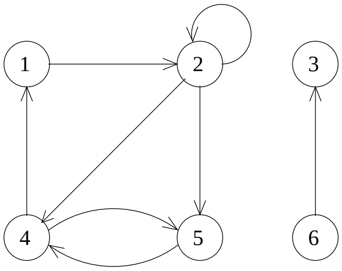
Graph basics

(CLRS B.4-B.5, 22.1)

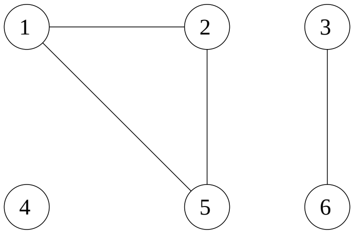
- A graph $G = (V, E)$ consists of a set of *vertices* V and a set of *edges* E.
- Vertices are objects, and edges model relationships. E.g: graphs can model a network of people, where edges mean friendships (the friendship graph).
- There are two types of graphs: directed and undirected.
- *Directed graph*: E is a set of ordered pairs of vertices (u, v) where $u, v \in V$



V = {1, 2, 3, 4, 5, 6}

E = {(1,2), (2,2), (2,4), (2,5)
(4,1), (4,5), (5,4), (6,3)

- *Undirected graph*: E is a set of unordered pairs of vertices $\{u, v\}$ where $u, v \in V$



V = {1, 2, 3, 4, 5, 6}

E = {{1,2}, {1,5}, {2,5}, {3,6}}

- Note: Some definitions do not allow self loops (graphs are simpler to work with).
- Often the edges (u, v) in a graph have weights $w(u, v)$, e.g.
  - Road networks (distances)
  - Cable networks (capacity)

Example

- Graphs appear all over the place in all kinds of applications.

  Example: the friendship graph: vertices are people, and edges are friendships. Some relevant questions:

  - is it directed or undirected? (if I am your friend, does it mean you are my friend??)
  - Are two people friends? How close?
  - Am I linked by some chain of friends to a star?
  - How close am I to a star?
  - Is there a path between any two people in the graph?
  - Who has the most friends?
  - What is the largest clique?

Terminology

- Edge (u, v) is said to be *incident* to u and v
- Undirected graph: *Degree* of vertex is the number of edges incident to it.
- Directed graph: *In (out) degree* of a vertex is the number of edges entering (leaving) it.
- Paths: A *path* from u_1 to u_2 is a sequence of vertices $< u_1{=}v_0, v_1, v_2, \cdots, v_k{=}u_2 >$ such that $(v_i, v_{i+1}) \in E$ (or $\{v_i, v_{i+1}\} \in E$). If $v_0 = v_k$ then it is a *cycle*. The length of a path is the number of edges on it.
- Reachability: if there is a path from u_1 to u_2 we write $u_1 \leadsto u_2$ and we say that u_2 is *reachable* from u_1 (directed and undirected graphs).
- An undirected graph is *connected* if every pair of vertices are connected by a path
  - The *connected components* are the equivalence classes of the vertices under the "reachability" relation. (All connected pair of vertices are in the same connected component).
- A directed graph is *strongly connected* if every pair of vertices are reachable from each other
  - The *strongly connected components* are the equivalence classes of the vertices under the "mutual reachability" relation.
- A graph is called a *tree* if there is a single path between any two vertices in G. It can be shown that a tree is connected, has no cycles, and the number of edges is precisely $|E| = |V| - 1$ (actually it can be shown that any two of these properties implies teh third one).

Graph Size

We will express algorithm running time (and memory) in terms of $|V|$ and $|E|$, often dropping the $|$'s. Sometimes $|V| = n$ and $|E| = m$.

