

COL 351 : ANALYSIS & DESIGN OF ALGORITHMS

LECTURE 8

GRAPH ALGORITHMS I : BASIC DEFINITIONS

AUG 07, 2024

|

ROHIT VAISH

REMINDERS

Sign up on **Teams** channel

Sign up on **Gradescope**

WHAT DOES POLYNOMIAL-TIME MEAN?

WHAT DOES POLYNOMIAL-TIME MEAN?



WHAT DOES POLYNOMIAL-TIME MEAN?



ALG is a **polynomial-time** algorithm if the worst-case running time of ALG is upper bounded by a **polynomial** function of the **size of the input**.

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Input size

Poly-time?

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Input size

Poly-time?

Given a number n ,
return its square.

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Given a number n ,
return its square.

Input size

n

Poly-time?

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Given a number n ,
return its square.

Input size

~~n~~

$\log n$

Poly-time?

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Given a number n ,
return its square.

Input size

~~n~~

$\log n$

Poly-time?

$O(n)$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n $\log n$	$O(n)$ X

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$ ✓

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$ ✓
		$O(n \log n)$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$ ✓
		$O(n \log n)$ X

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$ ✓
		$O(n \log n)$ X
		$O(\sqrt{\log n})$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem	Input size	Poly-time?
Given a number n , return its square.	n	$O(n)$ X
	$\log n$	$O(n^2)$ X
		$O(\log^2 n)$ ✓
		$O(n \log n)$ X
		$O(\sqrt{\log n})$ ✓

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Input size

Poly-time?

Find the largest entry
in an n -element array

$$A = \begin{array}{|c|c|c|c|} \hline a_1 & a_2 & \dots & a_n \\ \hline \end{array}$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2)$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lceil c \rceil)$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lceil c \rceil) \quad \checkmark$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lceil c \rceil) \quad \checkmark$$

$$O(n^c)$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lfloor c \rfloor) \quad \checkmark$$

$$O(n^c) \quad \times$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lceil c \rceil) \quad \checkmark$$

$$O(n^c) \quad \times$$

$$O(n^2 \cdot 2^c)$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \lceil c \rceil) \quad \checkmark$$

$$O(n^c) \quad \times$$

$$O(n^2 \cdot 2^c) \quad \times$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \sqrt{c}) \quad \checkmark$$

$$O(n^c) \quad \times$$

$$O(n^2 \cdot 2^c) \quad \times$$

$$O(a_3)$$

WHAT DOES POLYNOMIAL-TIME MEAN?

Problem

Find the largest entry
in an n -element array

$$A = \boxed{a_1} \boxed{a_2} \boxed{\dots} \boxed{a_n}$$

Input size

$n \cdot c$

$c = \#$ bits to encode
any entry a_i

Poly-time?

$$O(n^2 c^2) \quad \checkmark$$

$$O(n^{100} \sqrt{c}) \quad \checkmark$$

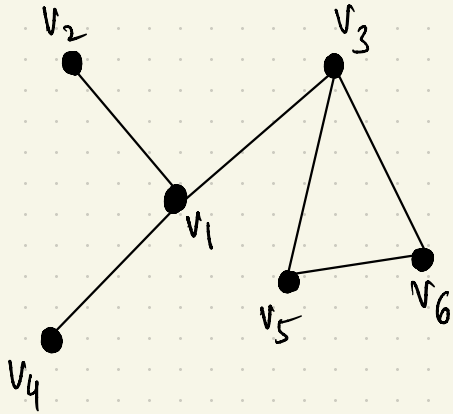
$$O(n^c) \quad \times$$

$$O(n^2 \cdot 2^c) \quad \times$$

$$O(a_3) \quad \times$$

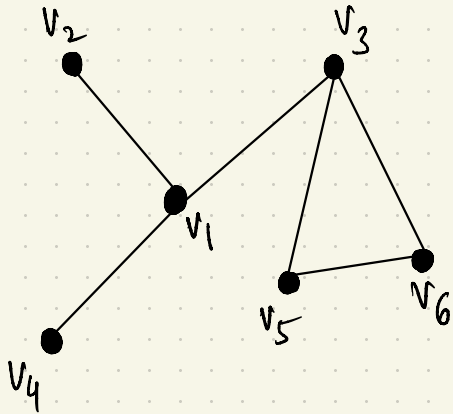
GRAPH ALGORITHMS

GRAPH

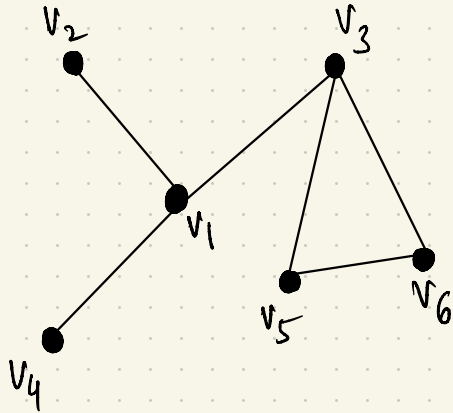


GRAPH

A graph G is a pair of sets (V, E)



GRAPH

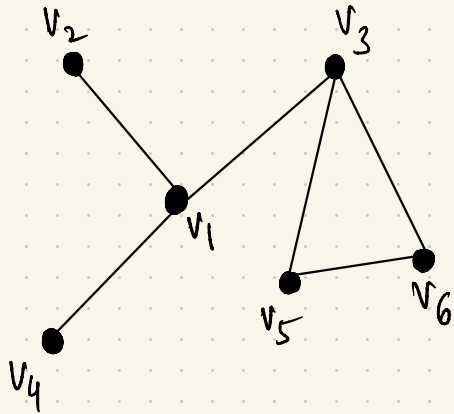


A graph G is a pair of sets (V, E)

V : non-empty set of items called
vertices / nodes

E : a (possibly empty) set of 2-item
subsets of V called edges

GRAPH



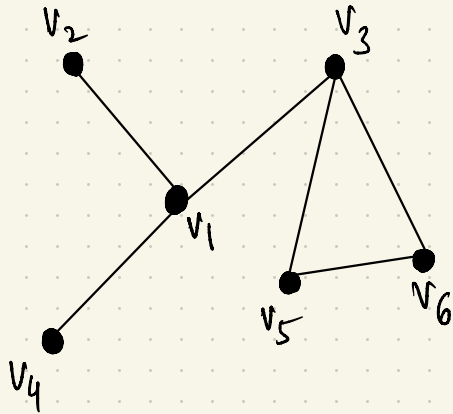
A graph G is a pair of sets (V, E)

