

CMPT-335 Discrete Structures

SLIGHT INTRO TO COMBINATORICS AND DISCRETE PROBABILITY

Permutations

- (General Definition) A **permutation** is an **ordered sequence** of elements selected from a given finite set, **without repetitions**, and **not necessarily using all elements** of the given set.
- For example, given the set of letters {C, E, G, I, N, R}, some permutations are ICE, RING, RICE, REIGN and CRINGE

Permutations

- The total number of different permutations of n elements of a set with the cardinality n is

$$P(n) = n \times (n-1) \times (n-2) \times \dots \times 2 \times 1 = n!$$

$$P(n) = n!$$

Permutations

- The number of different (ordered) **permutations** (**arrangements**) of ***r*** elements selected from ***n*** (***r*** -permutations) is

$$P(n, r) = A_r^n = n \times (n-1) \times \dots \times (n-r+1) = \frac{n!}{(n-r)!}$$

$$P(n, r) = A_r^n = \frac{n!}{(n-r)!}$$

Combinations

- A **combination** is an **un-ordered** collection of **distinct** elements, usually of a prescribed size and taken from a given set.
- Given a set S , a combination of elements of S is just a **subset** of S , where, as always for (sub)sets **the order of the elements is not taken into account** (two lists with the same elements in different orders are considered to be the same combination). Also, as always for (sub)sets, no elements can be repeated more than once in a combination; this is often referred to as a “**combination (collection) without repetition**”

Combinations

- The number of different (not ordered) combinations of r elements selected from n is the number of all possible permutations (arrangements) of r elements selected from n $P(n, r) = A_r^n$ divided by the number $r!$ of all possible permutations of r elements :

$$C(n, r) = C_r^n = \frac{P(n, r)}{r!} = \frac{A_r^n}{r!} = \frac{n!}{r!(n-r)!}$$

The number of subsets of a set

- **Theorem.** Let S be a set containing n elements, where n is a nonnegative integer. If r is an integer such that $0 \leq r \leq n$, then the number of subsets of S containing exactly r elements is

$$C(n, r) = C_r^n = \frac{P(n, r)}{r!} = \frac{A_r^n}{r!} = \frac{n!}{r!(n-r)!}$$

Discrete Probability

- When **probability** is applied to something, we usually mean an **experiment with certain outcomes**.
- An **outcome** is any one of the possibilities that may be expected from the experiment.
- The totality of all these outcomes forms a universal set which is called the **sample space**.

Sample Space

- For example, if we checked occasionally the number of people in this classroom on Monday from 12:00pm to 1-15pm, we should consider this an experiment having 32 possible outcomes $\{0,1,2,\dots,31\}$ that form a universal set.
- 0 – nobody is in the classroom, ... 31 – all students taking the Discrete Structures Class and the instructor are in the classroom

Sample Space

- A sample space containing at most a denumerable number of elements is called **discrete**
- A sample space containing a nondenumerable number of elements is called **continuous**

Sample Space

- A subset of a sample space containing any number of elements (outcomes) is called an **event**
- **Null event** is an empty subset. It represents an event that is **impossible** to occur
- An event containing all sample space elements is an event that is certain to occur

Sample Space

We toss a single die, what are the possible outcomes, which form the sample space?

$\{1,2,3,4,5,6\}$

We toss a pair of dice, what is the sample space?



Depends on what we're going to ask.

Often convenient to choose a sample space of equally likely events.

$\{(1,1),(1,2),(1,3),\dots,(6,6)\}$

Sample Space

- The following sets are subsets of the sample space $\{1, 2, 3, 4, 5, 6\}$ in the die-tossing experiments and therefore they are the events:
- $A = \{1, 2, 4, 6\}$
- $B = \{n: n \text{ is an integer and } 4 < n \leq 6 \}$
- $C = \{n: n \text{ is an even positive integer less than } 7\}$

The Probability

- The classical definition given by Laplace says that the probability is the ratio of the number of favorable events to the total number of possible events.
- All events in this definition are considered to be equally likely: e.g., throwing of a true die by an honest person under prescribed circumstances...
- ...but not checking for the number of people in the classroom.

The Probability

- According to the [Laplace definition](#), for any event E in a finite sample space S (recall that if E is an event then $E \subset S$) consisting of equally likely outcomes, the probability of E , which is denoted $P(E)$ is

$$P(E) = \frac{|E|}{|S|}$$

The Probability

- The following properties are important:

$$0 \leq |E| \leq M \leq N; 0 \leq |S| \leq N$$

$$0 \leq \frac{|E|}{|S|} \leq 1 \Rightarrow 0 \leq P(E) \leq 1$$

The Probability

- The following properties are important:

$$\text{If } P(E) = p \Rightarrow P(\bar{E}) = 1 - p$$

$$P(E \cup \bar{E}) = 1$$


CMPT-335 Discrete Structures

GRAPHS

Intuitive Concept of Graph

- A **graph** is a **discrete structure** consisting of a set of objects where some pairs of the objects are connected by links
- The interconnected objects are represented by mathematical abstractions called *vertices (nodes)*, and the links that connect some pairs of vertices are called *edges*

Intuitive Notion of Graph

- A **graph** is a bunch of vertices (or nodes) represented by circles • which are connected by edges, represented by line segments 
- In other words, a graph can be considered as a relation on its vertices set

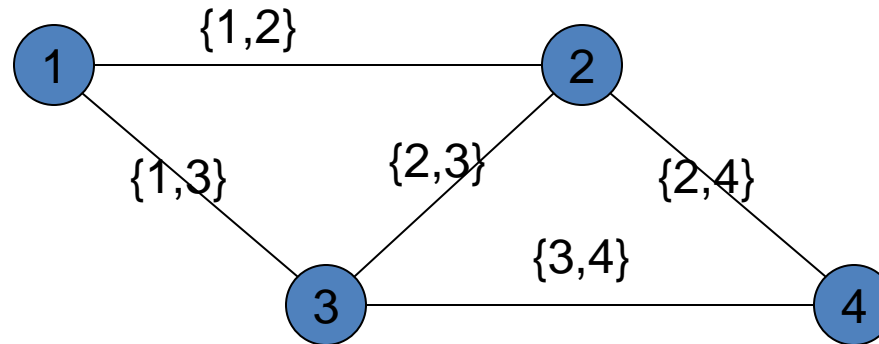
Applications of Graphs

- GPS navigation
- Interconnection of computers and mobile devices in local and global networks
- Electrical and Computer Engineering (to determine whether a circuit can be implemented on a planar circuit board)
- Planning of complex tasks and job assignments
- Coloring of maps
- Etc., etc., etc... many other applications

Definition of Graph

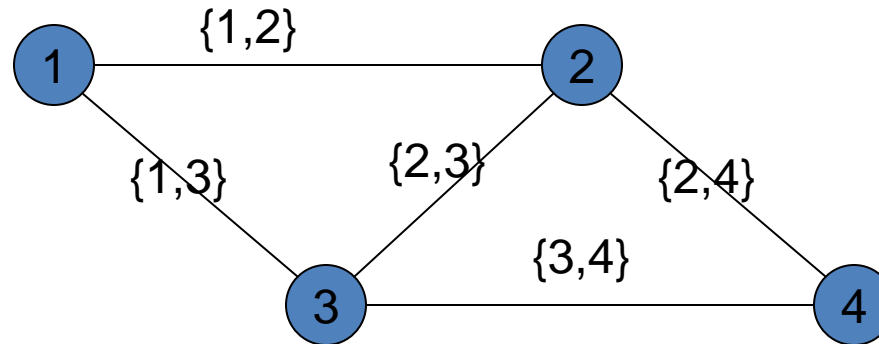
- A graph (an undirected or simple graph) $G = (V, E)$ is a nonempty finite set V (a set of vertices or nodes) together with a set E of edges, where each edge is a subset of V with cardinality 2 (an unordered pair).
- A simple graph is bidirectional (undirected) and has no loops (no “self-communication”).

Example



- $V=\{1,2,3,4\}$
- $E=\{ \{1,2\},\{1,3\},\{2,3\},\{2,4\},\{3,4\} \}$

Example



- Each edge can be viewed as the set of its two endpoints

Edges

- For a set V with n elements, how many possible edges are there?
- This is the number of pairs in V - the number of 2-element subsets of V :

$$C(n, 2) = \frac{n!}{2!(n-2)!} = n(n-1)/2$$

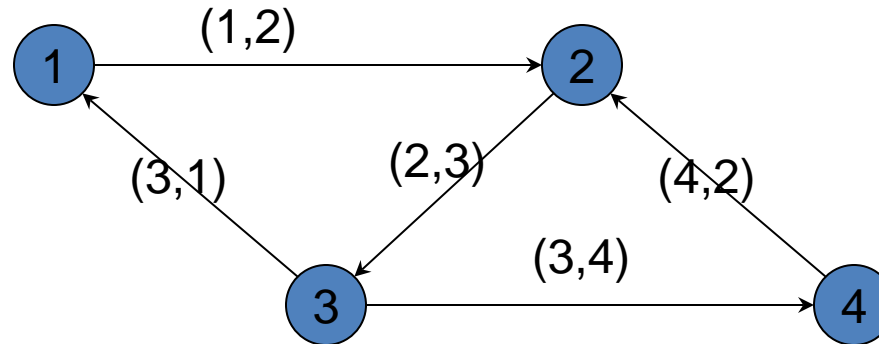
The number of graphs

- How many possible graphs are there for the same set of vertices V ?
- The number of subsets in the set of possible edges. There are $n \cdot (n - 1) / 2$ possible edges, therefore the number of graphs on V is $2^{n(n-1)/2}$

Directed Graph

- A **directed graph** $G = (V, E)$ is a nonempty finite set V (a set of **vertices** or **nodes**) together with a set E of **directed edges**, where each edge is an ordered subset of V with cardinality 2 (an ordered pair)
- The directed edge associated with the ordered pair (u, v) is said to **start** at u and **end** at v

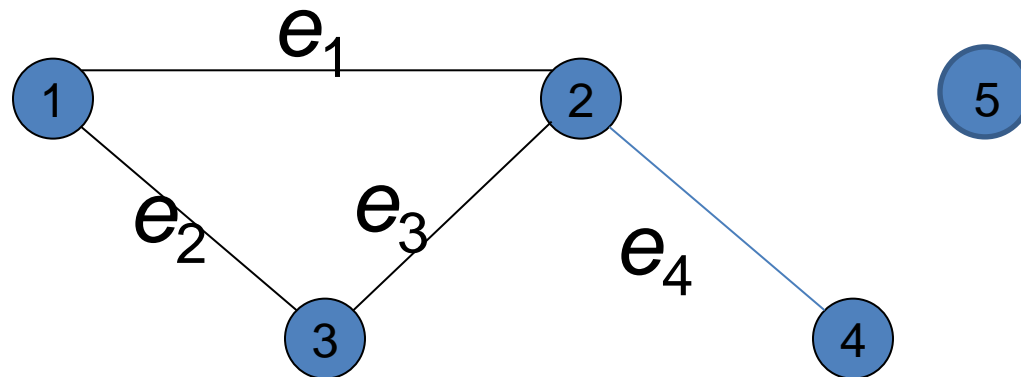
Directed Graph: Example



- $V=\{1,2,3,4\}$
- $E=\{ (1,2),(3,1),(2,3),(4,2),(3,4) \}$

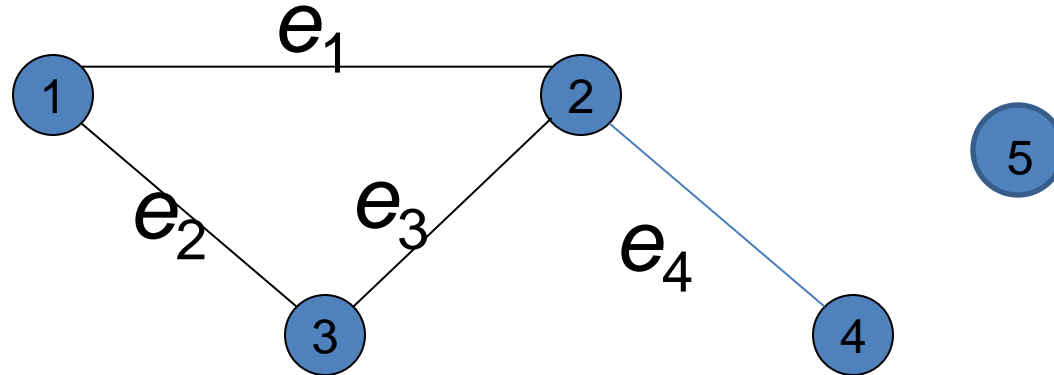
Adjacent Vertices

- Vertices are **adjacent** (or **neighbors**) if they are the endpoints of the same edge. This edge **joins** the adjacent vertices.