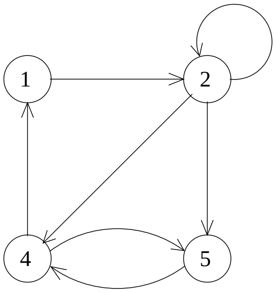
- the number of edges in a graph can be as liittle as 0 (not a very interesting graph): $|E| \geq 0$
- the largest number of edges in a graph is $|E| = O(|V|^2)$. The exact count depends on whether the graph is directed or not, and if self loops are allowed.
- if $|E| \sim |V|^2$ the graph is said to be dense
- if $|E| \sim |V|$ the graph is said to be sparse

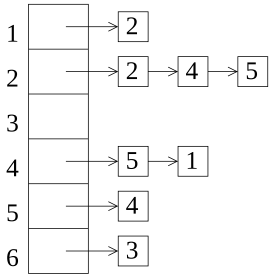
Graph representation

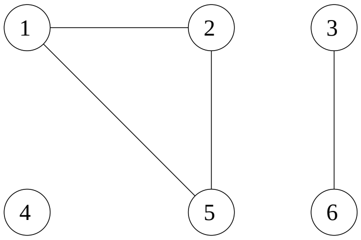
A graph can be repreented as adjacency list or matrix.

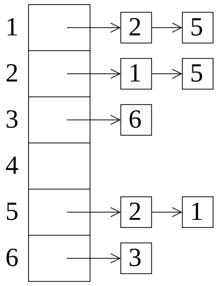
- *Adjacency-list* representation: Array of $|V|$ list of edges incident to each vertex.

  Examples:







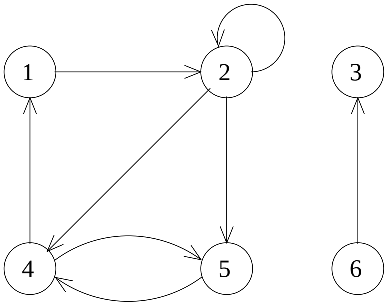


  - Note: For undirected graphs, every edge is stored twice.
  - If graph is weighted, a weight is stored with each edge.
  - How much space is required?

- *Adjacency-matrix* representation: $|V| \times |V|$ matrix A where

$$a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

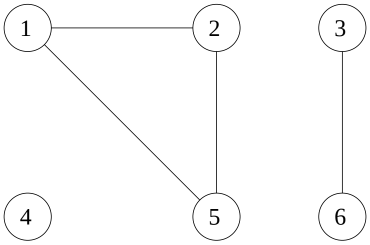
  Examples:



J

i

	1	2	3	4	5	6
1	0	1	0	0	0	0
2	0	1	0	1	1	0
3	0	0	0	0	0	0
4	1	0	0	0	1	0
5	0	0	0	1	0	0
6	0	0	1	0	0	0



$$\begin{array}{cc|c|c|c|c|c|c|}
 & \multicolumn{7}{c}{\text{j}} \\
 & & 1 & 2 & 3 & 4 & 5 & 6 \\ \cline{3-8}
 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ \cline{3-8}
 & 2 & 1 & 0 & 0 & 0 & 1 & 0 \\ \cline{3-8}
\text{i} & 3 & 0 & 0 & 0 & 0 & 0 & 1 \\ \cline{3-8}
 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ \cline{3-8}
 & 5 & 1 & 1 & 0 & 0 & 0 & 0 \\ \cline{3-8}
 & 6 & 0 & 0 & 1 & 0 & 0 & 0 \\ \cline{3-8}
\end{array}$$

- Note: For undirected graphs, the adjacency matrix is symmetric along the main diagonal ($A^T = A$).
- If graph is weighted, weights are stored instead of one's.
- How much space is required?

- Comparison of matrix and list representation:

Adjacency list	Adjacency matrix
$O(\lvert V\rvert + \lvert E\rvert)$ space	$O(\lvert V\rvert^2)$ space
Good if graph *sparse* ($\lvert E\rvert << \lvert V\rvert^2$)	Good if graph *dense* ($\lvert E\rvert \approx \lvert V\rvert^2$)
No quick access to (u, v)	$O(1)$ access to (u, v)

- We will use adjacency list representation unless stated otherwise ($O(|V| + |E|)$ space).
- Interesting: large graphs are sparse.