COT 2000

Foundations of Computing

Spring 2024

Lecture 20 – part 1

Lab 10 (Optional)
Homework 7 – 07/26/24
Exam 4 – 08/02/24

Lecture 20 – part 2

Review

Review

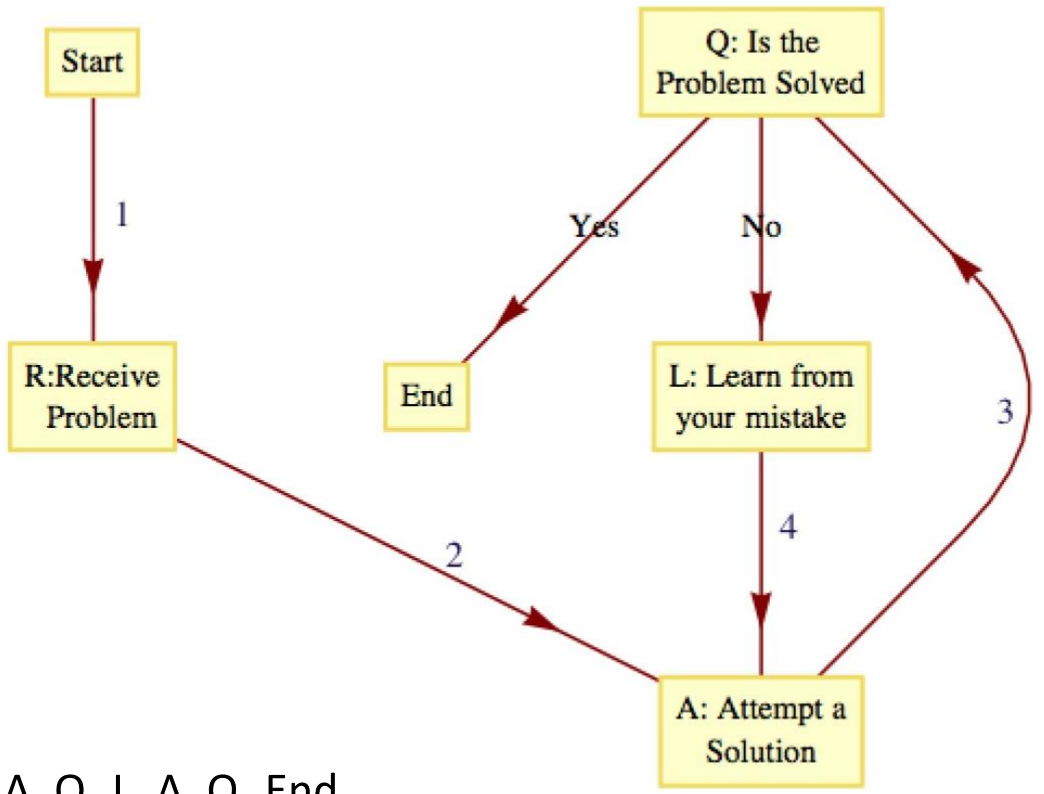
Graph Theory

- Directed, Undirected, Complete Undirected, Multigraph

Notation and Terminology

- Vertex, Edge
- Initial Vertex, Terminal Vertex
- Path, Vertex list
- Path Length
- Subpath, proper, improper
- Circuit, simple circuit
- Subgraphs, Induced subgraph, Spanning subgraph
- Connected Component
- Examples: Flowchart, Tournament Graph
- Graph Isomorphisms

Example:
A Labeled Graph. A flowchart is a common example of a simple graph that requires labels for its vertices and some of its edges.

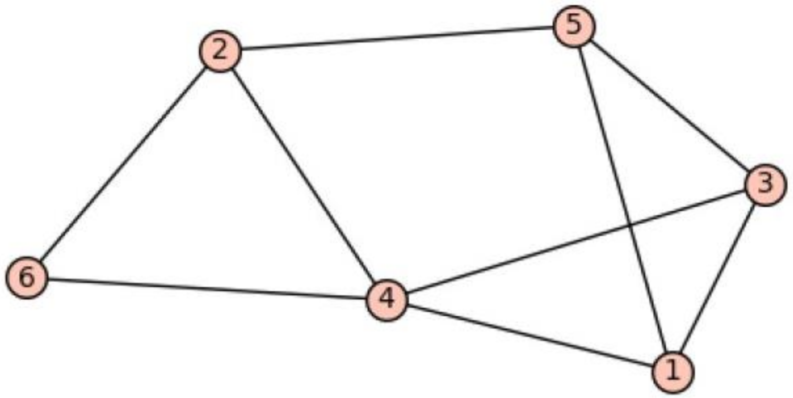


The sequence of vertices from “Start” to “End” is called a **path**.
The “Start” vertex is called the initial vertex of the path, while the “End” is called the final, or terminal, vertex.

