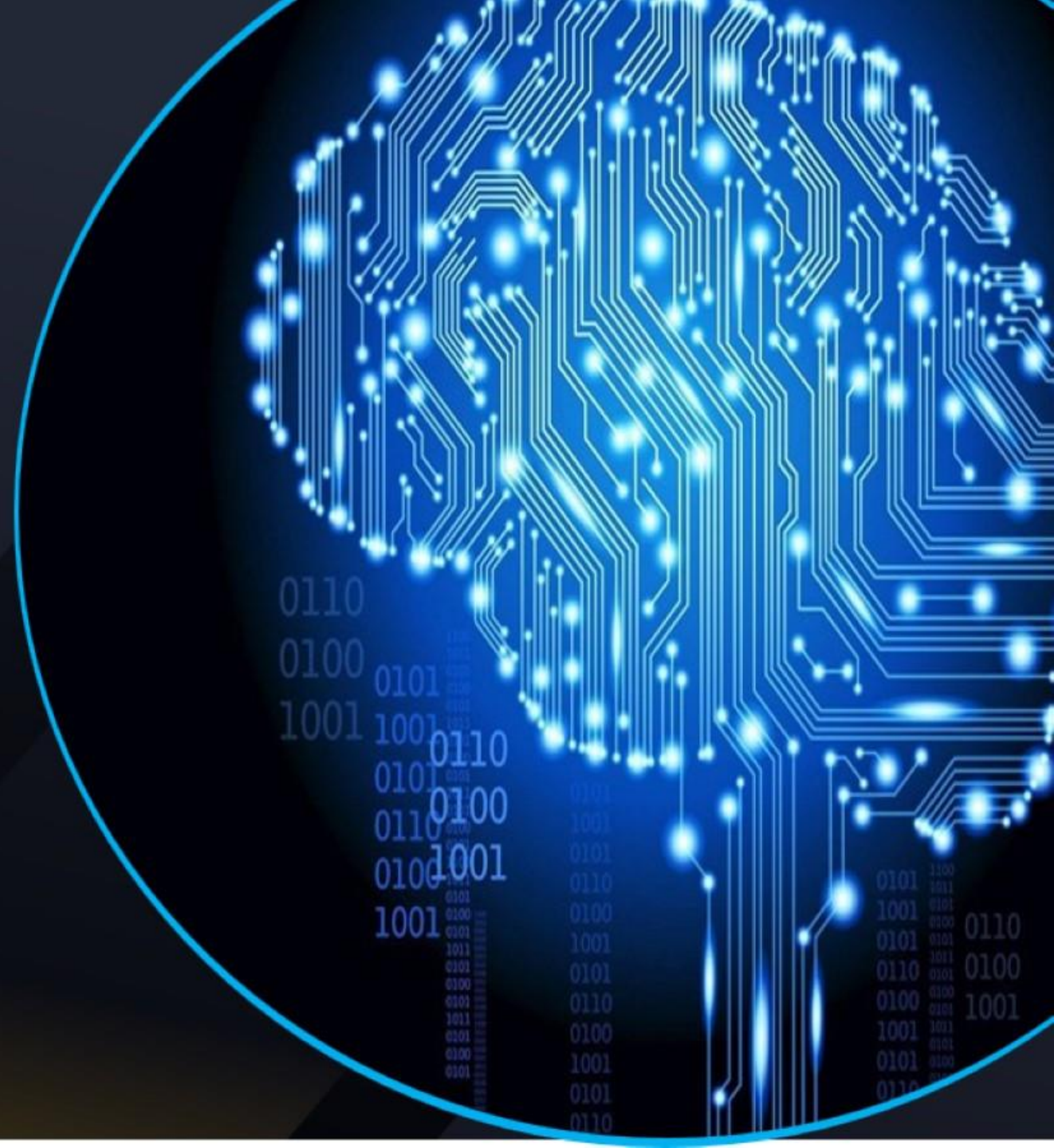


Discrete Mathematics

Dr. Shahira Ahmed

IT241

Lecture 5





Discrete Mathematics

Lecture 5

Graphs & Trees

Dr. Shahira Ahmed

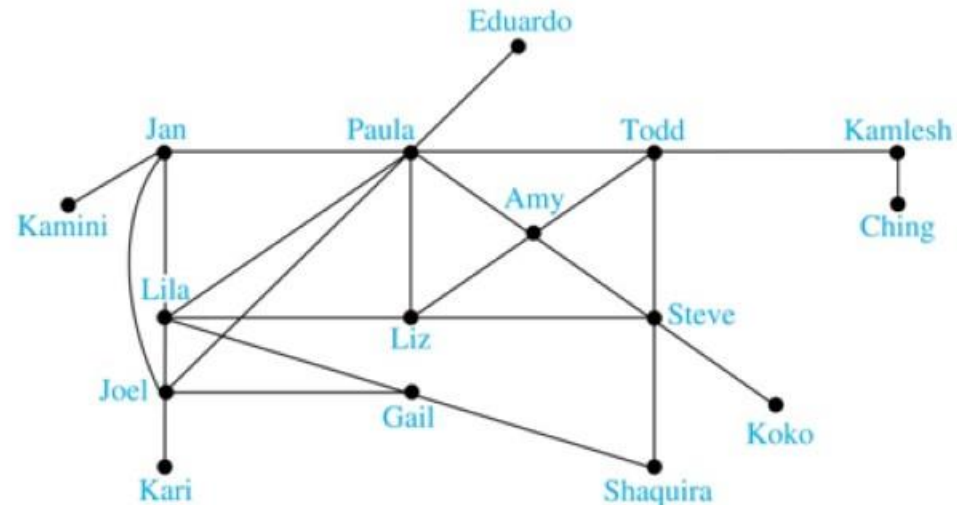


LECTURE OUTLINE

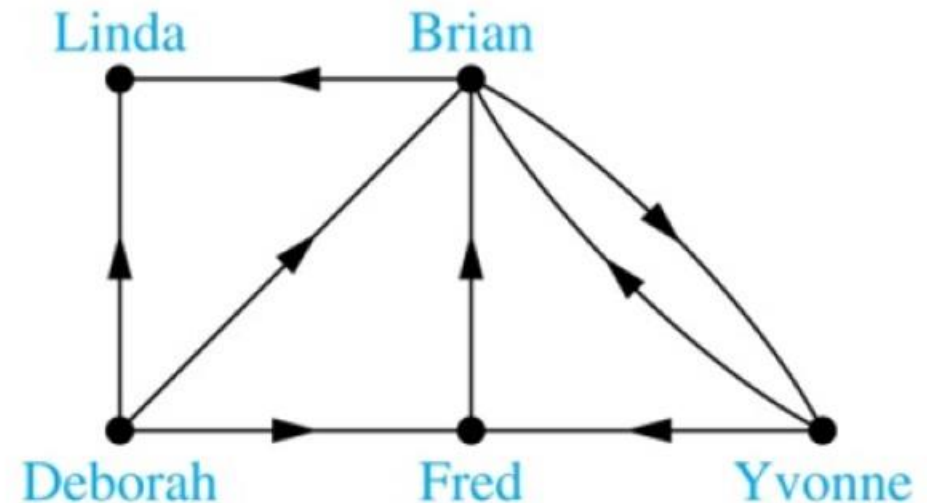
- ◉ Models of Graph
- ◉ Special Types of Simple Graph
- ◉ Connectivity
- ◉ Paths vs walk
- ◉ Cycle vs circuit
- ◉ Introduction to Trees

- Graph theory can be used in models of:
 - Social networks
 - Communications networks
 - Information networks
 - Software design
 - Transportation networks
 - Biological networks
- It's a challenge to find a subject to which graph theory has not yet been applied.
Can you find an area without applications of graph theory?

- **Facebook Friendship** graphs - **undirected** graphs where two people are connected if they are friends. Facebook model their users as a graph in which each vertex is a user profile. The edge between two persons is the fact that they are friends.



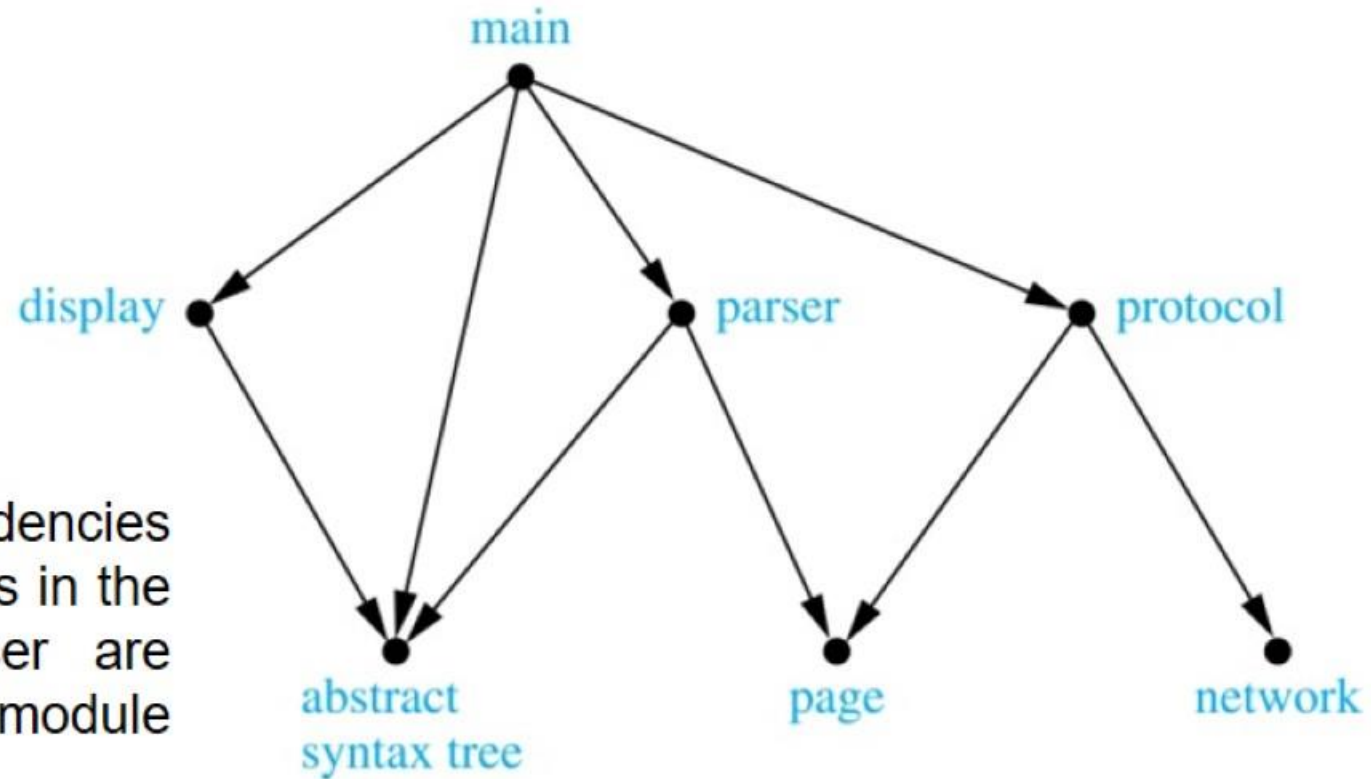
- **Twitter** is **Directed** graph- You follow the university account on Twitter, but the institution may not follow you back.



- **World Wide Web** : In a web graph, **web pages** are represented by vertices and **links** are represented by **directed** edges.

- Airline networks can be modeled using directed **multigraphs** where
 - **airports** are represented by **vertices**
 - each **flight** is represented by **a directed edge** from the vertex representing the departure airport to the vertex representing the destination airport
- Road networks can be modeled using graphs where
 - **vertices** represent **intersections** and **edges** represent **roads**.
 - **undirected** edges represent **two-way roads** and **directed** edges represent **one-way roads**.

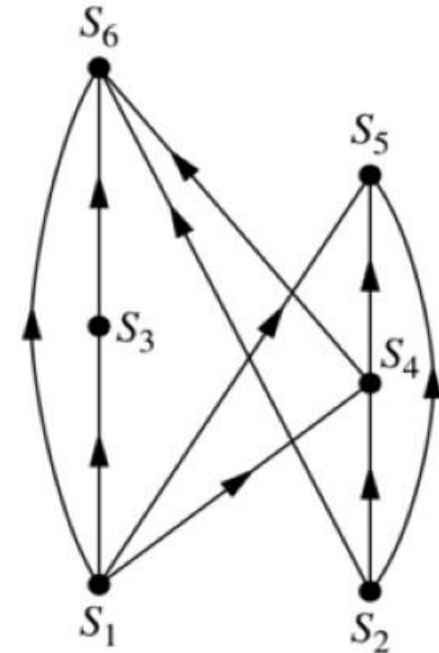
- The dependency between the **modules** of a **software application**.



Example: The dependencies between the seven modules in the design of a web browser are represented by this module dependency graph.

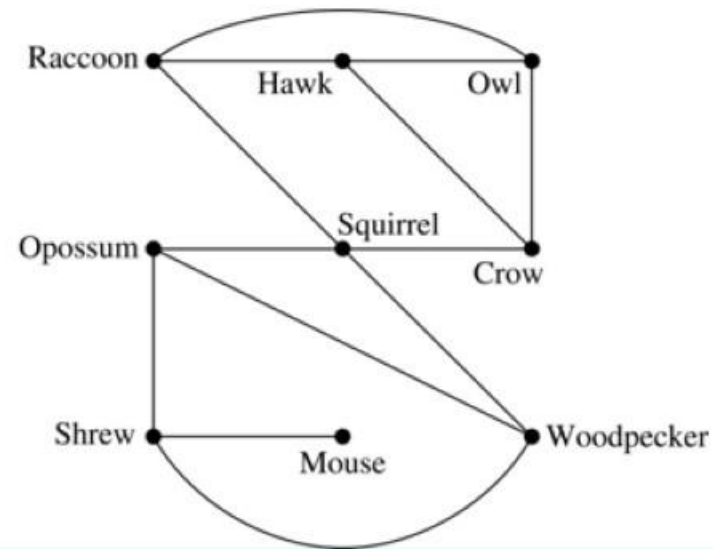
- *Precedence graph* : Precedence of statement
 - Vertices represent **statements** in a computer program
 - There is a directed edge from a vertex to a second vertex if the second vertex cannot be executed before the first

S_1 $a := 0$
 S_2 $b := 1$
 S_3 $c := a + 1$
 S_4 $d := b + a$
 S_5 $e := d + 1$
 S_6 $e := c + d$



- Graph models are used extensively in many areas of the biological science.
- *Niche overlap graphs* model competition between species in an ecosystem
 - Vertices represent species and an edge connects two vertices when they represent species who compete for food resources.

Example: This is the niche overlap graph for a forest ecosystem with nine species.



Special Types of Simple Graphs: Complete Graphs

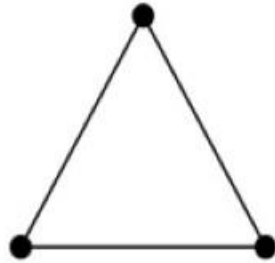
A *complete graph on n vertices*, denoted by K_n , is the simple graph that contains **exactly one** edge between **each pair** of distinct vertices.



