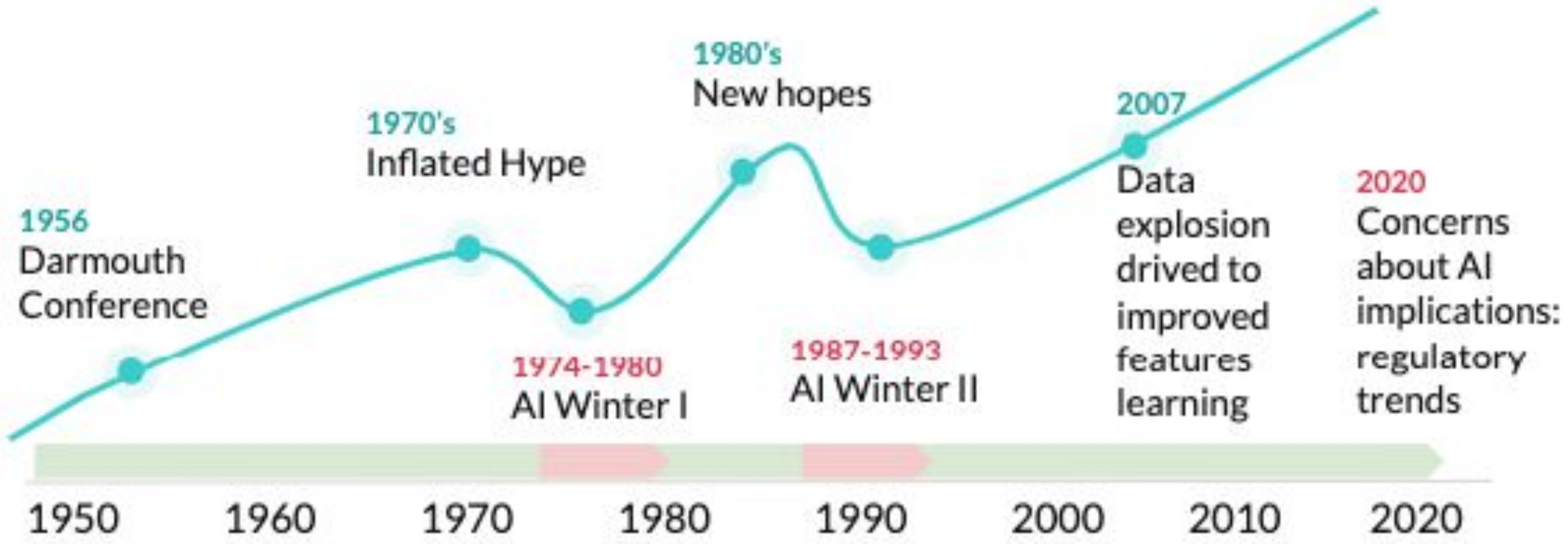
Artificial Intelligence

Kushal Shah @ Sitare


1956
Darmouth
Conference
1970's
Inflated Hype
1974-1980
AI Winter I
1980's
New hopes
1987-1993
AI Winter II
2007
Data
explosion
drived to
improved
features
learning
2020
Concerns
about AI
implications:
regulatory
trends
1950
1960
1970
1980
1990
2000
2010
2020