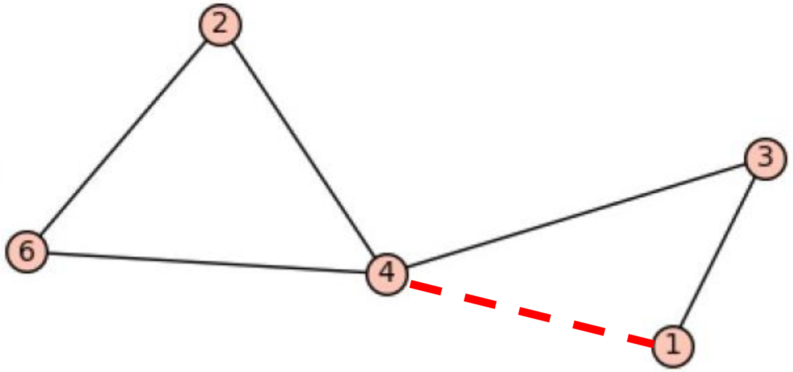
Path:
Start, R, A, Q, L, A, Q, End.
1, 2, 3, No, 4, 3,Yes. (is unambiguous)

Example:

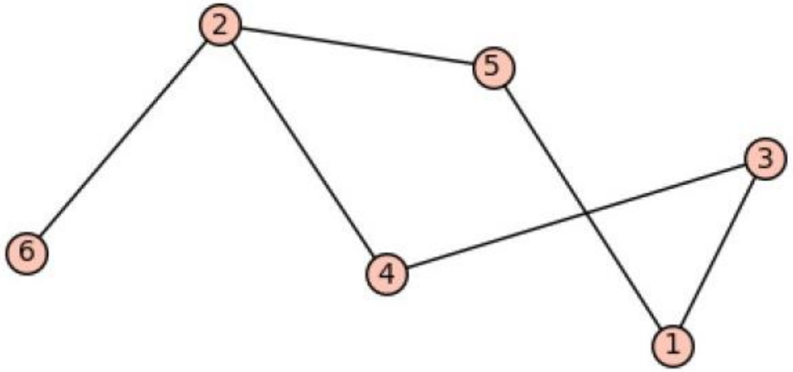
Graph



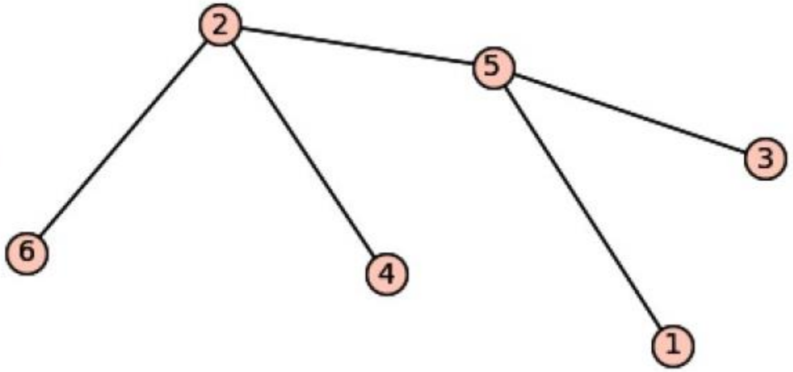
Subgraph



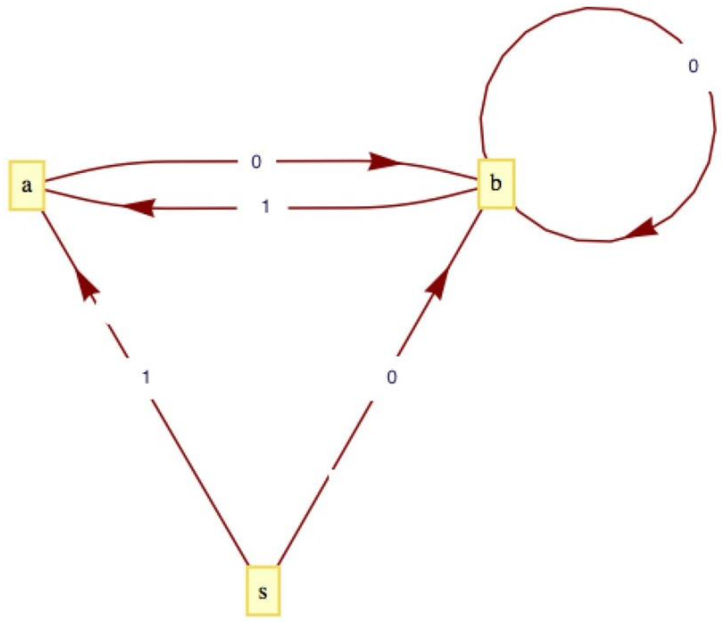
Induced subgraph



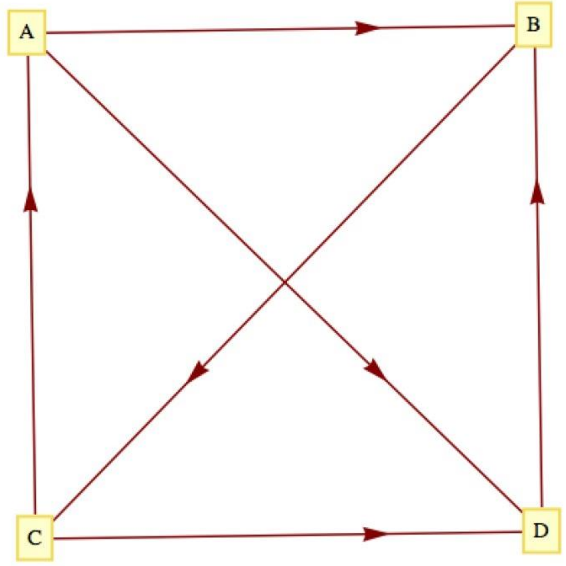
