Chapter 5

1 Section 5.1: Basic Notation and Terminology for Graphs

Definition 1.1. A **graph** G is a mathematical object consisting of two sets:

- a **vertex set** $V(G)$, and $V(G) = \{A, B, C, D, E, F, \cancel{G}\ X\}$
- an **edge set** $E(G)$ consisting of <u>unordered</u> pairs of elements of the vertex set.

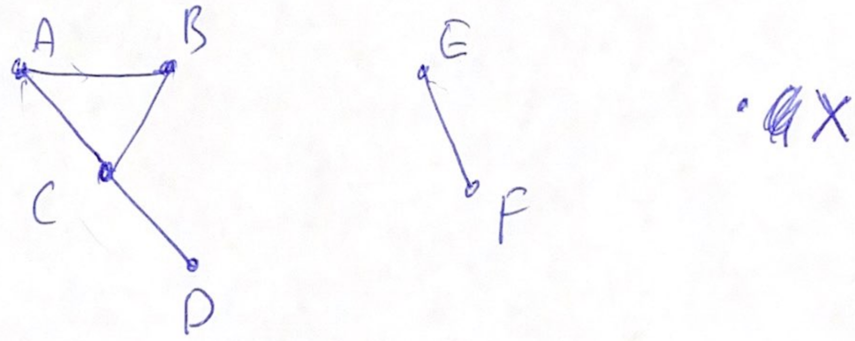
$E(G) = \{\{A,B\}, \{A,C\}, \{B,C\}, \{C,D\}, \{E,F\}\}$

We call elements of $V(G)$ **vertices** and elements of $E(G)$ **edges**.

We think of the vertices as being points in space, and the edges as lines connecting them.

$v \in G$

G



Definition 1.2. Two verices x and y of a graph G are **adjacent** if there is an edge (x, y). (The two vertices are connected by an edge.) If x and y are adjacent, we can also say that y is a **neighbor** of x (and vice versa).

Definition 1.3. The **degree** of a vertex x, given by the notation $d(x)$, is the number of edges it is contained in. A vertex is called a **leaf** if its degree is 1.

A is adjacent to B and C $\Leftrightarrow$ (A has B and C as neighbors)

A is not adjacent to D, E, F, X

~~deg~~ degree of A: $d(A) = 2 = d(B)$

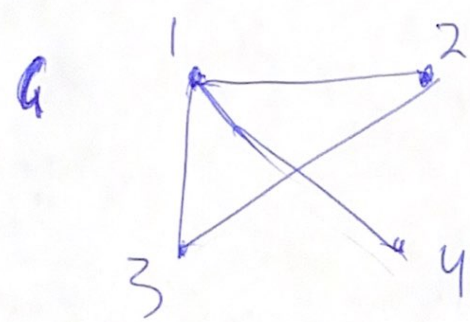
$d(C) = 3$

leaves are D, E, F $\longrightarrow d(D) = d(E) = d(F) = 1$

$d(X) = 0$

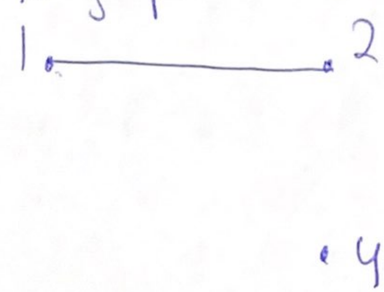
Definition 1.4.

- A graph H is a **subgraph** of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.
- A subgraph H of G is a **spanning subgraph** if $V(H) = V(G)$.
- If we take a subset of the vertices of the graph G, that is $X \subseteq V(G)$, the **induced subgraph** of G from X consists of X and all the edges of G that connect vertices in X.

