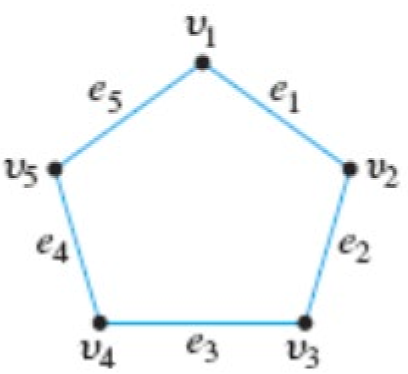
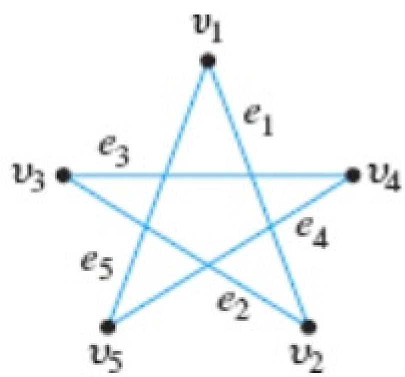
Section 10.3 – Isomorphism of Graphs

Partner up

Give the edge-endpoint table for the following two graphs:



(a) , G



(b)

==

(a)

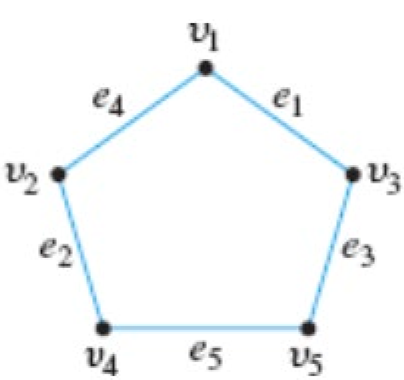
Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_2, v_3\}$
e_3	$\{v_3, v_4\}$
e_4	$\{v_4, v_5\}$
e_5	$\{v_1, v_5\}$

(b)

Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_2, v_3\}$
e_3	$\{v_3, v_4\}$
e_4	$\{v_4, v_5\}$
e_5	$\{v_1, v_5\}$

They're the same! So even though they are drawn differently, these two graphs are the same. A graph is completely defined by its edge-endpoint table, not how it is drawn.

Now, consider the third graph:



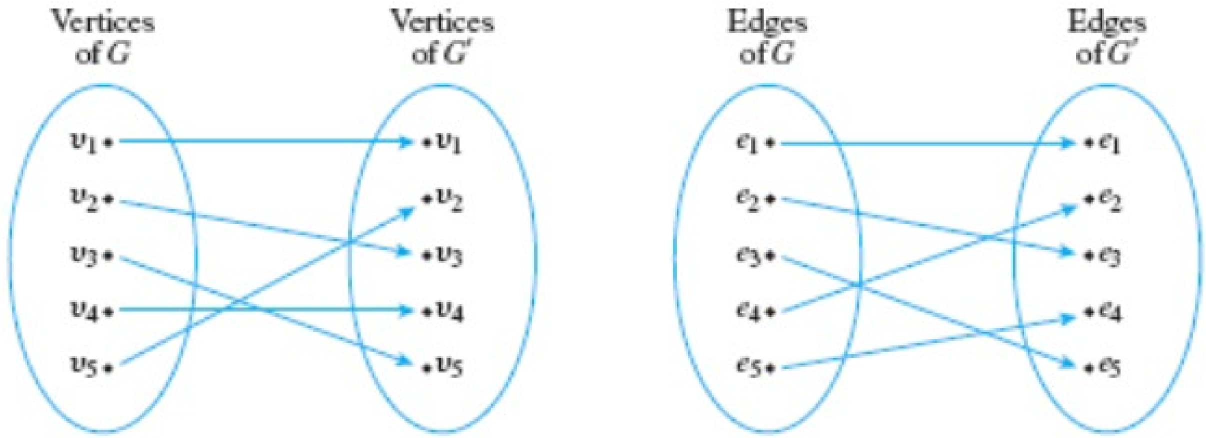
(c) G'

What is its edge-endpoint table?

Edge	Endpoints
e_1	$\{v_1, v_3\}$
e_2	$\{v_2, v_4\}$
e_3	$\{v_3, v_5\}$
e_4	$\{v_1, v_2\}$
e_5	$\{v_4, v_5\}$

(a) ***looks like*** (c), but they ***ARE NOT*** the same graph, since their edge-endpoint tables are different.

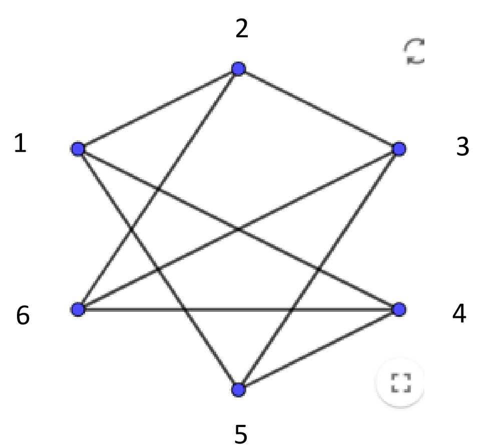
But in this section, we will explore graphs that are alike, but not the same. Specifically, we can find functions that map the edges from (a) to (c).