V : non-empty set of items called
vertices / nodes

e.g., $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$

E : a (possibly empty) set of 2-item
subsets of V called edges

GRAPH



A graph G is a pair of sets (V, E)

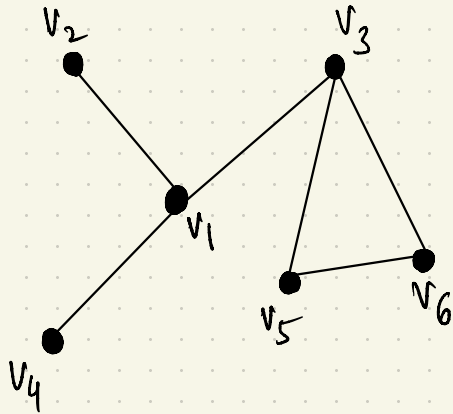
V : non-empty set of items called
vertices / nodes

e.g., $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$

E : a (possibly empty) set of 2-item
subsets of V called edges

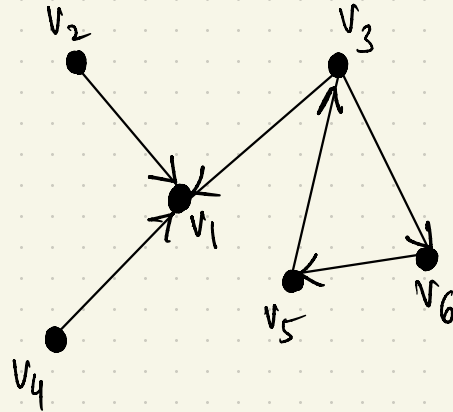
e.g., $E = \left\{ \{v_1, v_4\}, \{v_1, v_2\}, \{v_1, v_3\}, \right.$
 $\left. \{v_3, v_5\}, \{v_5, v_6\}, \{v_3, v_6\} \right\}$

GRAPH



Edges can be undirected (unordered pair)

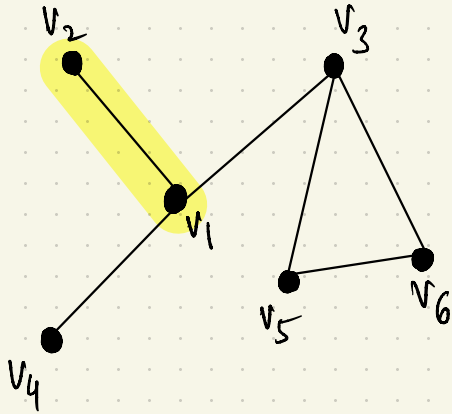
or directed (ordered pair).



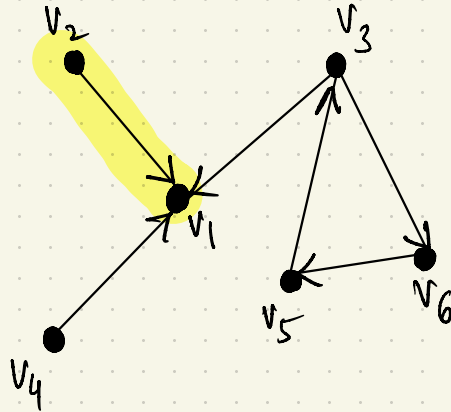
GRAPH

Edges can be undirected (unordered pair)

or directed (ordered pair).



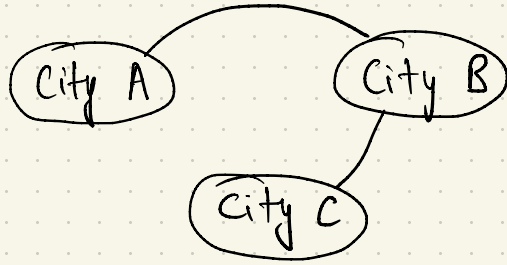
edge $\{v_1, v_2\}$ or $\{v_2, v_1\}$



edge (v_2, v_1)

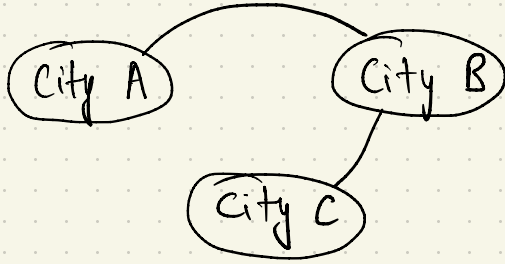
APPLICATIONS

APPLICATIONS

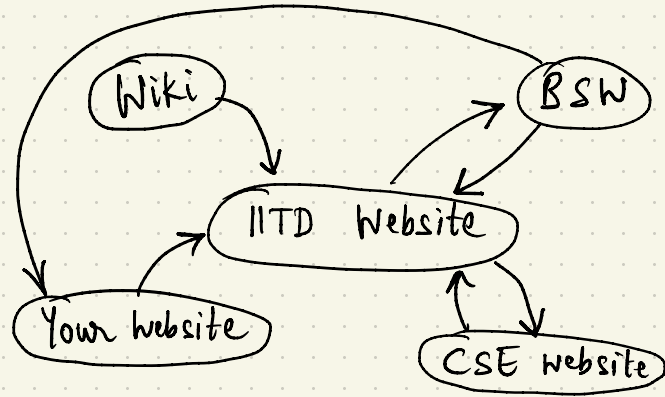


Road networks

APPLICATIONS

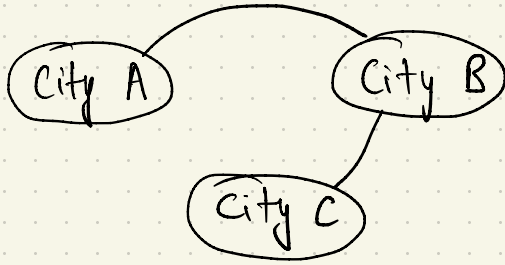


Road networks

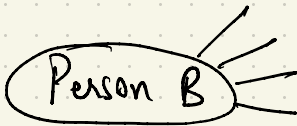
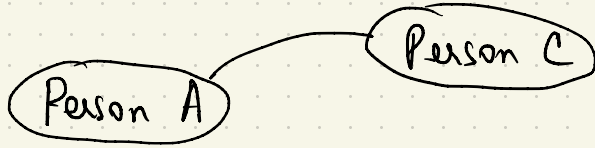