Eye-catching advances in some AI fields are not real

When tuned up, old algorithms can match the abilities of their successors

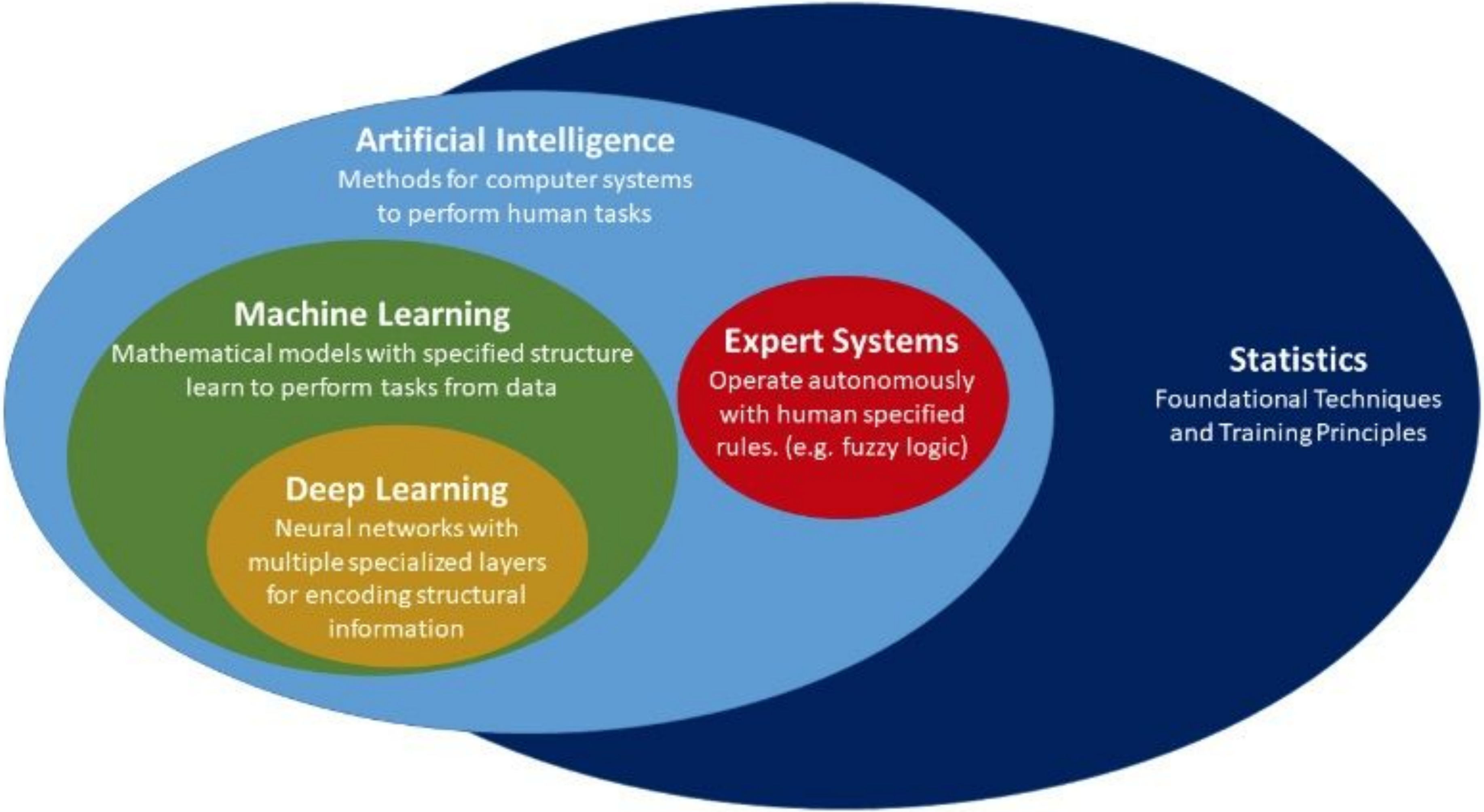
27 MAY 2020 · BY MATTHEW HUTSON

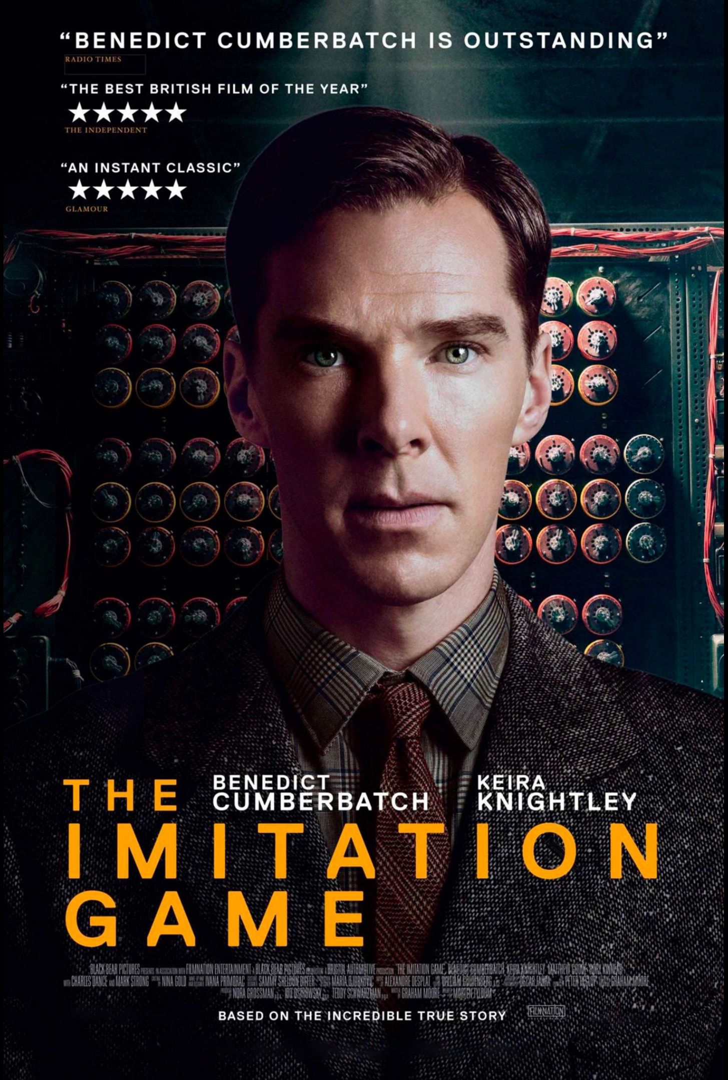
Researchers are waking up to the signs of shaky progress across many subfields of AI. A 2019 meta-analysis of information retrieval algorithms used in search engines concluded the "high-water mark … was actually set in 2009." Another study in 2019 reproduced seven neural network recommendation systems, of the kind used by media streaming services. It found that six failed to outperform much simpler, nonneural algorithms developed years before, when the earlier techniques were fine-tuned, revealing "phantom progress" in the field. In another paper posted on arXiv in March, Kevin Musgrave, a computer scientist at Cornell University, took a look at loss functions, the part of an algorithm that mathematically specifies its objective. Musgrave compared a dozen of them on equal footing, in a task involving image retrieval, and found that, contrary to their developers' claims, accuracy had not improved since 2006. "There's always been these waves of hype," Musgrave says.

Mastering the game of Go with deep neural networks and tree search

David Silver[1]*, Aja Huang[1]*, Chris J. Maddison[1], Arthur Guez[1], Laurent Sifre[1], George van den Driessche[1], Julian Schrittwieser[1], Ioannis Antonoglou[1], Veda Panneershelvam[1], Marc Lanctot[1], Sander Dieleman[1], Dominik Grewe[1], John Nham[2], Nal Kalchbrenner[1], Ilya Sutskever[2], Timothy Lillicrap[1], Madeleine Leach[1], Koray Kavukcuoglu[1], Thore Graepel[1] & Demis Hassabis[1]

The game of Go has long been viewed as the most challenging of classic games for artificial intelligence owing to its enormous search space and the difficulty of evaluating board positions and moves. Here we introduce a new approach to computer Go that uses 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. Without any lookahead search, the neural networks play Go at the level of state-of-the-art Monte Carlo tree search programs that simulate thousands of random games of self-play. We also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.


Artificial Intelligence
Methods for computer systems to perform human tasks
Machine Learning
Mathematical models with specified structure learn to perform tasks from data
Deep Learning
Neural networks with multiple specialized layers for encoding structural information
Expert Systems
Operate autonomously with human specified rules. (e.g. fuzzy logic)
Statistics
Foundational Techniques and Training Principles



"BENEDICT CUMBERBATCH IS OUTSTANDING"
RADIO TIMES
"THE BEST BRITISH FILM OF THE YEAR"
★★★★★
THE INDEPENDENT
"AN INSTANT CLASSIC"
★★★★★
GLAMOUR
BENEDICT CUMBERBATCH
KEIRA KNIGHTLEY
THE IMITATION GAME
BASED ON THE INCREDIBLE TRUE STORY
FILMNATION

Can machines think?

Turing Test

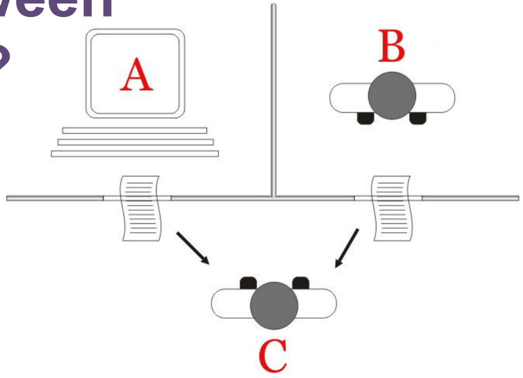
[Alan Turing 1950]

Can machines think?

