Professor Hamlin
Day 15

Agenda

1. Admin - HW 5 probability due Friday
2. Review
3. Graph Definitions
4. Graph representation
   - list of lists
   - adjacency matrix
5. Graph equivalence (isomorphism)

Happy Halloween!

Review

Bernoulli: - outcome of event w/ success/failure probability

Binomial: odds of suceeding k out of N times
Assumptions 1) same p for every trial
2) trials are independent

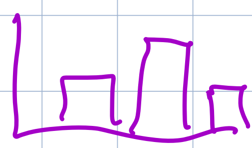
Poisson: odds of k events happening in time x (λ)
Assumpt 1) events are independent
2) rate had to be constant

Exercise | 1. "Probability of drawing 5 red cards from 52 card deck"
Which distribution & does it satisfy assumptions?
Binomial: but p is not the same!

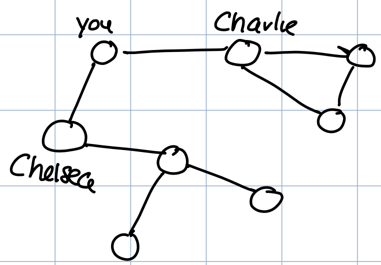
2. "Odds of 4 students coming to my office hours?"

Poisson : yes assumptions

Graphs :

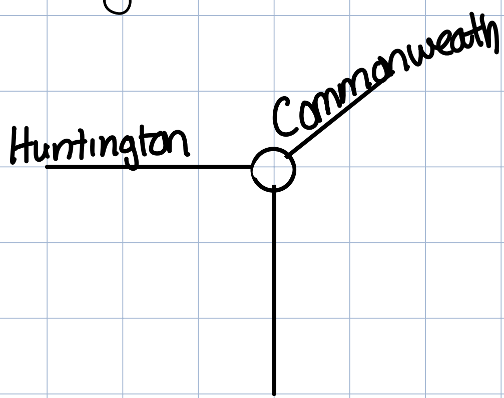


Real life examples:



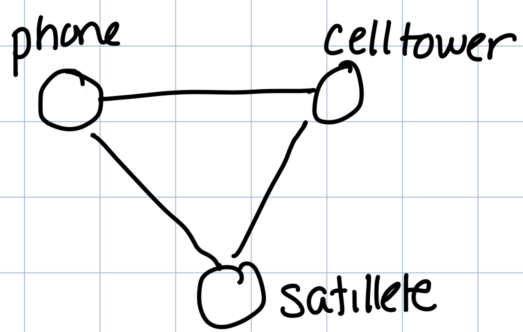
Social networks

people connected by friendship



Streets

intersections connected by roads



Internet

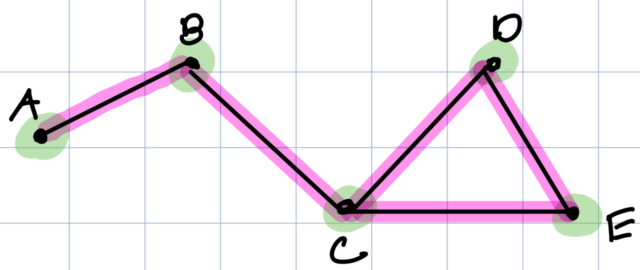
routers & devices connected by communication

8 8

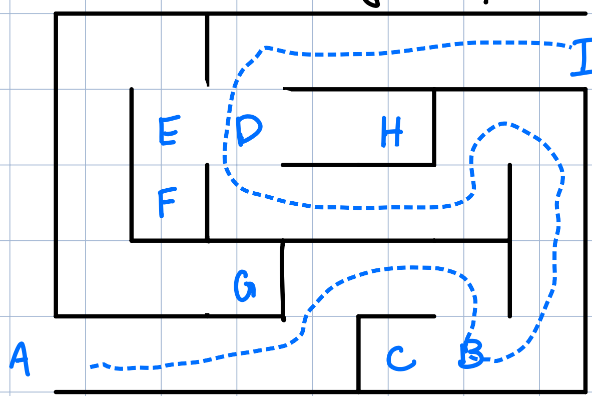
> Graphs are a fundamental way of representing real data & structures