Spanning subgraph

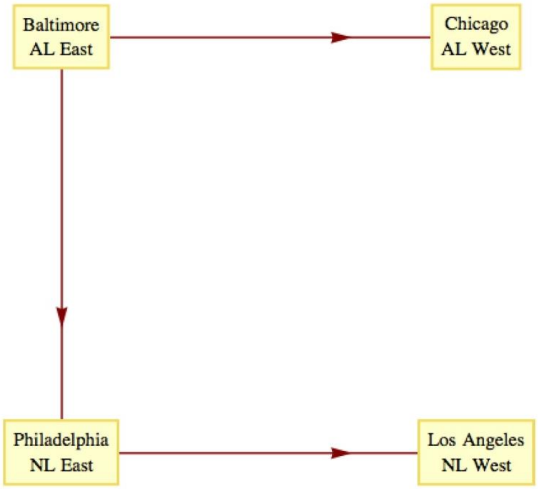


Spanning subgraph
Spanning subtree



vertex list (s, a, b, b, a, b, b, a, b)
output of 10010010.

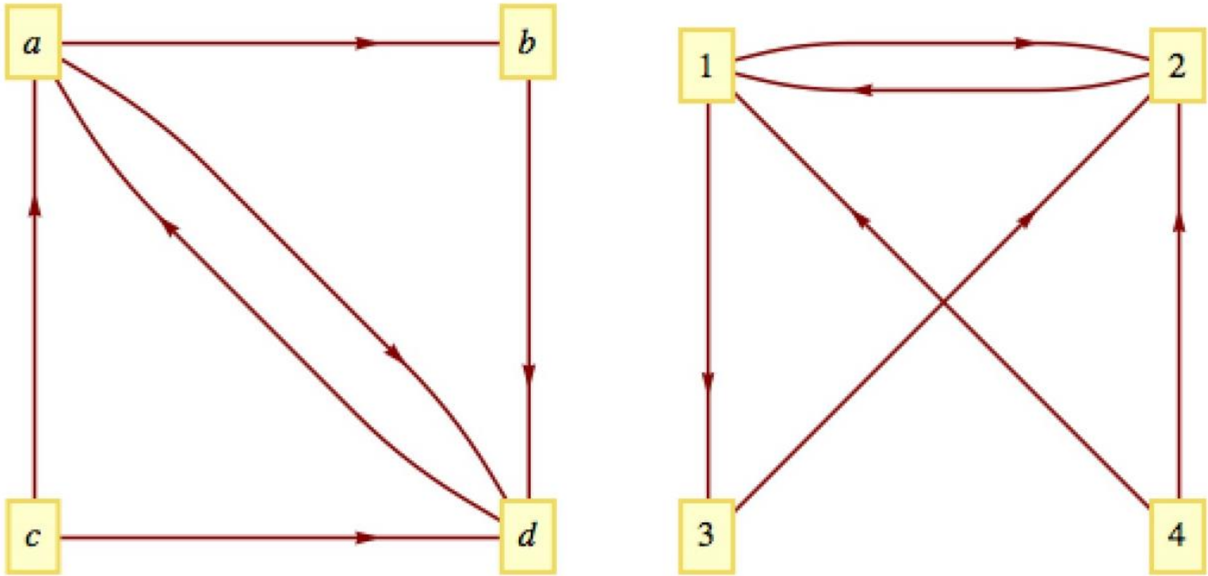




There are many types of tournaments and they all can be modeled by different types of graphs.