The web

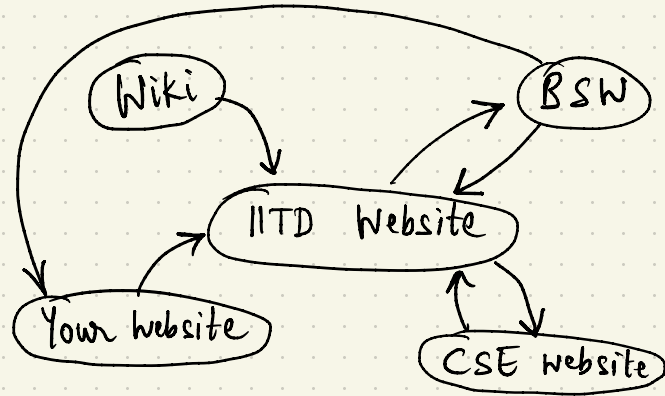
APPLICATIONS



Road networks

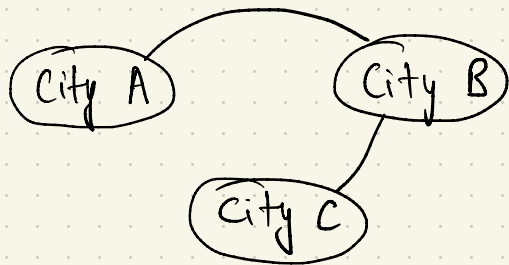


Social networks

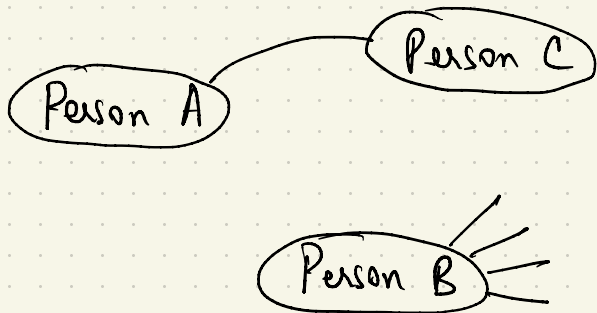


The web

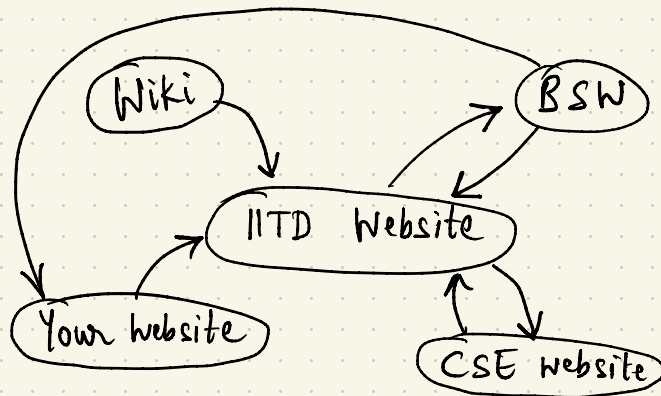
APPLICATIONS



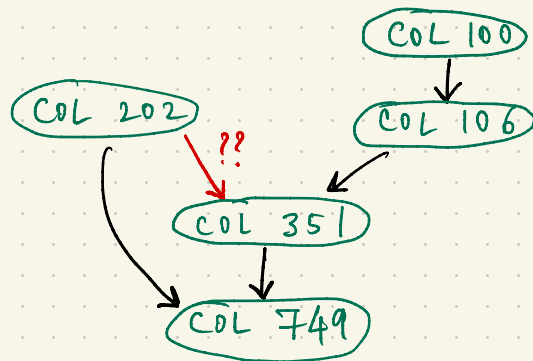
Road networks



Social networks

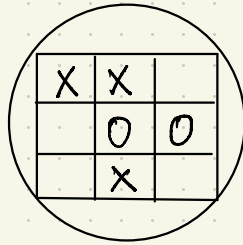


The web



Precedence constraints

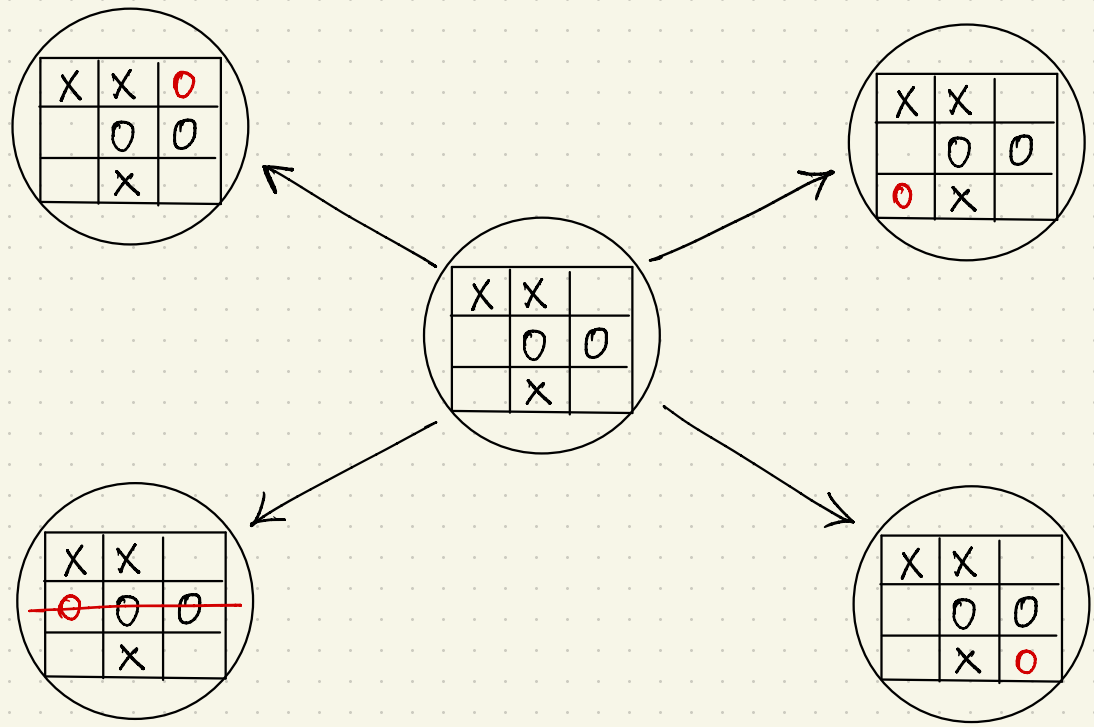
APPLICATIONS



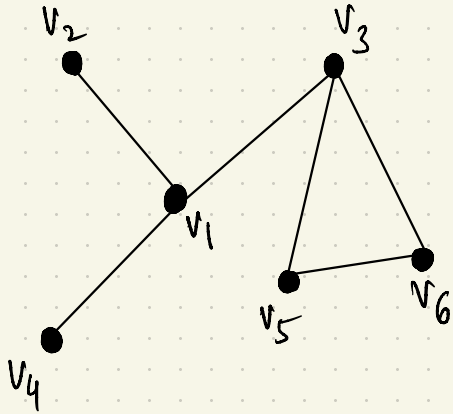
A 3x3 grid is centered on the page, enclosed within a circle. The grid contains the following characters:

X	X	
	O	O
	X	

APPLICATIONS

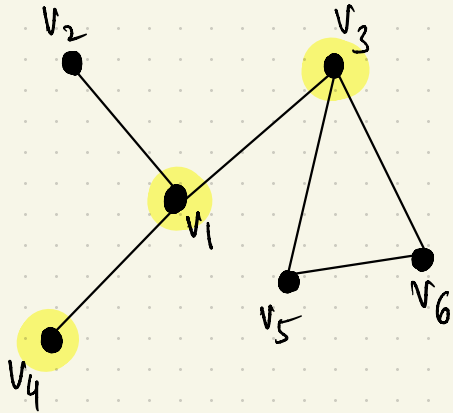


GRAPH



Two vertices v_i and v_j are adjacent
if $\{v_i, v_j\} \in E$.

GRAPH



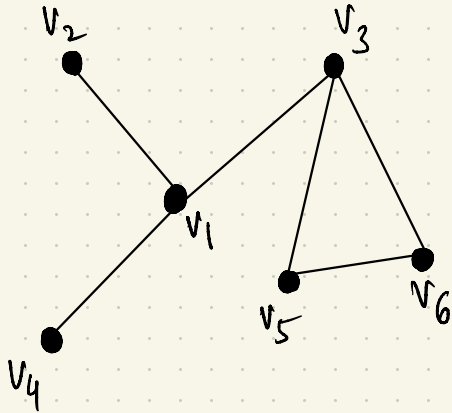
Two vertices v_i and v_j are adjacent

if $\{v_i, v_j\} \in E$.

e.g., v_1 and v_3 are adjacent

v_3 and v_4 are NOT adjacent

GRAPH

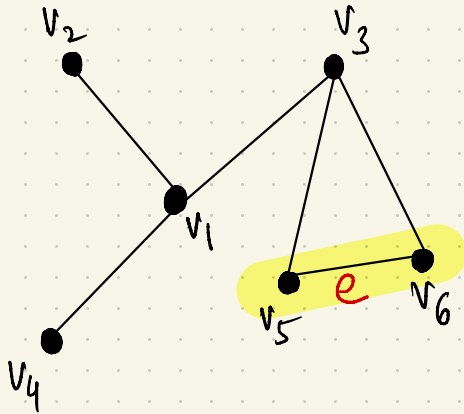