Can humans tell the difference between machines and one of their kind?

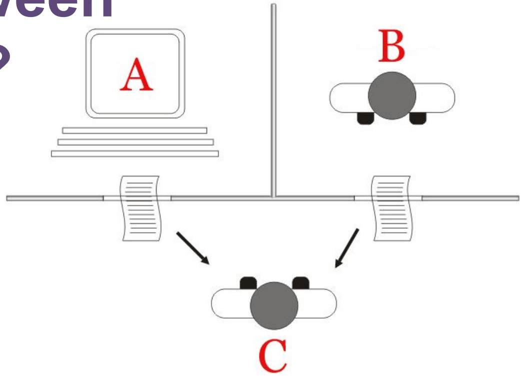
Turing Test

[Alan Turing 1950]



Can machines think?

Can humans tell the difference between machines and one of their kind?



Turing Test

[Alan Turing 1950]

Is the mind merely an information processing system operating on formal symbols?

Chinese Room Argument

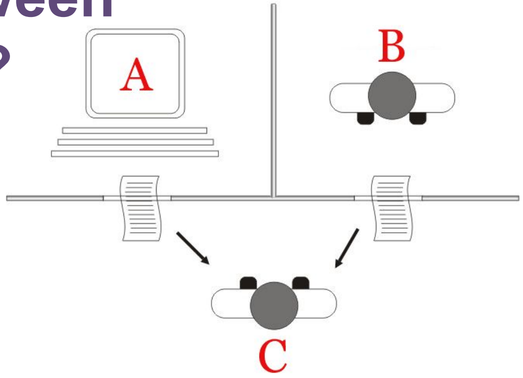
[John Searle 1980]

Can machines think?

Can humans tell the difference between machines and one of their kind?

Turing Test

[Alan Turing 1950]



Is the mind merely an information processing system operating on formal symbols?

Chinese Room Argument

[John Searle 1980]

Passing the Turing Test is not equivalent to Intelligence!

Intelligence measures an agent's ability to achieve goals in a wide range of environments. Intelligence is not really the ability to do anything in particular, rather it is a very general ability that affects many kinds of performance.

[S. Legg and M. Hutter 2007]

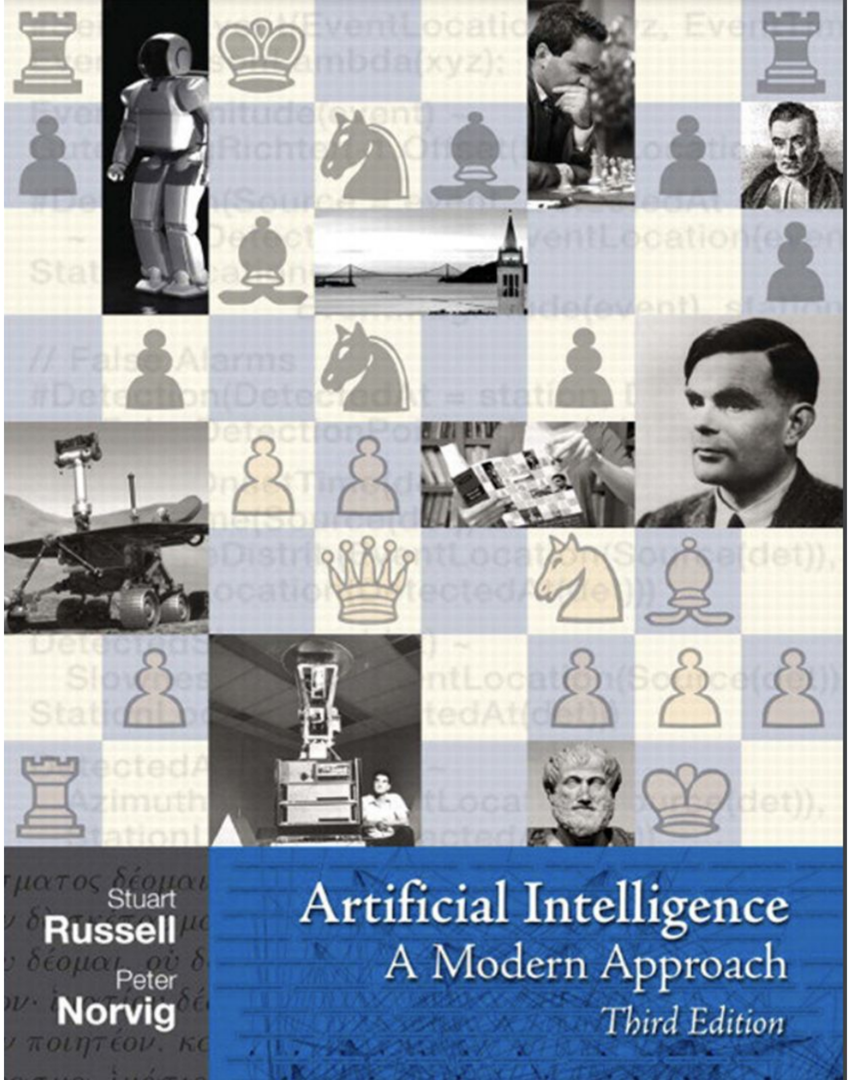
Intelligence measures an agent's ability to achieve goals in a wide range of environments. Intelligence is not really the ability to do anything in particular, rather it is a very general ability that affects many kinds of performance.

[S. Legg and M. Hutter 2007]

Turing Test and several other measures of Intelligence conflate computing power with Intelligence!

The question of whether Machines Can Think . . .
is about as relevant as the question
of whether Submarines Can Swim.