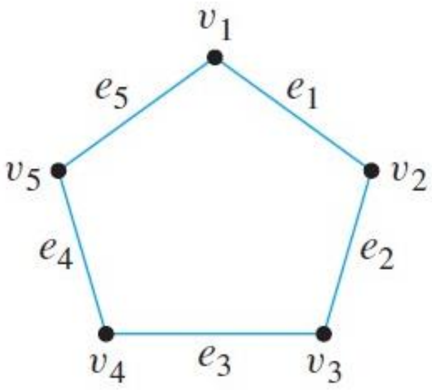
Graph Isomorphisms

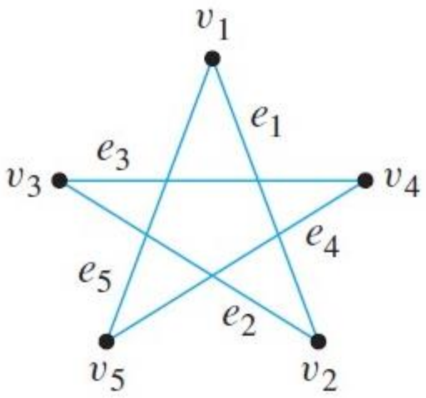
- In simpler terms, if you can rename the vertices of one graph to get the other graph without changing which vertices are connected by edges, then the two graphs are isomorphic.

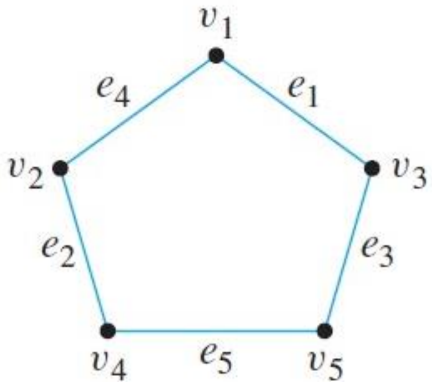


Lecture 20 – part 3

More on Graphs







V = {v1,v2,v3,v4,v5}

E = {e1,e2,e3,e4,e5}

E = {{v1,v2},{v2,v3},{v3,v4},{v4,v5},{v1,v5}}

V = {v1,v2,v3,v4,v5}

E = {e1,e2,e3,e4,e5}

E = {{v1,v2},{v2,v3},{v3,v4},{v4,v5},{v1,v5}}

V = {v1,v2,v3,v4,v5}

E = {e1,e2,e3,e4,e5}

E = {{v1,v3},{v2,v4},{v3,v5},{v1,v2},{v4,v5}}

Different sets, but ....
when relabeled, they yield the same form !.

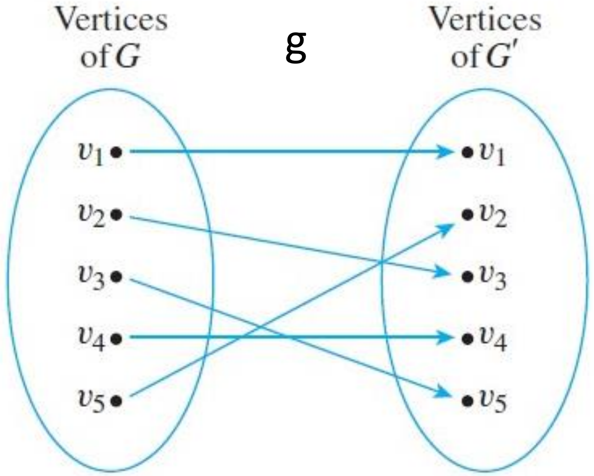
Two graphs that are the same except for the labeling of their vertices and edges are called **isomorphic** (meaning Same form).
Isomorphic graphs are those that have essentially the same form.

