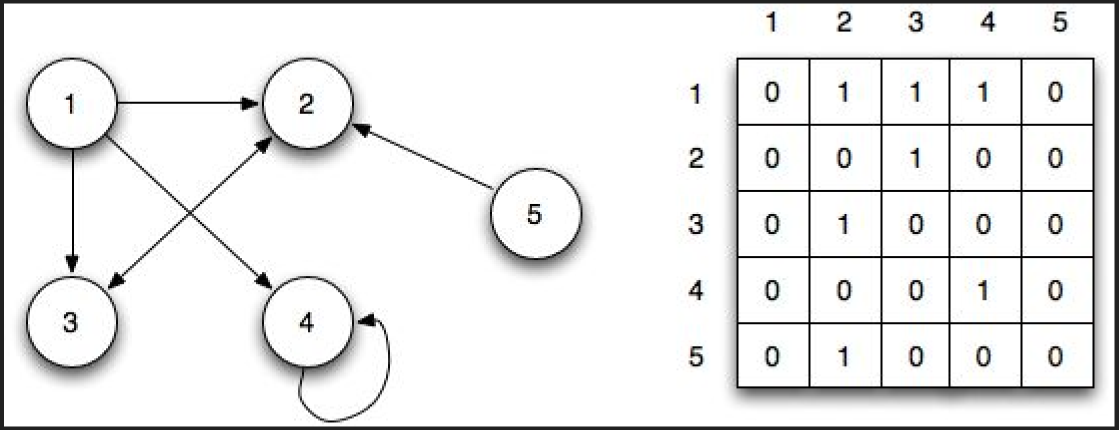
A **edge** is defined using the following properties:

- ➔ The 2 vertices it connects, and
- ➔ The direction of the edge

(e.g. A bidirectional edge connecting Alice and Bob)

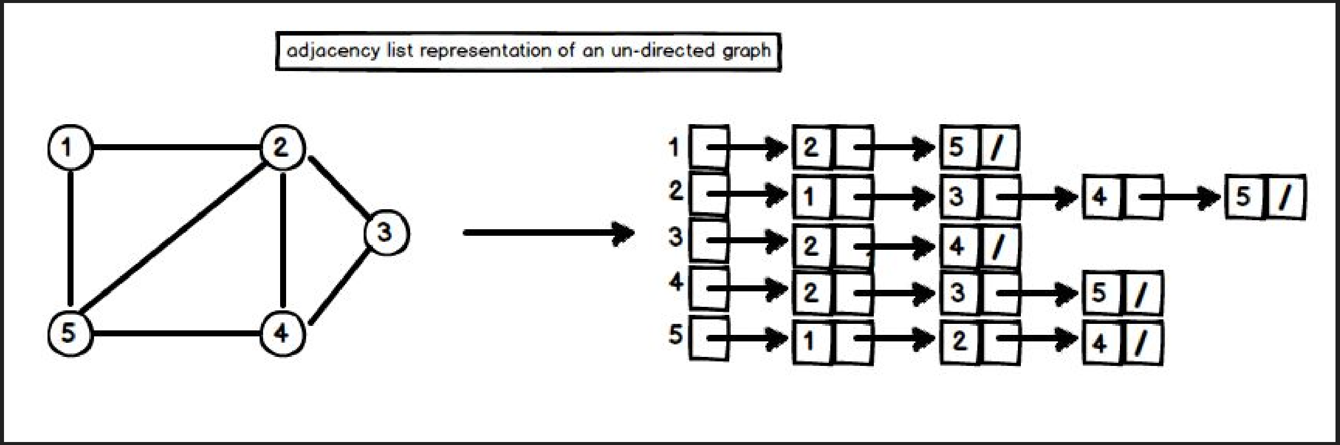
Representing Graphs

- Adjacency Matrix



	1	2	3	4	5
1	0	1	1	1	0
2	0	0	1	0	0
3	0	1	0	0	0
4	0	0	0	1	0
5	0	1	0	0	0

- Adjacency List


adjacency list representation of an un-directed graph

Graph Traversal Methods

Breadth-First Search (BFS)