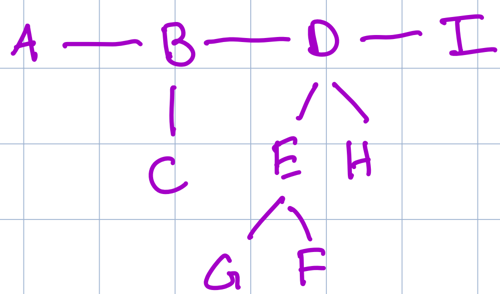
What are graphs (the data structure)

A set of vertices (nodes) and a set of edges connecting pairs of vertices



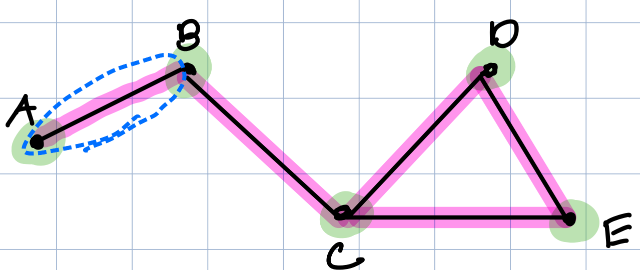
Example Node - intersection, deadend
Edge - possible movement





Clear representation helps us solve the problem

Formally representing graphs $G = (V, E)$



Vertices

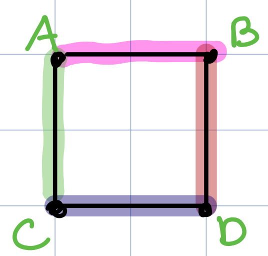
$V = \{A, B, C, D, E\}$

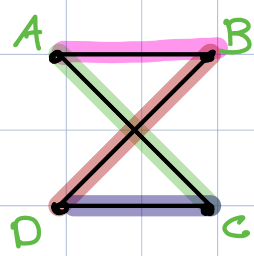
Edges

$E = \{(A,B), (B,C), (C,D), (C,E), (D,E)\}$

V = set of vertex names
E = set of tuples (pairs) of vertex names representing edges

Graph can be drawn differently and still be the same graph





$G = (V, E)$

$V = \{A, B, C, D\}$

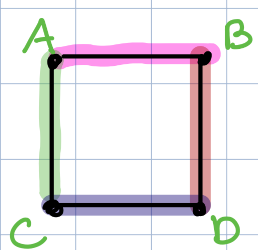
$E = \{(A,B), (A,C), (B,D), (C,D)\}$

Warning: lots of vocab, most is intuitive but check just to make sure

Definitions

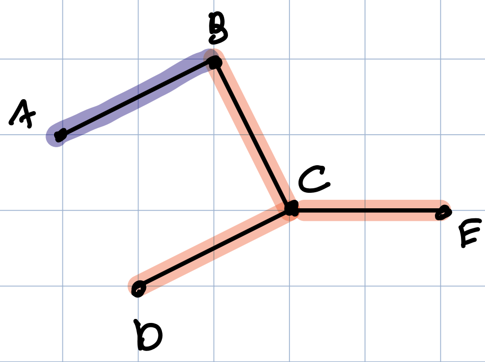
Adjacent: • two vertices are adjacent in a graph if they are connected by an edge

Adjacent: A & B   Not: A & D
A & C   C & B



- a vertex and edge are adjacent if vertex is in edge
  Adjacent: A & (A,B)   Not D & (A,C)
- two edges are adjacent if they share a vertex
  adjacent: (A,B) & (B,D)   Not: (A,B)
  (C,D)

Degree of vertex: number of edges which are adjacent to it.