Two vertices v_i and v_j are adjacent if $\{v_i, v_j\} \in E$.

e.g., v_1 and v_3 are adjacent
 v_3 and v_4 are NOT adjacent

An edge $e = \{v_i, v_j\}$ is incident to the vertices v_i and v_j .

GRAPH



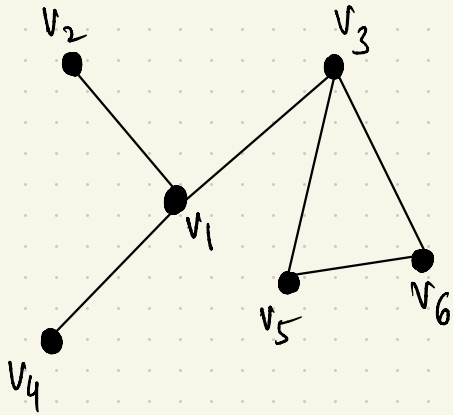
Two vertices v_i and v_j are adjacent
if $\{v_i, v_j\} \in E$.

e.g., v_1 and v_3 are adjacent
 v_3 and v_4 are NOT adjacent

An edge $e = \{v_i, v_j\}$ is incident
to the vertices v_i and v_j .

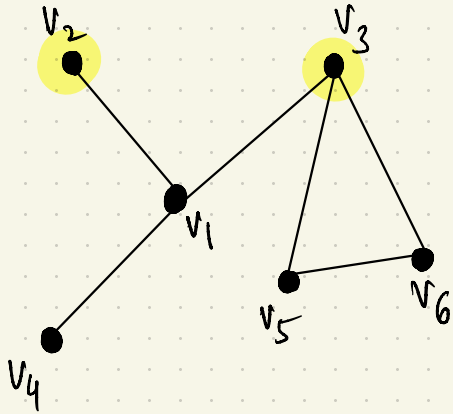
e.g., e is incident to v_5 and v_6
but not v_3 .

GRAPH



The number of edges incident to a vertex is called the *degree*.

GRAPH

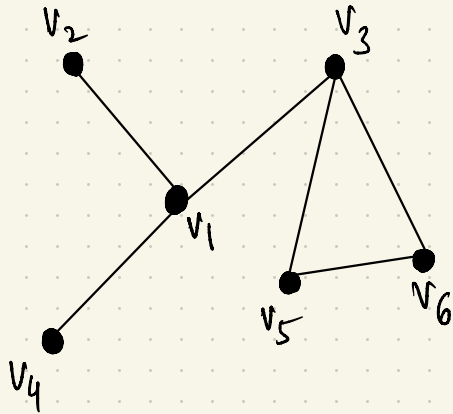


The number of edges incident to a vertex is called the **degree**.

e.g., $\deg(v_3) = 3$

$\deg(v_2) = 1$

GRAPH

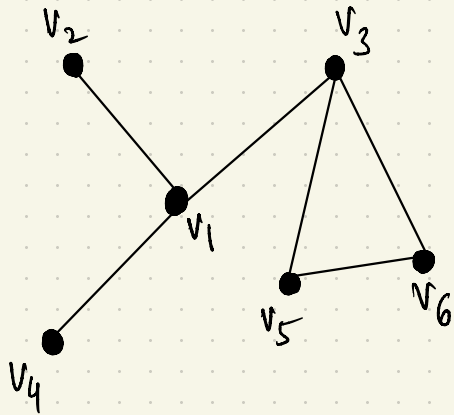


The number of edges incident to a vertex is called the **degree**.

e.g., $\deg(v_3) = 3$ $\deg(v_2) = 1$

A **simple** graph has no loops or multiedges.