~ Edsger Dijkstra ~


Stuart
Russell
Peter
Norvig
Artificial Intelligence
A Modern Approach
Third Edition


FOREVER


Messenger
Pinterest
Facebook
Instagram
Twitter
Snapchat

amazon

Flipkart

Myntra

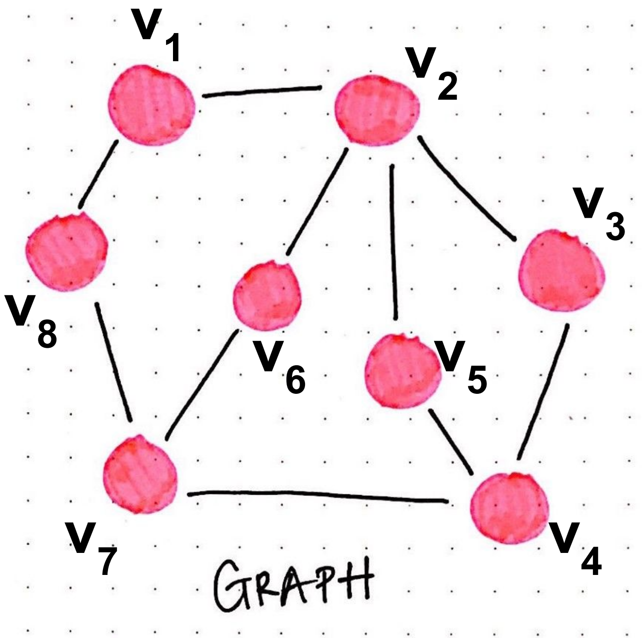
JABONG COM

SHOPCLUES.COM

snapdeal

Paytm mall

Graph

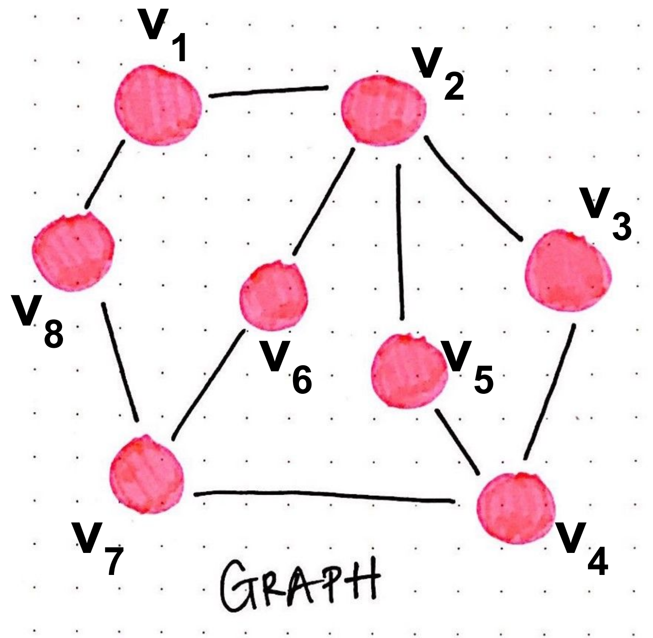


Graph G(V, E)

- Finite, non-empty set of vertices or nodes, V(G)
- Finite, possibly empty set of edges, E(G)

Elements of E(G) are pairs of vertices :

(v_1, v_2), (v_2, v_6), and so on.



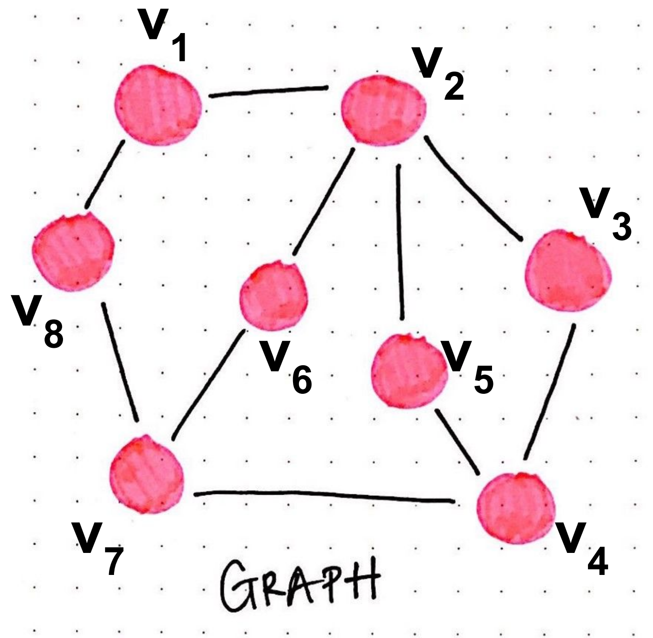
Graph G(V, E)

Which nodes in this graph are adjacent?

- Finite, non-empty set of vertices or nodes, V(G)

- Finite, possibly empty set of edges, E(G)

Elements of E(G) are pairs of vertices :
(v_1,v_2), (v_2, v_6), and so on.



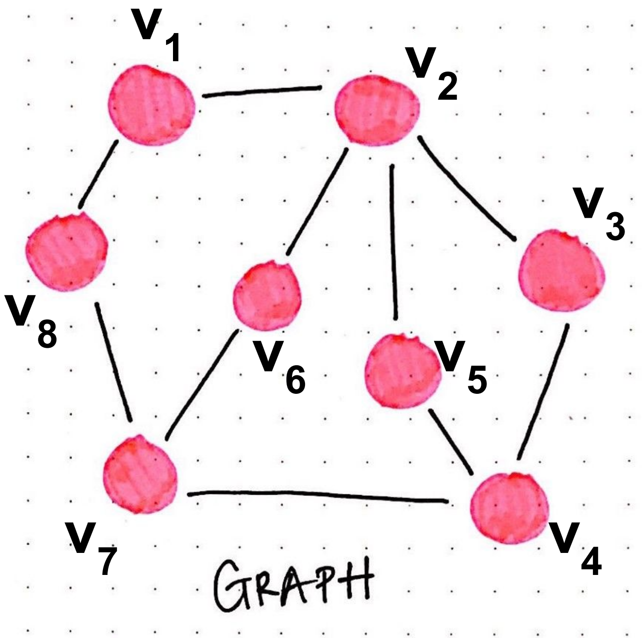
Graph G(V, E)

What is the degree of vertex v_2?

- Finite, non-empty set of vertices or nodes, V(G)

- Finite, possibly empty set of edges, E(G)

Elements of E(G) are pairs of vertices :
(v_1, v_2), (v_2, v_6), and so on.