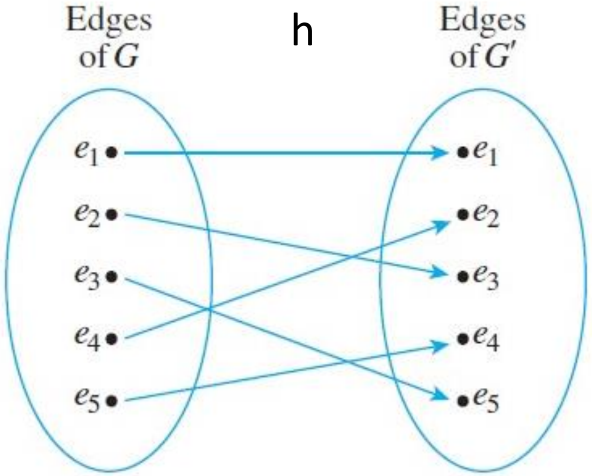
Graph Isomorphisms

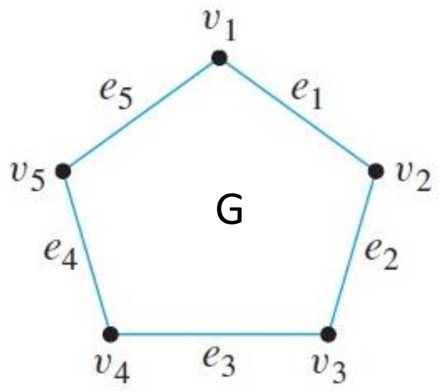
• Definition

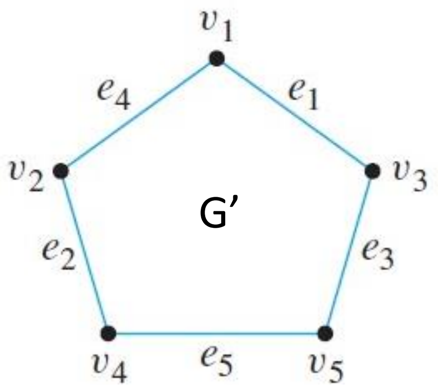
Let G and G' be graphs with vertex sets $V(G)$ and $V(G')$ and edge sets $E(G)$ and $E(G')$, respectively. **G is isomorphic to G'** if, and only if, there exist one-to-one correspondences $g\colon V(G) \to V(G')$ and $h\colon E(G) \to E(G')$ that preserve the edge-endpoint functions of G and G' in the sense that for all $v \in V(G)$ and $e \in E(G)$,

$$v \text{ is an endpoint of } e \quad \Leftrightarrow \quad g(v) \text{ is an endpoint of } h(e). \qquad 10.4.1$$



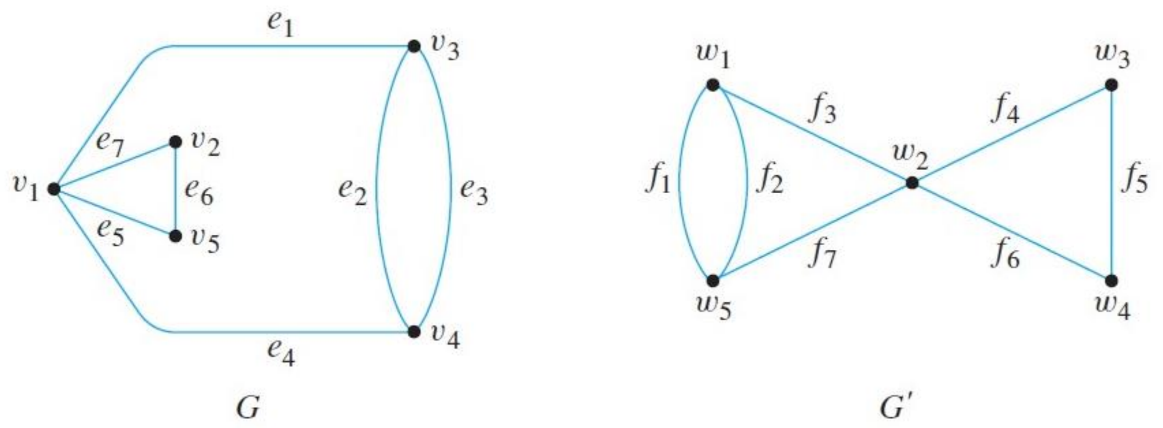






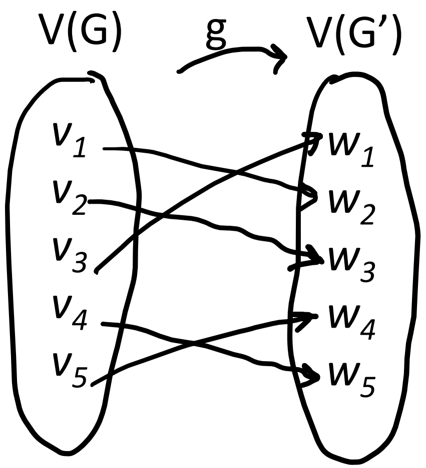
Note that these relabeling functions are one-to-one and onto.

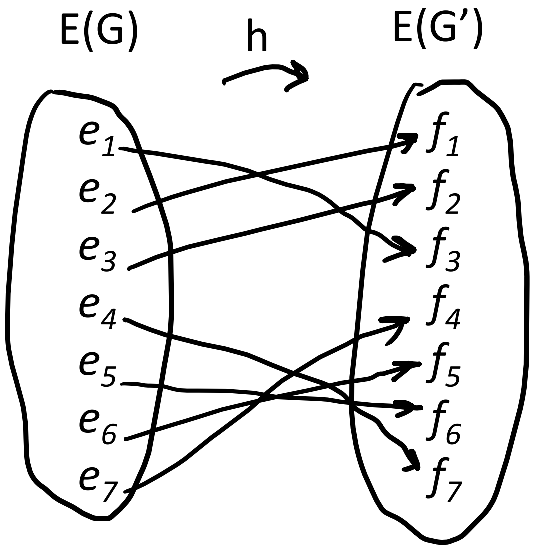
Exercise: Show that the following two graphs are isomorphic.



Solution:

To solve this problem, you must find functions **g:** V(G) → V(G’) and **h:** E(G) → E(G’)
such that for all **v** ∈ V(G) and **e** ∈ E(G), **v** is an endpoint of **e** if, and only if, **g(v)** is an endpoint of **h(e)**.





Conclusion:
There is isomorphism between *G* and *G’*.

