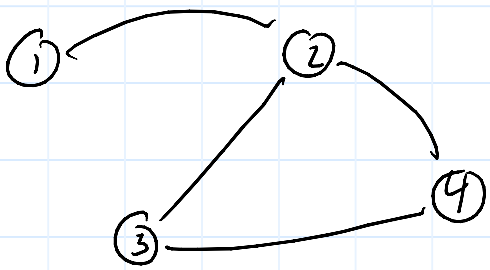
Note Title 5/14/2013

Graph Terminology

An undirected Graph is a pair (V, E)

Where V is a set of vertices
and E is a set of unordered pairs from V.
The elements of E are called edges.



$$V = \{1, 2, 3, 4\}$$

$$E = \{\{1,2\}, \{2,3\}, \{2,4\}, \{3,4\}\}$$

for $v \in V$ $w \in V$, if $\{v, w\} \in E$

we say v is adjacent to w.

v and w are incident to $\{v, w\}$

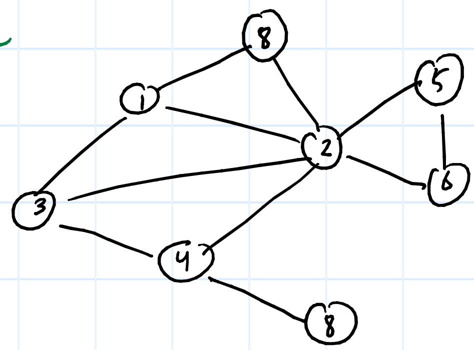
can also write vw for $\{v, w\}$

A subgraph of $G:(V,E)$ is a graph

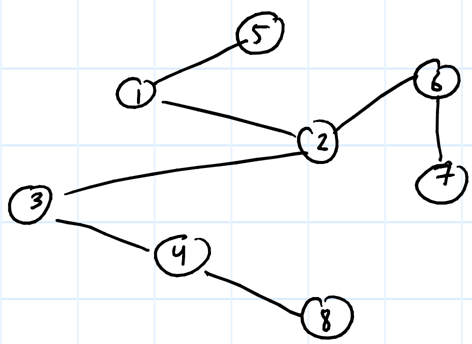
$G':(V',E')$ $\quad V'\subseteq V \quad E'\subseteq E$

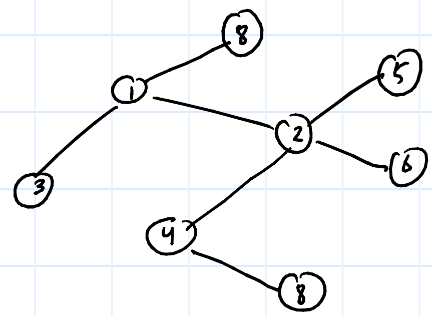
Example

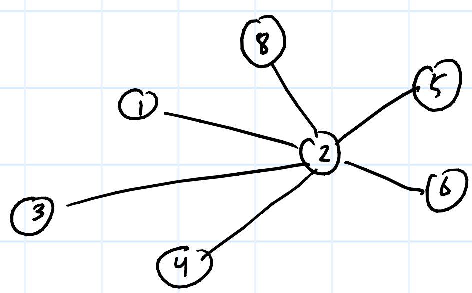
$G:$

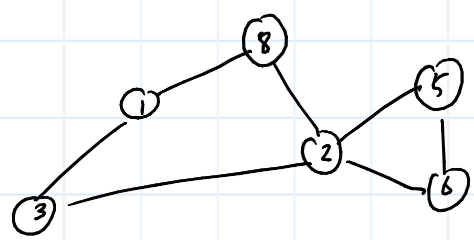


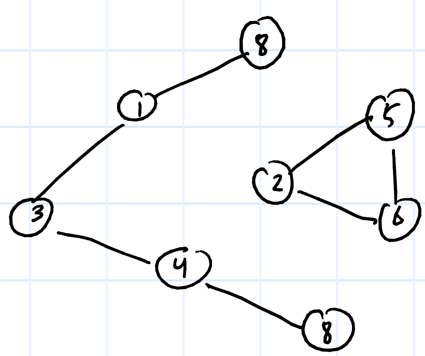
some subgraphs of G

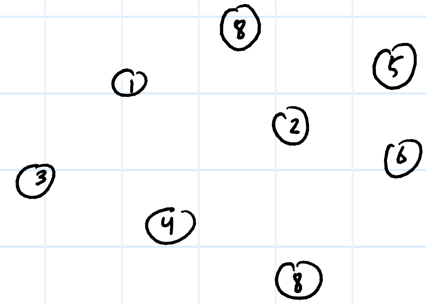












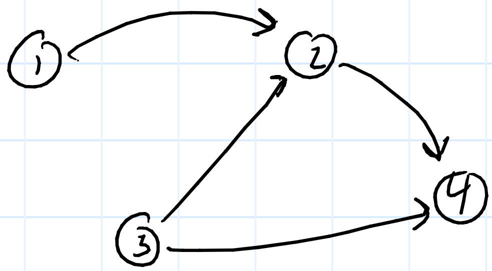
A directed graph ("digraph")