Q: Which vertices are adjacent to 1, 2, 3, 4, and 5?

Adjacent Vertices



1 is adjacent to 2 and 3

2 is adjacent to 1, 3, and 4

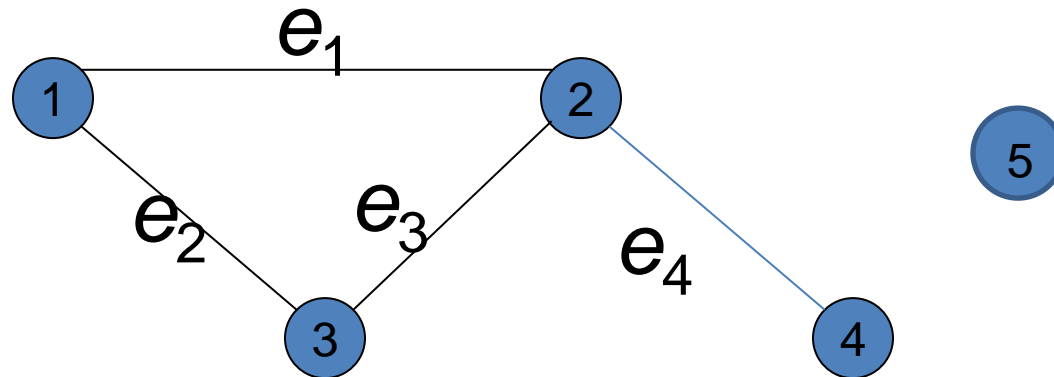
3 is adjacent to 1 and 2

4 is adjacent to 2

5 is not adjacent to any vertex

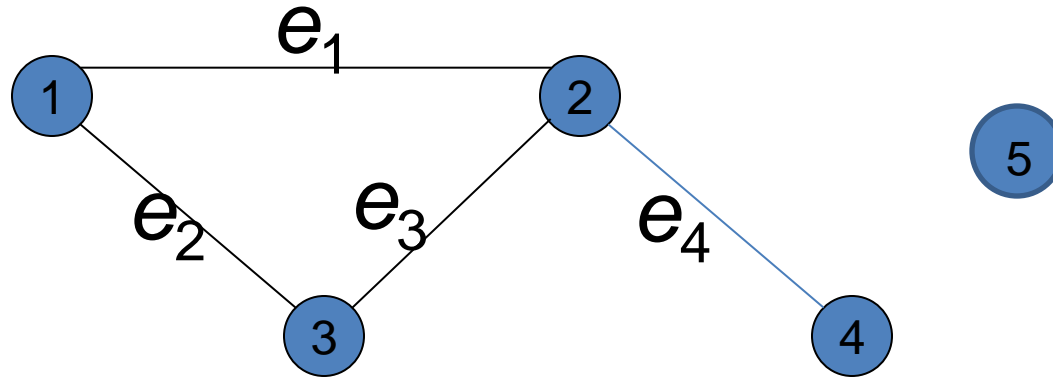
Incident Vertices and Edges

- A vertex is **incident** with an edge (and the edge is **incident** with the vertex) if it is the endpoint of the edge.



- Which edges are incident to 1, 2, 3, 4, and 5?

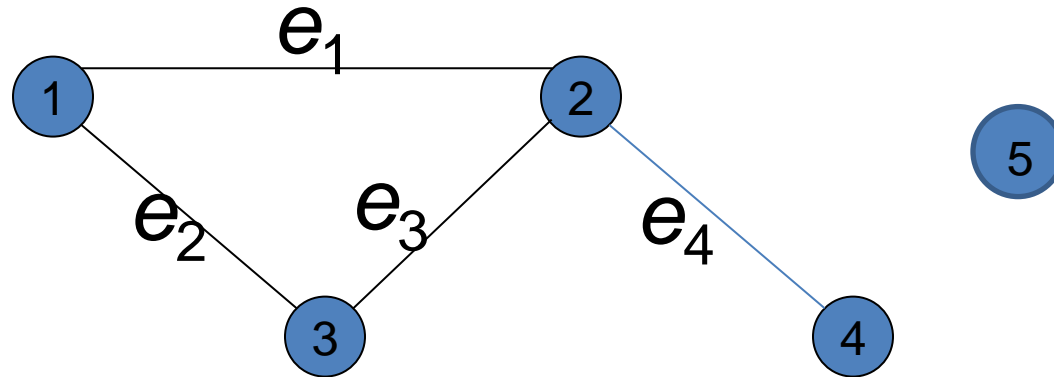
Incident Vertices and Edges



1 is incident with e_1, e_2
 e_1, e_2 are incident with 1
 e_1, e_3, e_4 are incident with 2
2 is incident with e_1, e_3, e_4
3 is incident with e_2, e_3
4 is incident with e_4
5 is not incident with any edge

Degree of a Vertex

- The number of edges incident with a vertex is called the **degree** of this vertex: $\deg(A)$ is the degree of A .



- $\deg(1)=2$; $\deg(2)=3$; $\deg(3)=2$; $\deg(4)=1$; $\deg(5)=0$
- A vertex of degree zero is called **isolated**
- The Handshaking Theorem:** In a graph, the sum of degrees of the vertices equals twice the number of edges

$$\sum_{v \in V} \deg(v) = 2e$$

Degree of a Vertex in a Directed Graph

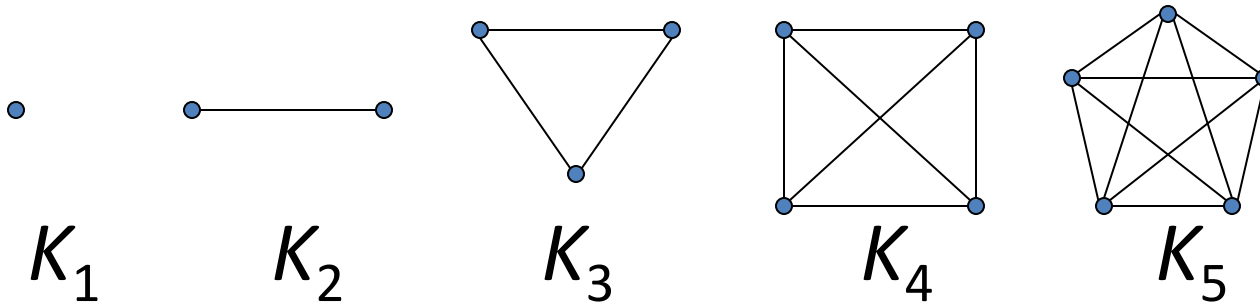
- In a directed graph, the **in-degree** of a vertex A $\deg^-(A)$ is the number of edges with A as their terminal vertex. The **out-degree** of a vertex A $\deg^+(A)$ is the number of edges with A as their initial vertex
- **Theorem:** Let $G=(V, E)$ is a directed graph.

Then

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = |E|$$

Complete Graph

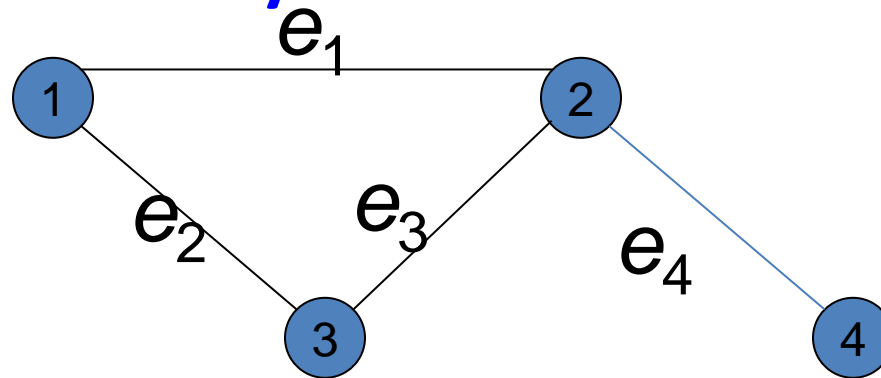
A simple graph is **complete** if every pair of distinct vertices share an edge. The notation K_n denotes the complete graph on n vertices.



Adjacency Matrix

- For the graph $G = (V, E)$ define a binary matrix A_G by:
- Rows, Columns –one for each vertex in V
- Value at i^{th} row and j^{th} column is
 - **1** if i^{th} vertex connects to j^{th} vertex ($i \rightarrow j$)
 - **0** otherwise

Adjacency Matrix - Example

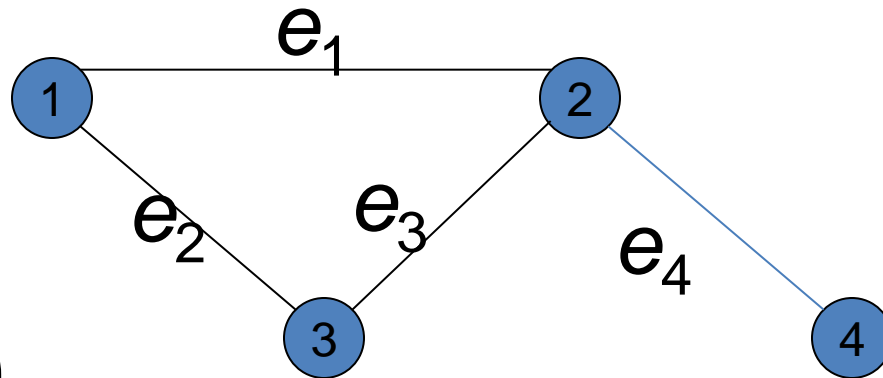


$$A_G = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Important property: for simple graphs A_G is *symmetric*, while for directed graphs it is not

Adjacency Matrix

- **Theorem.** The sum of the entries in row i of the adjacency matrix of a simple graph is the degree of the i^{th} vertex.