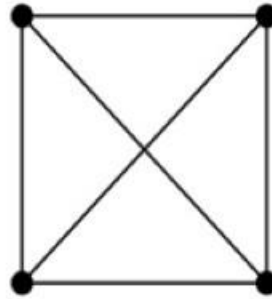
K_1



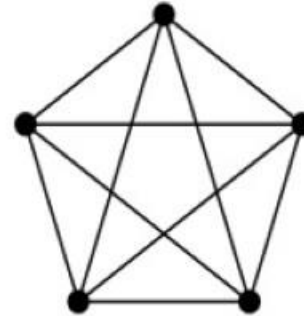
K_2



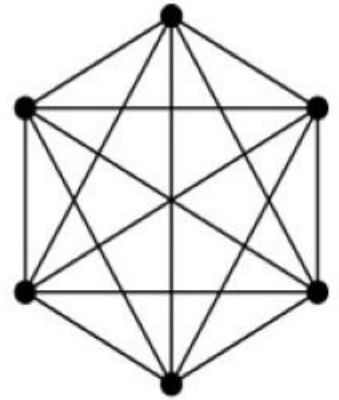
K_3



K_4



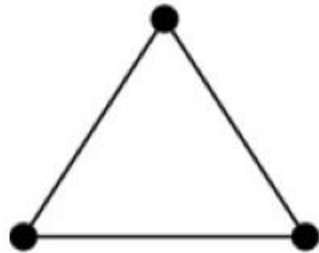
K_5



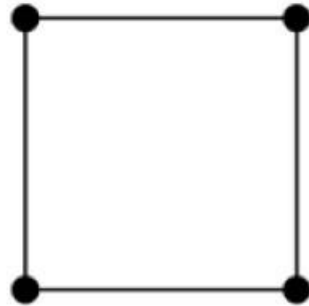
K_6

Special Types of Simple Graphs: Cycles

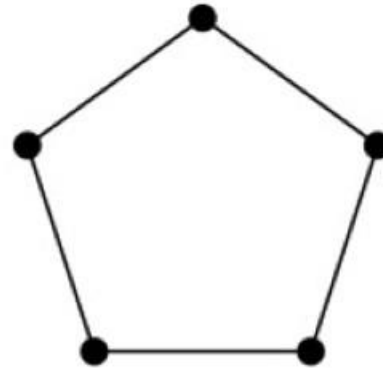
A cycle C_n for $n \geq 3$ consists of n vertices $v_1, v_2, \dots, v_n$, and edges $\{v_1, v_2\}, \{v_2, v_3\}, \dots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.



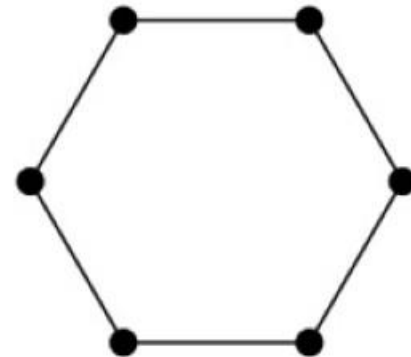
C_3



C_4



C_5

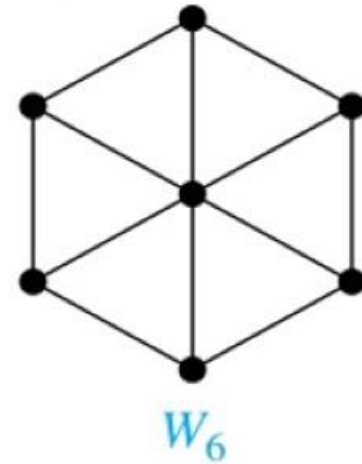
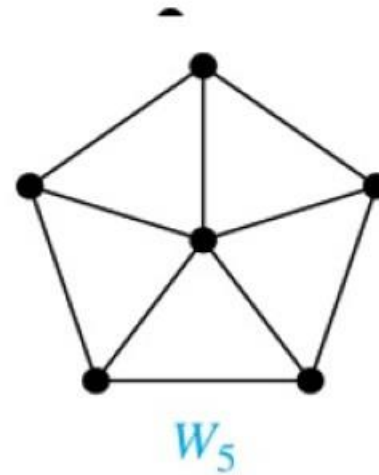
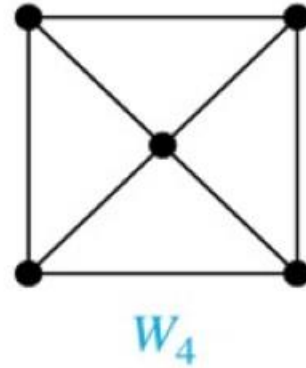
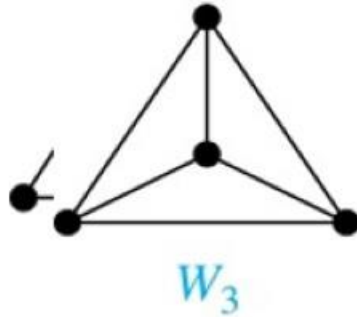


C_6

Special Types of Simple Graphs: Wheels

A *wheel* W_n is obtained by

1. **adding** an additional vertex to a cycle C_n for $n \geq 3$
2. **connecting** this new vertex to each of the n vertices in C_n by new edges.



An n -dimensional hypercube, or n -cube, Q_n , is a graph with 2^n vertices representing all bit strings of length n , where there is an edge between two vertices that differ in **exactly one** bit position.

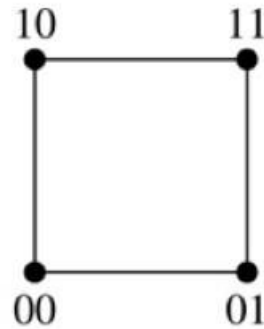
An n -dimensional hypercube, or n -cube, Q_n , is a graph with 2^n vertices representing all bit strings of length n , where there is an edge between two vertices that differ in **exactly one** bit position.

Special Types of Simple Graphs: n -Cubes

An n -dimensional hypercube, or n -cube, Q_n , is a graph with 2^n vertices representing all bit strings of length n , where there is an edge between two vertices that differ in **exactly one** bit position.



Q_1



Q_2

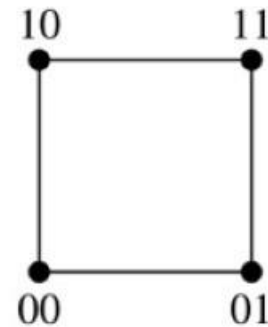
Special Types of Simple Graphs: n -Cubes

An n -dimensional hypercube, or n -cube, Q_n , is a graph with 2^n vertices representing all bit strings of length n , where there is an edge between two vertices that differ in **exactly one** bit position.

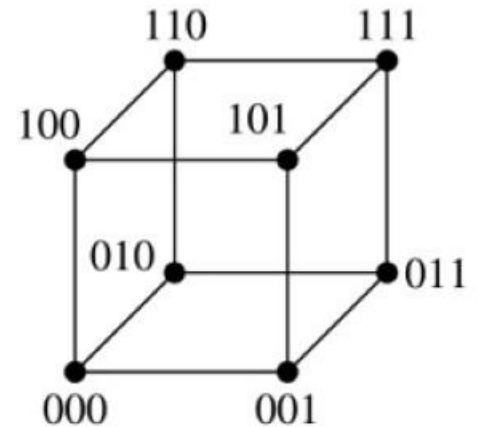
Note that you can construct the $(n + 1)$ -cube Q_{n+1} from the n -cube Q_n by making **two copies** of Q_n , prefacing the labels on the vertices with a **0** in one copy of Q_n and with a **1** in the other copy of Q_n , and adding edges connecting two vertices that have labels differing only in the first bit.



Q_1

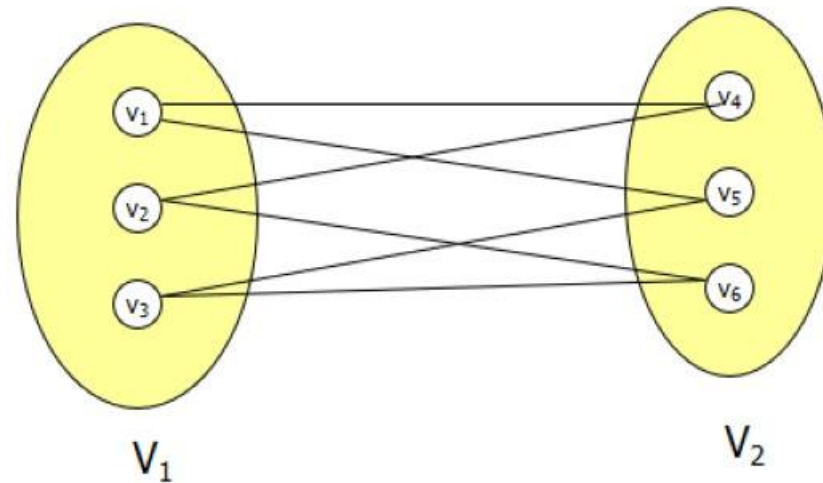


Q_2

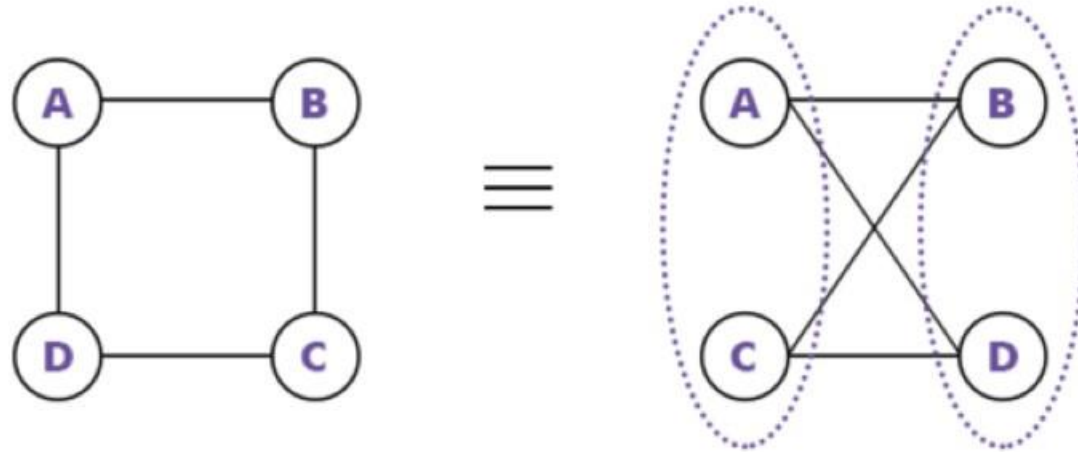


Q_3

Definition: A **simple graph** G is bipartite if V can be partitioned into two **disjoint subsets** V_1 and V_2 such that every edge connects a vertex in V_1 and a vertex in V_2 . In other words, there are **no edges** which connect two vertices in V_1 or in V_2 .

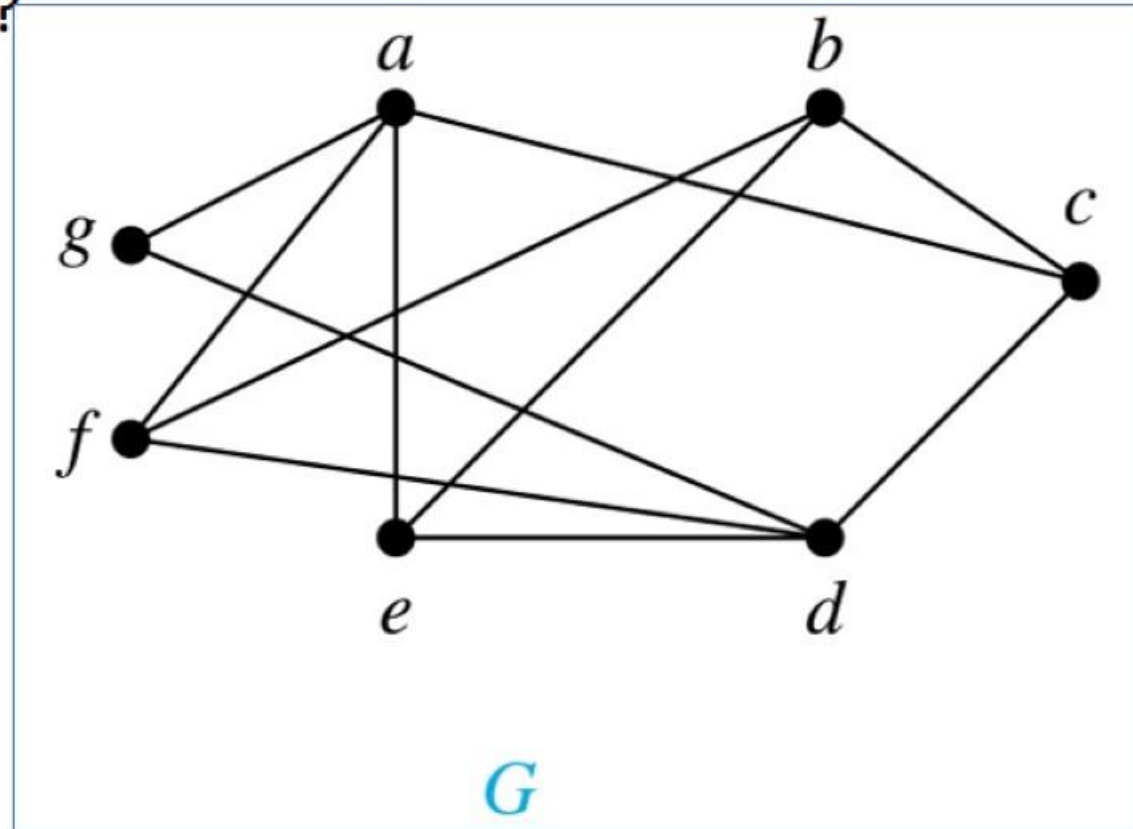


Bipartite Graphs



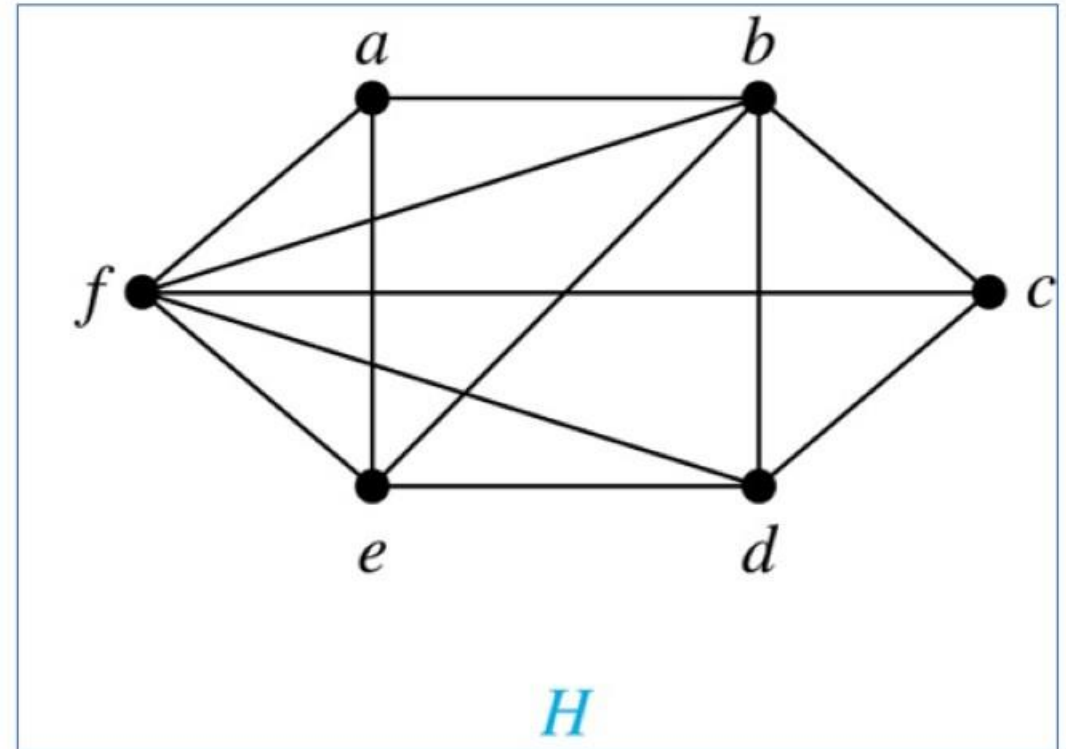
The following graph is bipartite?