A directed Graph is a pair (V, E)

Where V is a set of vertices

and E is a set of ordered pairs

The elements of E are called edges.



$V = \{1, 2, 3, 4\}$

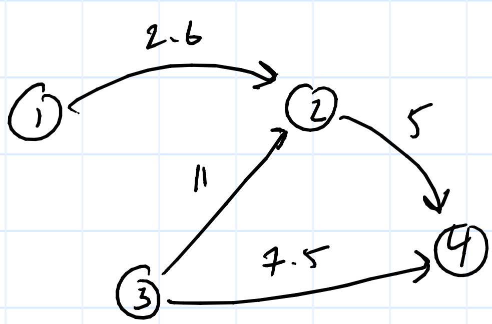
$E = \{(1,2), (3,2), (2,4), (3,4)\}$

A weighted graph is a triple (V, E, w)

where (V, E) is a graph and w

is a function from E to $\mathbb{R}$.

Example



$w((1,2)) = 2.6$

$w((2,4)) = 5$

$w((3,2)) = 11$

$w((3,4)) = 7.5$

A path of length k from v to w

is a sequence of edges $v_0v_1, v_1v_2, \dots, v_{k-1}v_k$

where $v_0 = v$ and $v_k = w$

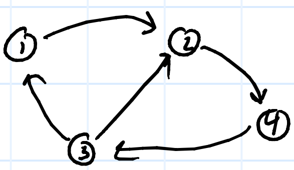
Example $(1,2),(2,4)$ is a path of length 2 in the digraph above.

Connected Graph: An undirected graph is connected iff $\forall v, w \in V$ there is a path from v to w.

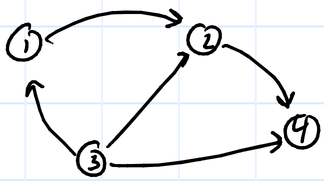
There's an analogous definition for digraphs in which case the digraph is strongly connected

If the associated undirected graph is connected then the digraph is weakly connected

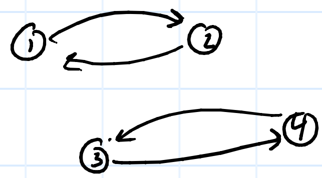
Example



strongly connected



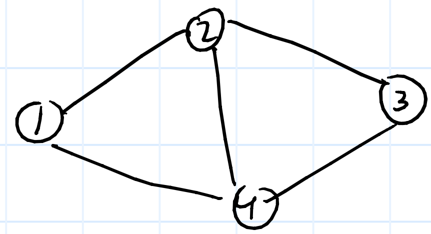
weakly connected



neither

A **cycle** is a path where first & last vertices are the same.