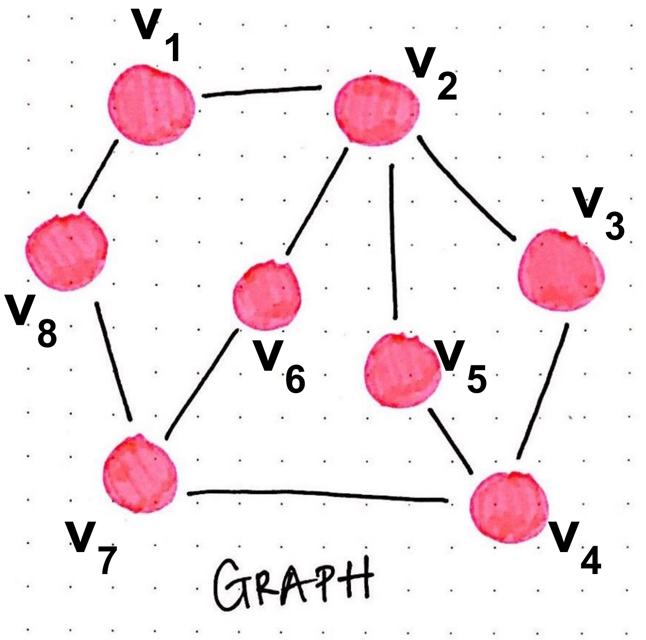
Types of Graph

Directed Graph

$$(v_1,v_2) \neq (v_2,v_1)$$

Undirected Graph

$$(v_1,v_2) = (v_2,v_1)$$



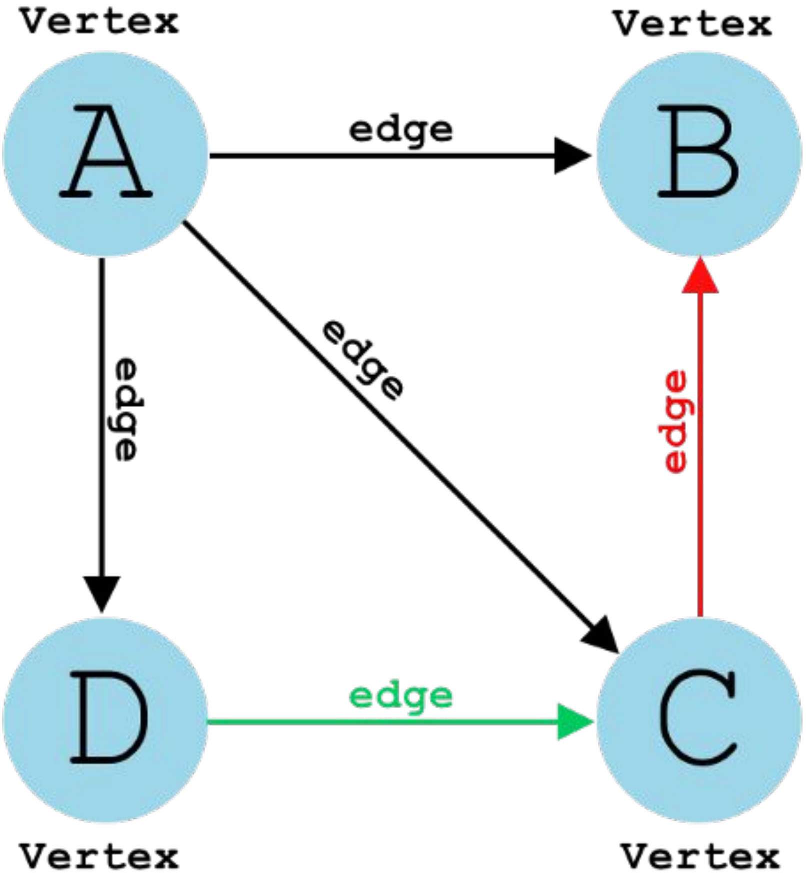
Types of Graph

Directed Graph

$$(v_1,v_2) \neq (v_2,v_1)$$

Undirected Graph

$$(v_1,v_2) = (v_2,v_1)$$



Types of Graph

Directed Graph

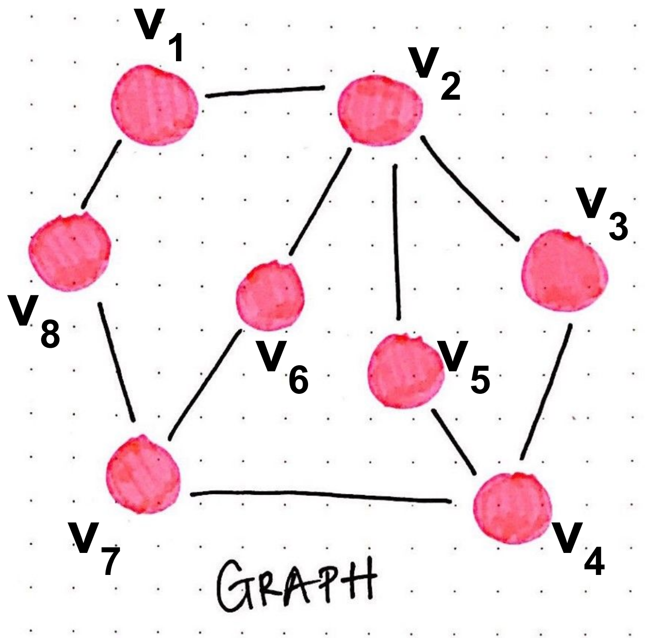
$$(v_1,v_2) \neq (v_2,v_1)$$

Undirected Graph

$$(v_1,v_2) = (v_2,v_1)$$

Complete Graph

A graph that has the maximum number of edges



Types of Graph

How many edges will this graph have if we make it complete?