G is **bipartite**



The following graph is bipartite?

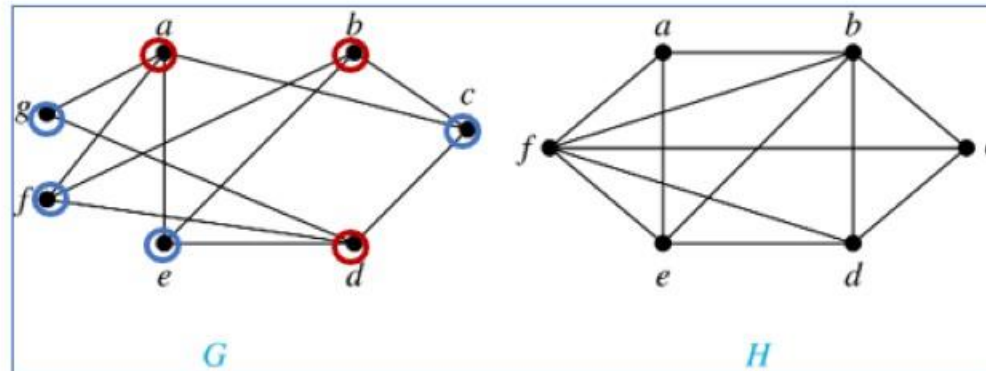
H is not bipartite



Bipartite Graphs

a bipartite graph is a graph where it is possible to **color** the vertices **red** or **blue** so that **no two adjacent vertices are the same color**.

G is
bipartite

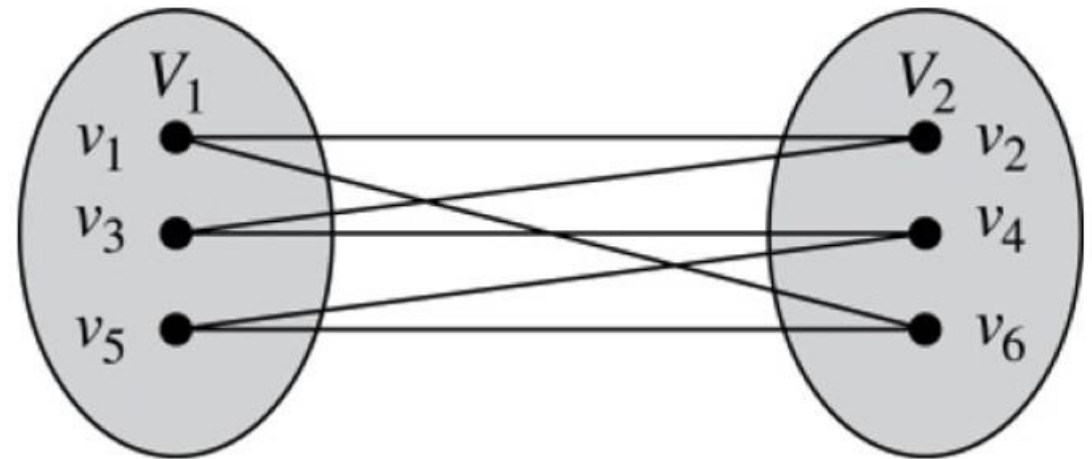
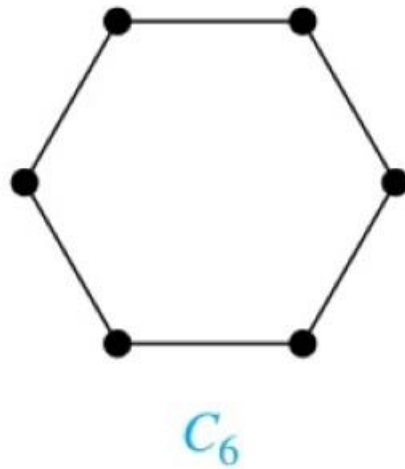


H is not bipartite
since if we color a red,
then the adjacent
vertices f and b must
both be blue.

Show that C_6 is bipartite.

Example: Show that C_6 is bipartite.

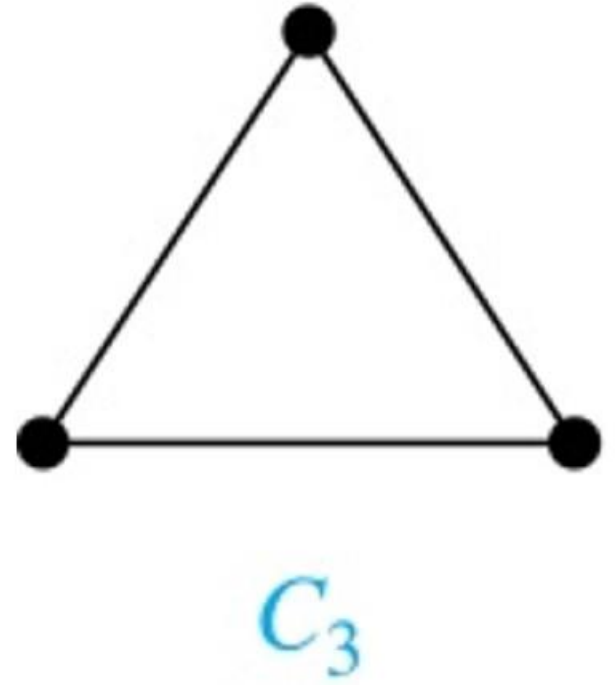
Solution: We can partition the vertex set into $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$ so that every edge of C_6 connects a vertex in V_1 and V_2 .



Example: Show that C_3 is not bipartite.

Example: Show that C_3 is **not** bipartite.

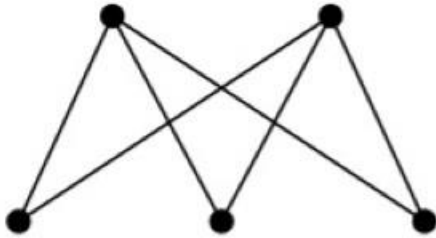
Solution: the two vertices in the same partition are connected. Hence, C_3 is not bipartite.



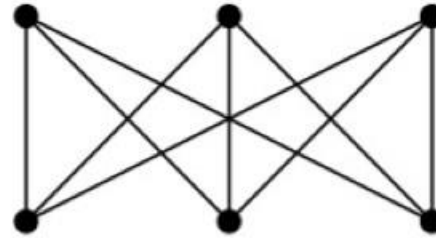
Definition: A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into **two subsets** V_1 of size m and V_2 of size n such that there is an edge from **every vertex in V_1 to every vertex in V_2** .

Complete Bipartite Graphs

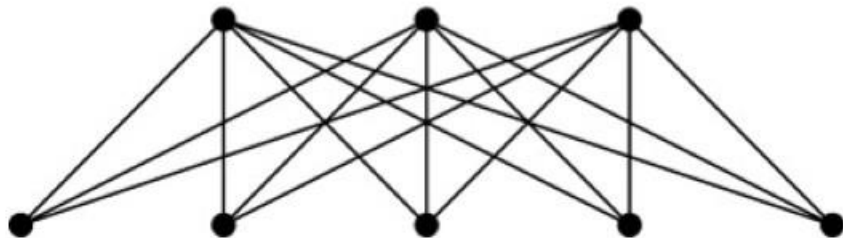
Example: We display four complete bipartite graphs here.



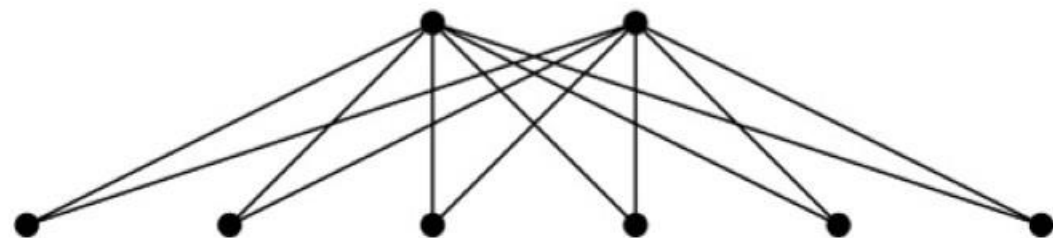
$K_{2,3}$



$K_{3,3}$

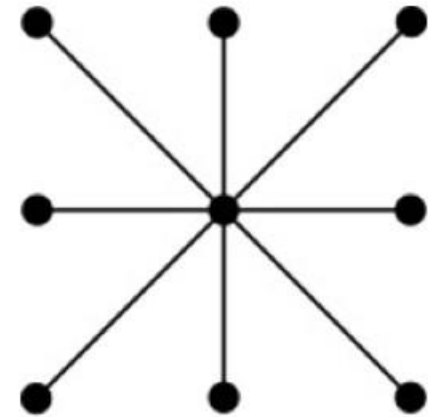


$K_{3,5}$



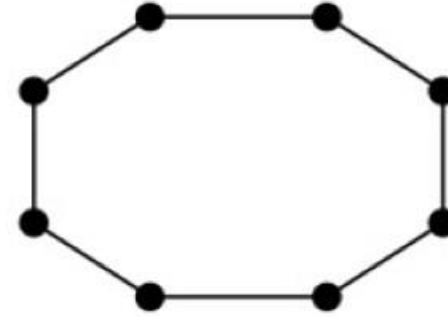
$K_{2,6}$

Various special graphs play an important role in the design of **computer networks**.



- Some local area networks use a **star topology**, which is a complete bipartite graph $K_{1,n}$. All devices are connected to a central control device.

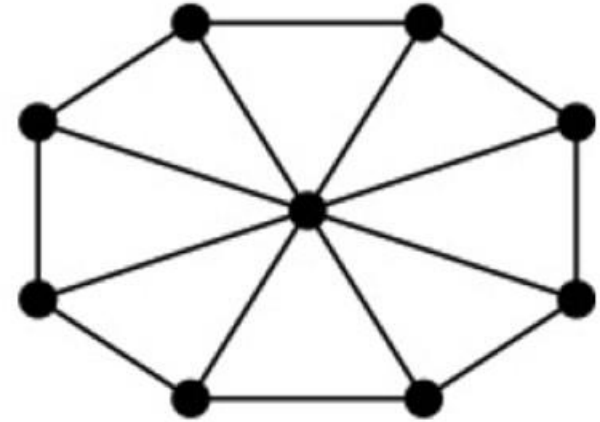
Various special graphs play an important role in the design of **computer networks**.



(b)

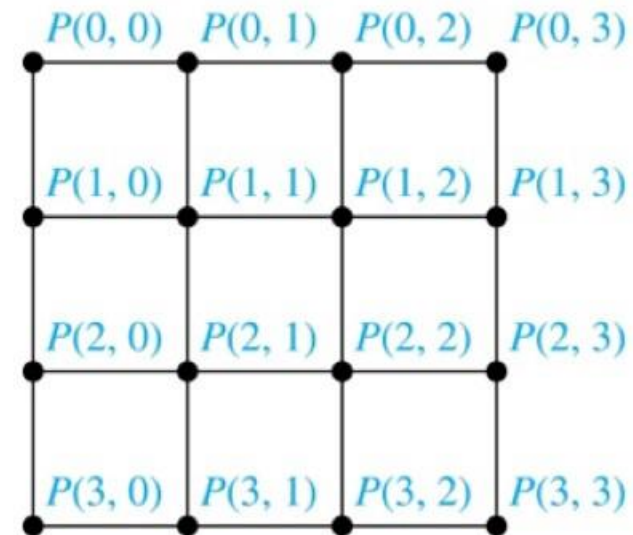
- Other local networks are based on a **ring topology**, where each device is connected to exactly two others using C_n .

Various special graphs play an important role in the design of **computer networks**.



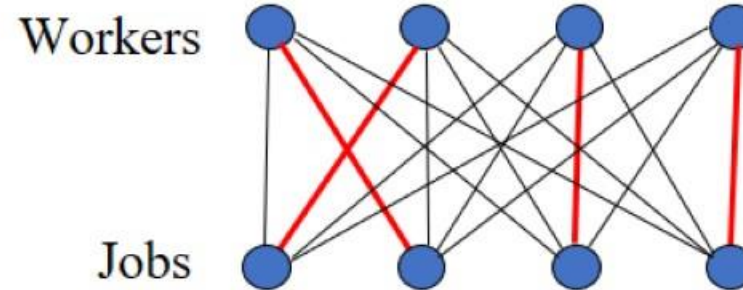
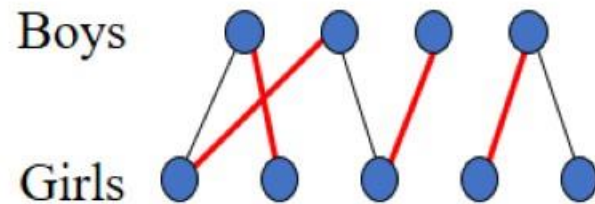
- use a W_n – based topology, combining the features of a star topology and a ring topology.

- Various special graphs also play a role in parallel processing where processors need to be interconnected as one processor may need the output generated by another.
 - The n -dimensional hypercube, or n -cube, Q_n , is a common way to connect processors in parallel, e.g., Intel Hypercube.
 - Another common method is the *mesh* network for 16 processors.