visits the child vertices before visiting the sibling vertices

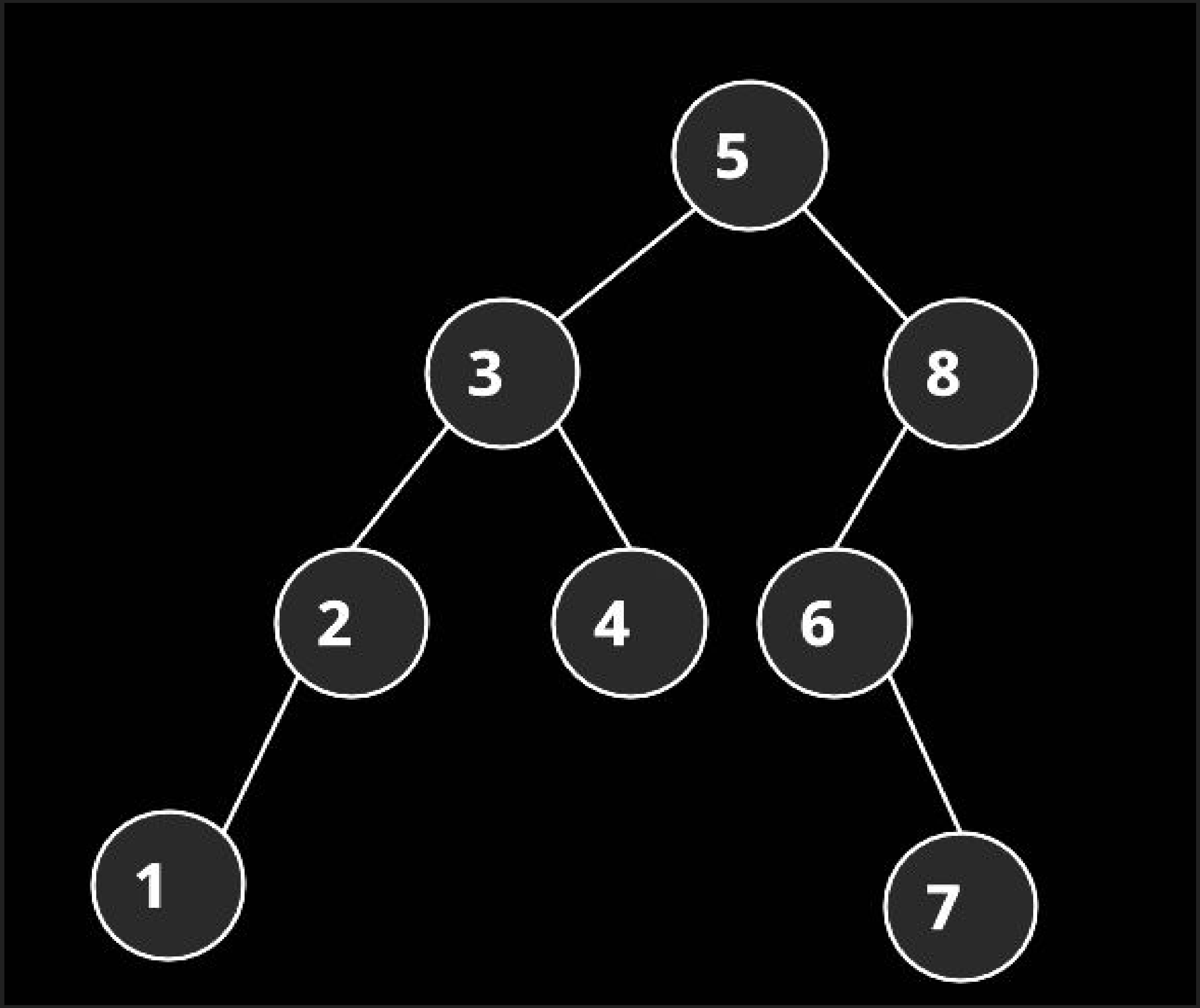
uses queues as data structure

Depth-First Search (DFS)

visits the neighbor vertices before visiting the child vertices

uses stacks as data structure

Demo



Graph Theory Algorithms

- **Shortest Path Problems**
  - Dijkstra, Bellman-Ford, SPFA, A*, Floyd-Warshall
- **Minimum Spanning Trees**
  - Prim’s, Kruskal’s
- **Strongly-Connected Components (SCC)**
  - Tarjan’s Algorithm
- **Maximum Bipartite Matching**
  - Hungarian Algorithm, Ford-Fulkerson
- etc...