Degree of a vertex

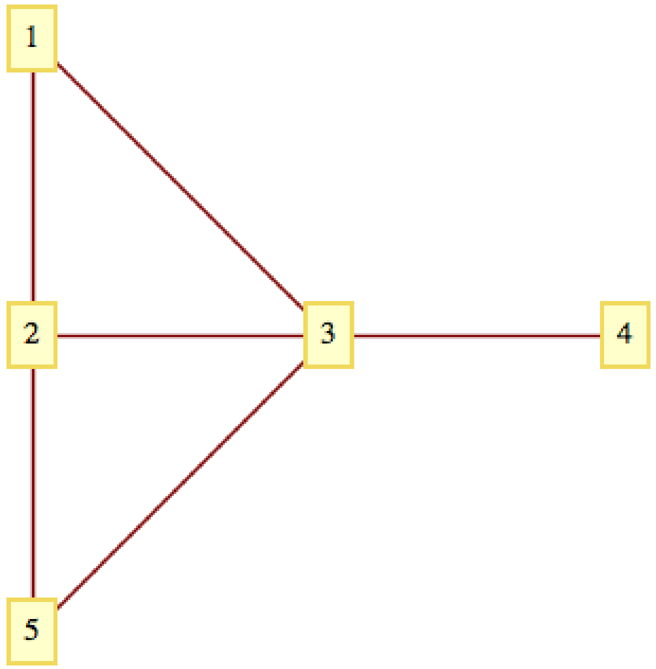
(a) Let v be a vertex of an **undirected graph**.

- The **degree** of v, denoted deg(v), is the number of edges that connect v to the other vertices in the graph.

(b) If v is a vertex of a **directed graph:**

- Then the **outdegree** of v, denoted outdeg(v), is the number of edges of the graph that initiate at v.
- The **indegree** of v, denoted indeg(v), is the number of edges that terminate at v.

Degree Sequence of a Graph: The degree sequence of a simple undirected graph is the non-increasing sequence of its vertex degrees



The degrees of vertices 1 through 5 in Figure are:
2, 3, 4, 1, and 2, respectively.

The degree sequence of the graph is (4, 3, 2, 2, 1).

Isomorphic Invariant

A property that is preserved by graph isomorphism is called an **isomorphic invariant.**

Example:

- if you are given two graphs, one with 16 vertices and the other with 17, you can immediately conclude that the two are not isomorphic**.**

• Definition

A property P is called an **invariant for graph isomorphism** if, and only if, given any graphs G and G', if G has property P and G' is isomorphic to G, then G' has property P.

Theorem 10.4.2

Each of the following properties is an invariant for graph isomorphism, where n, m, and k are all nonnegative integers:

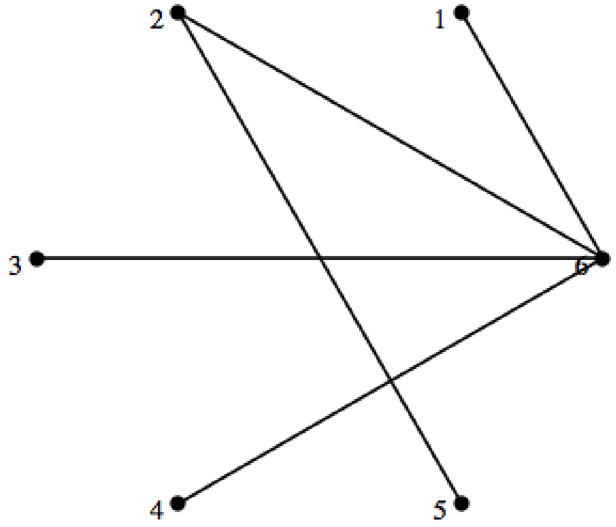
1. has n vertices;
2. has m edges;
3. has a vertex of degree k;
4. has m vertices of degree k;
5. has a circuit of length k;
6. has a simple circuit of length k;
7. has m simple circuits of length k;
8. is connected;
9. has an Euler circuit;
10. has a Hamiltonian circuit.

Traversals

Graphic Sequence

- A finite nonincreasing <u>sequence</u> of integers **d1, d2, . . . , dn** is **graphic** if there exists a simple undirected graph with n vertices having the sequence as its degree sequence.

Example: Is this sequence 4, 2, 1, 1, 1, 1 a graphic sequence ?



The sequence 4, 2, 1, 1, 1, 1 is **graphic** because the degrees of the graph in the figure match these numbers.
Note: There is no connection between the vertex number and its degree in this graph.

Basic Data Structures for Graphs

Assume a graph with n vertices that can be indexed by the integers $1, 2, \ldots, n$.

Adjacency Matrix:

An adjacency matrix, G, where $G_{ij} = 1$ if and only if vertex i is connected to vertex j in the graph.