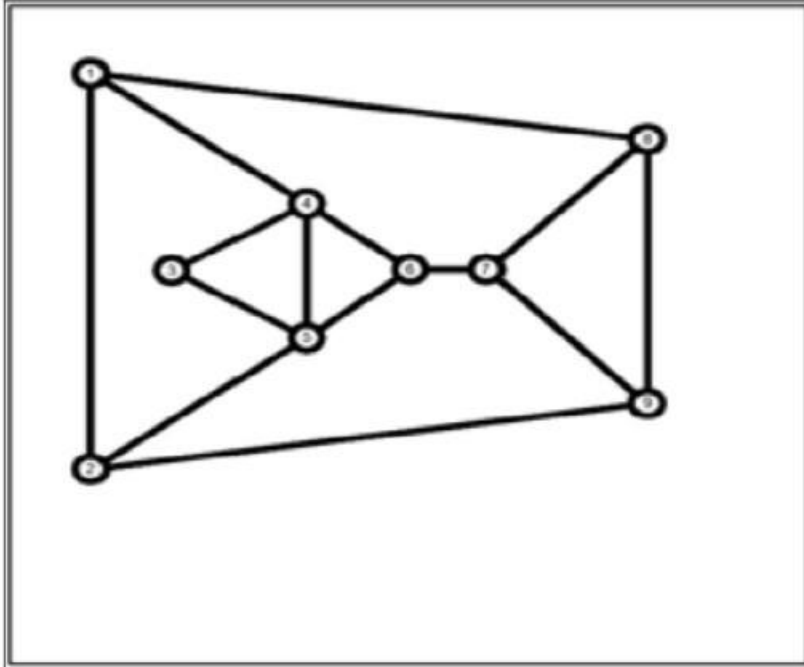


Applications of bipartite graph

- Matching Problem
- Job Assignment Problem

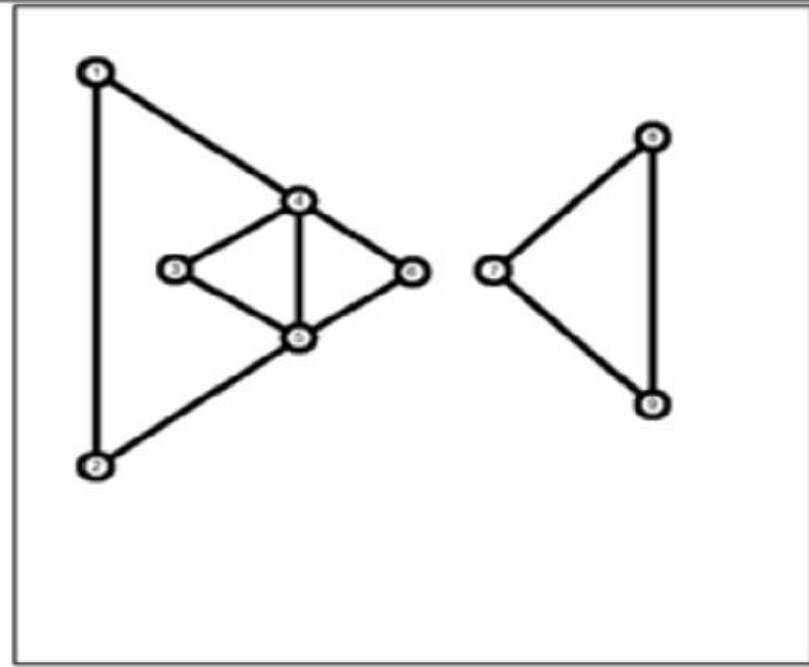


- Basic Idea: In a Graph Reachability among **vertices** by **traversing** the **edges**
- Application Example:
 - In a city to city road-network, if one city can be reached from another city.
 - whether a message can be sent between two computers.
 - efficiently planning routes for mail delivery.



(a)

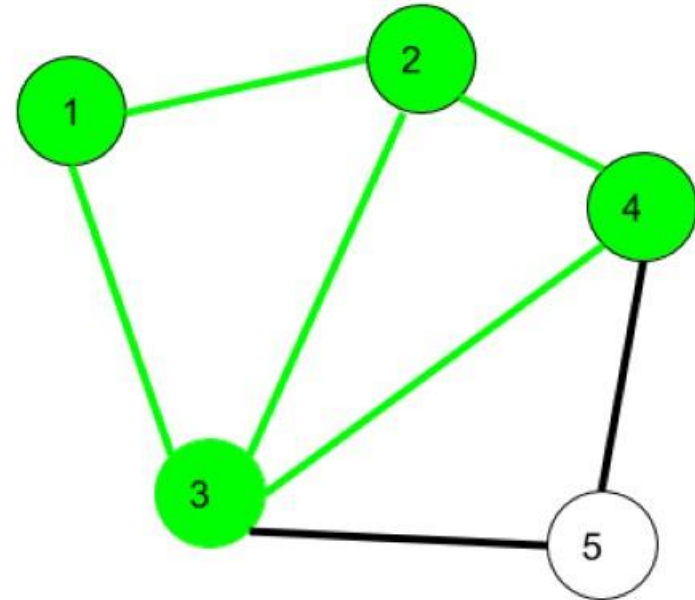
Connected graph



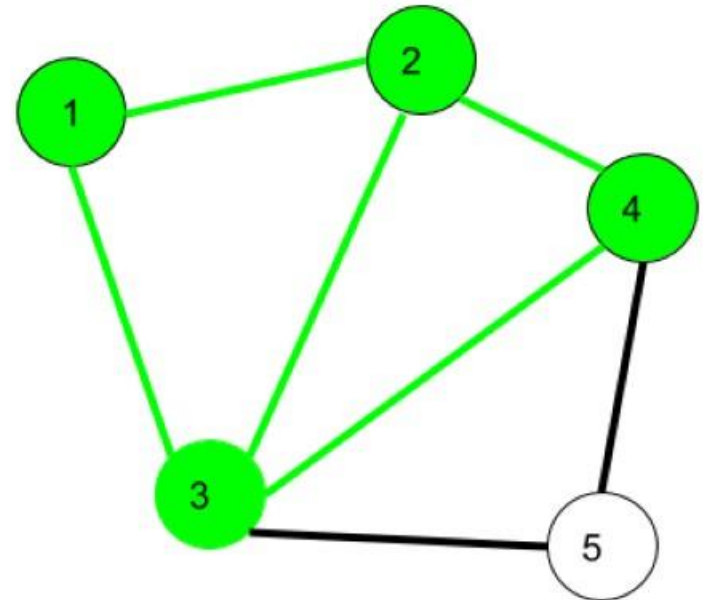
(b)

Disconnected graph

- A walk is a sequence of vertices and edges of a graph i.e. if we **traverse** a graph then we get a walk.
- **Edge** and **Vertices** both can be **repeated**.
 - 1,2,3,4,2,1,3 is a walk.



- **Open walk:** A walk is said to be an open walk if the starting and ending vertices are **different**.
 - **Closed walk:** A walk is said to be a closed walk if the starting and ending vertices are the **same**.
- 1,2,3,4,5,3 is an open walk.
 - 1,2,3,4,5,3,1 is a closed walk.

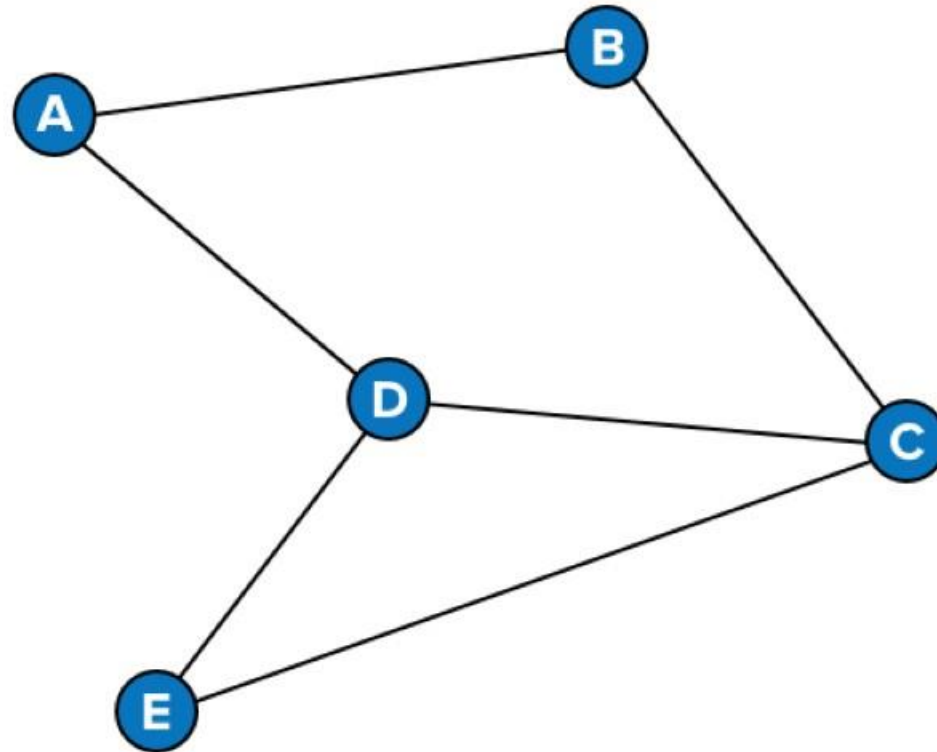


Choose (open walk-closed walk-not a walk)

1. A,B,E,A,B,D
2. A,D,C,B,A
3. A,B,C

Solution

1. not a walk
2. Closed walk
3. Open walk



- Walk with **no repeated edge**
- Closed trail (also called a **circuit**)

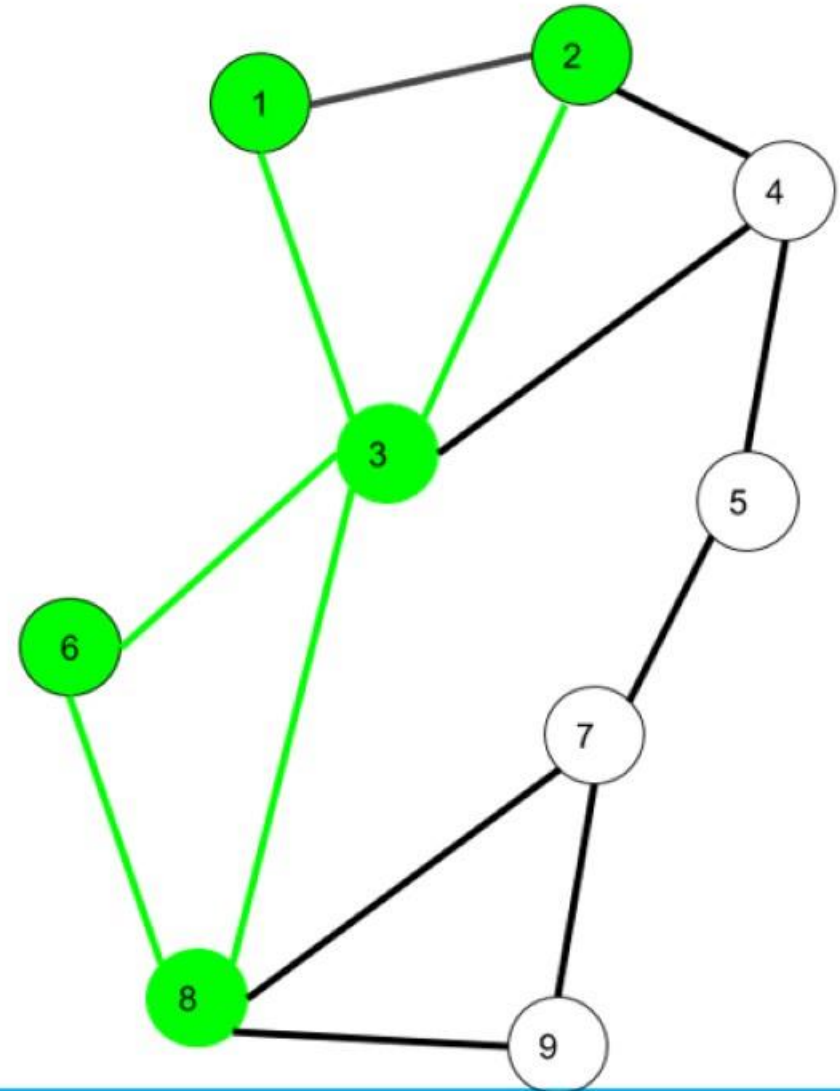
- trail with **no repeated vertex**
- Closed path (also called a **Cycle**)

- Choose (open trail-circuit-a walk)

1. 1,3,6,8,3,1
2. 1,3,8,6,3,2
3. 1,3,8,6,3,2,1

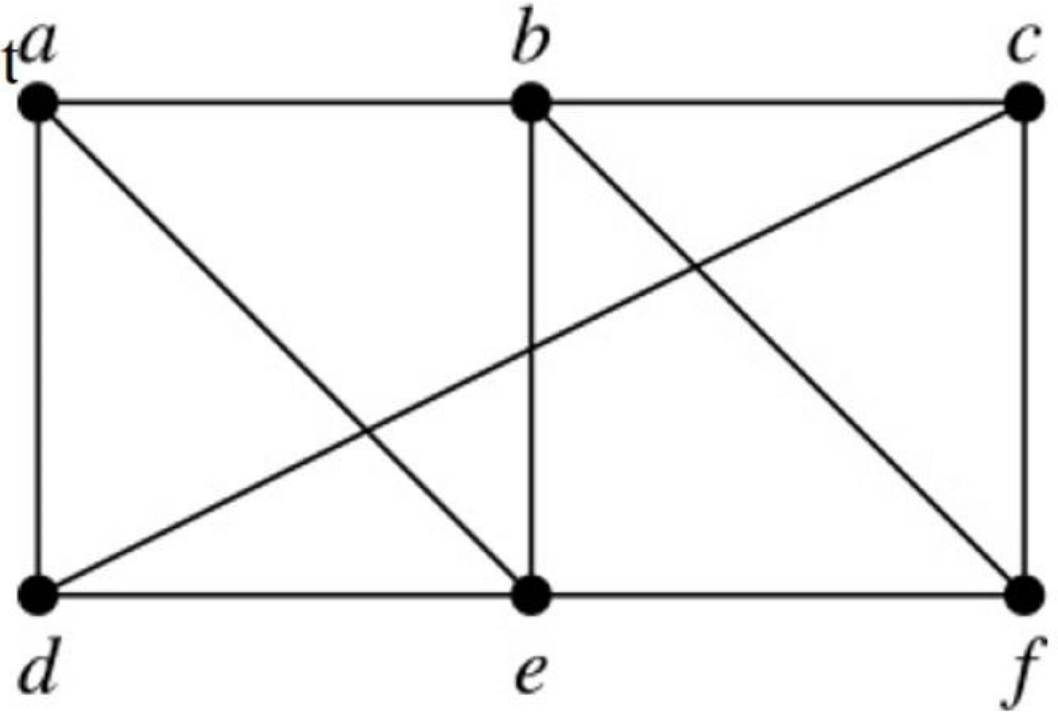
- Solution

- Walk
- Open trail
- Closed trail (circuit)