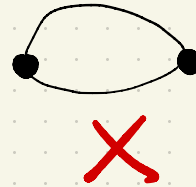
GRAPH



The number of edges incident to a vertex is called the **degree**.

e.g., $\deg(v_3) = 3$ $\deg(v_2) = 1$

A **simple** graph has no self-loops or multiedges.



WALKS AND PATHS

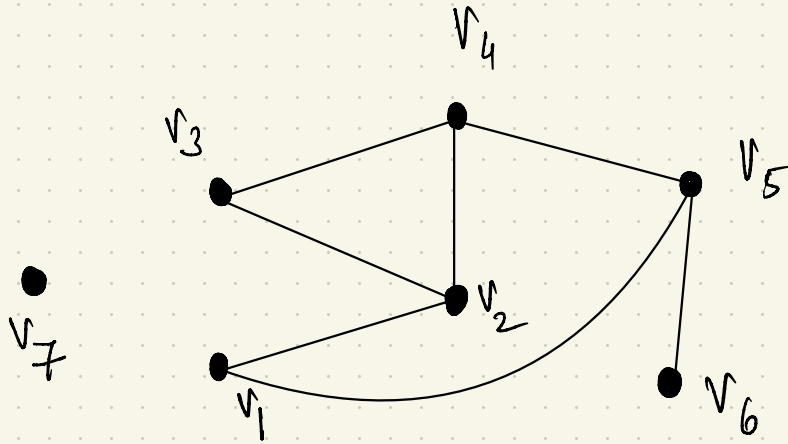
WALKS AND PATHS

A **walk** is a sequence of vertices connected by edges.

WALKS AND PATHS

A **walk** is a sequence of vertices connected by edges.

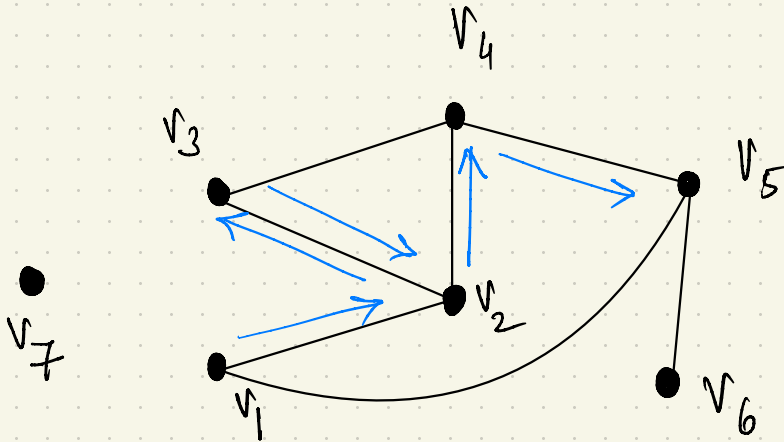
E.g.,



WALKS AND PATHS

A **walk** is a sequence of vertices connected by edges.

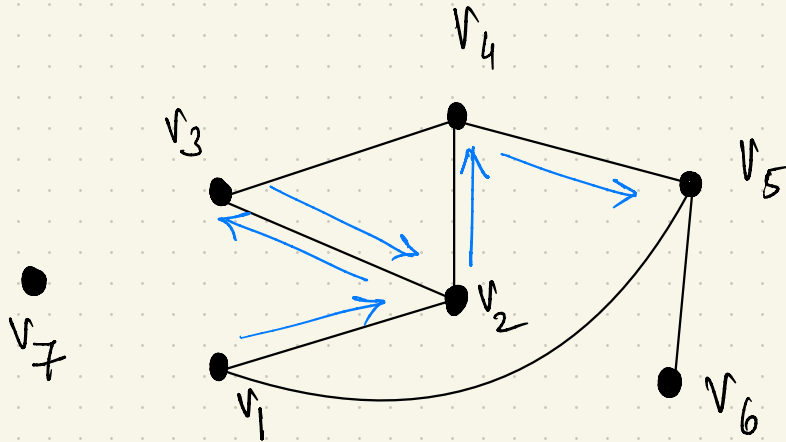
E.g., v_1 — v_2 — v_3 — v_2 — v_4 — v_5
start end



WALKS AND PATHS

A **walk** is a sequence of vertices connected by edges.

E.g., $v_1 \text{ --- } v_2 \text{ --- } v_3 \text{ --- } v_2 \text{ --- } v_4 \text{ --- } v_5$ } walk of length 5
start end



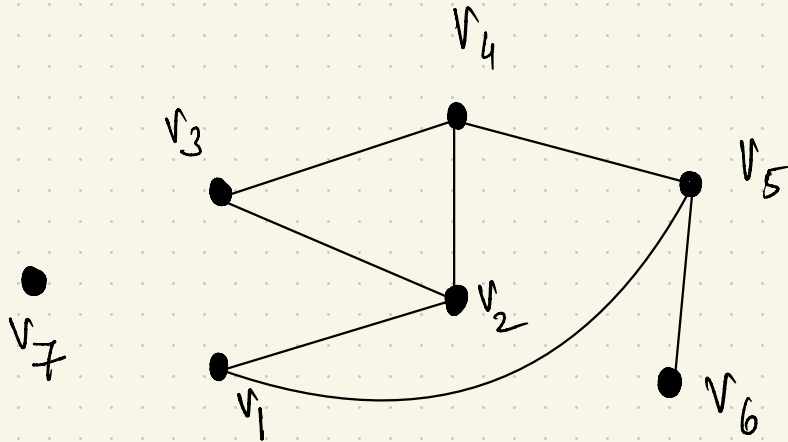
WALKS AND PATHS

A **path** is a walk where all vertices are different.