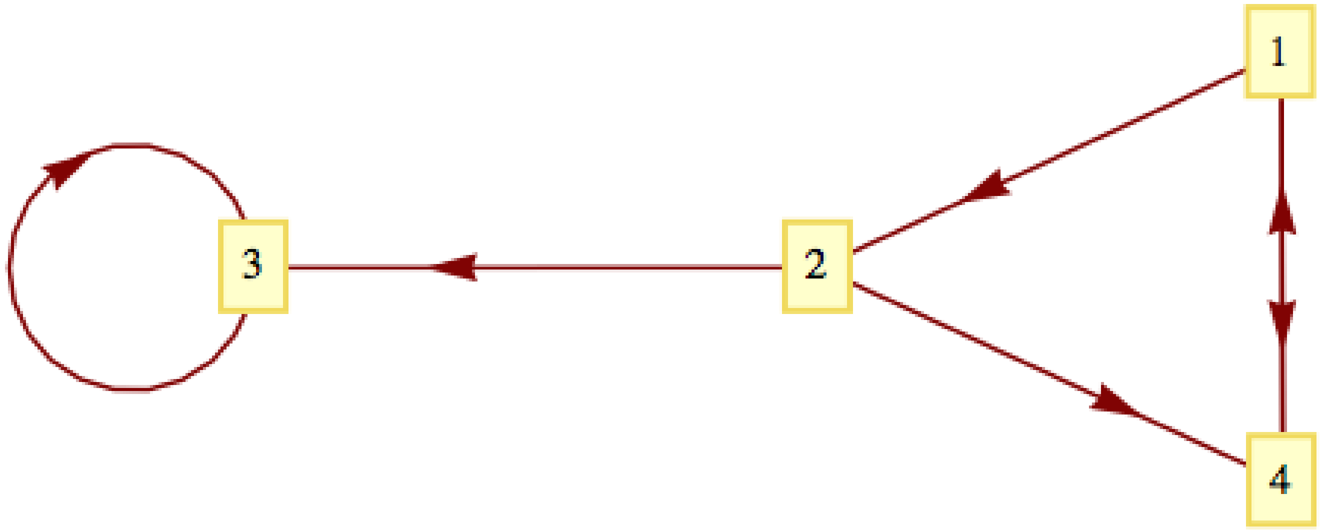
Edge Dictionary:

For each vertex in the graph, we maintain a list of edges that initiate at that vertex. If G represents the graph's edge information, then we denote by G_i the list of vertices that are terminal vertices of edges initiating at vertex i.

Edge List:

A simple way to represent the edges is to maintain a **list of ordered pairs, or two element sets**, depending on whether the graph is intended to be **directed or undirected**.

Example:



Adjacency Matrix

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

Edge Dictionary

G = { 1:[2,4], 2:[3,4], 3:[3], 4:[1] }

Edge List

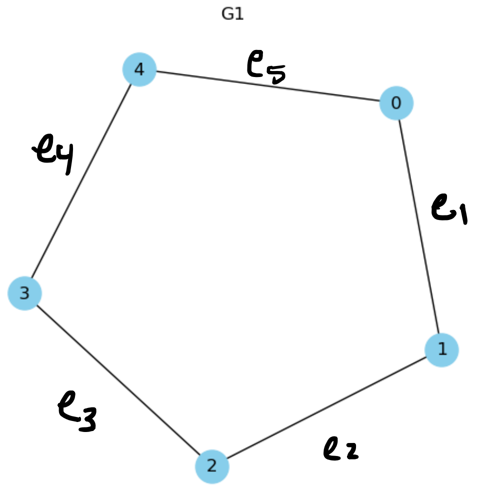
G = [(1,2),(1,4),(2,3),(2,4),(3,3),(4,1)]

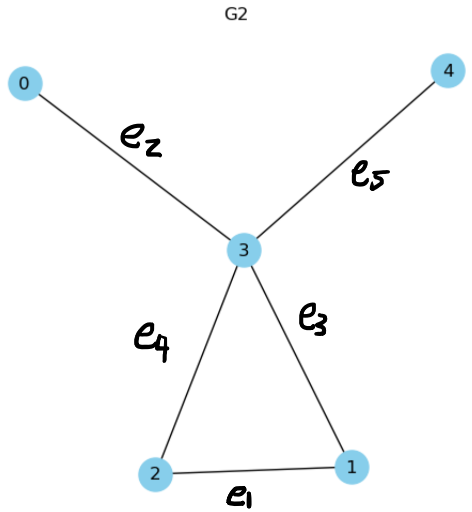
Exercise:

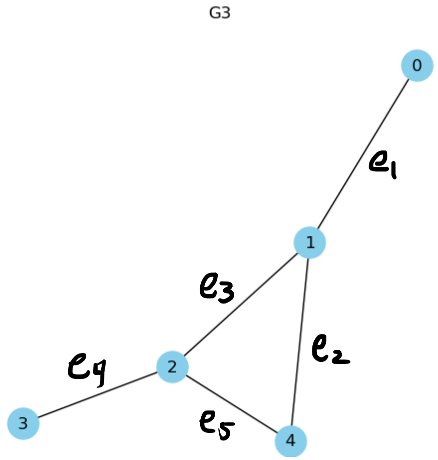
Directed graphs $G_1, \ldots, G_6$, each with vertex set $\{1, 2, 3, 4, 5\}$ are represented by the matrices below. Which graphs are isomorphic to one another?