Directed Graph

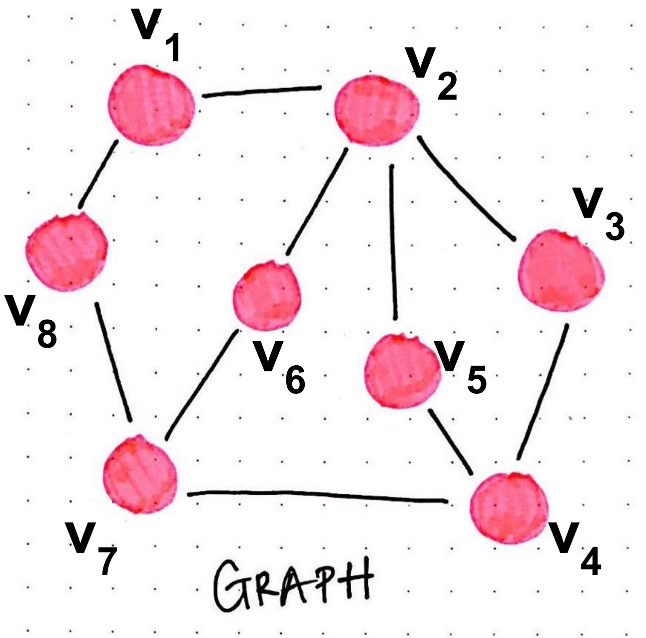
$$(v_1,v_2) \neq (v_2,v_1)$$

Undirected Graph

$$(v_1,v_2) = (v_2,v_1)$$

Complete Graph

A graph that has the maximum number of edges



Types of Graph

Simple Path

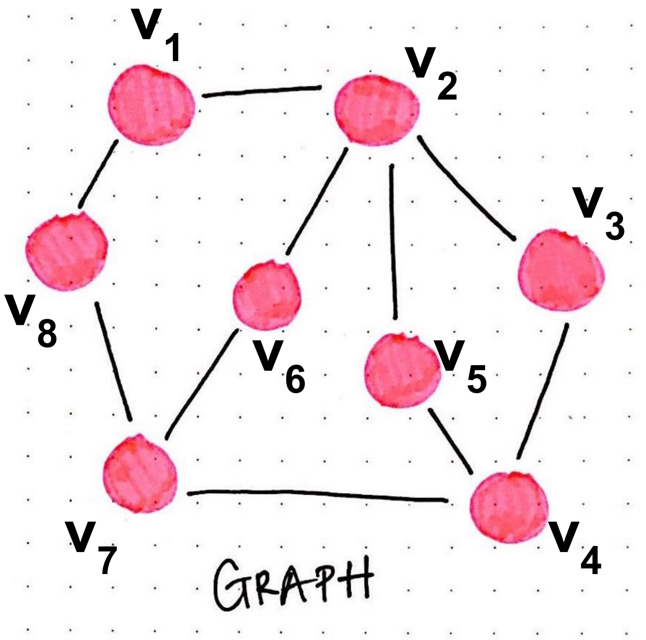
Path in which all vertices, except possibly the first and last are distinct

Cycle

Simple path in which the first and last vertices are the same

Connected Vertices

Vertices between which there is a path



Types of Graph

What is a connected graph?

Simple Path

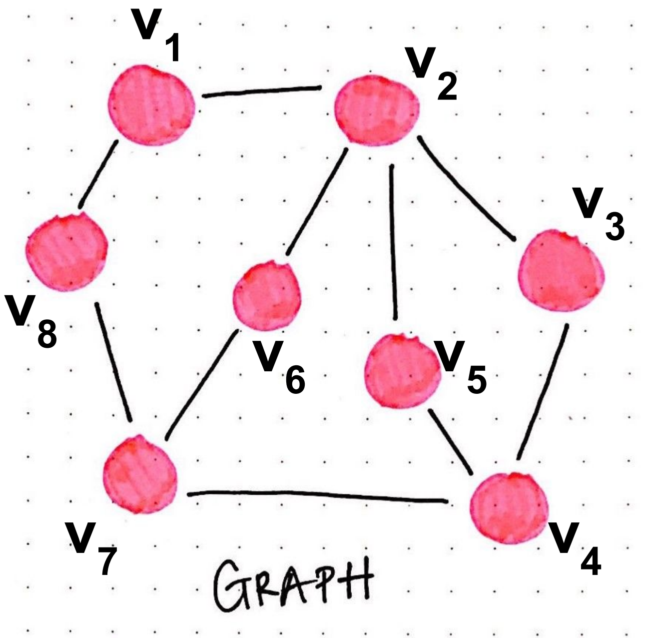
Path in which all vertices, except possibly the first and last are distinct

Cycle

Simple path in which the first and last vertices are the same

Connected Vertices

Vertices between which there is a path



Types of Graph

What is the difference between connected and complete graph?

Simple Path

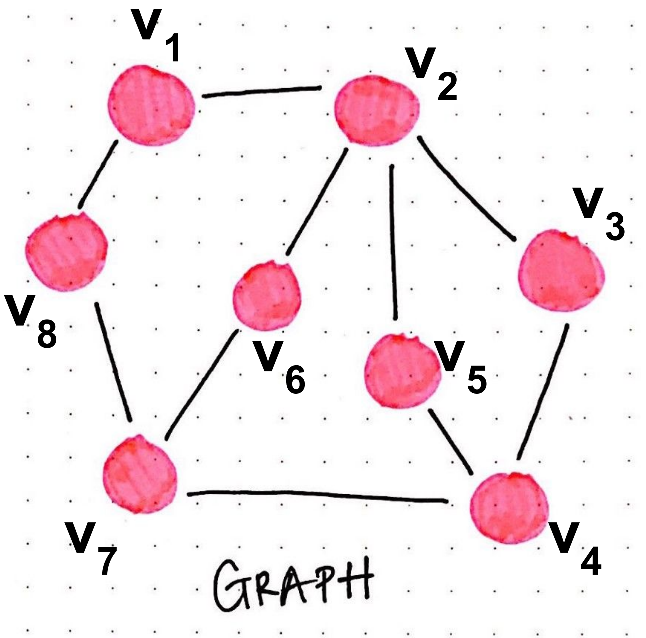
Path in which all vertices, except possibly the first and last are distinct