A **simple cycle** a cycle with no repeated vertices



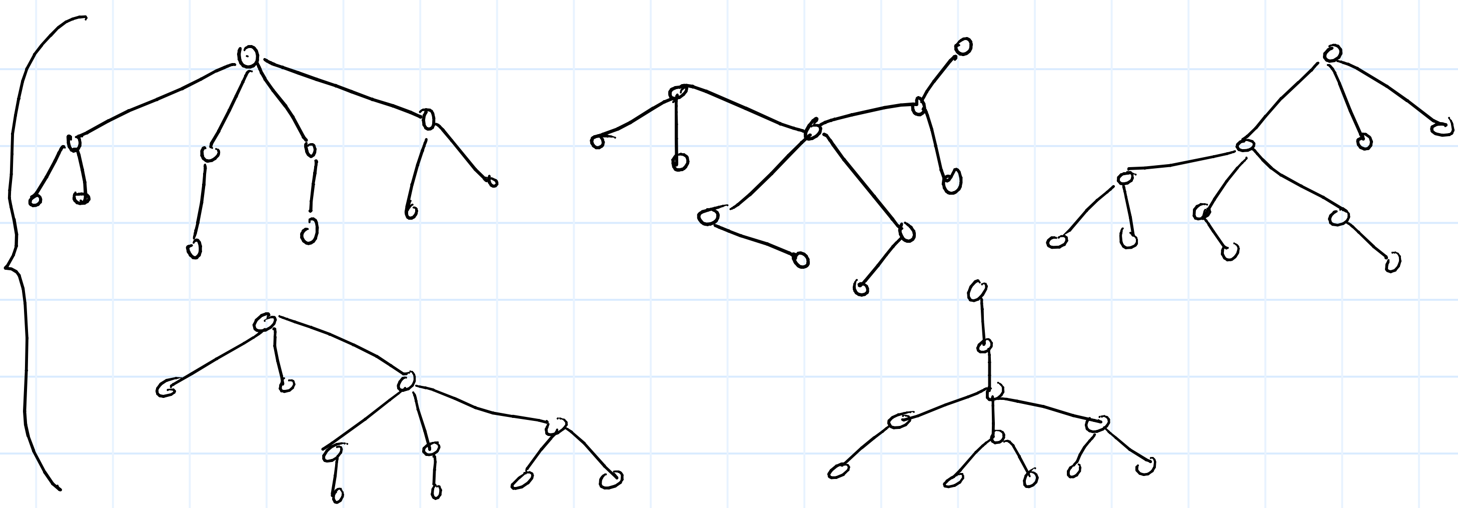
simple cycle: (1,2), (2,3), (3,4), (4,1)

cycle that's not simple (1,2), (2,3), (3,4), (4,2), (2,1)

A graph with no cycles is acyclic.

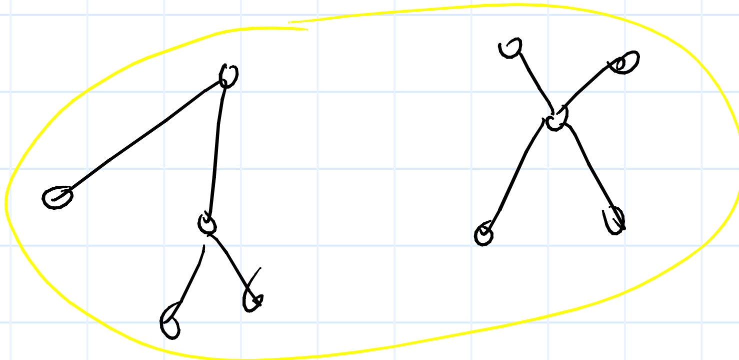
A **tree** is a connected acyclic graph.

All the same tree



Notice: This version of a tree has no root. If we designate a certain vertex as the root then it's a **rooted tree**.

A group of disconnected trees is a **forest**.



forest

To represent a graph on a computer there are two usual methods:

1) Adjacency Matrix
2) Adjacency Lists

Adj. matrix : A

$$\begin{array}{c c} & \begin{array}{cccc} 1 & 2 & \cdots & n \end{array} \\ \begin{array}{c} 1 \\ 2 \\ \vdots \\ n \end{array} & \left(\begin{array}{cccc} \; & \; & \; & \; \\ & & & \\ & & & \\ & & & \end{array} \right) \end{array}$$

$G = (V, E)$

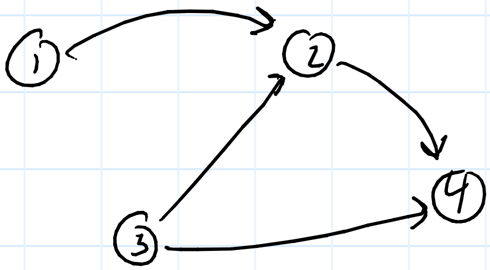
where $|V| = n$

$A(G)$ is an $n \times n$ matrix where

$A_{\underset{\text{row}}{i}\,\underset{\text{column}}{j}} = 1$ if $(v_i, v_j) \in E$.

and 0 otherwise

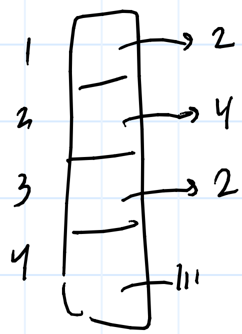
$G: \quad V = \{1,2,3,4\} \qquad E: \{(1,2),(3,2)$
$|V| = 4 \qquad\qquad\qquad (2,4), 3,4)\}$



$$
A(G) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}
\begin{array}{c}
\begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\
\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}
\end{array}
$$