WALKS AND PATHS

A **path** is a walk where all vertices are different.

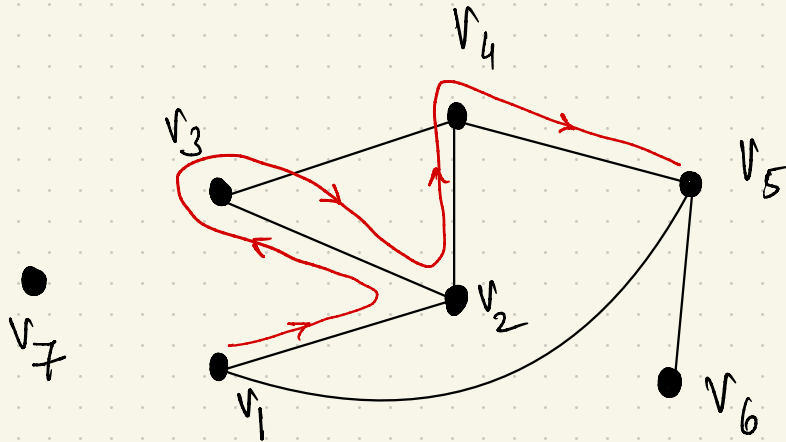
E.g.,



WALKS AND PATHS

A **path** is a walk where all vertices are different.

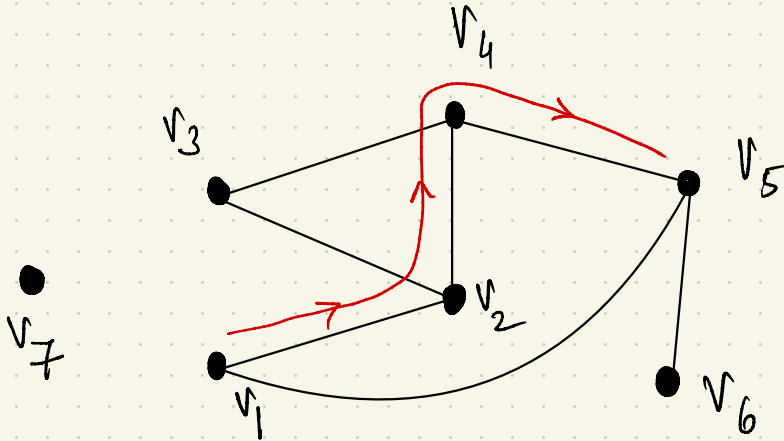
E.g., $v_1 - v_2 - v_3 - v_2 - v_4 - v_5$ X Not a path



WALKS AND PATHS

A **path** is a walk where all vertices are different.

E.g., $v_1 - v_2 - v_4 - v_5$ A valid path (length = 3)



Lemma: For any two distinct vertices u and v ,

there exists a **walk** between u and v \iff there exists a **path** between u and v

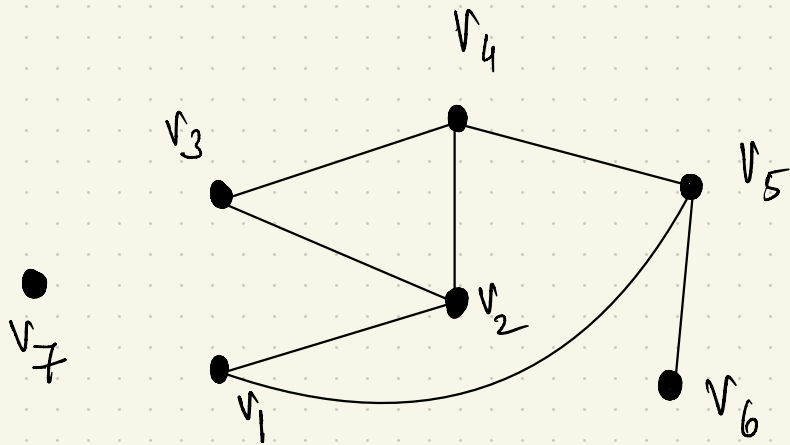
Proof: Exercise

CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .

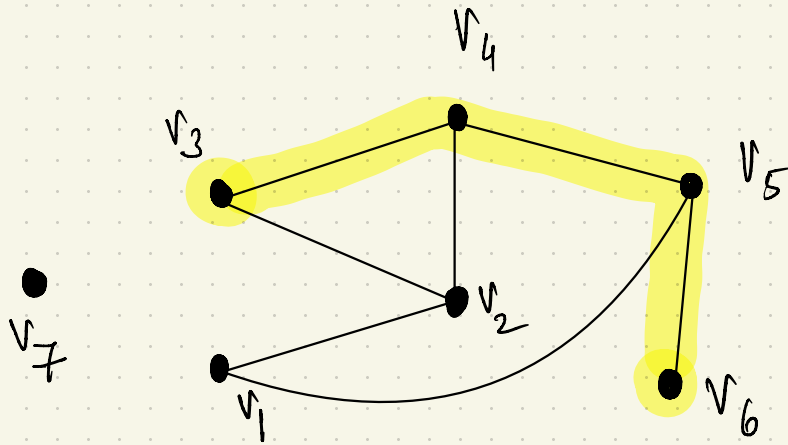
CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .



CONNECTIVITY

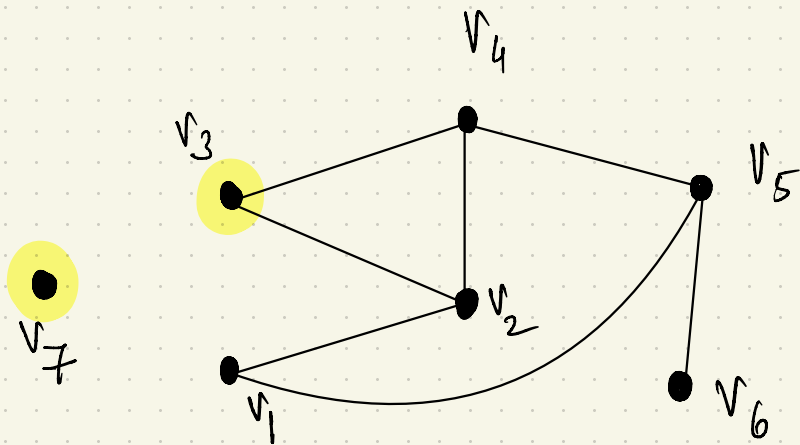
A pair of vertices u and v are **connected** if there is a path between u and v .