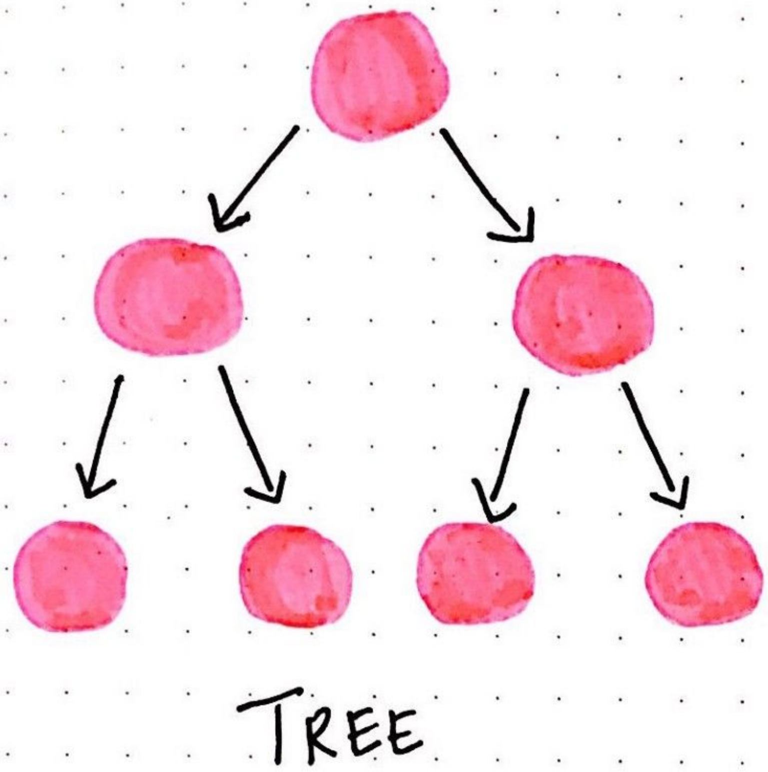
Cycle

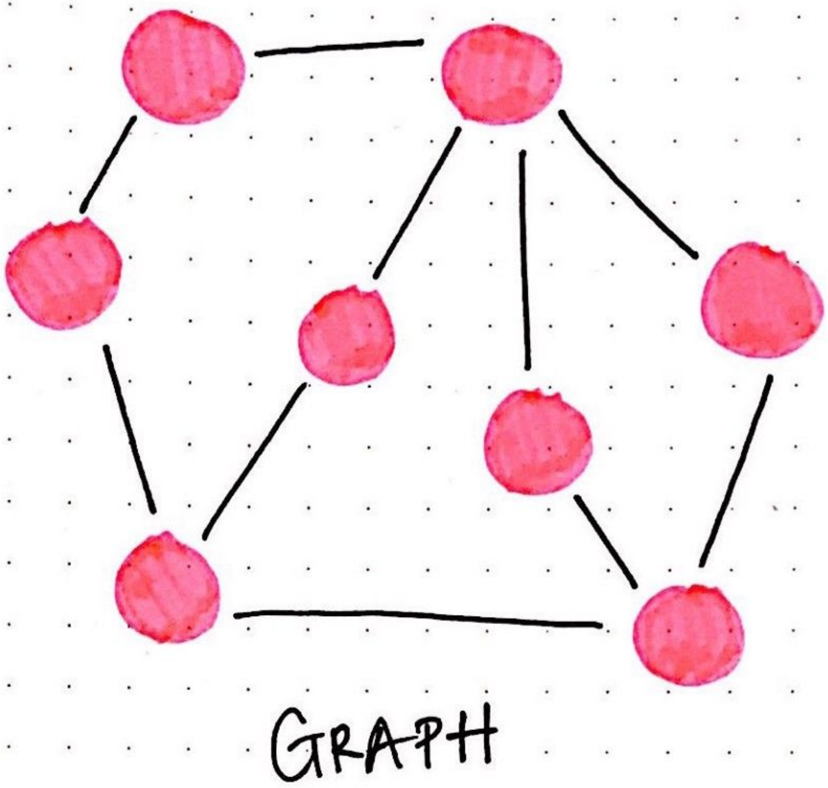
Simple path in which the first and last vertices are the same