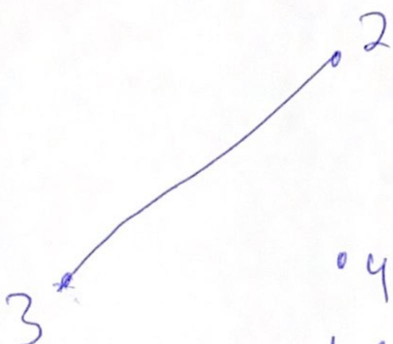


subgraph H of G

one possible subgraph of G

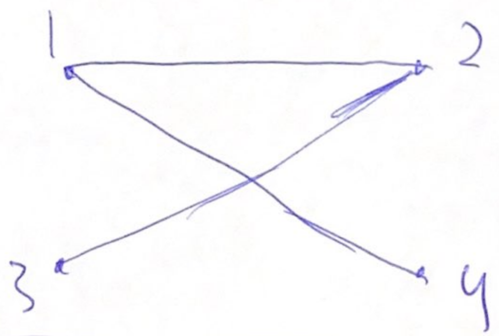


subgraph induced by 2, 3, 4

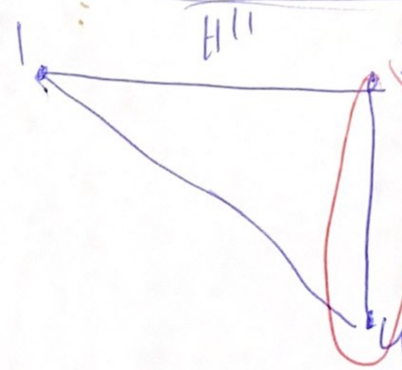


all other edges of G touch vertex 1, which is not in this induced subgraph, so I don't include these

spanning subgraph H' of G (choose all vertices of G for H')

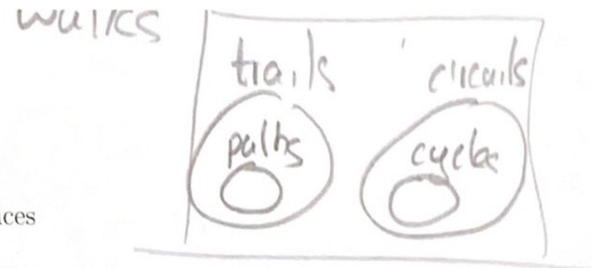


H''



NOT a subgraph

where did this edge come from?



Definition 1.5. A **walk** is a sequence of vertices

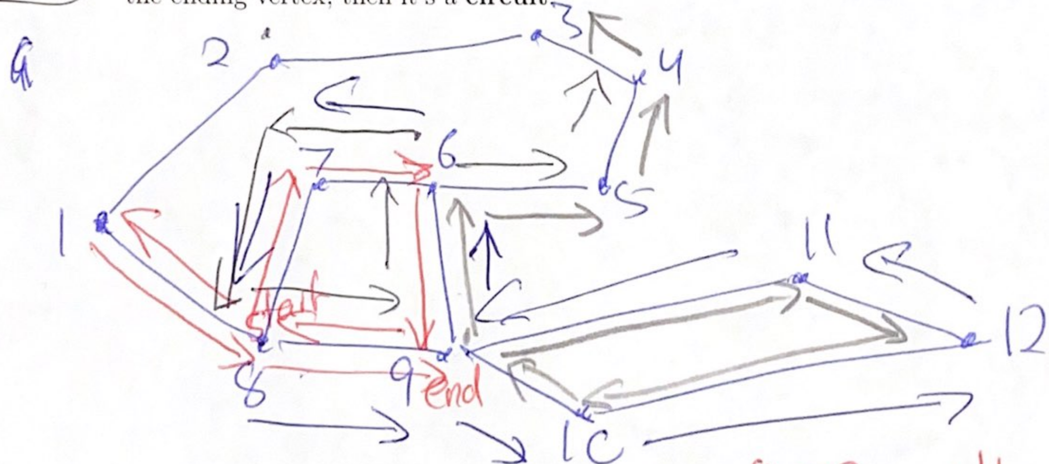
$$x_1 \to x_2 \to \cdots \to x_{m-1} \to x_m$$

such that there is an edge between each pair of consecutive vertices x_i and x_{i+1}.

length is the number of edges that we traverse

If a walk:

- does not repeat edges and does not repeat vertices, then it's a **path**, (every path is a trail)
- does not repeat edges and does not repeat vertices except that the starting vertex is the same as the ending vertex, then it's a **cycle**, (every cycle is a circuit)
- does not repeat edges but can repeat vertices, then it's a **trail**,
- does not repeat edges but can repeat vertices, and the starting vertex is the same as the ending vertex, then it's a **circuit**.