Adjacency Lists

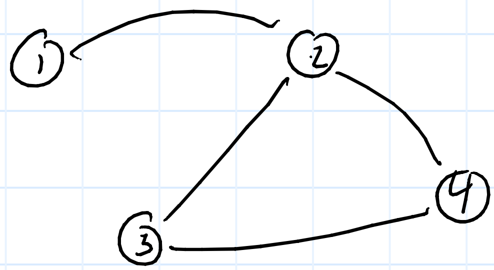
Array of lists indexed by vertex number.



vertex adjacent to 1
vertex adjacent to 2
two vertices adjacend to 3
no vertices adjacent to 4

Careful: The arrows represent **links** in a linked list - <u>not</u> edges in the graph

G:



$A(G)$:

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & & \\ 2 & & & & \\ 3 & & & & \\ 4 & & & & \end{array}\quad \begin{array}{c} \\ 1\\2\\3\\4 \end{array}\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 & 0 \\ \boxed{1 \;\; 0 \;\; 1 \;\; 1} \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}}_{A} = \underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & \boxed{3} & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{pmatrix}}_{A^2}$$

$1 \times 1 + 0 \times 0 + 1 \times 1 + 1 \times 1 = 3$

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ \boxed{0 \;\; 1 \;\; 0 \;\; 1} \\ 0 & 1 & 1 & 0 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{pmatrix}}_{A^2} = \underbrace{\begin{pmatrix} 0 & 3 & 1 & 1 \\ 3 & 2 & 4 & 4 \\ 1 & \boxed{4} & 2 & 3 \\ 1 & 4 & 3 & 2 \end{pmatrix}}_{A^3}$$

$0 \times 0 + 1 \times 3 + 0 \times 1 + 1 \times 1 = 4$

See if you can find the four paths from v_2 to v_4 of length 3.

A^{k}_{ij} = # paths of length k from v_i to v_j

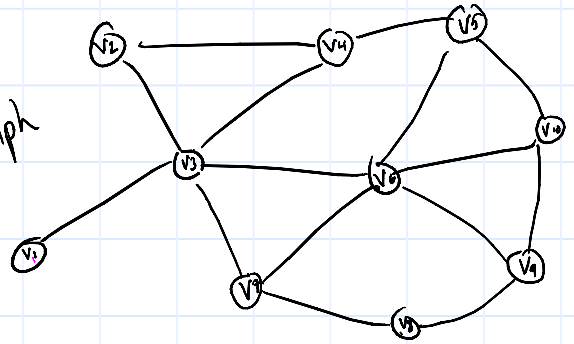
power

row

column

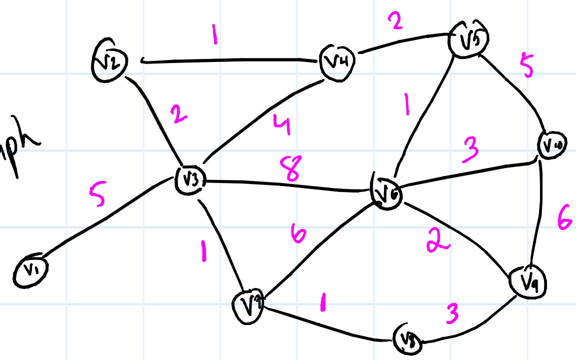
Later we'll see algorithms to find shortest path between any two vertices.
For weighted graph This is Dijkstra's Algorithm. For unweighted graph it's simply a breadth first search.