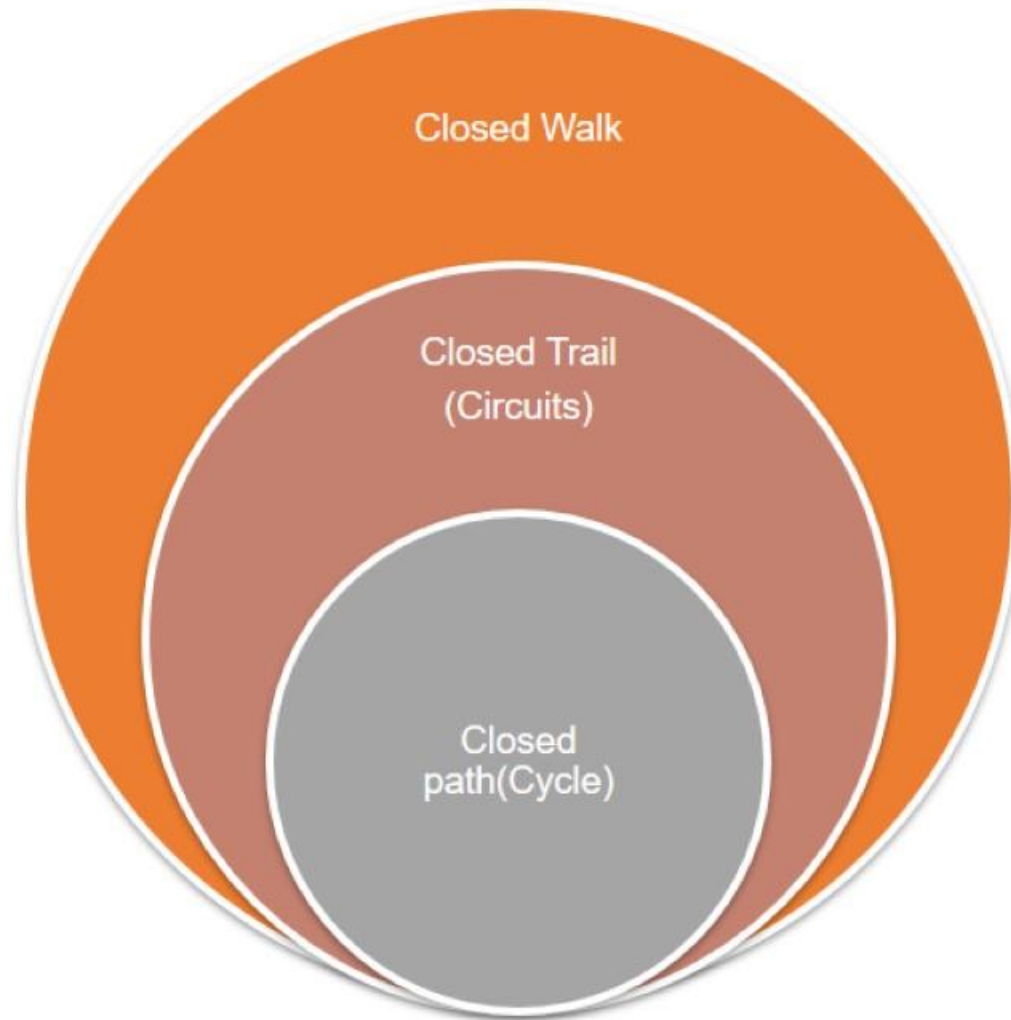
Determine if a path, circuit simple or not

- a, d, c, f, e
- d, e, c, a .
- b, c, f, e, b
- a, b, e, d, a, b

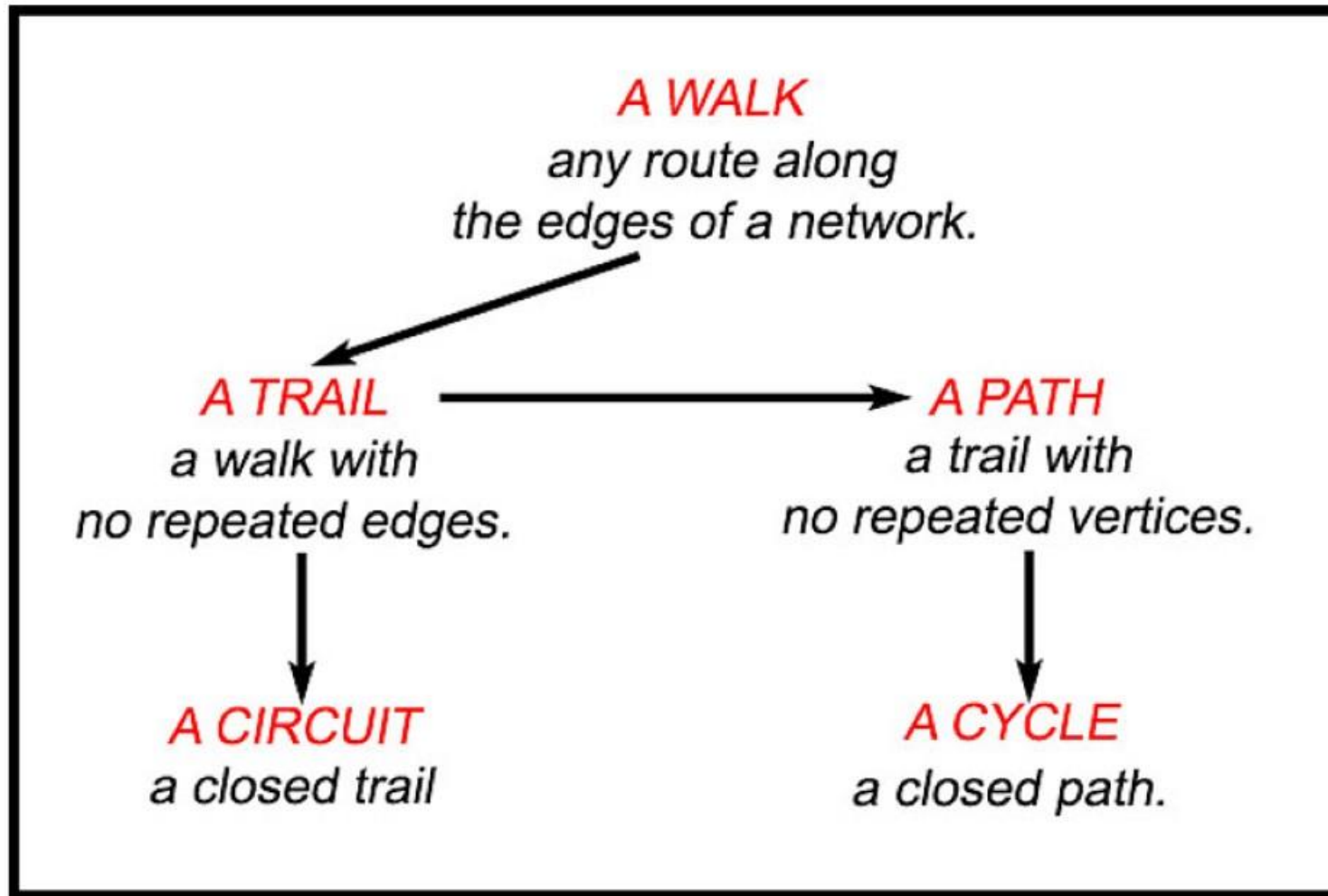


- a, d, c, f, e is a simple path of length 4.
- d, e, c, a is not a path because e is not connected to c .
- b, c, f, e, b is a cycle of length 4.
- a, b, e, d, a, b is a walk.

Wal vs trail vs path

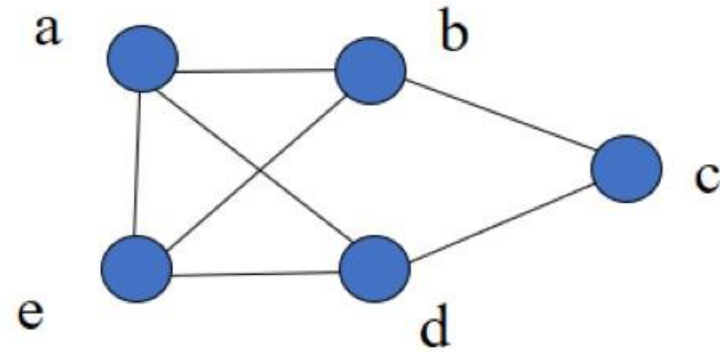


Walk vs Trail vs Path



Choose(path,cycle,walk)

- *a, d, c, b, e*
- *a, b, e, d, c, b, e, d*
- *a, d, c, b, e, a*

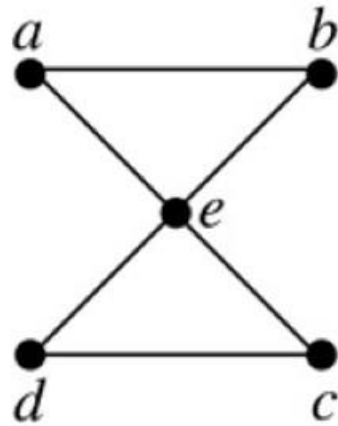


- Solution
 - *a, d, c, b, e*) is a path
 - *(a, b, e, d, c, b, e, d)* is not a path; it is a walk.
 - *a, d, c, b, e, a)* is a cycle

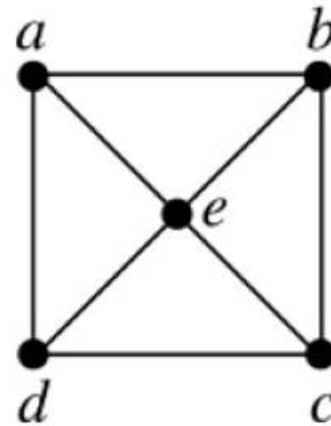
Definition: An *Euler circuit* in a graph G is a **simple circuit containing every edge of G .**

An *Euler path* in G is a **simple path containing every edge of G .**

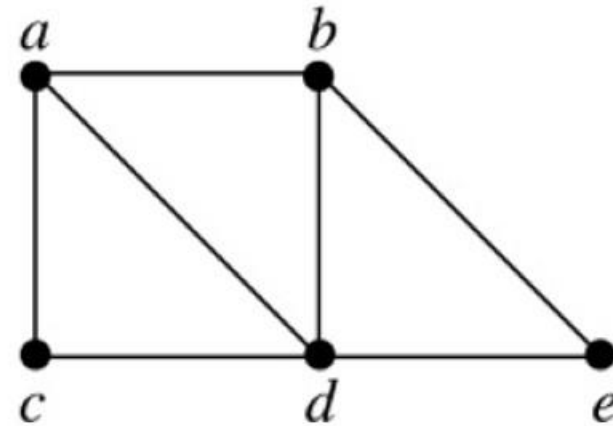
Example: Which of the undirected graphs G_1 , G_2 , and G_3 has a Euler circuit?



G_1



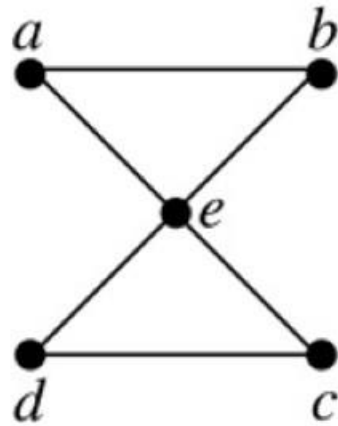
G_2



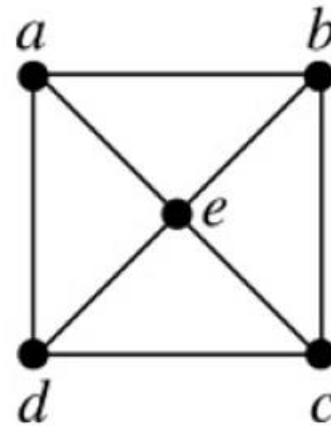
G_3

The graph G_1 (e.g., a, e, c, d, e, b, a)

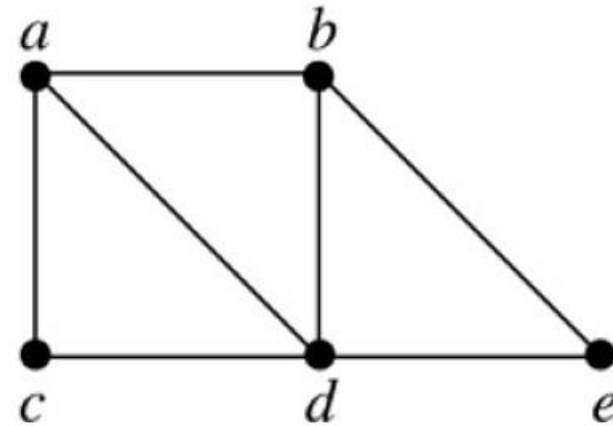
Example: Which of the undirected graphs G_1 , G_2 , and G_3 has a Euler path?



G_1



G_2

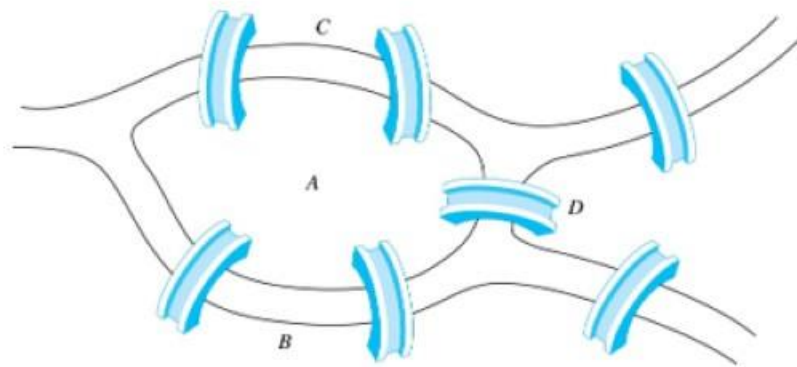


G_3

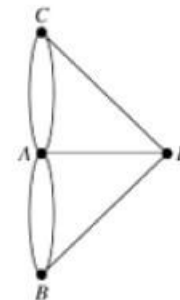
The graph G_3 (e.g., a, c, d, e, b, d, a, b),

Euler Paths and Circuits

- The town of Königsberg, Prussia (now Kalingrad, Russia) was divided into four sections by the branches of the Pregel river. In the 18th century seven bridges connected these regions.
- People wondered whether whether it was possible to follow a path that crosses each bridge exactly once and returns to the starting point.
- The Swiss mathematician Leonard Euler proved that no such path exists. This result is often considered to be the first theorem ever proved in graph theory.