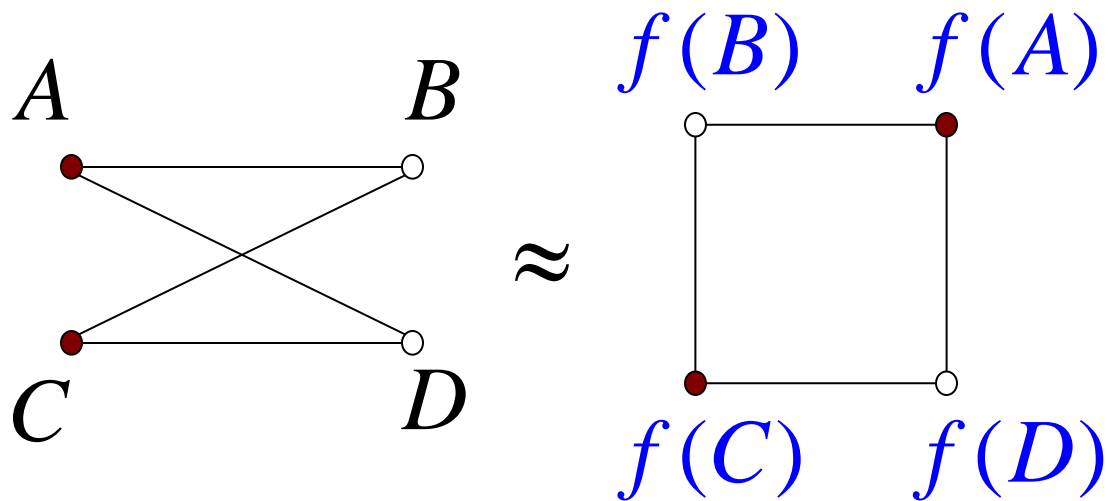


$$A_G = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \rightarrow \deg(1) = 2; \deg(2) = 3; \deg(3) = 2; \deg(4) = 1$$

Graph Isomorphism

- A graph G_1 is **isomorphic** to a graph G_2 , when there is a **one-to-one correspondence** f between the vertices of G_1 and G_2 such that vertices A and B are adjacent in G_1 if and only if the vertices $f(A)$ and $f(B)$ are adjacent in G_2 .
- The function f is called an **isomorphism** of G_1 with G_2 .

Graph Isomorphism



Graph Isomorphism Invariant

- A property is said to be a **graph isomorphism invariant** if, whenever G_1 and G_2 are isomorphic graphs and G_1 has this property, then so does G_2 . The properties are:
 - has n vertices
 - has e edges
 - has a vertex of degree k

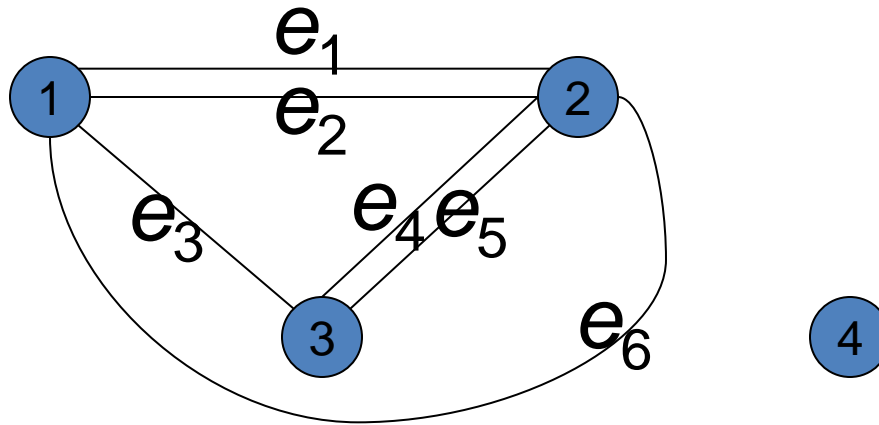
Graph Isomorphism and Degrees of Vertices

- **Theorem.** Let f be an isomorphism of graphs G_1 and G_2 . For any vertex A in G_1 , the degrees of A and $f(A)$ are equal.
- To prove that two graphs G_1 and G_2 are isomorphic, it is necessary to show that there is a one-to-one correspondence (bijection) between their vertices (to build such a correspondence) and to show that vertices A and B are adjacent in G_1 if and only if the vertices $f(A)$ and $f(B)$ are adjacent in G_2 .

Multigraphs

- If we need to reach some place B from the place A , there may be several routes to choose from (GPS navigator, etc.)
- If computers are connected via internet, there also may be several routes to choose from for each connection
- Depending on traffic and possibly other factors, one route could be better than another. Makes sense to allow multiple edges:

Multigraphs



Edge-labels distinguish between edges sharing same endpoints.

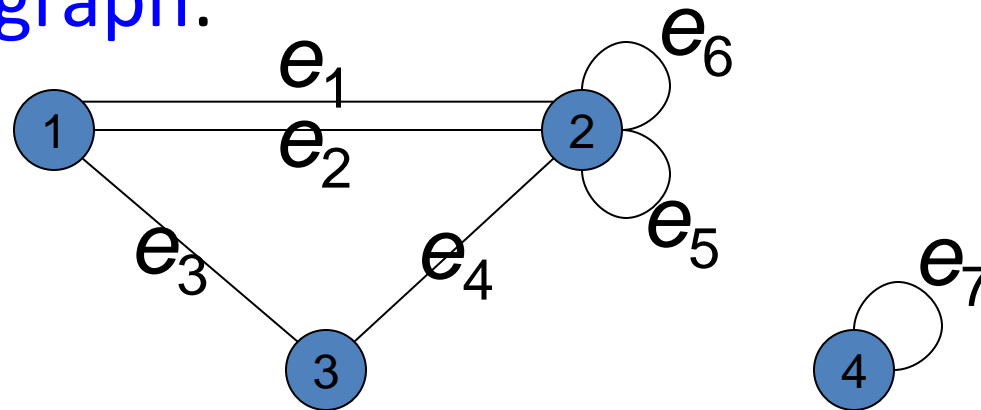
$$e_1 \rightarrow \{1,2\}, e_2 \rightarrow \{1,2\}, e_3 \rightarrow \{1,3\}, e_4 \rightarrow \{2,3\}, \\ e_5 \rightarrow \{2,3\}, e_6 \rightarrow \{1,2\}$$

Multigraphs

- A **multigraph** $G = (V, E, f)$ consists of a non-empty set V of **vertices**, a set E (possibly empty) of **edges** and a function f with domain E and codomain, which is the set of pairs of vertices taken from V

Pseudographs

If self-loops in a multigraph are allowed, we get a **pseudograph**:



Now edges may be associated with a single vertex, when the edge is a loop

$$e_1 \rightarrow \{1,2\}, e_2 \rightarrow \{1,2\}, e_3 \rightarrow \{1,3\},$$

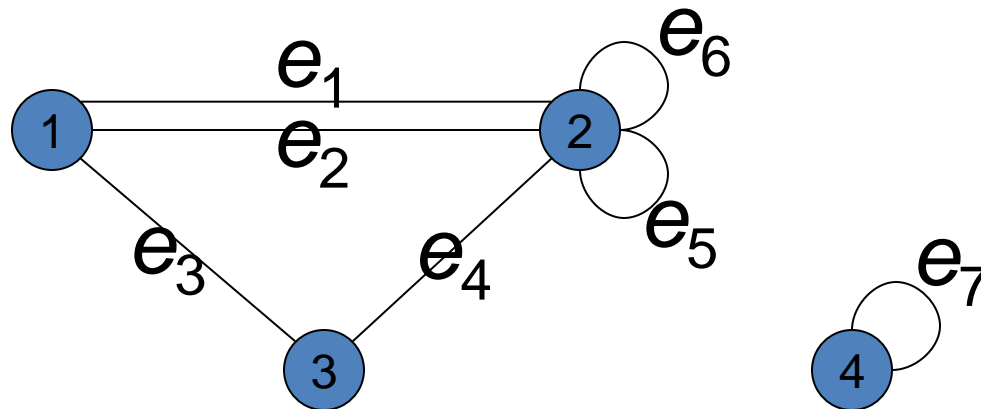
$$e_4 \rightarrow \{2,3\}, e_5 \rightarrow \{2\}, e_6 \rightarrow \{2\}, e_7 \rightarrow \{4\}$$

Pseudographs

- A **pseudograph** $G = (V, E, f)$ consists of a non-empty set V of **vertices**, a set E (possibly empty) of **edges** and a function f whose domain is E and whose codomain is pairs of vertices taken from V

Degree of a Vertex

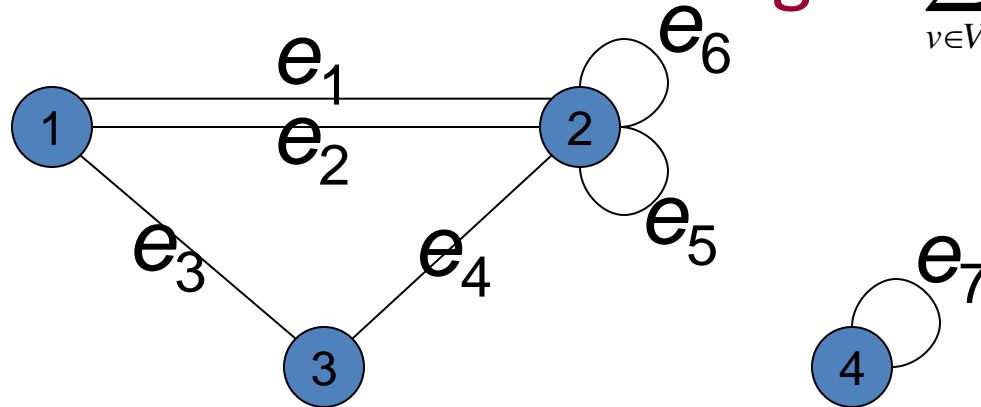
- The number of edges incident with a vertex is called the **degree** of this vertex: $\deg(A)$ is the degree of A.



- A loop on a vertex A is counted twice in $\deg(A)$
- $\deg(1)=3$; $\deg(2)=7$; $\deg(3)=2$; $\deg(4)=2$

Degree of a Vertex

- The Handshaking Theorem: In a multi (pseudo) graph, the sum of degrees of the vertices equals twice the number of edges $\sum_{v \in V} \deg(v) = 2e$



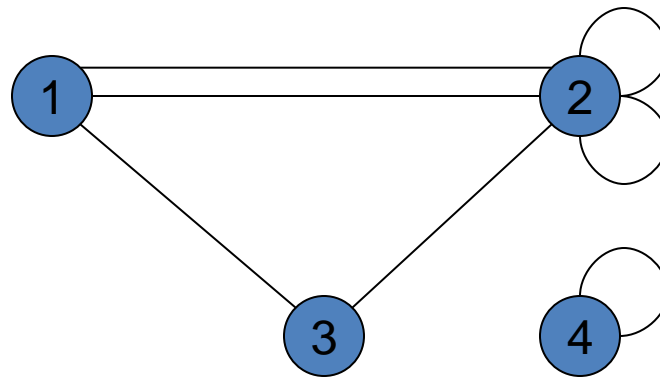
- $\deg(1)=3$; $\deg(2)=7$; $\deg(3)=2$; $\deg(4)=2$
- There are **7** edges and $\sum_{i=1}^4 \deg(i) = 3 + 7 + 2 + 2 = \mathbf{14}$

Multigraphs: Adjacency Matrix

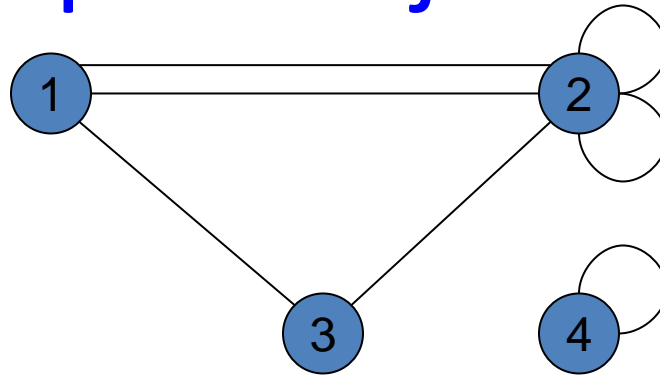
- For an undirected multi (pseudo) graph $G = (V, E, f)$ define the matrix A_G by:
 - Rows, Columns –one for each element of V
 - Value at i^{th} row and j^{th} column is the number of edges incident with vertices i and j .