$v_3, v_6 \rightarrow$ Connected

CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .

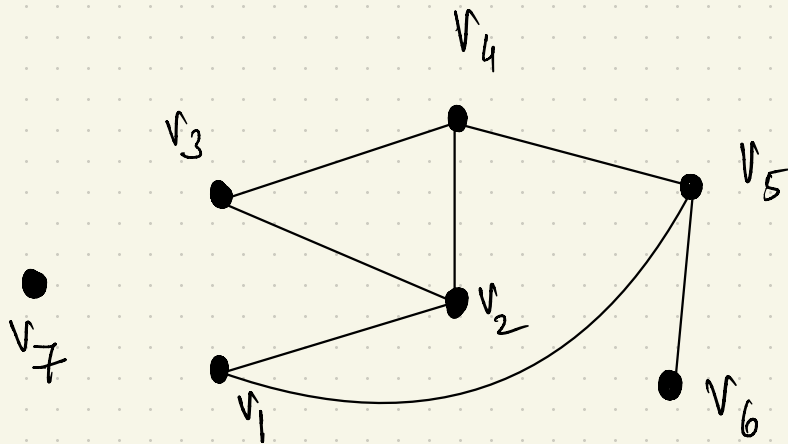


$v_3, v_6 \rightarrow$ Connected

$v_3, v_7 \rightarrow$ Not connected

CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .



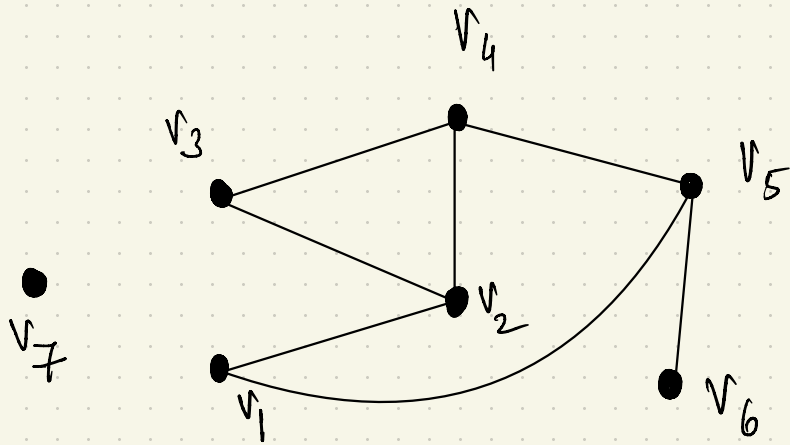
$v_3, v_6 \rightarrow$ Connected

$v_3, v_7 \rightarrow$ Not connected

A graph is **connected** if every pair of vertices are connected.

CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .

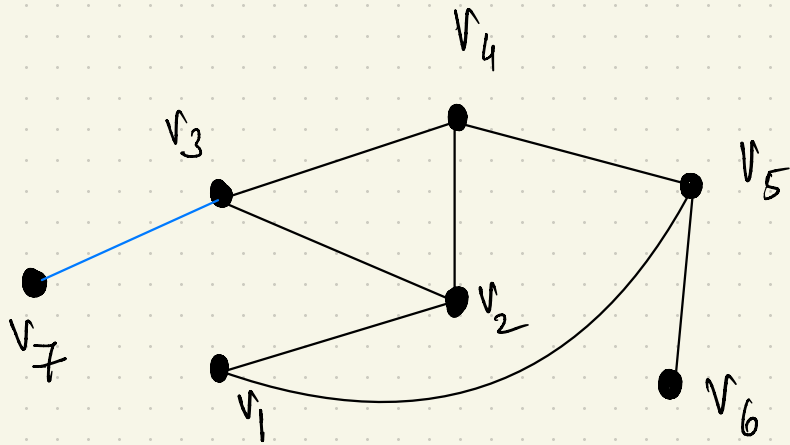


Not a connected graph

A graph is **connected** if every pair of vertices are connected.

CONNECTIVITY

A pair of vertices u and v are **connected** if there is a path between u and v .



Connected

A graph is **connected** if every pair of vertices are connected.

CONNECTIVITY

Given an undirected and connected graph on n vertices

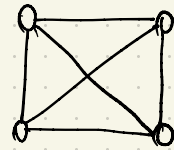
Maximum # edges =

Minimum # edges =

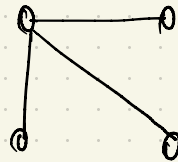
CONNECTIVITY

Given an undirected and connected graph on n vertices

Maximum # edges = ${}^n C_2$



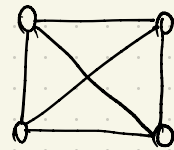
Minimum # edges = $n - 1$



CONNECTIVITY

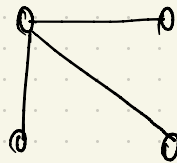
Given an undirected and connected graph on n vertices

Maximum # edges = ${}^n C_2$



complete graph

Minimum # edges = $n - 1$



tree

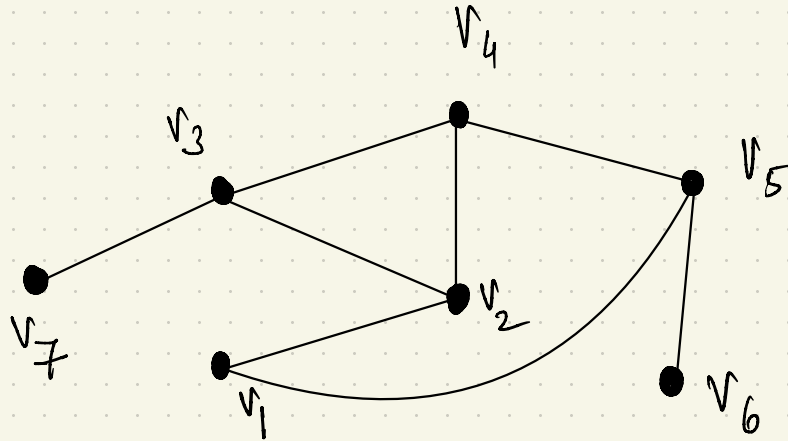
CYCLES AND CLOSED WALKS

CYCLES AND CLOSED WALKS

A closed walk is a walk that starts and ends at the same vertex.