Deg(A) = 1

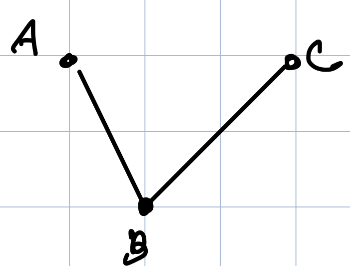
Deg(C) = 3

Deg(B) = 2

total degree: sum of the degrees of all the vertices

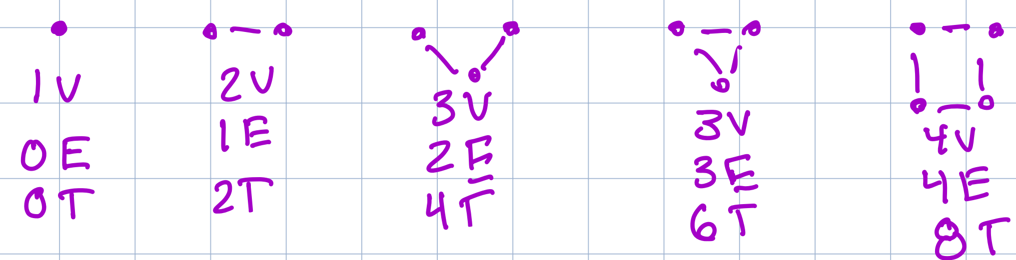
$$\text{Total Deg} = \underbrace{Deg(A)}_{1} + \underbrace{Deg(B)}_{2} + \underbrace{Deg(C)}_{3} + \underbrace{Deg(D)}_{1} + \underbrace{Deg(E)}_{1} = 8$$

Exercise | Draw a graph were the total degree is odd (or argue why it isn't possible)



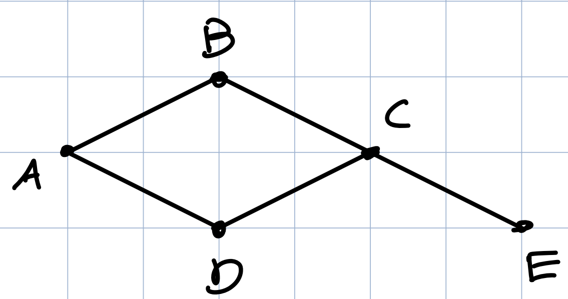
$$\underset{1}{Deg(A)} + \underset{2}{Deg(B)} + \underset{1}{Deg(C)} = 4$$

What is the relationship between the total degree and the number of edges in the graph? (try drawing some examples)



$$\boxed{\text{Total Degree} = 2 \cdot (\text{Total \# Edges})}$$

-



Walk: a sequence (ordered list) of adjacent edges (or equivalently adjacent vertices)

(A, B) (B, C) (C, B) = A B C B

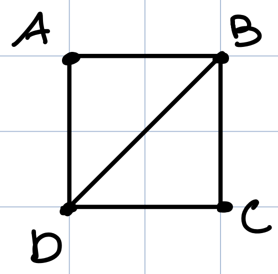
Path: a walk where each vertex is unique

Path: A B C E   Not: A B C B

Cycle: a path which starts & ends at same vertex (only one allowed to be not-unique)

Cycle: A B C D A   Not: A B C E

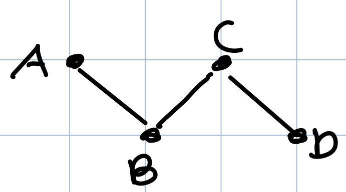
Exercise)



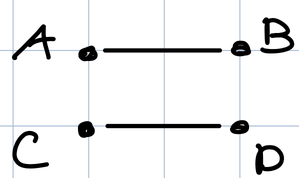
Identify the following as: cycle, path, or walk

1. A B D C
   Path (Walk)
2. A D C B D
   Walk
3. A B D A
   Walk, Cycle

Connected Graph: a graph is connected if there is a path from every node to every other node