Multigraphs: Adjacency Matrix

Q: What's the adjacency matrix?



Multigraphs: Adjacency Matrix

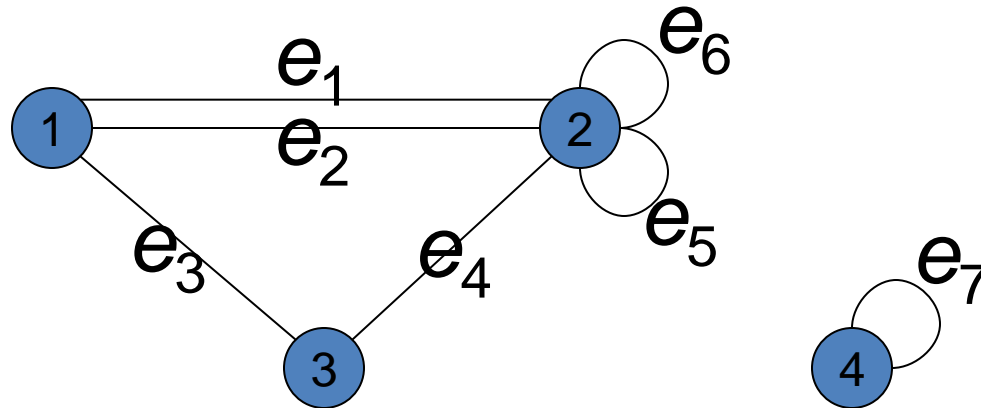


$$A_G = \begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 4 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

Important property: A_G is *symmetric* for undirected graphs, while for directed graphs it is not

Paths

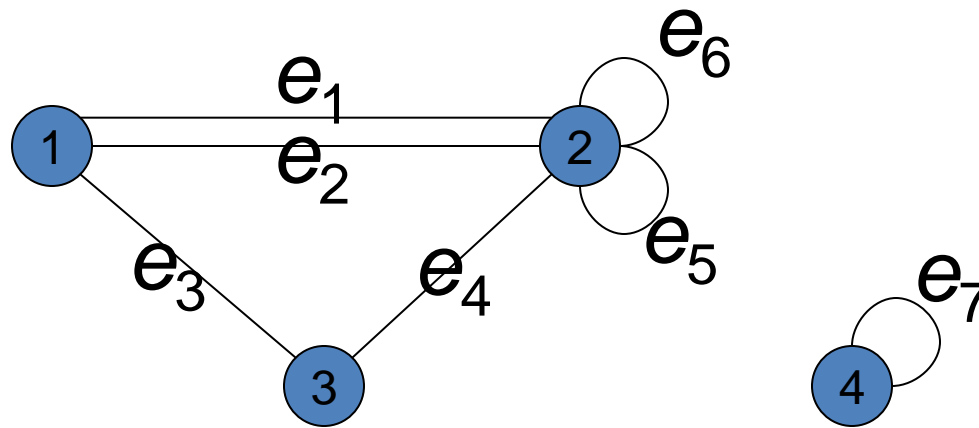
- A **path** in a multi(pseudo) graph is a continuous way of getting from one vertex to another by using a sequence of edges.



- Could get from 1 to 3 circuitously as follows:
- $1-e_1 \rightarrow 2-e_1 \rightarrow 1-e_3 \rightarrow 3-e_4 \rightarrow 2-e_6 \rightarrow 2-e_5 \rightarrow 2-e_4 \rightarrow 3$
- Or simpler: $1-e_1 \rightarrow 2-e_4 \rightarrow 3$

Paths

- If a path contains of n edges, then the length of the path is n .



- $1-e_1 \rightarrow 2-e_1 \rightarrow 1-e_3 \rightarrow 3-e_4 \rightarrow 2-e_6 \rightarrow 2-e_5 \rightarrow 2-e_4 \rightarrow 3 \Rightarrow 7$
- Or simpler: $1-e_1 \rightarrow 2-e_4 \rightarrow 3 \Rightarrow 2$
- 4 is the path to itself with the length 0.
- However, there is another path $4-e_7 \rightarrow 4 \Rightarrow 1$

Paths

- In a path, the vertices need not be distinct, and some of the edges can be the same.
- When there is no chance of confusion, a path can be represented by the vertices V_1, V_2, \dots, V_{n+1} only or by the edges e_1, e_2, \dots, e_n only.

Paths – Another Definition

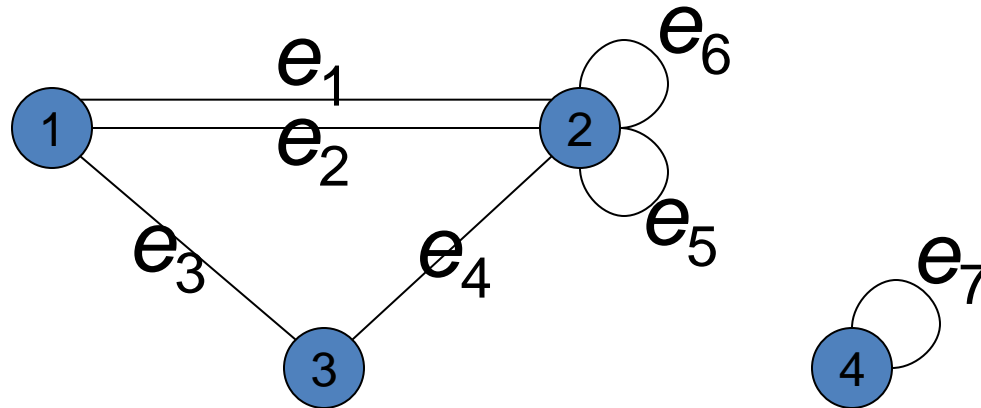
- A **path** of length n in a **multi (pseudo) graph** is a sequence of n edges e_1, e_2, \dots, e_n such that each consecutive pair e_i, e_{i+1} share a common vertex
- In a **simple graph**, one may instead define a path of length n as a sequence of $n+1$ vertices $v_0, v_1, v_2, \dots, v_n$ such that each consecutive pair v_i, v_{i+1} are adjacent
- Paths of length 0 are also allowed according to this definition

Simple Paths and Cycles

- A **simple path** contains no duplicate edges (though duplicate vertices are allowed)
- A **cycle** (or **circuit**) C_n , $n \geq 3$ is a path which starts and ends at the same vertex and where **all the vertices (except the first/last one) and all the edges are distinct**
- Note: Simple paths need not be in simple graphs. E.g., they may contain loops and therefore they may be found in multi (pseudo) graphs

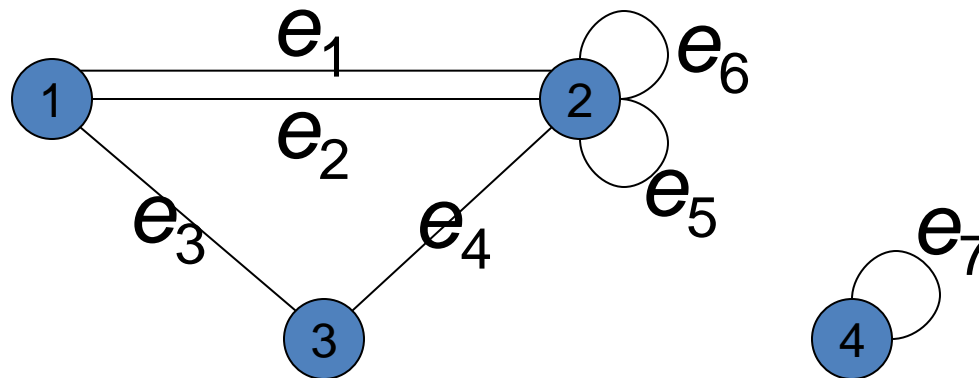
Simple Paths and Circuits

- Find a longest possible simple path in the following graph:



Simple Paths and Circuits

- Solution: The path $e_1, e_5, e_6, e_2, e_3, e_4$ from 1 to 2 is a maximal simple path because
- **simple**: each of its edges appears exactly once
- **maximal**: because it contains every edge except the unreachable edge e_7

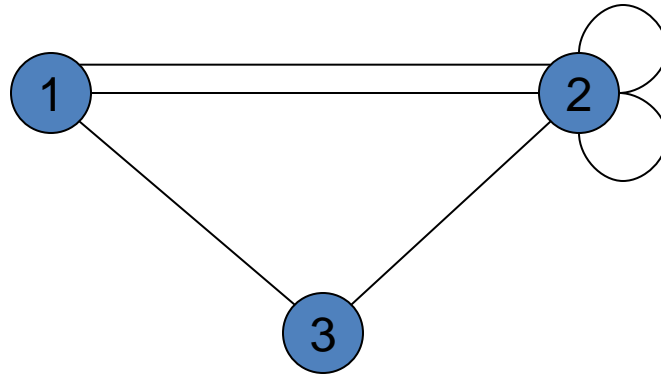
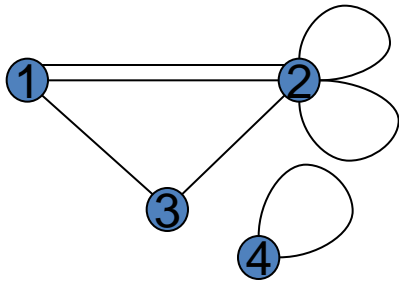


Connectivity

- Let G be a multi (pseudo) graph. Let U and V be vertices. U and V are **connected** *to each other* if there is a path in G which starts at U and ends at V
- Graph G is said to be **connected** if all vertices are connected to each other.
- **Remark** : Any vertex is automatically connected to itself via the empty path
- For example, any two computers in a network can communicate if and only if the graph of this network is connected

Connectivity

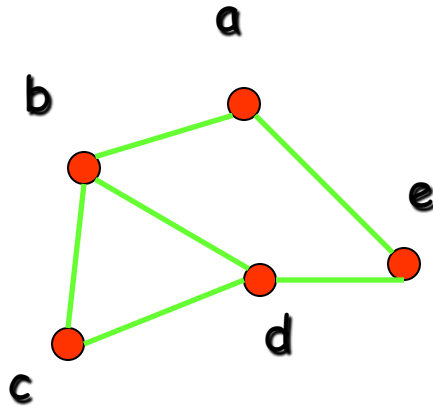
- Which of the following graphs are connected?