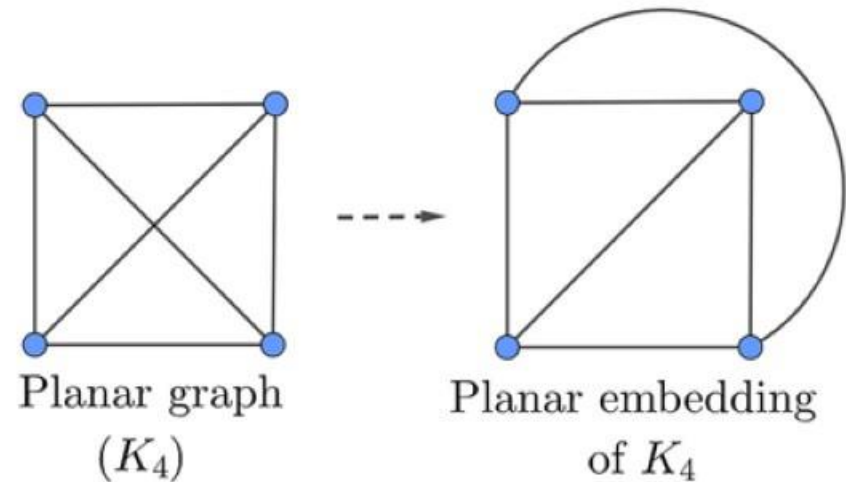
The 7 Bridges of Königsberg



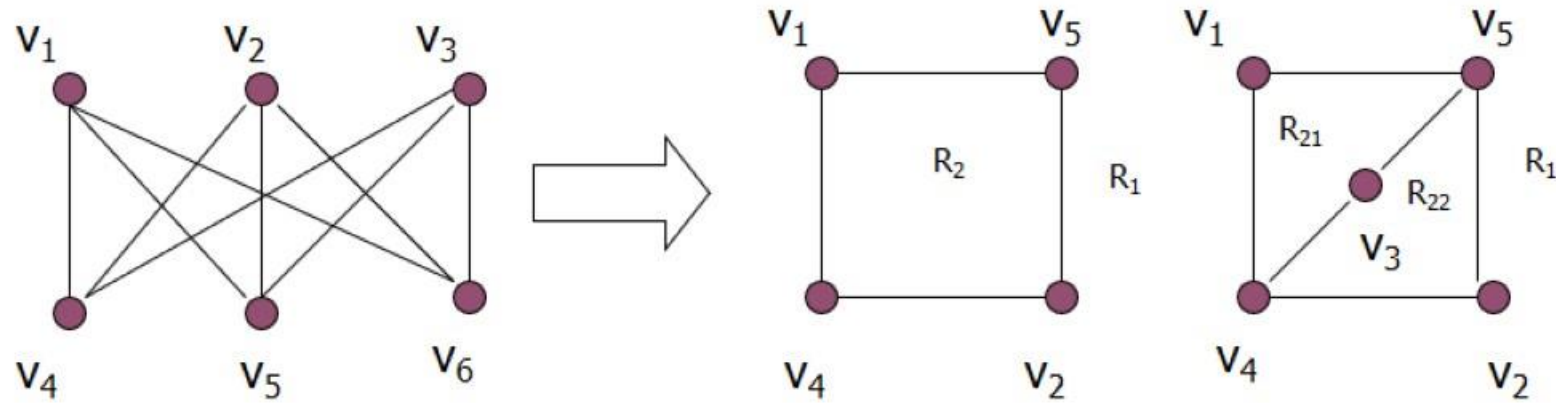
Multigraph Model of
the Bridges of
Königsberg

- A **graph** (or **multigraph**) G is called *planar* if G can be drawn in the plane with its edges intersecting only at vertices of G , such a drawing of G is called an *embedding* of G in the plane.

Application Example: VLSI design (overlapping edges requires extra layers), Circuit design (cannot overlap wires on board)



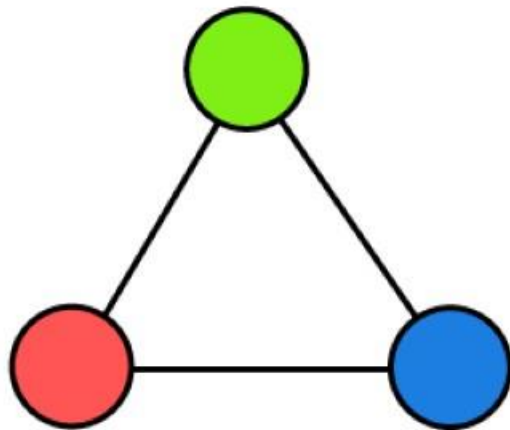
- Representation examples: $K_{3,3}$ is Nonplanar



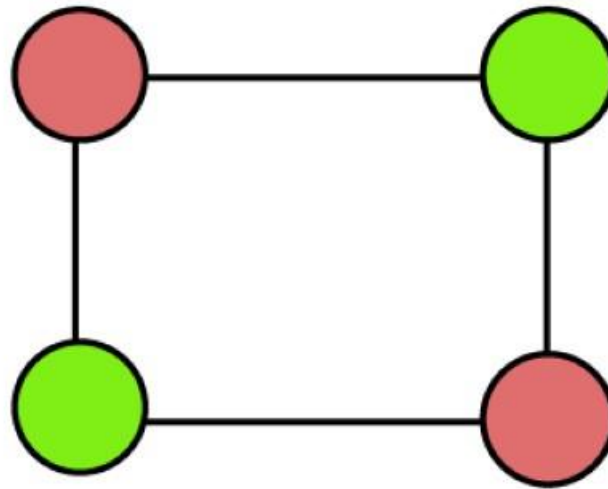
- **Graph coloring** is an assignment of "*colors*", almost always taken to be consecutive integers starting from 1 without loss of generality, to certain objects in a graph. Such objects can be vertices, edges, faces, or a mixture of the above.
- Application examples: scheduling, register allocation in a microprocessor, frequency assignment in mobile radios, and pattern matching

Vertex Coloring Problem

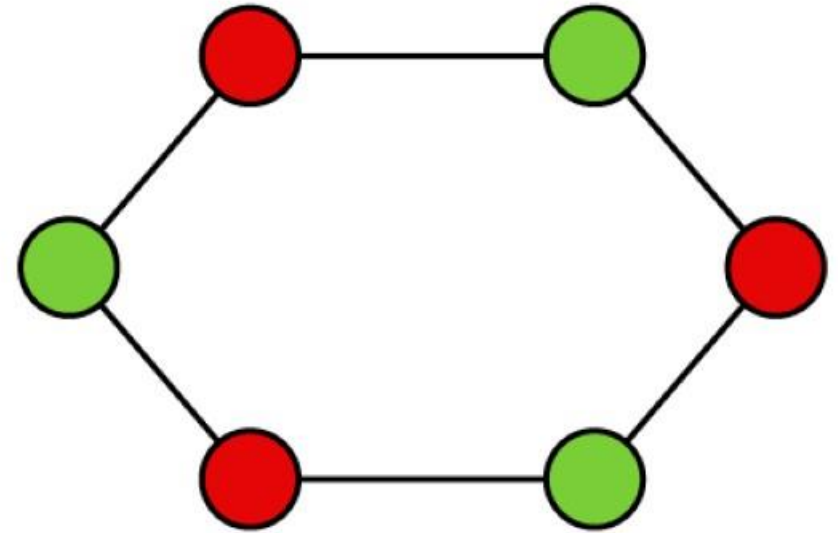
- Assignment of colors to the vertices of the graph such that proper coloring takes place (no two **adjacent vertices are assigned the same color**)
- **Chromatic number**: least number of colors needed to color the graph



Chromatic number=3



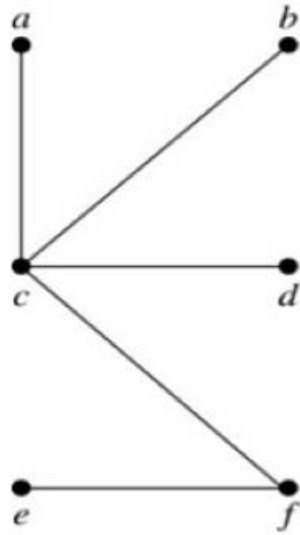
Chromatic number=2



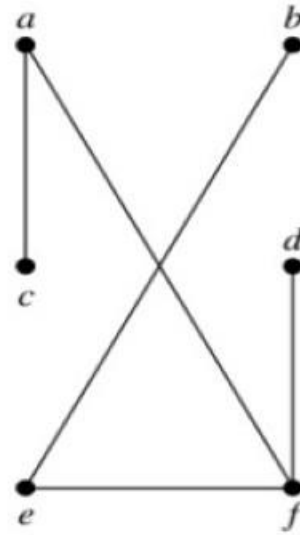
Chromatic number=2

Definition: A *tree* is a connected undirected graph with no simple circuits.

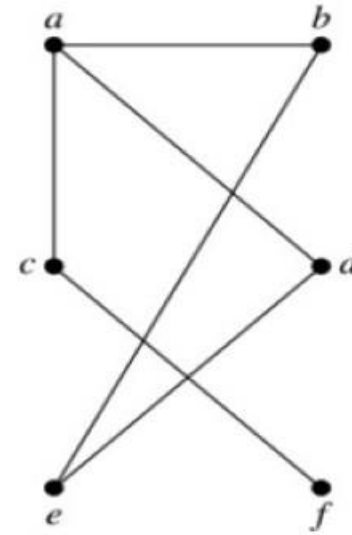
Example: Which of these graphs are trees?



G_1

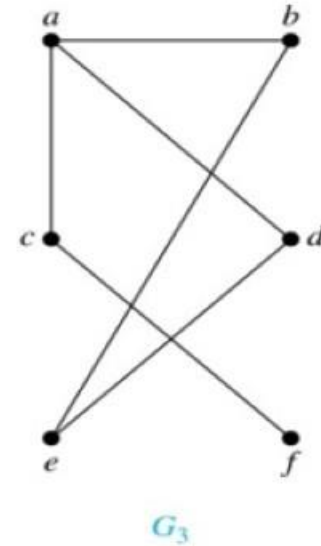
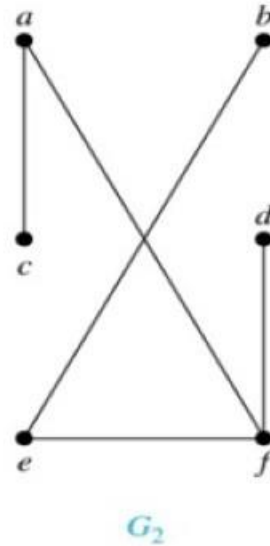
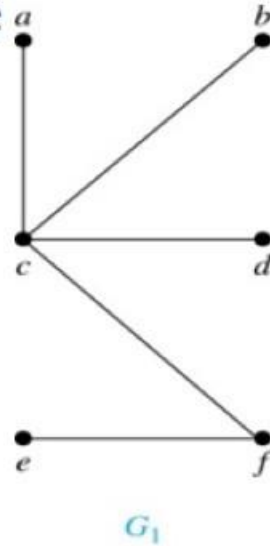


G_2



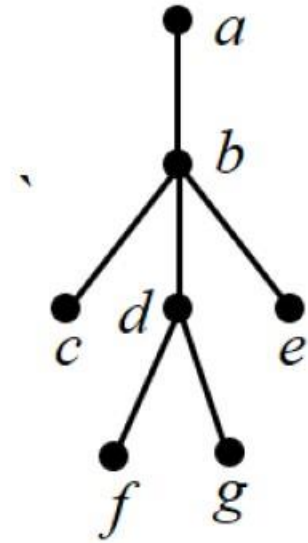
G_3

Example: Which of these graphs are trees?



Solution: G_1 and G_2 are trees - both are connected and have no simple circuits

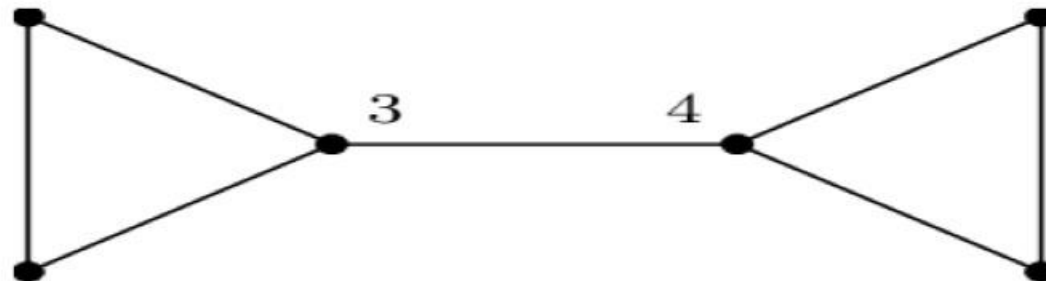
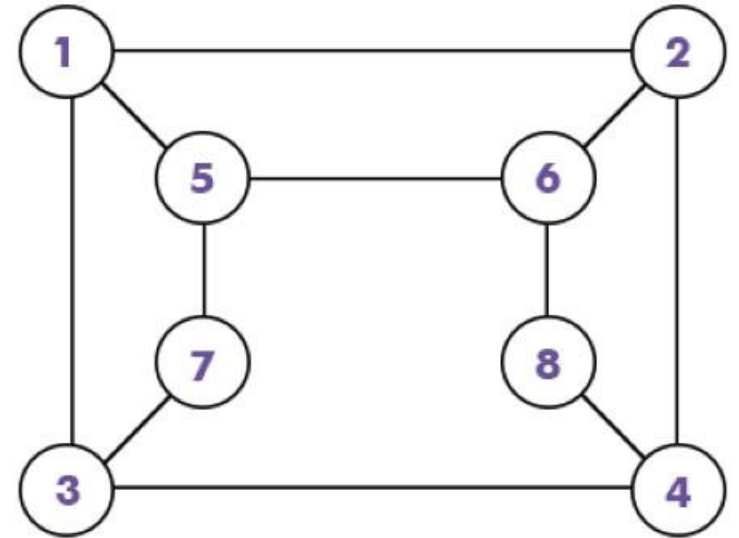
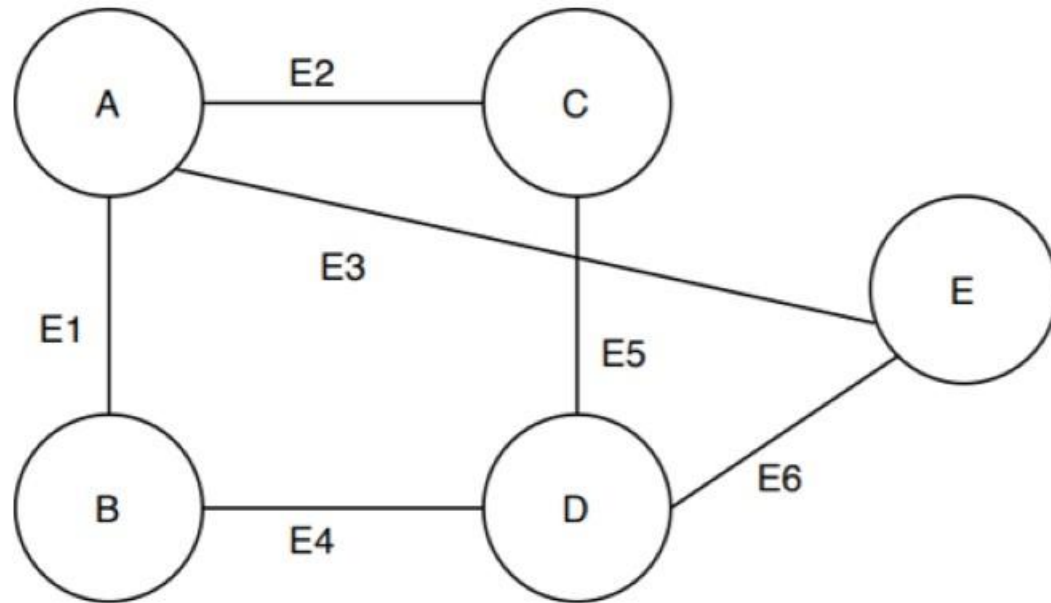
Because e, b, a, d, e is a simple circuit, G_3 is not a tree.