unweighted graph



a shortest path from v_1 to v_{10} is
$v_1 \, v_3 \, v_6 \, v_{10}$

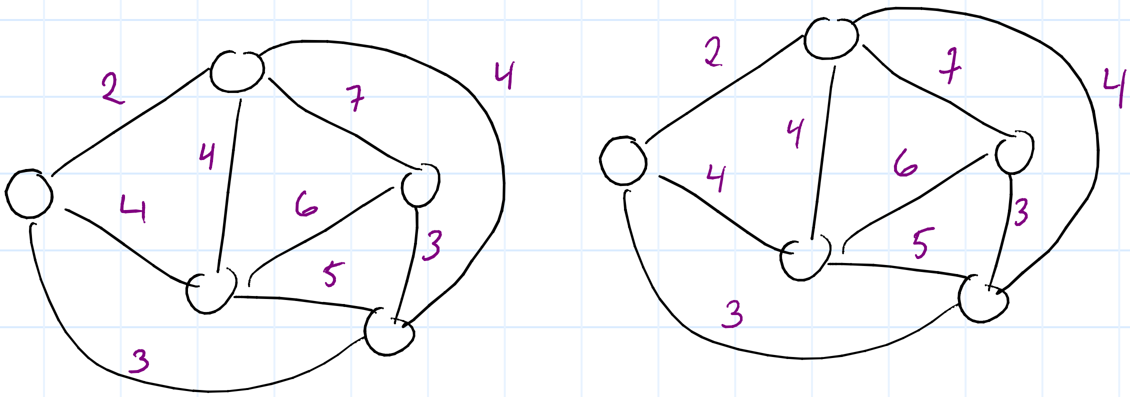
weighted graph



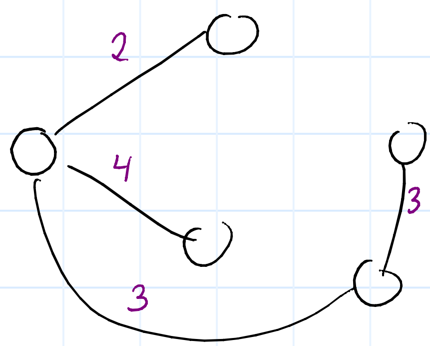
what is shortest path from v_1 to v_{10} in This graph?

A minimum spanning tree is a connected subgraph of connected undirected weighted graph G that has minimum total weight and contains all the vertices of G

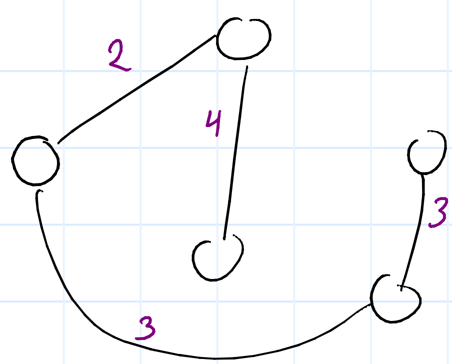
G:



an MST of G:

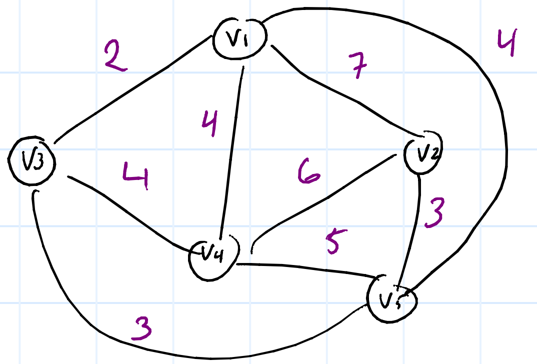


another MST of G



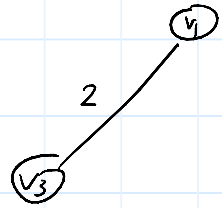
To find a MST use Prim's Algorithm:

Always choose edge of lowest weight to add to current tree, until all vertices are added. Start with any vertex for the initial tree.

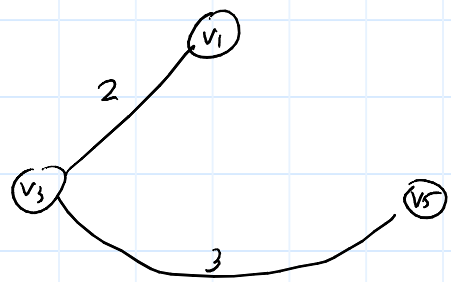


Start at V_1.

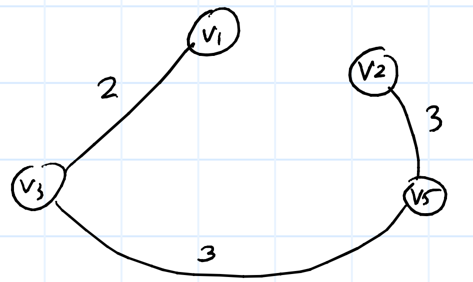
Add V_1V_3



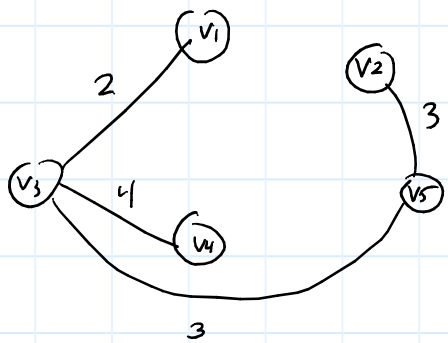
Add V_3V_5



Add V_5V_2



Add V_3V_4
(alternatively add V_1V_4)



No matter which vertex you start with and which choices you make (when there is a choice), the MST will have same total weight.