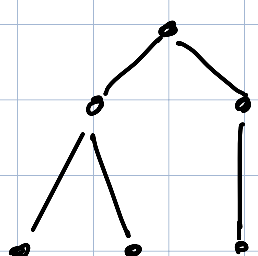


Connected

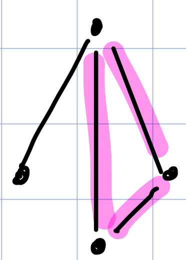


Not connected

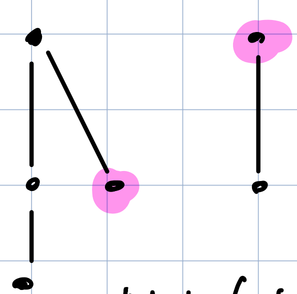
Tree: a connected graph without any cycles (acyclic)



Tree



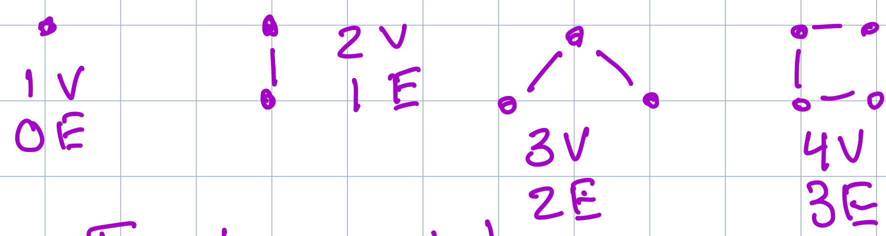
Not (has cycle)



Not (disconnected)

Trees are super common kinds of graphs

<u>Exercise1</u> What is the relationship between the $|V|$ (number of vertices) and $|E|$ (number of edges)?

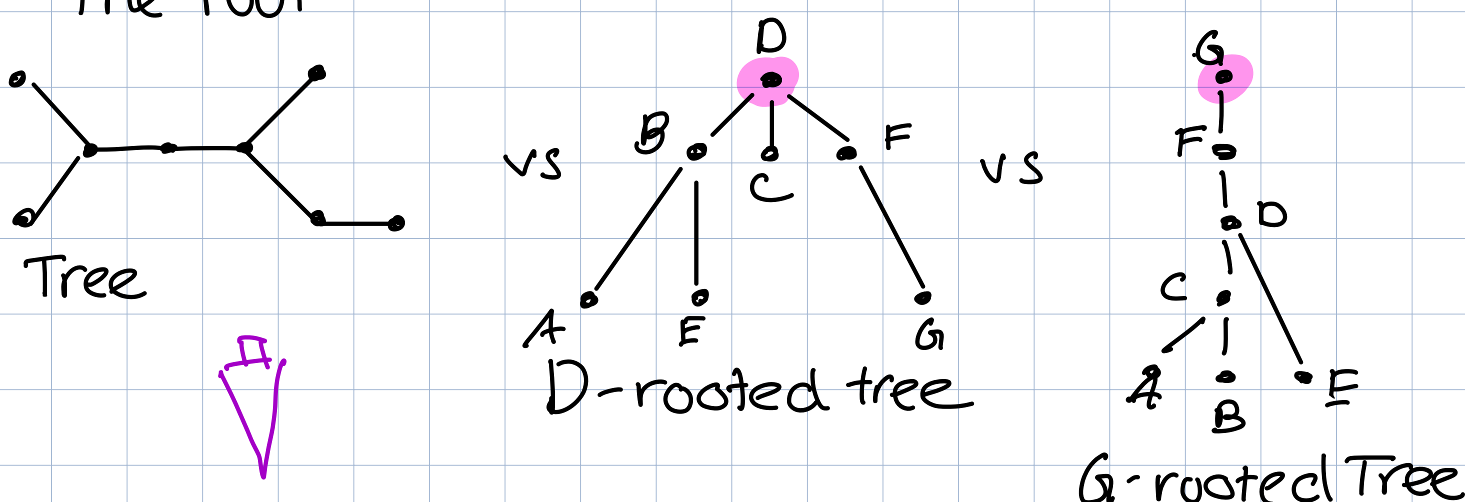


For trees only!