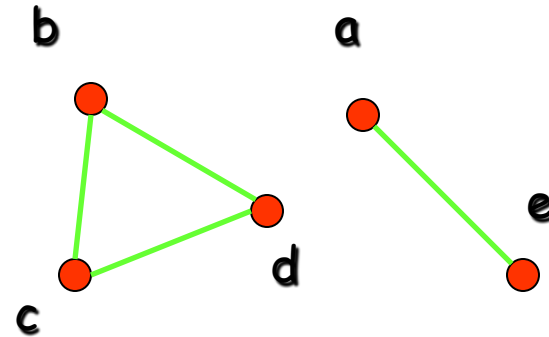
- The first is disconnected (there is no path to (from) 4 from (to) other vertices), the second is connected

Connectivity

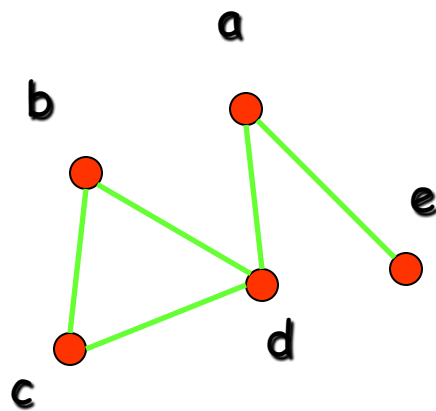
- Example: Are the following graphs connected?



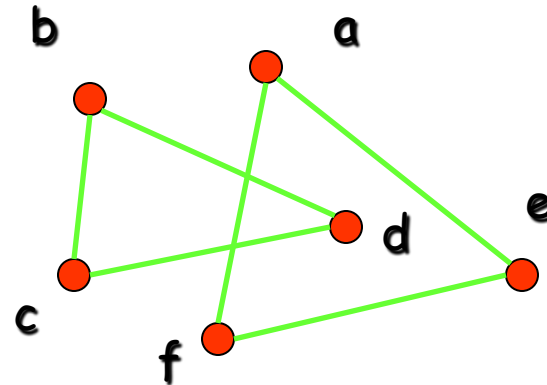
Yes.



No.



Yes.



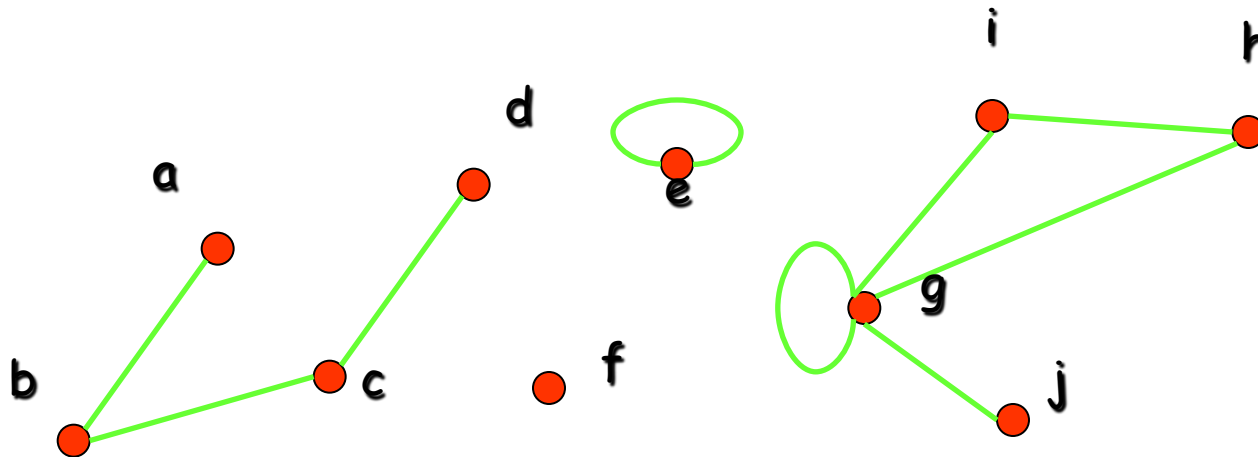
No.

Connectivity

- **Theorem:** There is a simple path between every pair of distinct vertices of a connected undirected graph
- A graph that is not connected is the union of two or more connected subgraphs, each pair of which has no vertex in common. These disjoint connected subgraphs are called the **connected components** of the graph

Connectivity

- Example: What are the connected components in the following graph?



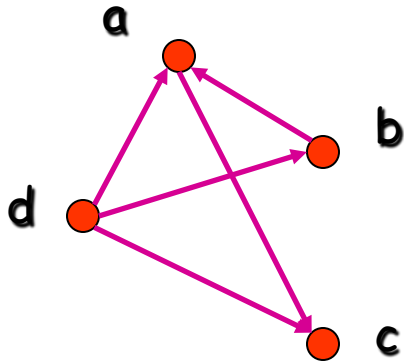
Solution: The connected components are the graphs with vertices $\{a, b, c, d\}$, $\{e\}$, $\{f\}$, $\{g, h, i, j\}$.

Connectivity in Directed Graphs

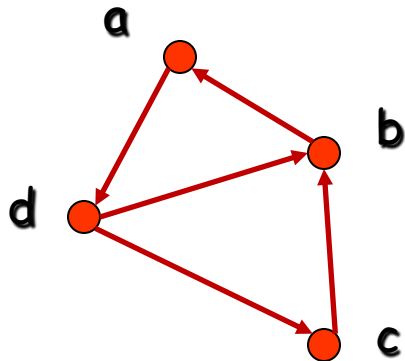
- A directed graph is **strongly connected** if there is a path from a to b and from b to a whenever a and b are vertices in the graph.
- A directed graph is **weakly connected** if there is a path between any two vertices only in the underlying undirected graph (that is only when considering it as an undirected graph) and this underlying undirected graph is connected), but there is a missing path between at least two vertices in one of the directions.

Connectivity in Directed Graphs

- Example: Are the following directed graphs strongly or weakly connected?



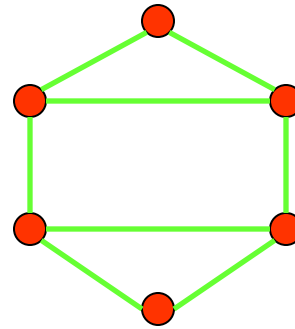
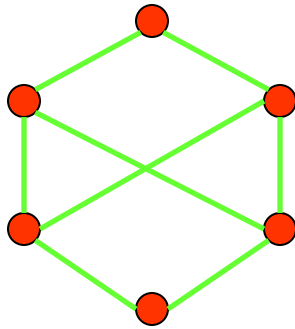
Weakly connected, because, for example, there is no path from b to d.



Strongly connected, because there are paths between all possible pairs of vertices.

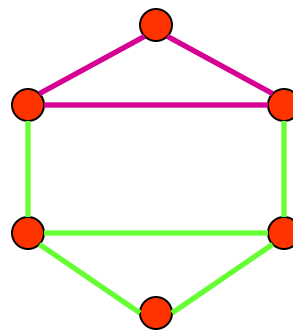
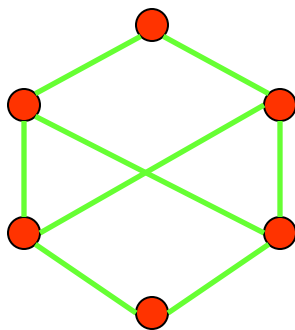
Connectivity and Isomorphism

- The number and size of connected components and circuits are further invariants with respect to isomorphism of simple graphs.
- Example: Are these two graphs isomorphic?



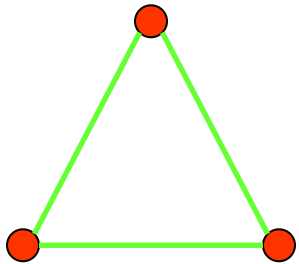
Connectivity and Isomorphism

- The number and size of connected components and circuits are further invariants with respect to isomorphism of simple graphs.
- Example: Are these two graphs isomorphic?

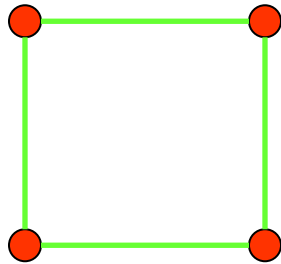


Solution: No, because the right graph contains circuits of length 3, while the left graph does not.

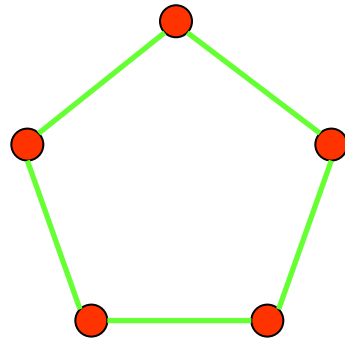
Cycles and Cyclic Graphs



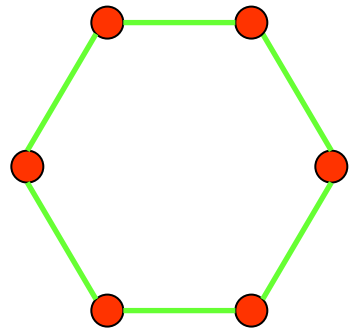
C_3



C_4



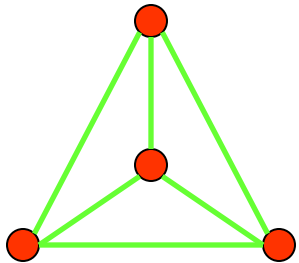
C_5



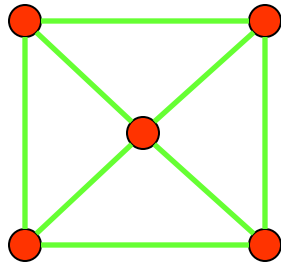
C_6

Wheels

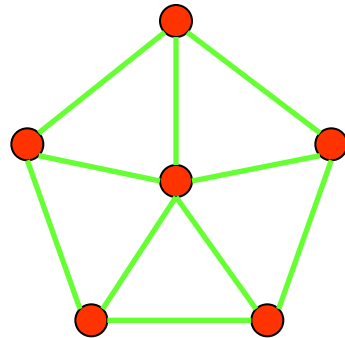
- We obtain the **wheel** W_n when we add an additional vertex to the cycle C_n , for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n by adding new edges



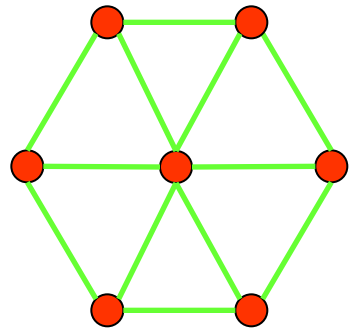
W_3



W_4



W_5



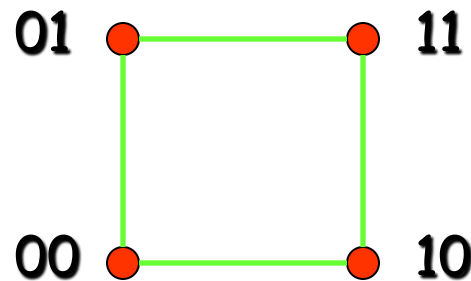
W_6

n -cube

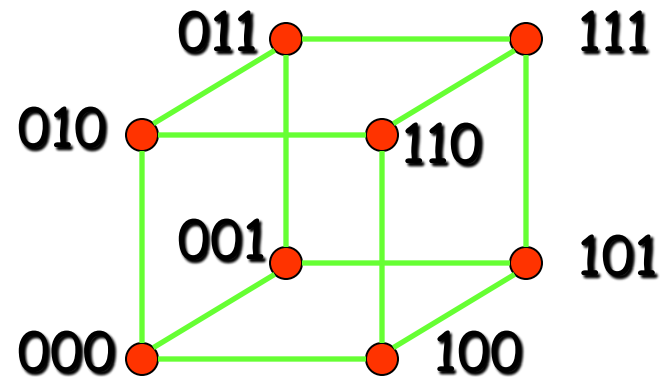
- The n -cube, denoted by Q_n , is the graph that has vertices representing the 2^n bit strings of length n . Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position.