$$G_1 : \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_2 : \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_3 : \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$G_4 : \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad G_5 : \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad G_6 : \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$







G1

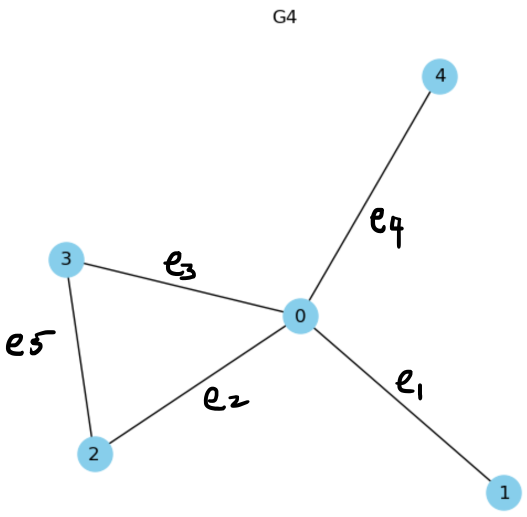
	v0	v1	v2	v3	v4
v0	0	1	0	0	0
v1	0	0	1	0	0
v2	0	0	0	1	0
v3	0	0	0	0	1
v4	1	0	0	0	0

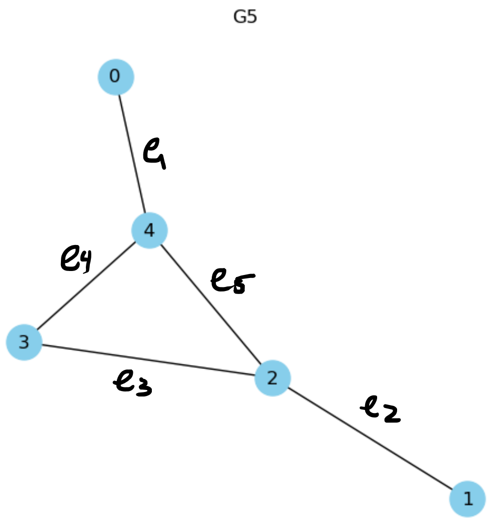
G2

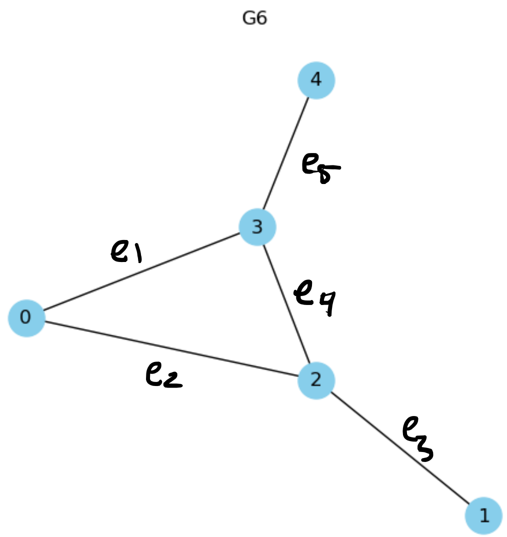
	v0	v1	v2	v3	v4
v0	0	0	0	0	0
v1	0	0	1	0	0
v2	0	0	0	0	0
v3	1	1	1	0	1
v4	0	0	0	0	0

G3

	v0	v1	v2	v3	v4
v0	0	0	0	0	0
v1	1	0	0	0	1
v2	0	1	0	0	0
v3	0	0	1	0	0
v4	0	0	1	0	0







G4

	v0	v1	v2	v3	v4
v0	0	1	1	1	1
v1	0	0	0	0	0
v2	0	0	0	0	0
v3	0	0	1	0	0
v4	0	0	0	0	0

G5

	v0	v1	v2	v3	v4
v0	0	0	0	0	1
v1	0	0	0	0	0
v2	0	1	0	1	0
v3	0	0	0	0	1
v4	0	0	1	0	0

G6

	v0	v1	v2	v3	v4
v0	0	0	0	1	0
v1	0	0	0	0	0
v2	1	1	0	0	0
v3	0	0	1	0	0
v4	0	0	0	1	0

Python code

```
{'G1 and G2': False,
 'G1 and G3': False,
 'G1 and G4': False,
 'G1 and G5': False,
 'G1 and G6': False,
 'G2 and G3': False,
 'G2 and G4': True,
 'G2 and G5': False,
 'G2 and G6': False,
 'G3 and G4': False,
 'G3 and G5': True,
 'G3 and G6': True,
 'G4 and G5': False,
 'G4 and G6': False,
 'G5 and G6': True}
```

✓