Total of # edges = # of vertices − 1

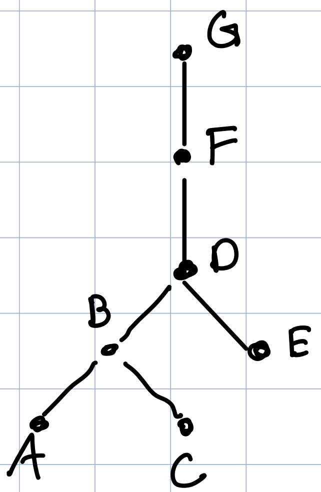
$$|E| = |V| - 1$$

<u>Rooted Trees:</u> a tree (connected, acyclic graph) which has a specific vertex identified by the root



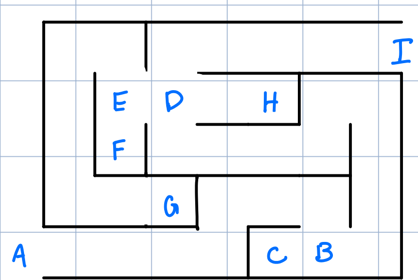
Convention: root of tree is drawn on top

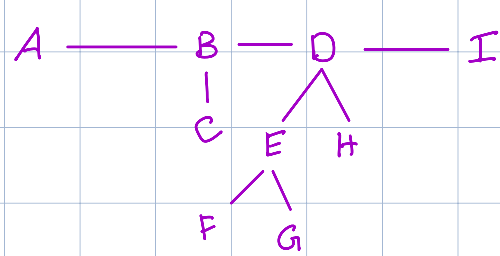
Why care about rooted trees? Allows us to define relationships between vertices



- **Parent** (of x): next vertex on path to the root (which has no parent)
  e.g. D is parent of B
- **Children** (of x): set of all vertices that x is a parent of
  e.g. {B, E} are D's children
- **Leaf**: a vertex with no children
  e.g. A, C, E are leaves
- **Sibling** (of x): set of vertices that have same parent as x
  e.g. C is A's sibling
- **Ancestor** (of x): all vertices along path from x to the root
  e.g. B, D, F, G are A's ancestors
- **Descendent** (of x): all nodes that have x as an ancestor
  e.g. B, E, A, C are D's descendants

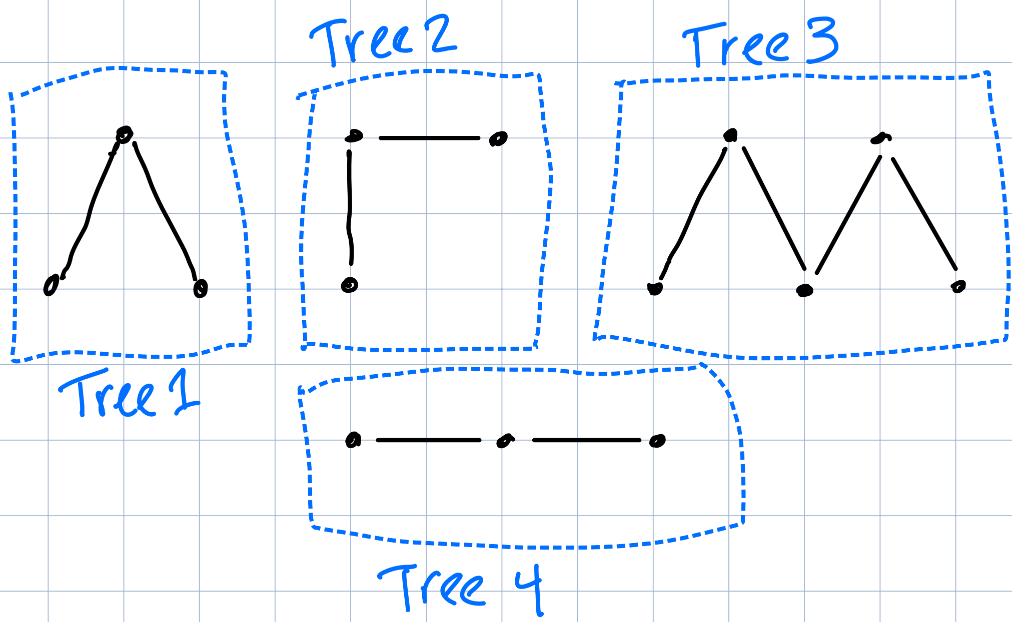
Our maze was a rooted tree w/ A as a root!