Q_1



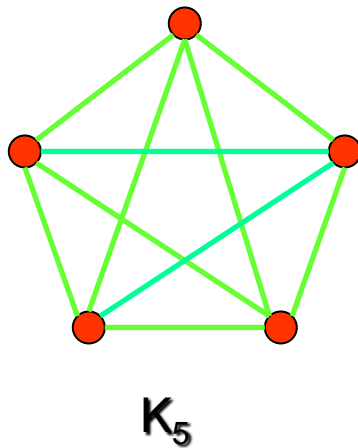
Q_2



Q_3

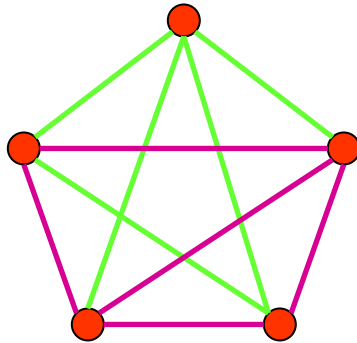
Operations on Graphs

- A **subgraph** of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq E$
- **Remark:** Of course, H is a valid graph

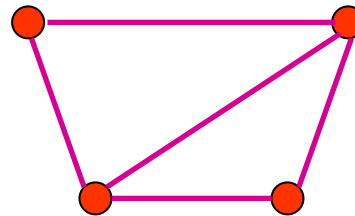


Operations on Graphs

- A **subgraph** of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq E$
- **Remark:** Of course, H is a valid graph



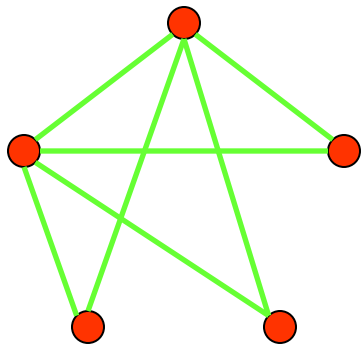
K_5



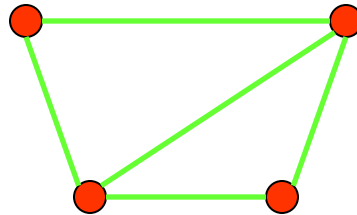
subgraph of K_5

Union of Simple Graphs

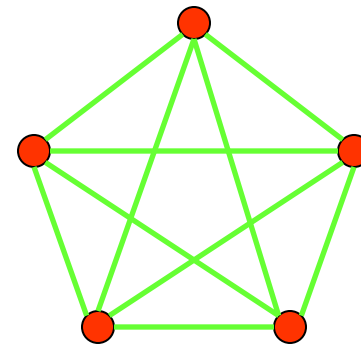
- The **union** of two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$
- The union of G_1 and G_2 is denoted by $G_1 \cup G_2$



G_1



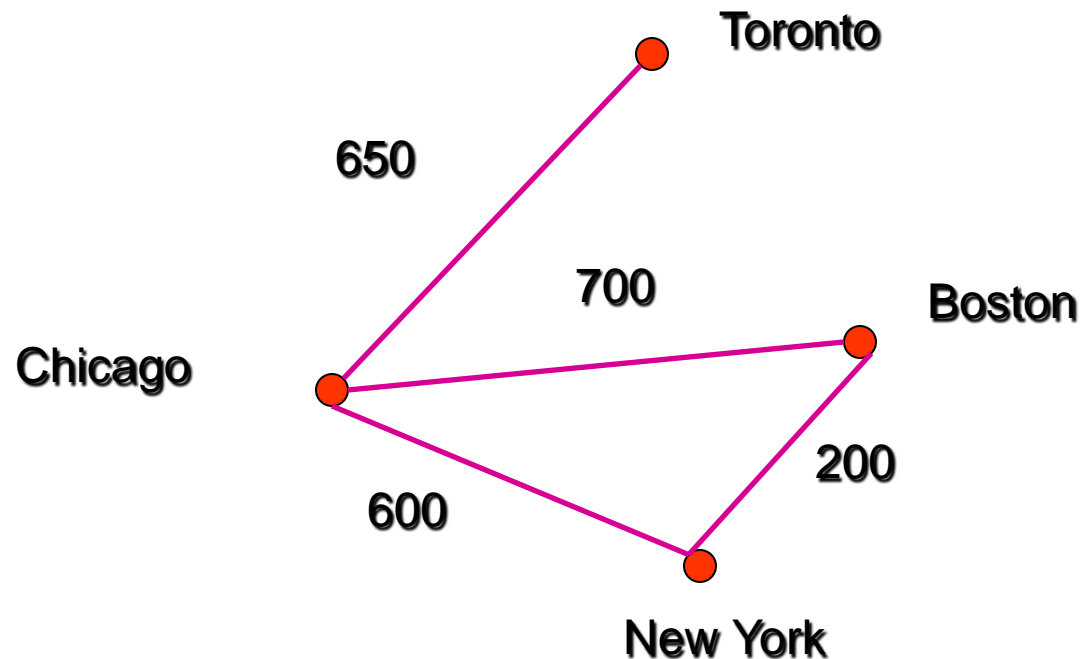
G_2



$G_1 \cup G_2 = K_5$

Shortest Path Problem

- We can assign weights to the edges of graphs, for example to represent the distance between cities or a cost of flight tickets between them



Shortest Path Problem

- Such weighted graphs can also be used to model computer networks with response times or costs as weights
- One of the most interesting questions that we can investigate with such graphs is:
 - What is the **shortest path** between two vertices in the graph, that is, the path with the **minimal sum of weights** along the way?
 - This corresponds to the shortest connection between cities or the fastest connection in a computer network

Dijkstra's Algorithm

- Dijkstra's algorithm is an iterative procedure that finds the shortest path between vertices a and z in a weighted graph
- It proceeds by finding the length of the shortest path from a to successive vertices and adding these vertices to a distinguished set of vertices S .
- The algorithm terminates once it reaches the vertex z .

Dijkstra's Algorithm

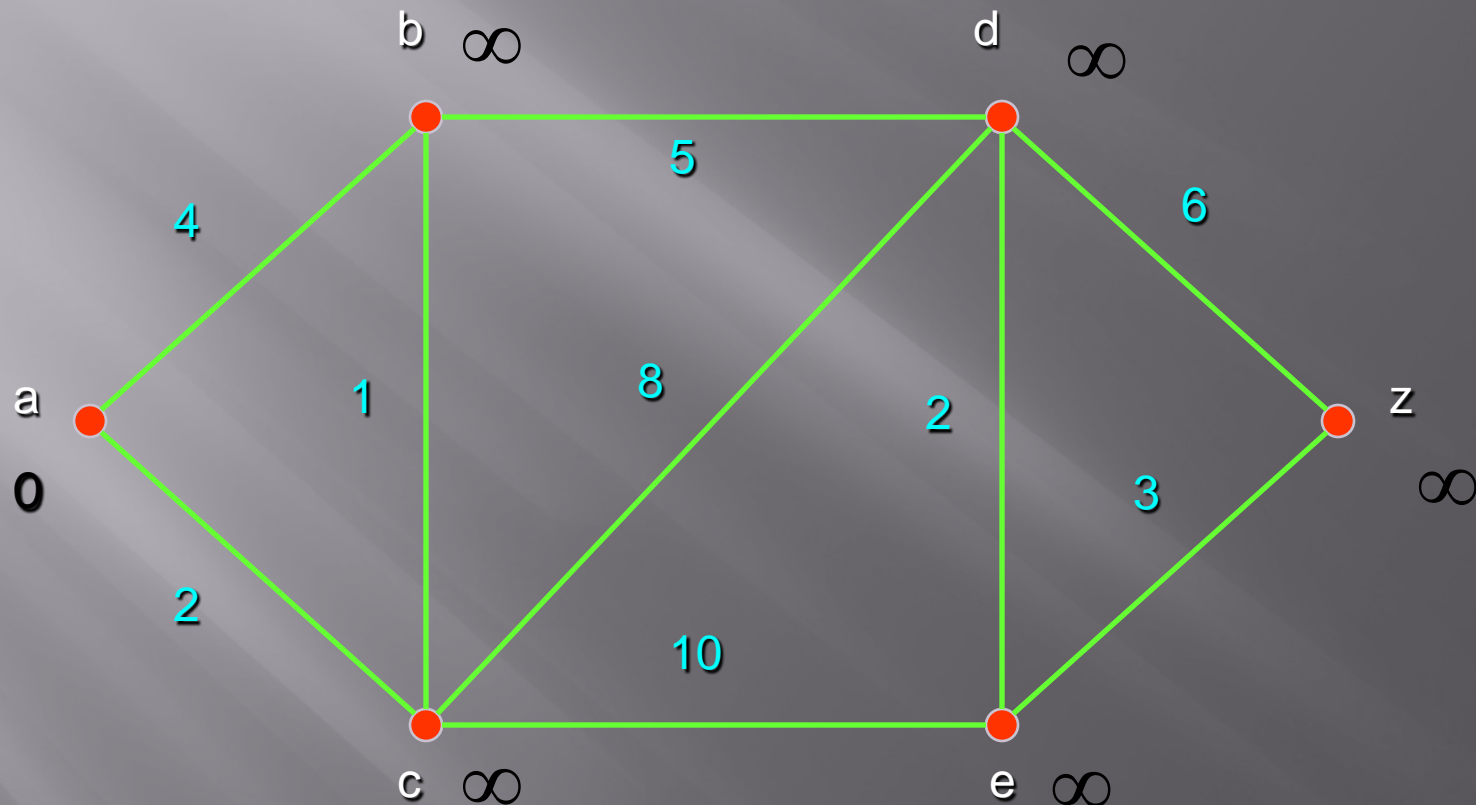
- **procedure** Dijkstra(G : weighted connected simple graph with vertices $a = v_0, v_1, \dots, v_n = z$ and positive weights $w(v_i, v_j)$,
where $w(v_i, v_j) = \infty$ if $\{v_i, v_j\}$ is not an edge in G)
- **for** $i = 1$ **to** n
 - $L(v_i) = \infty$
 - $L(a) = 0$
 - $S = \emptyset$
- {the labels are now initialized so that the label of a is zero and all other labels are ∞ , and the distinguished set of vertices S is empty}

Dijkstra's Algorithm

- **while** $z \notin S$
- **begin**
- u = the vertex not in S with minimal $L(u)$
- $S = S \cup \{u\}$
- **for** all vertices v not in S
- **if** $L(u) + w(u, v) < L(v)$ **then** $L(v) = L(u) + w(u, v)$
- {this adds a vertex to S with minimal label and updates the labels of vertices not in S }
- **end** { $L(z)$ = length of shortest path from a to z }