CYCLES AND CLOSED WALKS

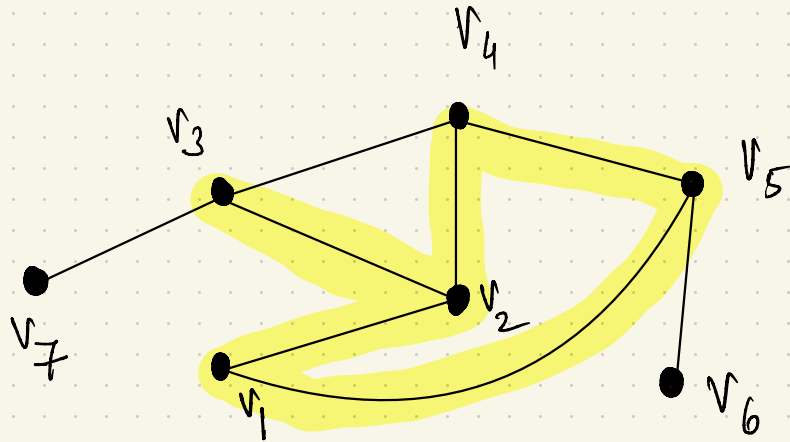
A closed walk is a walk that starts and ends at the same vertex.



CYCLES AND CLOSED WALKS

A **closed walk** is a walk that starts and ends at the same vertex.

$v_1 - v_2 - v_3 - v_2 - v_4 - v_5 - v_1$




CYCLES AND CLOSED WALKS

A **closed walk** is a walk that starts and ends at the same vertex.

A **cycle** is a closed walk in which all vertices (except start and end) are different.

CYCLES AND CLOSED WALKS

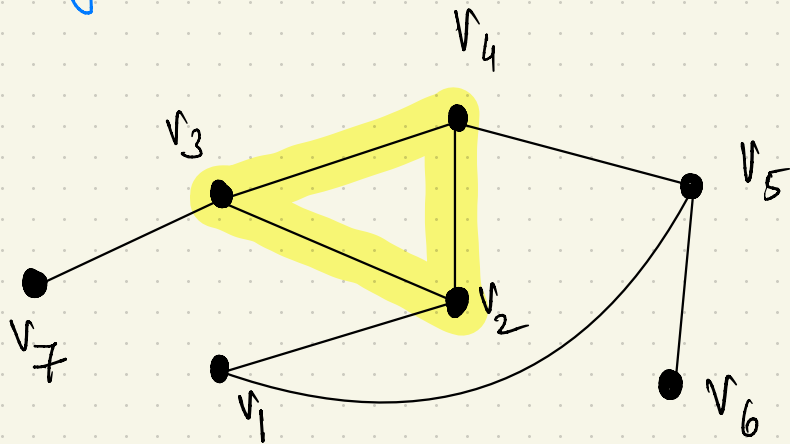
A closed walk is a walk that starts and ends at the same vertex.

A cycle is a closed walk in which all vertices (except start and end) are different.  disallow single vertices or single edges

CYCLES AND CLOSED WALKS

A **closed walk** is a walk that starts and ends at the same vertex.

A **cycle** is a closed walk in which all vertices (except start and end) are different. ↙ disallow single vertices or single edges

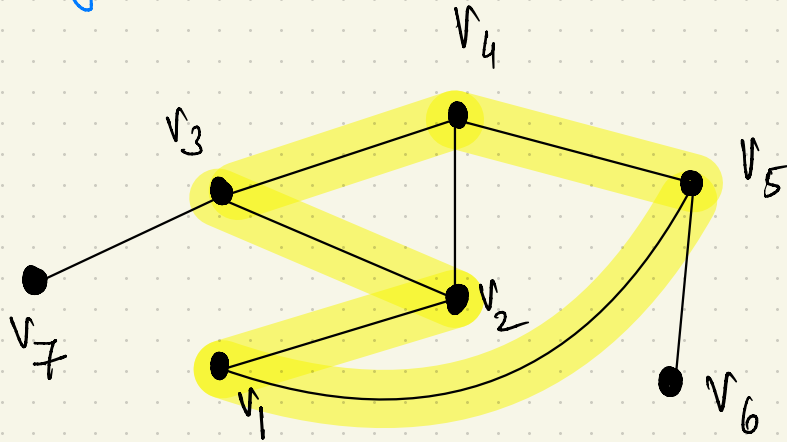


$v_2 - v_3 - v_4 - v_2$ is a cycle of length three

CYCLES AND CLOSED WALKS

A **closed walk** is a walk that starts and ends at the same vertex.

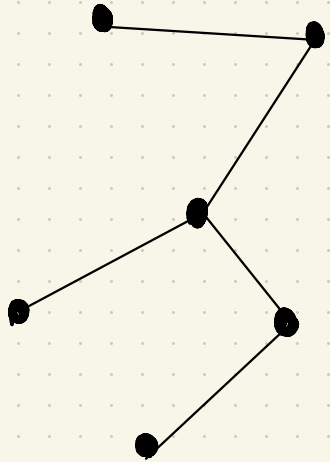
A **cycle** is a closed walk in which all vertices (except start and end) are different.
 disallow single vertices or single edges



$v_1 - v_2 - v_3 - v_4 - v_5 - v_1$ is a cycle of length five

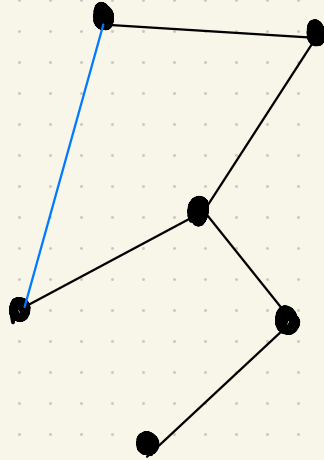
TREES

A tree is a connected and acyclic graph.



TREES

A tree is a connected and acyclic graph.



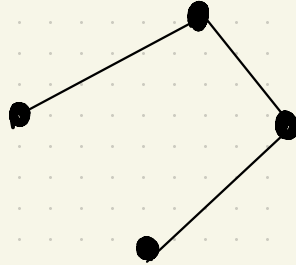
Not a tree

TREES

A tree is a connected and acyclic graph.



Not a tree



(It's a forest!)