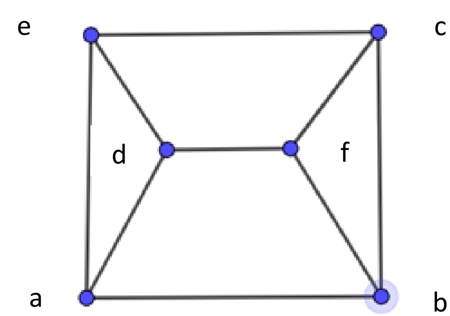


Note that these relabeling functions are ***one-to-one*** AND ***onto***. This means they are one-to-one correspondences and invertible. If you can map one way, you can map back to the original set of vertices and edges.

Two graphs that are the same except for the labeling of their vertices and edges are called ***isomorphic***, which is derived from Greek meaning "same form."

Are these two graphs isomorphic?





Yes!, we can map the vertices! 1→a 2→b 3→c 4→d 5→e 6→f

This was pretty complicated, it helps to have a tool, like Doug Ensley's Graph Isomorphism: https://www.geogebra.org/m/snvmfqxk to do the manipulations.

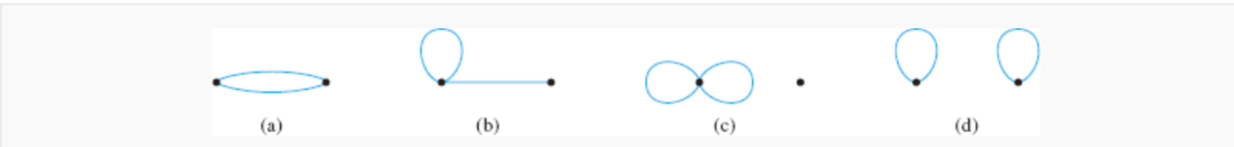
> It is not hard to show that graph isomorphism is an equivalence relation on a set of graphs; in other words, it is reflexive, symmetric, and transitive.

Theorem 10.3.1 **Graph Isomorphism Is an Equivalence Relation**

> Let S be a set of graphs and let R be the relation of graph isomorphism on S. Then R is an equivalence relation on S.

Practically, this means that if we just consider the "shape" of graphs, and not their edge or vertex labels, we kind find equivalence classes. So, for example, Find all nonisomorphic graphs that have two vertices and two edges. In other words, find a collection of representative graphs with two vertices and two edges such that every graph with two vertices and two edges is isomorphic to one in the collection.

There are four nonisomorphic graphs that have two vertices and two edges. These can be drawn without vertex and edge labels because any two labelings give isomorphic graphs.



Remember that previously we defined ***simple graphs***:

One important class of graphs consists of those that do not have any loops or parallel edges. Such graphs are called *simple*. In a simple graph, no two edges share the same set of endpoints, so specifying two endpoints is sufficient to determine an edge.

If you are checking for isomorphism in simple graphs, you only need to map vertices, since edges are unique to each pair of vertices.

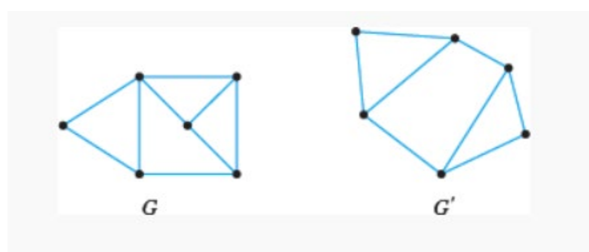
Properties preserved by isomorphism, where $m, n, and\ k$ are all non-negative integers:

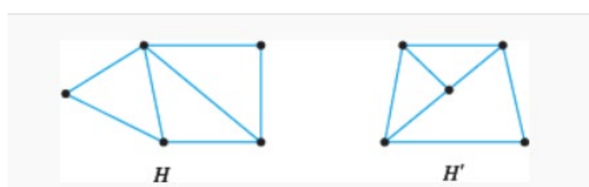
1. has n vertices
2. has m edges
3. has a vertex of degree k
4. has m vertices of degree k
5. has a circuit of length k
6. has a simple circuit of length k
7. has m simple circuits of length k
8. is connected
9. has an Euler circuit
10. has a Hamiltonian circuit.

So, these are reasons that two graphs wouldn't be isomorphic. Generally the first 4 properties are used, although there are a few homework problems where properties 6 and 7 come into play.

Partner up

Show the following pairs of graphs are not isomorphic by finding an isomorphic invariant that they do not share:





==

G has 9 edges, G' only has 8 edges.
H has a vertex of degree 4 edges, H' does not have any vertices of degree 4.