Dijkstra's Algorithm: Example

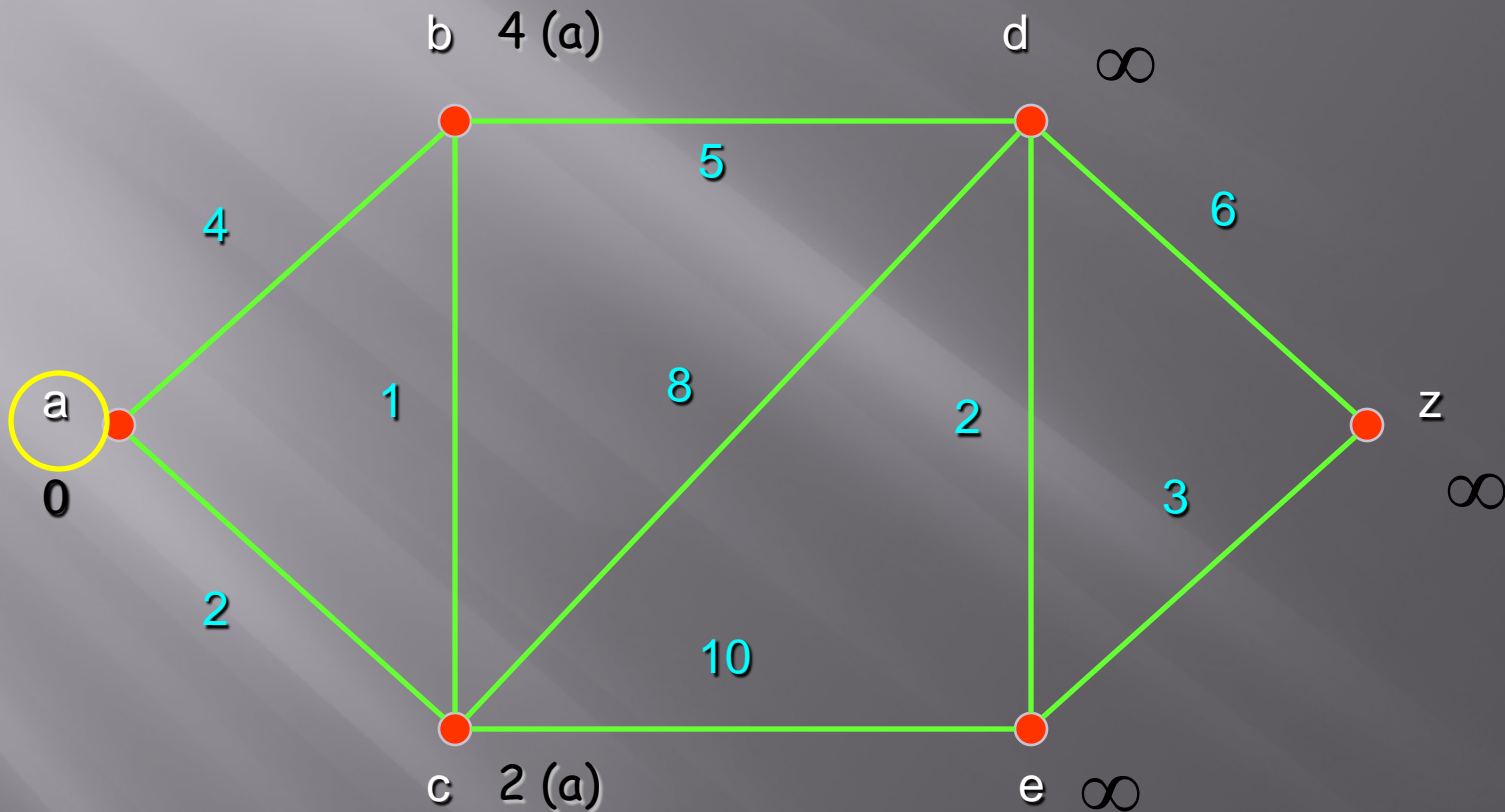
Example:



Step 0

Dijkstra's Algorithm: Example

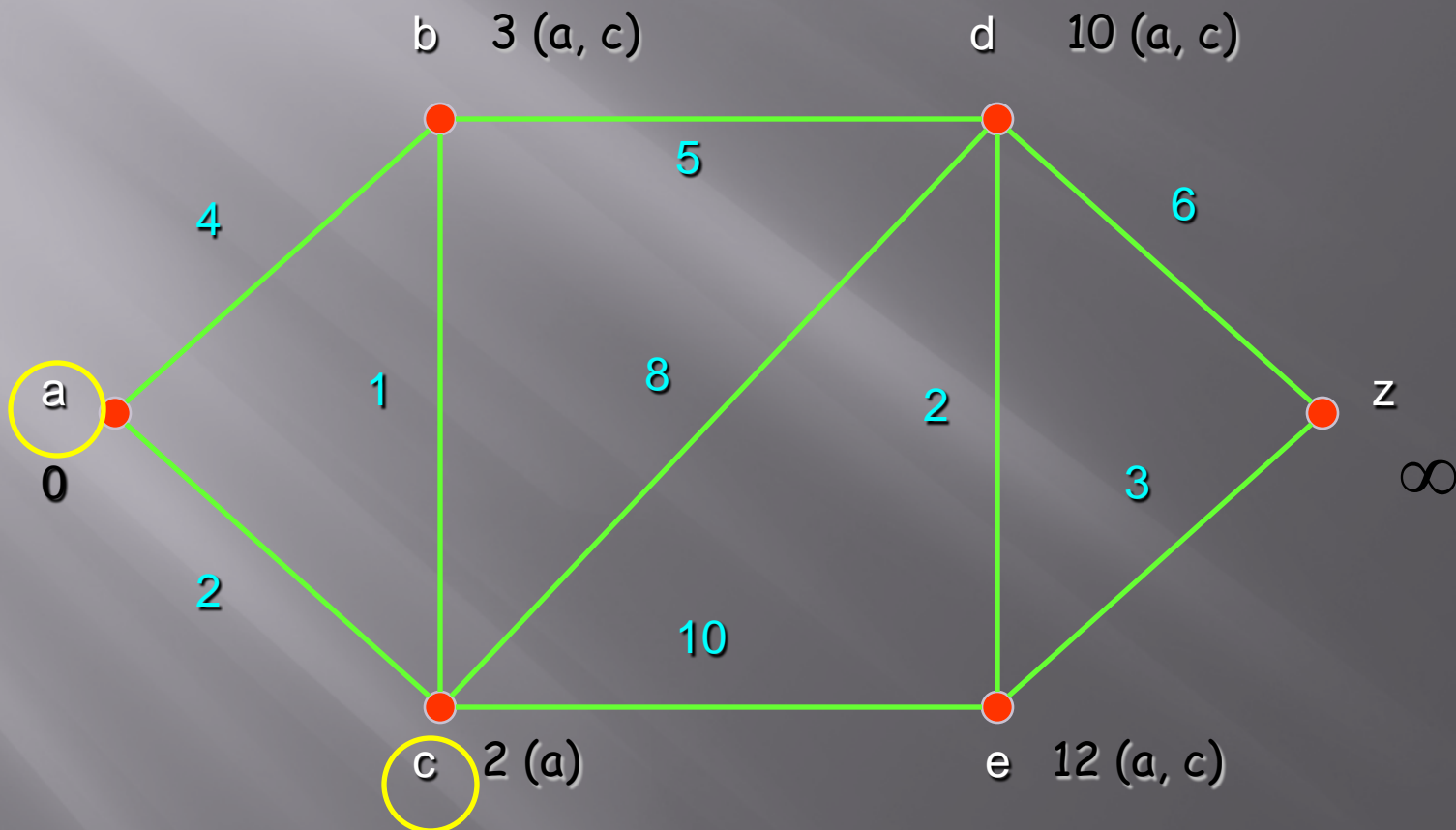
Example:



Step 1

Dijkstra's Algorithm: Example

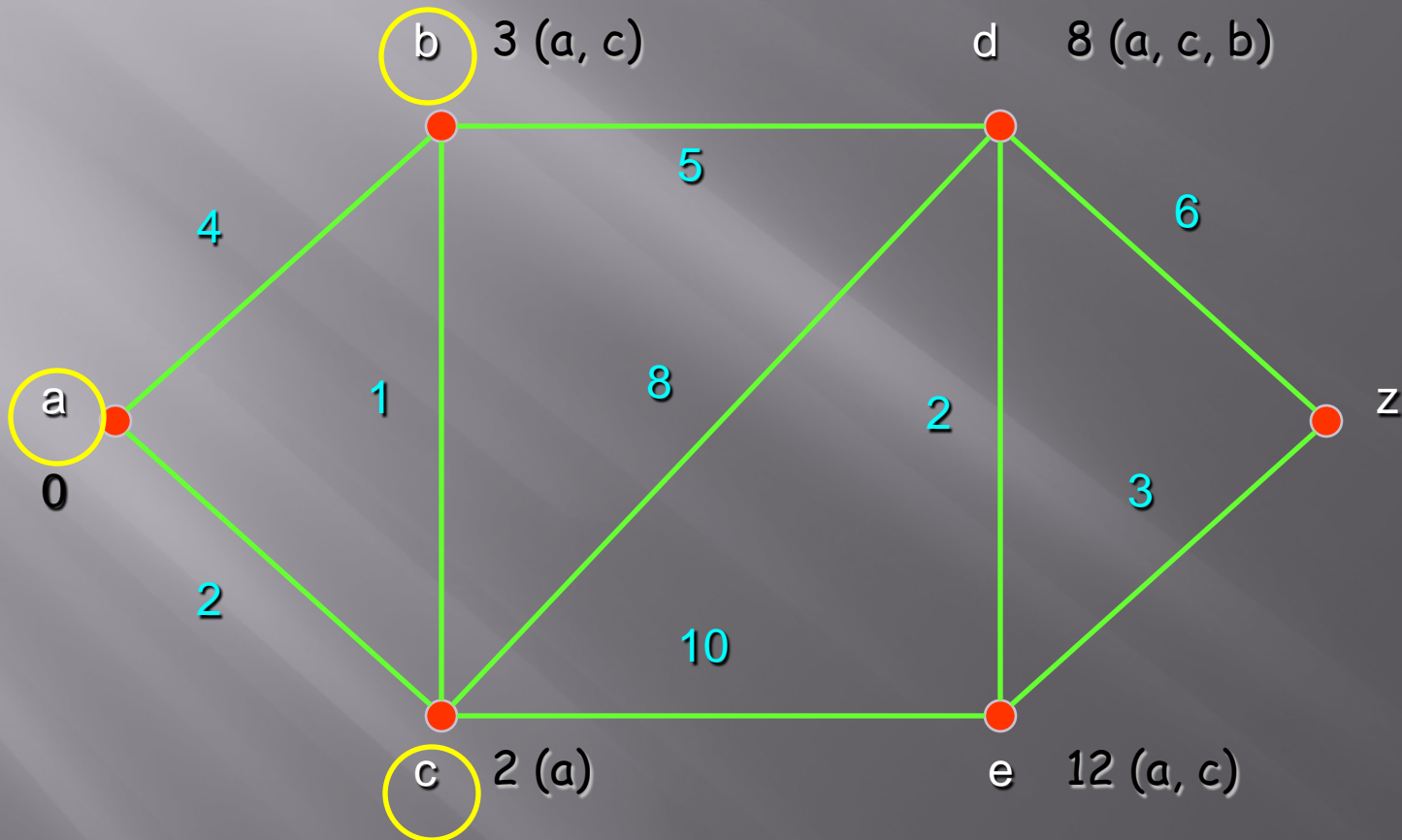
Example:



Step 2

Dijkstra's Algorithm: Example

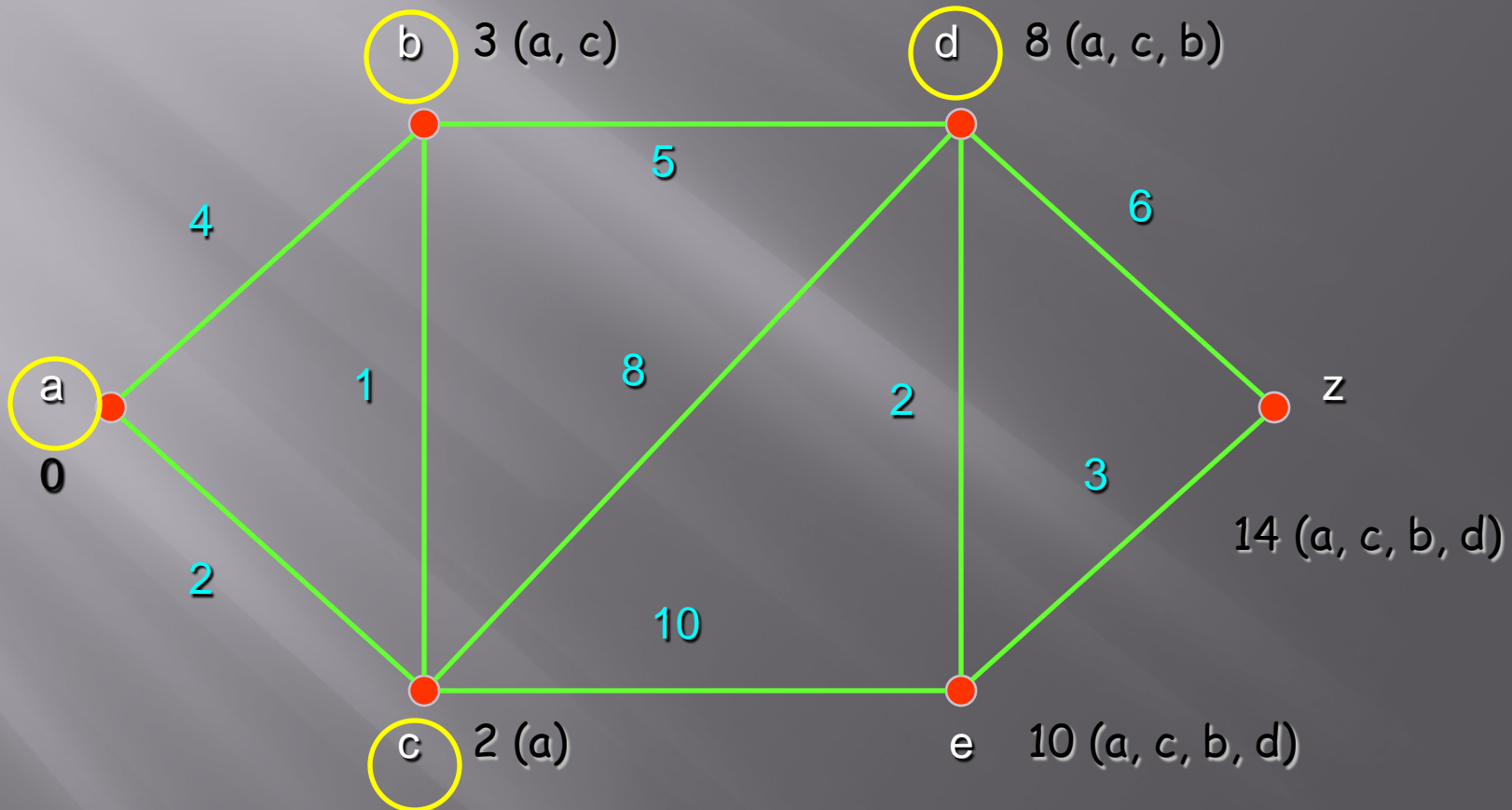
Example:



Step 3

Dijkstra's Algorithm: Example

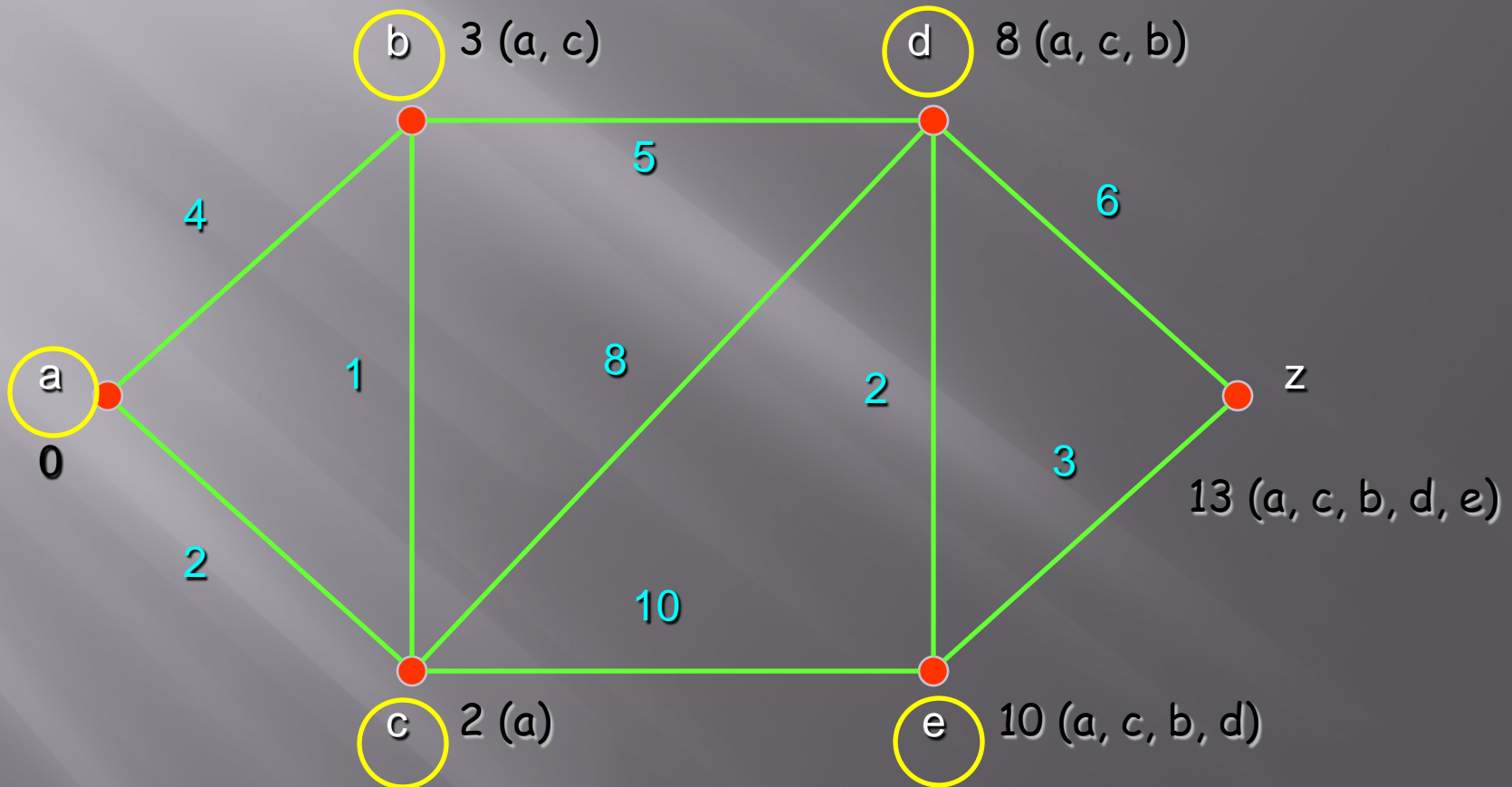
Example:



Step 4

Dijkstra's Algorithm: Example

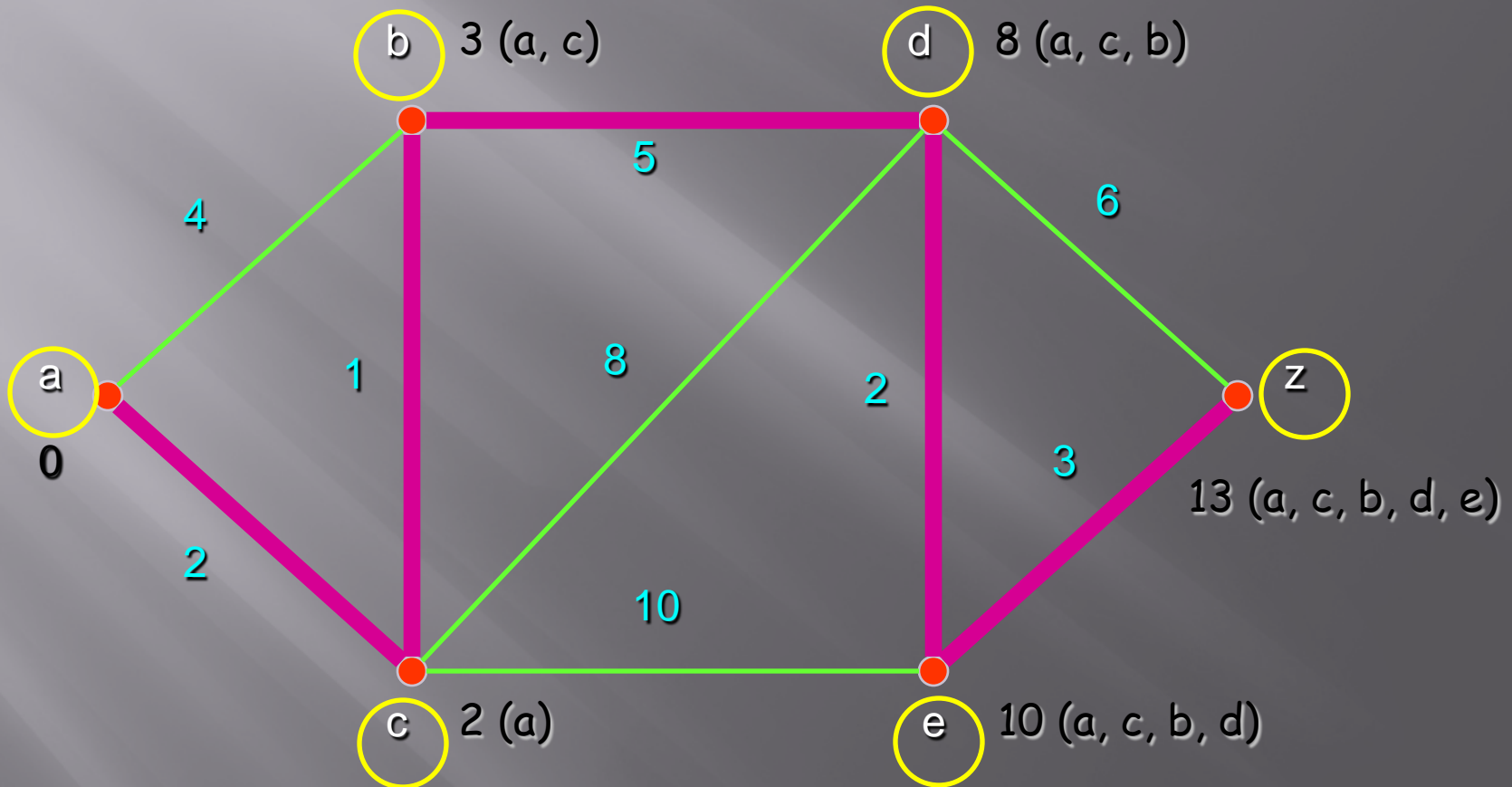
Example:



Step 5

Dijkstra's Algorithm: Example

Example:



Step 6