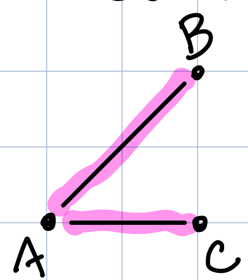


We can also have a graph made of several trees

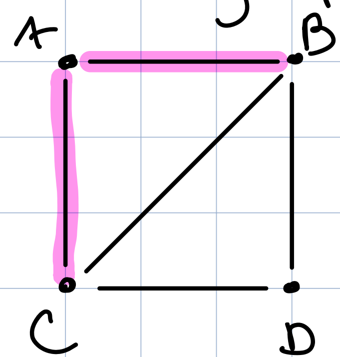
Forest: any acyclic graph (notice doesn't have to be connected!)



Subgraph: a graph whose vertices & edges are contained within another graph



is subgraph



$G_1 = (V_1, E_1)$

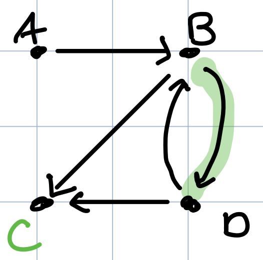
$G_2 = (V_2, E_2)$

$V_1 \subseteq V_2$

$E_1 \subseteq E_2$

We so far have been defining general graphs but there are a few special cases

Directed: each edge has direction (e.g. one way roads)

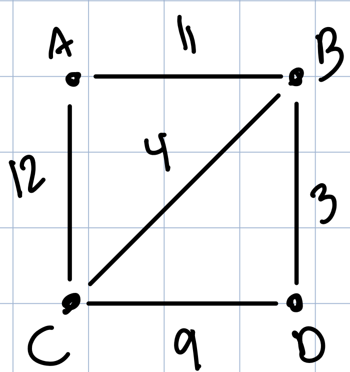


$V = \{A, B, C, D\}$

$E = \{(A,B), (B,C), (D,C), (B,D), (D,B)\}$

order indicates directi

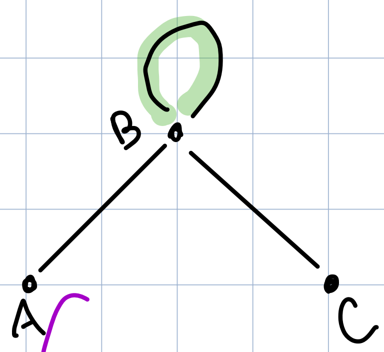
Weighted: each edge has a weight



$V = \{A, B, C, D\}$

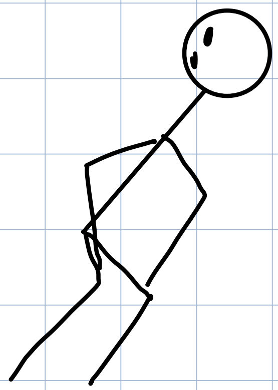
$E = \{(A,B,11), (A,C,12), (C,B,4), (C,D,9), (B,D,3)\}$

Non-simple: edge may start/end at same vertex



$V = \{A, B, C\}$

$E = \{(A,B), (B,B), (B,C)\}$

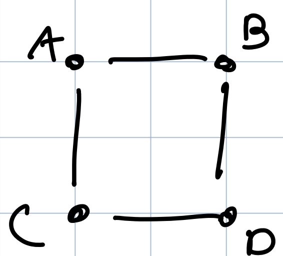


Let's take a breath, a lot of vocab

<u>Good news:</u> almost done
can just look back if you have questions

<u>Bad news:</u> sometimes def'ns are inconsistent (vertex vs. node, is a vertex it's own ancester)

<u>Neighbors:</u> two vertices are neighbors if they are adjacent (connected by an edge)