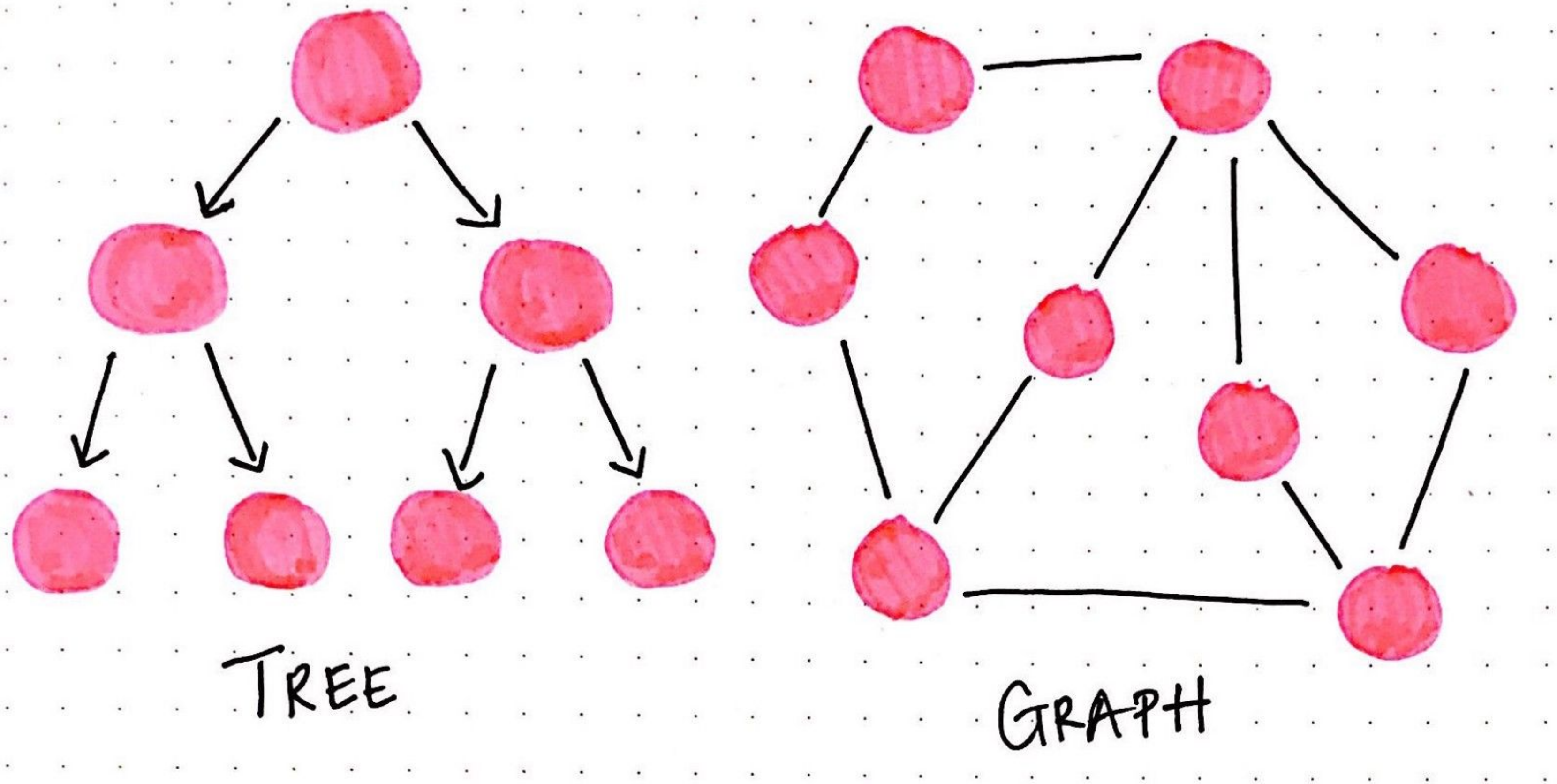
Connected Vertices

Vertices between which there is a path

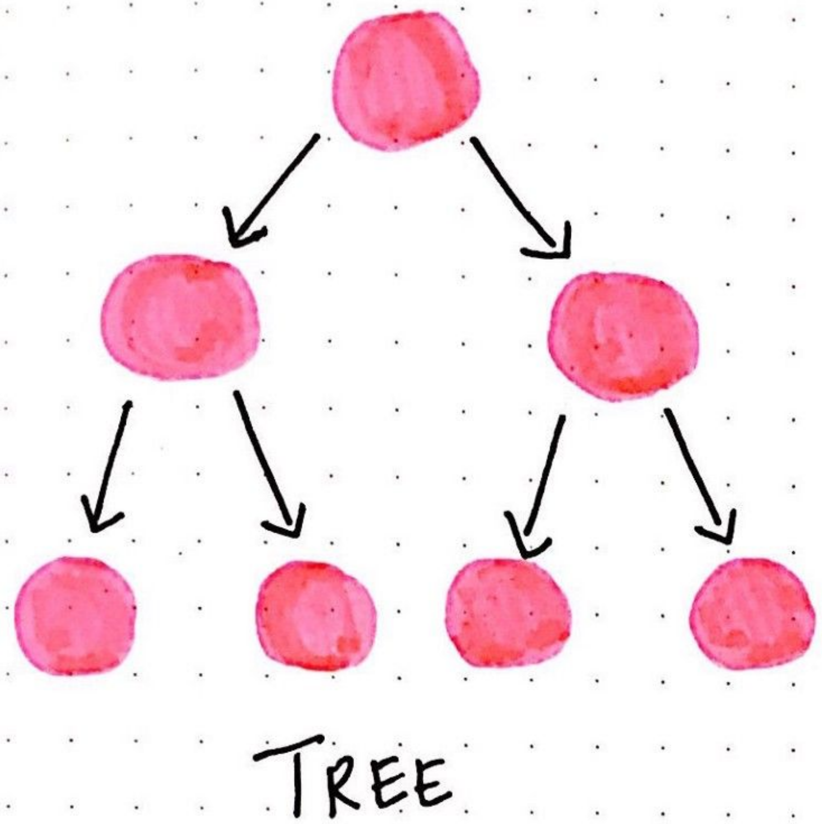



TREE

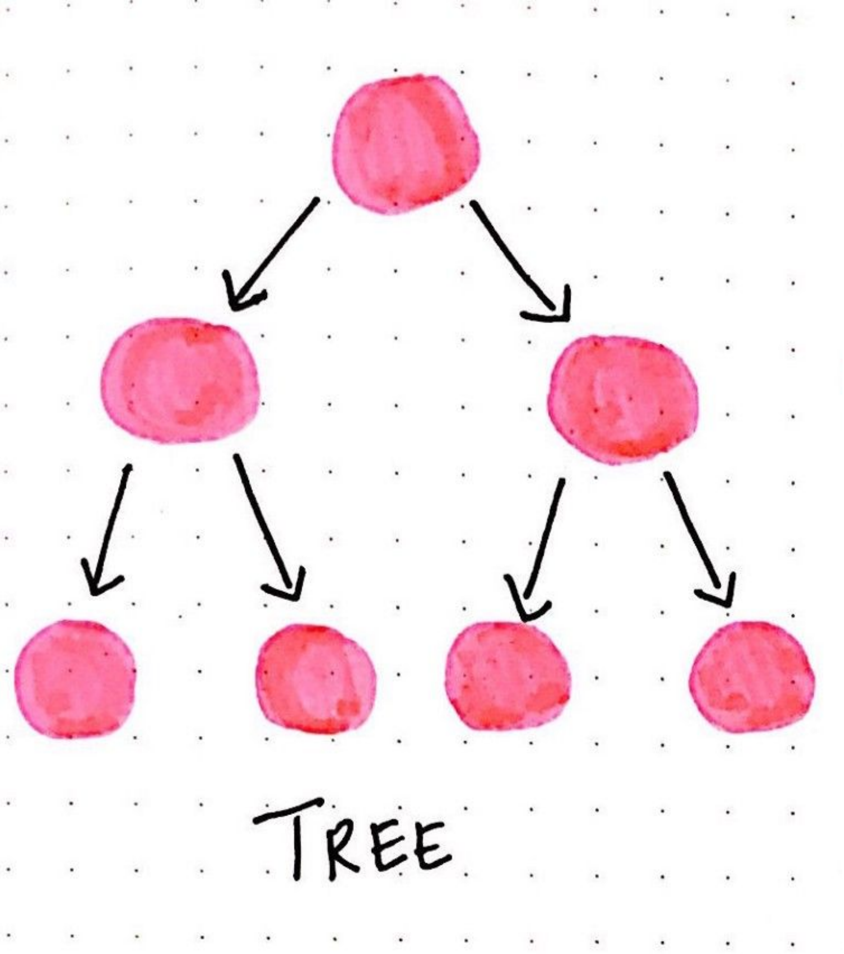

GRAPH



A **tree** is a directed acyclic graph [DAG] that is connected and in which each node has only one parent.



What is the in-degree of a node in this tree?



What is the out-degree of a node in this tree?

Adjacency Matrix

```
graph = [[0, 1, 2],
         [2, 0, 5],
         [4, 5, 0]]
```

Adjacency List

$$A \rightarrow [(B, 4), (C, 1)]$$

Edge List

```
graph = [(C, A, 4), (A, C, 1), (B, C, 6),
         (A, B, 4), (C, B, 1), (C, D, 2)]
```