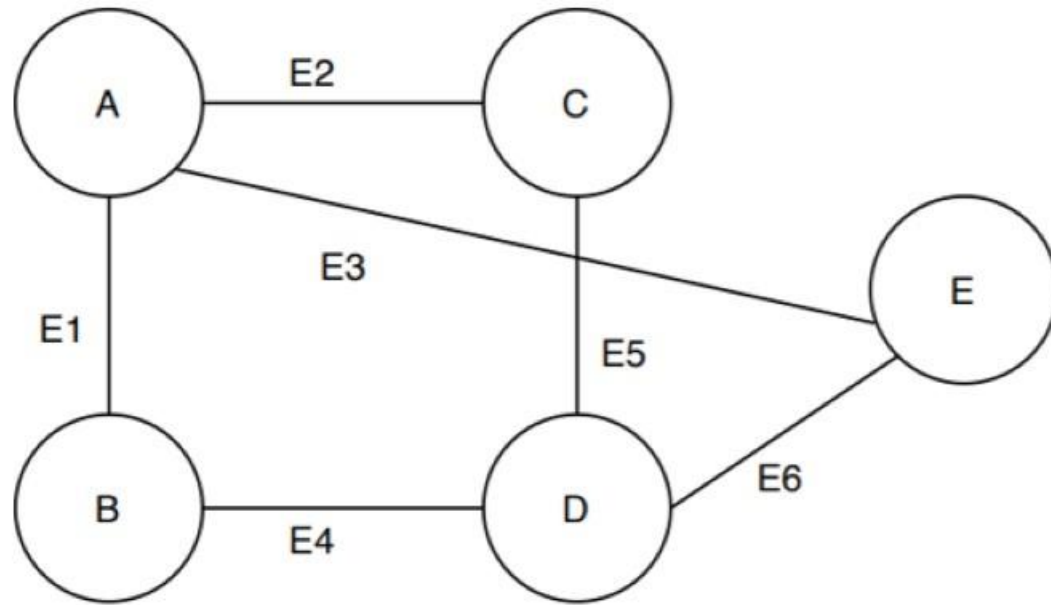
a is the **parent** of b ,
 b is the **child** of a ,
 c, d, e are **siblings**,
 c, e, f, g are **leaves**

Recap

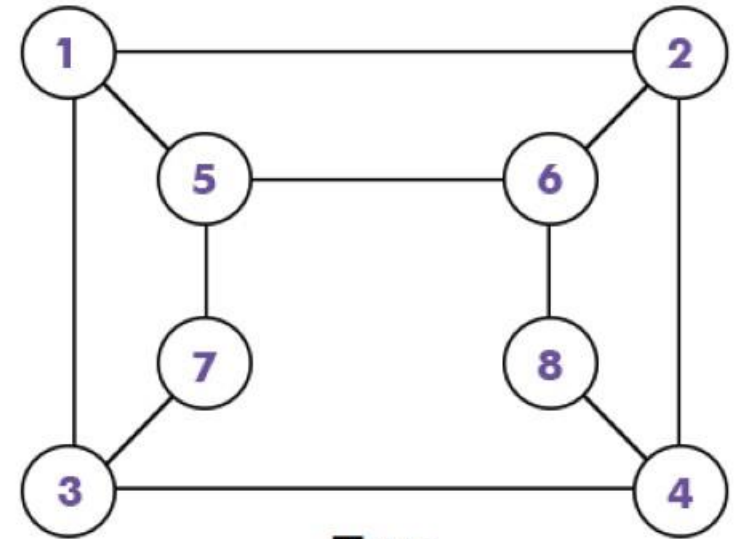
- Bipartite graph ?



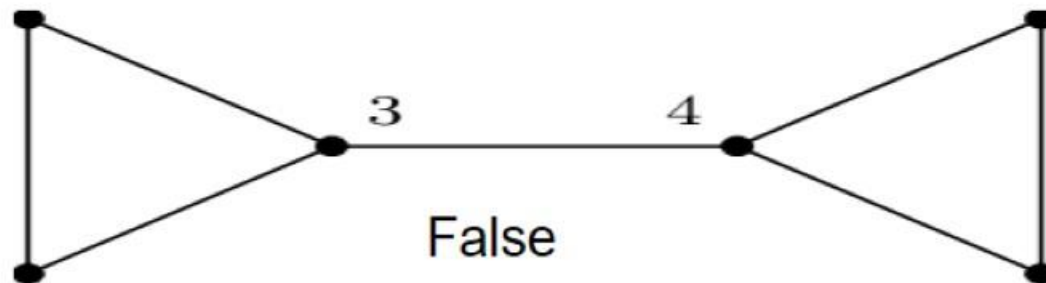
- Bipartite graph ?



True

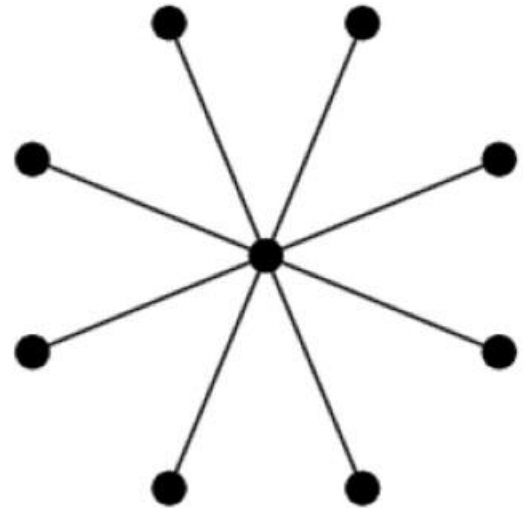
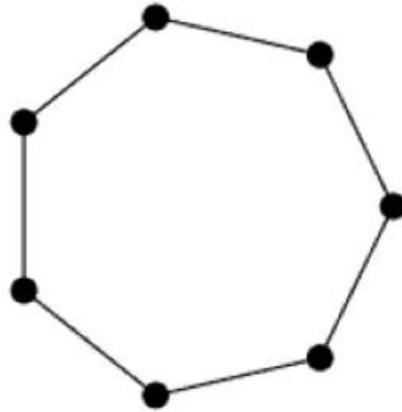
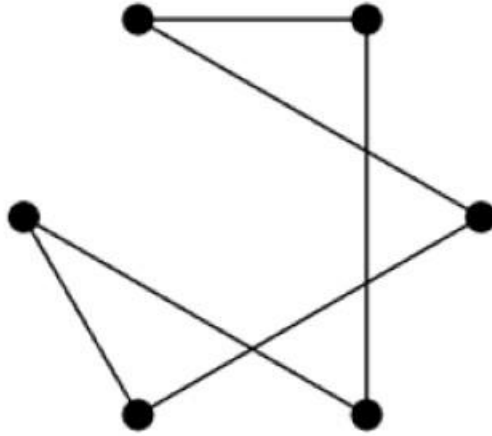
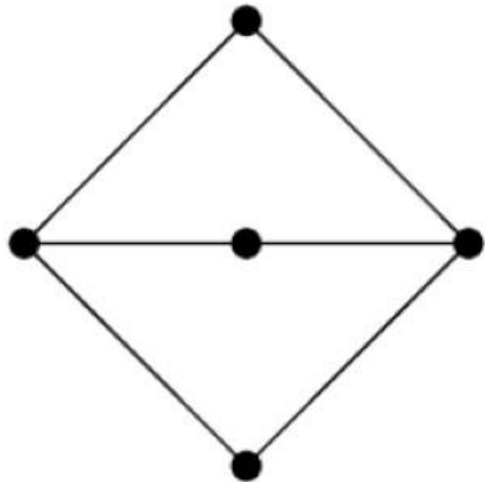


True

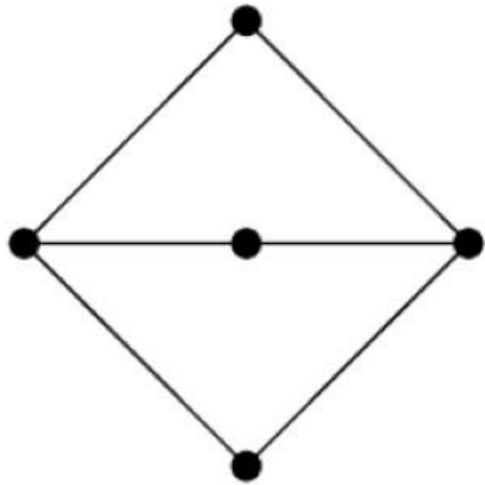


False

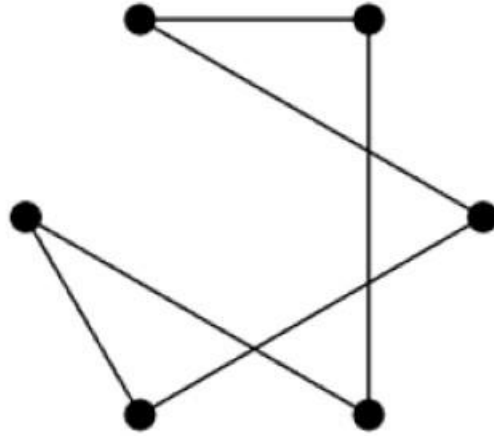
Bipartite graph ?



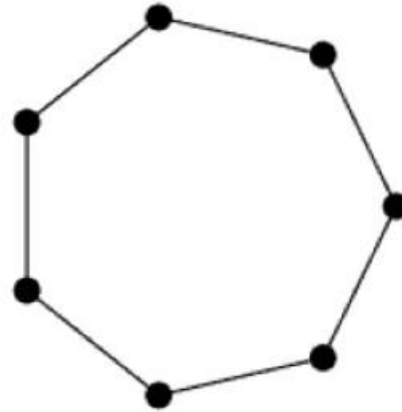
Bipartite graph ?



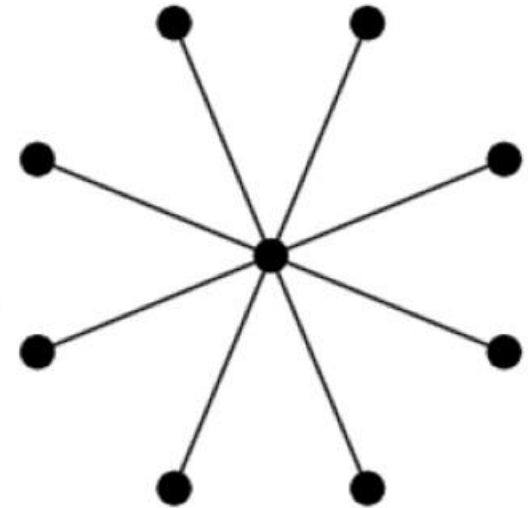
True



True

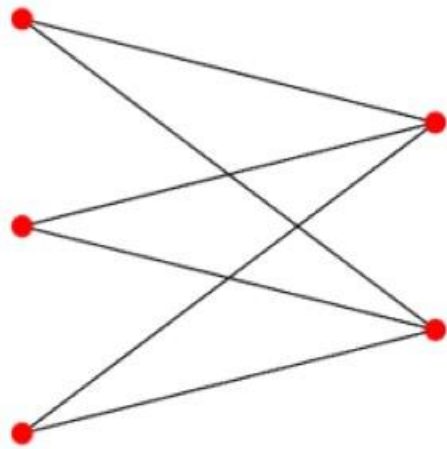


False

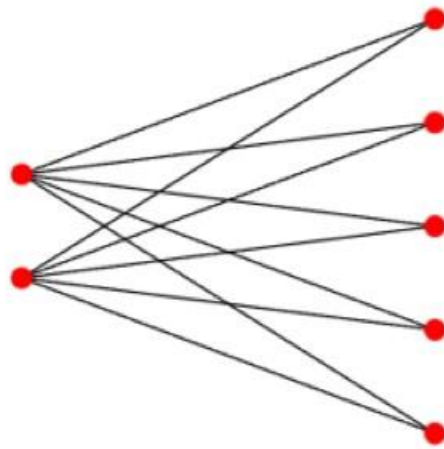


True

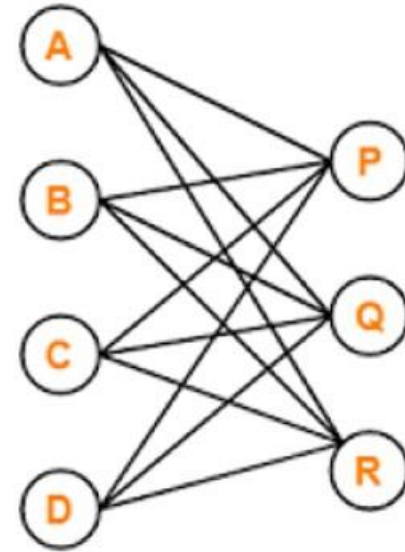
Bipartite graph ?



True

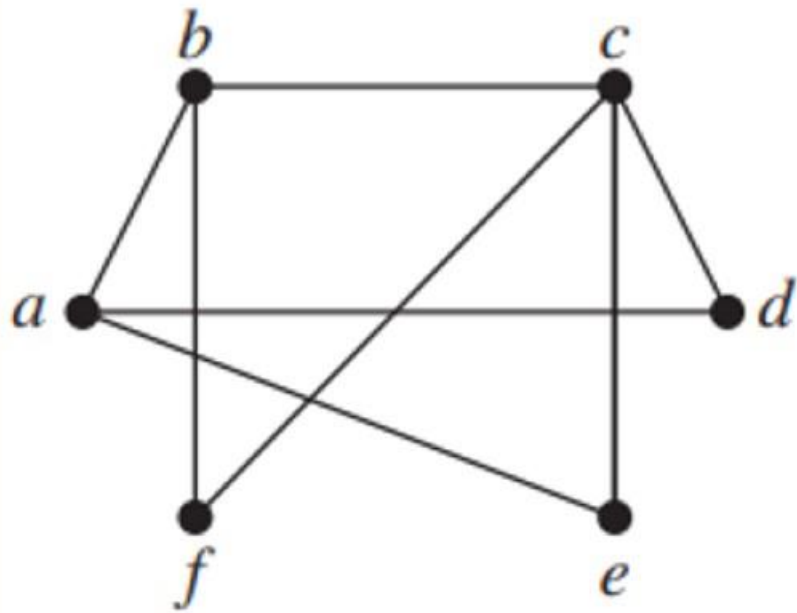


True

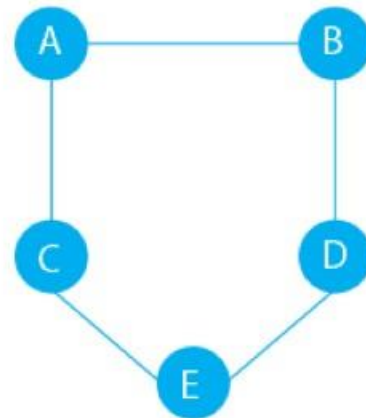


True

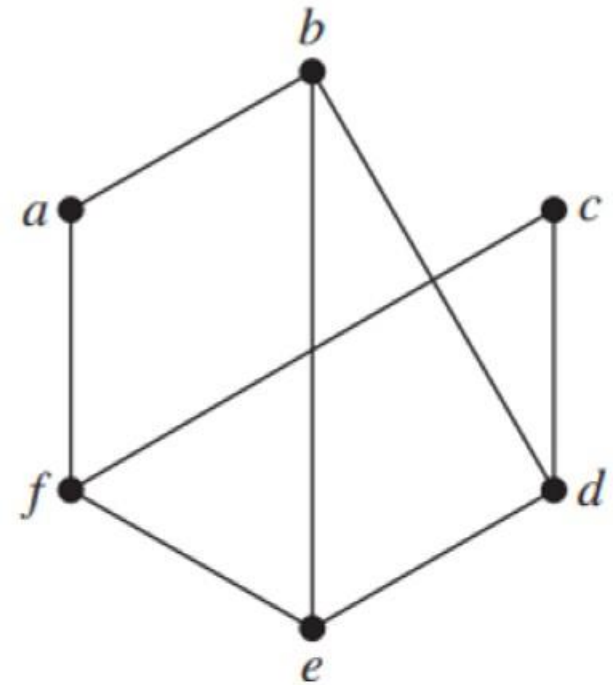
- Bipartite graph ?



