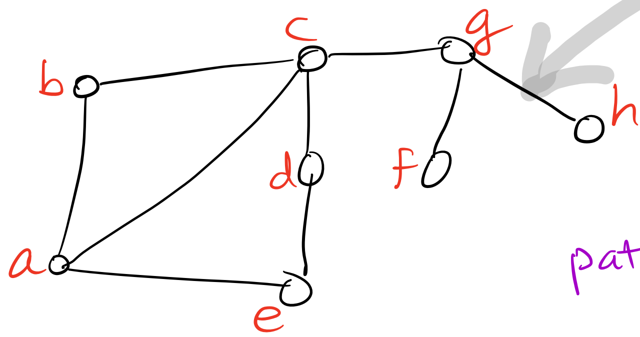
# Graphs

A graph is a set of vertices $V$ and an edge set $E \subseteq V \times V$.

$\{1,2\} \times \{a,b\}$
$= \{(1,a),(2,a),(1,b),(2,b)\}$

$\{1,2\} \times \{1,2\}$
$= \{(1,2),(1,1),(2,1),(2,2)\}$

(Undirected) graph $G = (V, E)$




path $a, b, c, g$

$a$

$V = \{a, b, c, d, e, f, g, h\}$

$(b, a$

$E = \{(a,b), (a,e), (b,c), (d,c)\ (e,d),$
$(c,g), (f,g), (g,h), (a,c)\}$

look like ordered pairs, but if graph is undirected, it same as $(h,g)$

sequence $v_1, v_2, \ldots v_k$

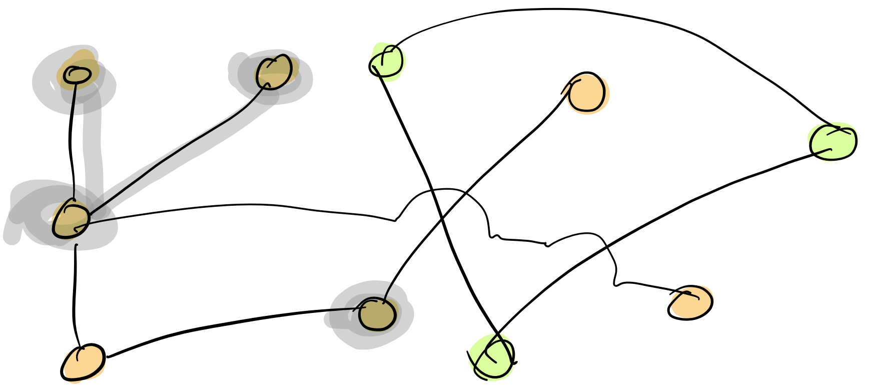
where $(v_i, v_{i+1}) \in E$ and

Path, simple cycle, each vertex appears no more than once.

length of path or cycle =

cycle = $v_1 v_2 \cdots v_k$ where $v_1 = v_k$ and $\nexists$ any (other) repeated vertices

Defn: A graph is <u>connected</u>



Defn: A graph is connected if $\forall u, v \in V$

$\exists$ a $u$-$v$ path in $G$.

Defn: $u$-$v$ path is a path from $u$ to $v$ in $G$.

Defn: A connected component of a graph $G$ is a maximal subgraph of $G$ that is connected.

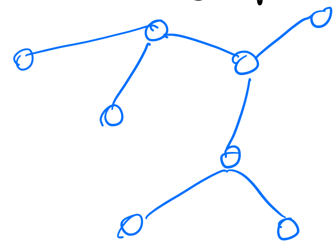
Defn: A <u>subgraph</u> of $G = (V, E)$ is

a graph $G'(V', E')$ such that

$V' \subseteq V$ and $E' \subseteq E$. (also, $E' \subseteq V' \times V'$)

Defn: An induced subgraph $G'' = (V'', E'')$ of $G = (V, E)$ has $V'' \subseteq V$, and $E'' = E \cap (V'' \times V'')$

Def$^{\underline{n}}$ A <u>tree</u> is an acyclic connected graph.

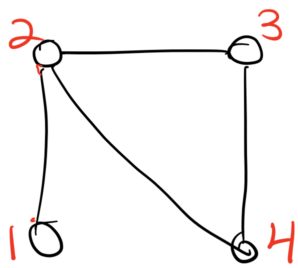


How many edges are in a <u>tree</u>?

$|V| = n$ $\qquad m = n-1$

$|E| = m$

## Graph Representation.




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

Adjacency Matrix representation.

Space $= \Theta(n^2)$.

How to determine if a graph is <u>Connected</u>

## Depth-First Search

Want to <u>visit</u> all nodes in the graph.
(Suppose they contain data, and you want to print that out)

```
Search_G_by_dfs (G)
    for  i = 1..n     marked [i] = false
    for  i = 1..n
         if  marked [i] = false
              dfs (G, i)


dfs (G, i)
    visit(i)
    marked [i] = true
    for each j where (i, j) ∈ E
         dfs (j)
```

BFS

# Breadth-First Search.