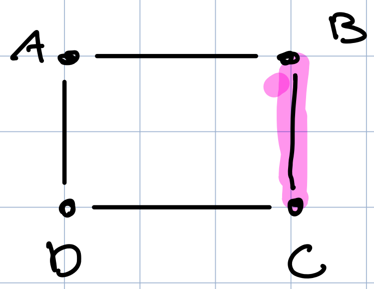
A & B Neighbors

A & D Not

How do we represent graphs on computers? (remember computers think in 0/1)

<u>Approach 1</u>: Adjacency list

idea: just list neighbors for each vertex

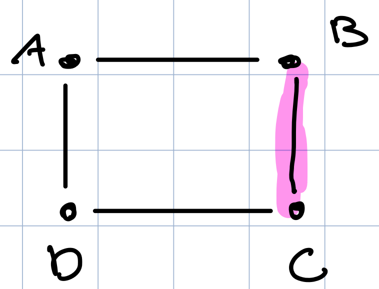


A: [B, D]
B: [A, C]
D: [A, C]
C: [B, D]

Good for graphs w/ large # of vertices and small # of edges

Approach 2: Adjacency Matrix

idea: have matrix, $|V| \times |V|$, 1 in row i, column j means edge between i & j



	A	B	C	D
A	0	1	0	1
B	1	0	1	0
C	0	1	0	1
D	1	0	1	0

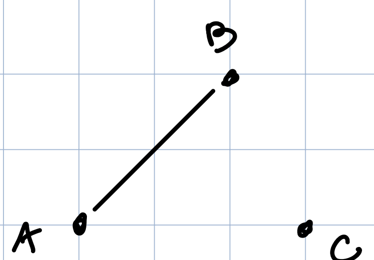
→ Typo in notes was 0

There is edge between B, C

Convention: a node is not it's own neighbor

Notice the symmetry

Exercise | Given one representation of a graph construct the other two (image, list, matrix)

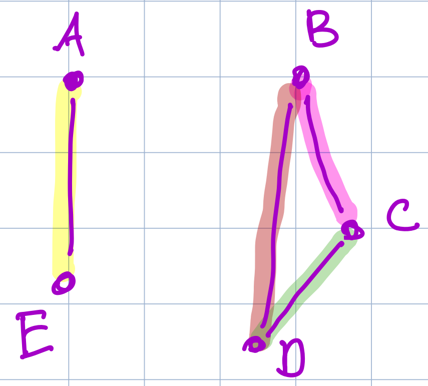


A: [B]
B: [A]
C: [ ]

	A	B	C
A	0	1	0
B	1	0	0
C	0	0	0

	A	B	C	D	E
A	0	0	0	0	1
B	0	0	1	1	0
C	0	1	0	1	0
D	0	1	1	0	0
E	1	0	0	0	0

A: [E]
B: [C, D]
C: [B, D]
D: [B, C]
E: [A]

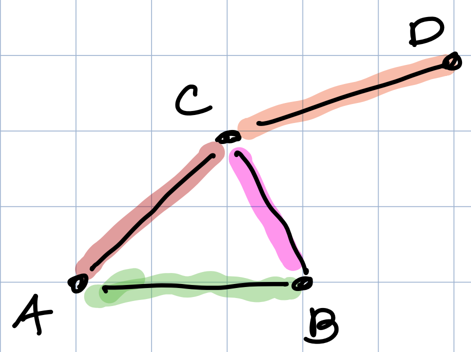


Graph isomorphism:

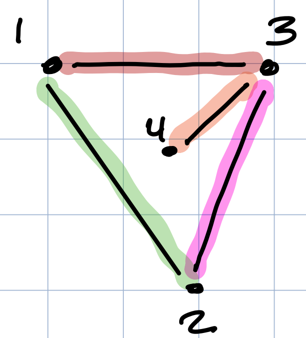
iso morphic
"same" "shape"

<u>Idea</u>: two graphs are isomorphic when they have the same shape

e.g. when we can rename* the nodes of one to get another



A = 1
B = 2
C = 3
D = 4



*rename = one-to-one mapping (bijection)