False

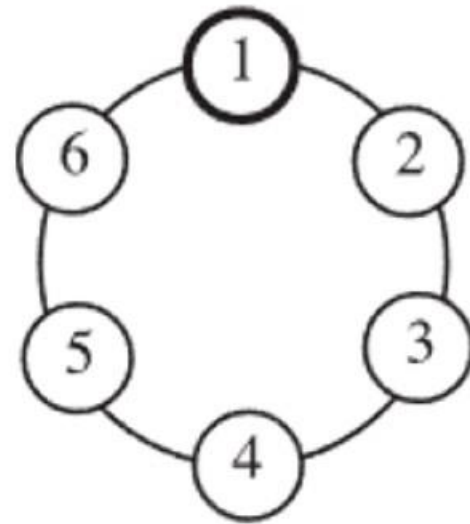
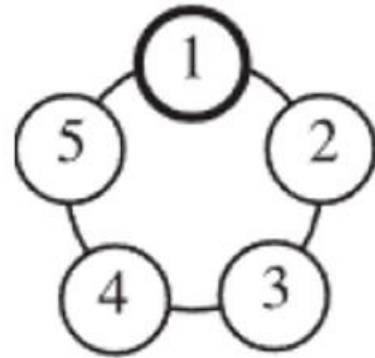
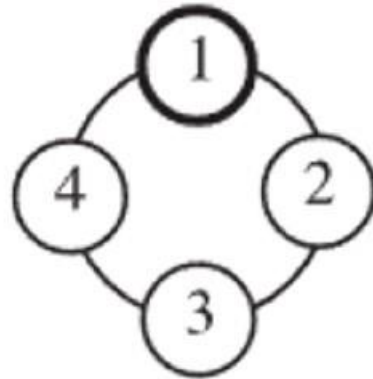
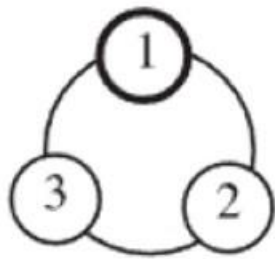


False



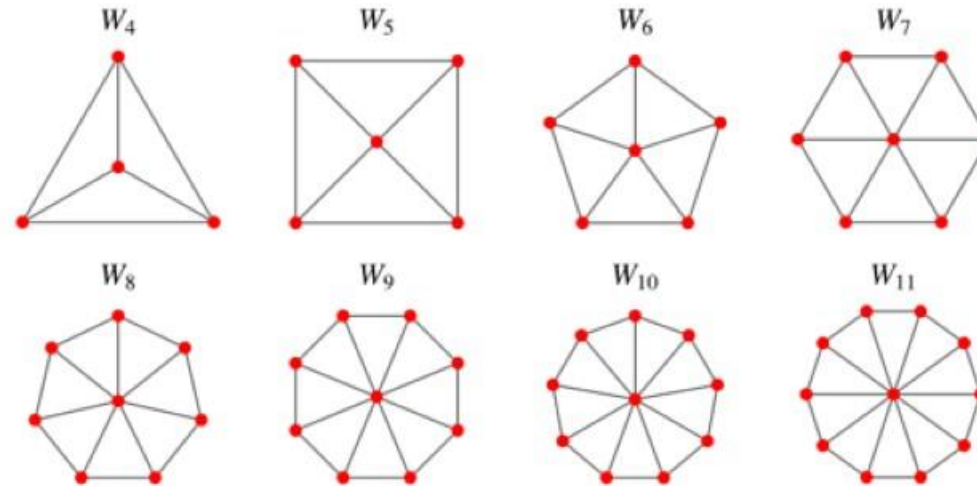
False

- Cycle or wheel or complete graph?



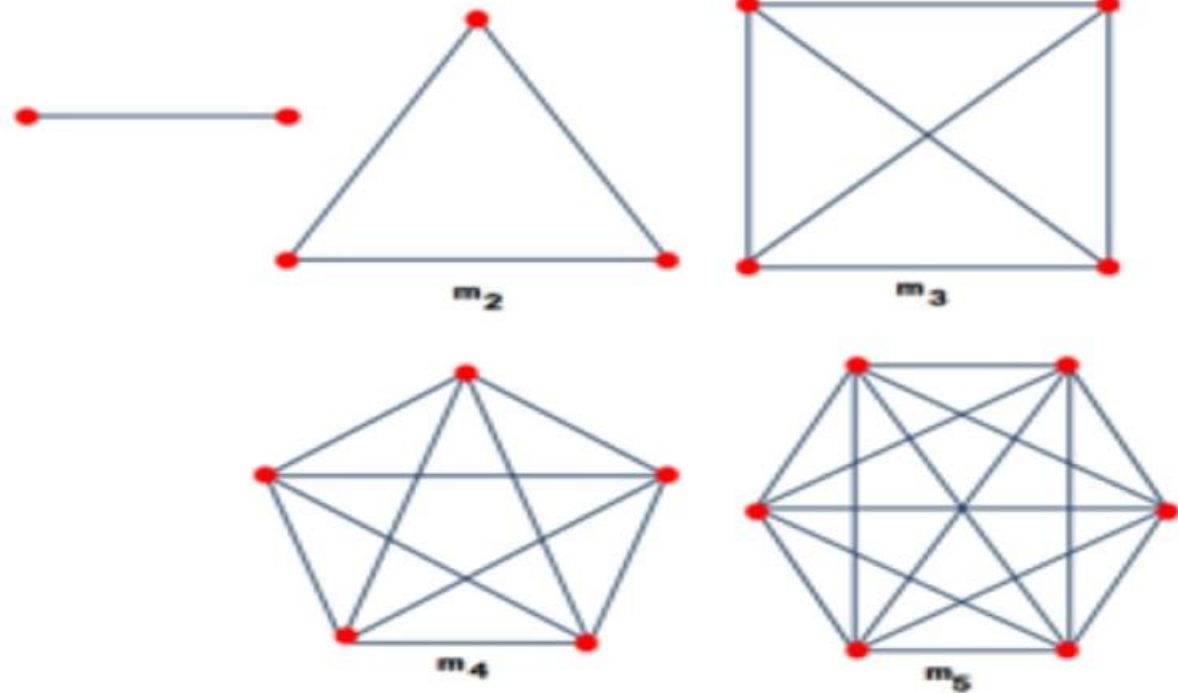
- Cycle

- Cycle or wheel or complete graph?



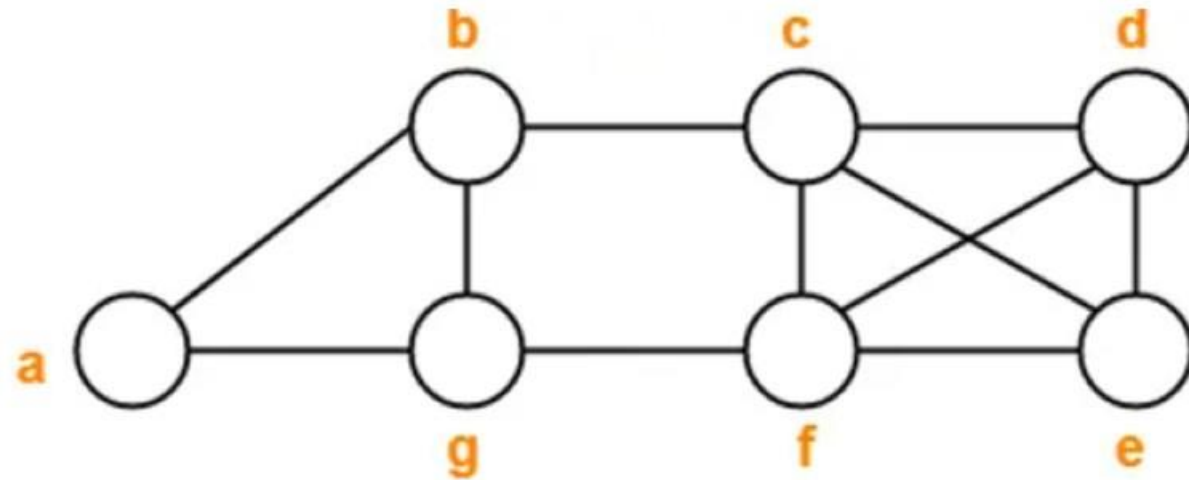
- Wheel

- Cycle or wheel or complete graph?



- A complete graph is a graph in which every vertex has an edge to all other vertices

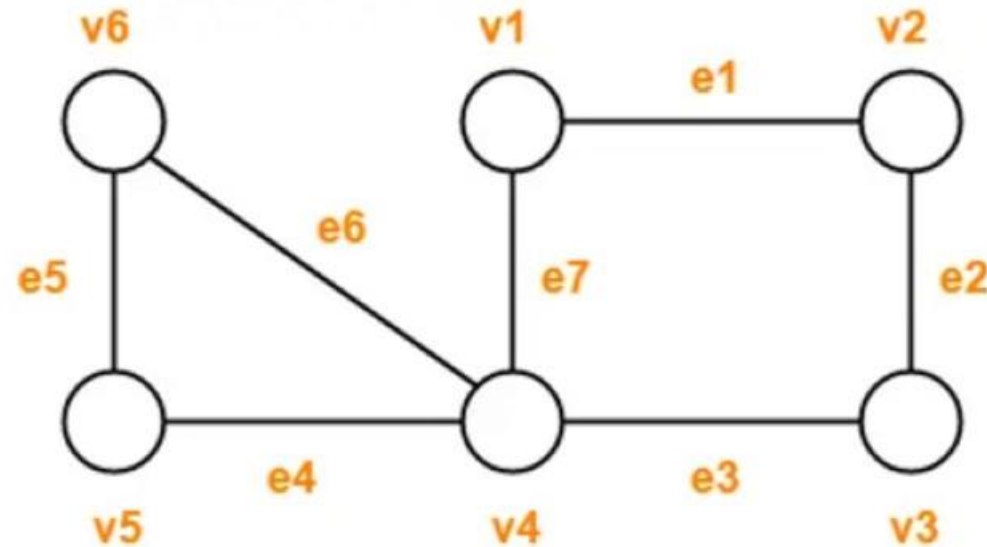
- Walk, circuit, a path, a cycle or a trail?



1. a , b , g , f , c , b
2. b , g , f , c , b , g , a
3. c , e , f , c
4. c , e , f , c , e
5. a , b , f , a
6. f , d , e , c , b

1. Trail
2. Walk
3. Cycle
4. Walk
5. Not a walk
6. Path

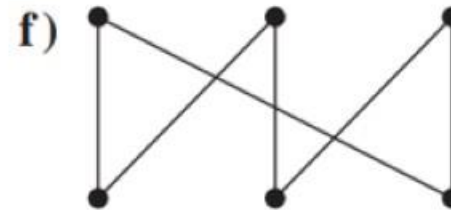
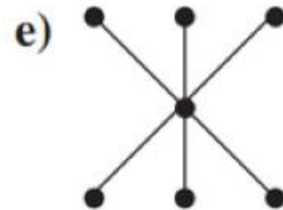
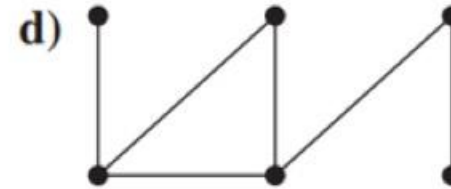
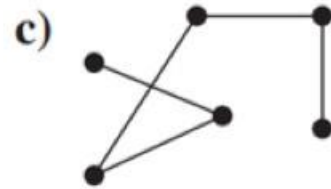
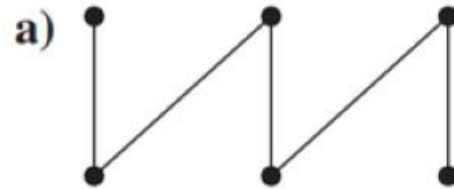
- Walk, circuit, a path, a cycle or a trail?



1. Open walk
2. Trail
3. Path
4. Cycle
5. Circuit

1. $v_1v_2v_3v_2$
2. $v_4v_1v_2v_3v_4v_5$
3. $v_1v_2v_3v_4v_5$
4. $v_1v_2v_3v_4v_1$
5. $v_6v_5v_4v_3v_2v_1v_4v_6$

- Which of these graphs are trees?



a,e,c

1. A complete graph on n vertices, denoted by K_n , is the simple graph that contains **exactly** ----- edge between **each pair** of distinct vertices.
2. **Close Path** called
3. **Closed trail** called
4. Facebook Friendship graphs - -----graphs
5. Twitter is -----graph-
6. **World Wide Web** : In a web graph, **web pages** are represented by vertices and **links** are represented by ----- edges.
7. An **Euler circuit** in a graph G is a **simple circuit** containing --- edge of G .
8. An **Euler path** in G is a **simple** path **containing** -- edge of G .

- 1) One
- 2) Cycle
- 3) Circuit
- 4) Undirected
- 5) Directed
- 6) Directed
- 7) Every
- 8) Every

1. wheel is obtained by -----an additional vertex to a cycle C_n for $n \geq$ ---
2. A trail is a walk where ----- edge is repeated.
3. An open trail starts and ends on ----- vertices.
4. A closed trail (circuit) starts and ends on the -----vertex.
5. Path -----repeated edge ,-----repeated vertices
6. Trail-----repeated edge ,-----repeated vertices
7. Walk -----repeated edge ,-----repeated vertices of G .
8. A tree is a connected undirected graph with ----- simple circuits.
9. A planer graph (or multigraph) G is called planar if G can be drawn in the plane with its edges intersecting only at -----

Quiz

1. Add,3
2. no.
3. different
4. A same
5. No No
6. No has
7. has has .
8. no.
9. vertices

*Thank
you*