8→7→6→9→8→1→8→9 walk length 7

1→2→5→4 not a walk

no edge between 2 and 5

9→6→5→4→3 path (and a trail) length 4

9→11→12→10→9 cycle (and a circuit) length 4

6→7→8→9→6→5→4 trail that is not a path length 6

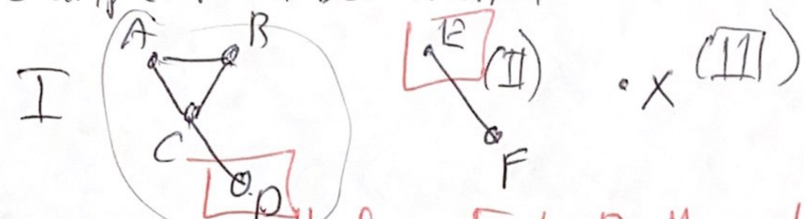
8→9→10→12→11→9→6→7→8 circuit that is not a cycle length 8

2 length 0 walk

above A→B→C→A→D→E→A not a cycle because A appears in the middle

Definition 1.6. A graph G is **connected** if for any two vertices there is a walk between them. A graph is **disconnected** if it is not connected.

- Graph example from Definition 1.5 is connected
- Graph example from Definition 1.1 is disconnected

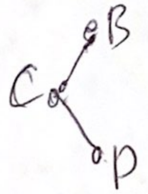


There is no walk from E to D, thus this graph is disconnected

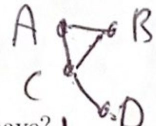
Definition 1.7. A **component** C of a graph G is a maximal connected subgraph (meaning that there is no bigger connected subgraph of G that contains C).

three components of above graph:

- induced subgraph on A, B, C, D (I)
- " " on E, F (II)
- " " on X (III)



is a connected subgraph, but it's not a component, because there is a bigger connected subgraph



that contains it.

Question 1.8.

- How many components does a connected graph have? 1
- Is the graph consisting of a single vertex connected? yes
- How many components can one vertex belong to? 1

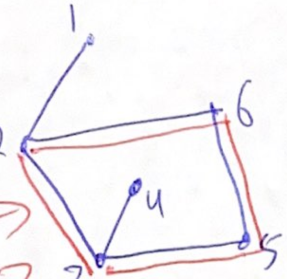
if two vertices are in different components, you can't have a walk between them

Definition 1.9.

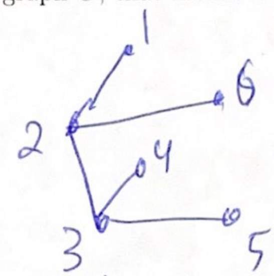
- A graph is **acyclic** if it does not have any cycles.
- An acyclic graph is a **tree** if it is connected, and a **forest** if it is not connected.
- If H is a **spanning tree** of a graph G, that means it is both a spanning subgraph of G and a tree.

acyclic graphs

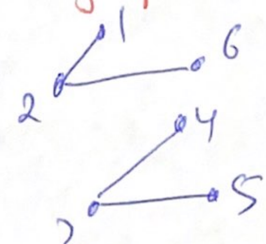
cycle →



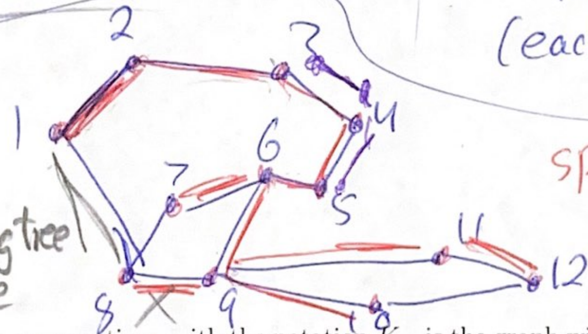
this graph has a cycle, so it is not acyclic



tree



forest (each component is a tree)



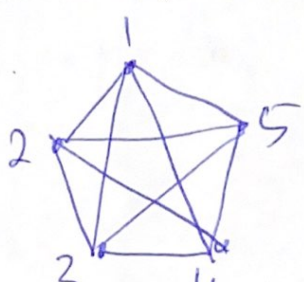
spanning tree of this graph ← (edges of tree marked in red)

graphs can have more than one spanning tree

could get a different tree if I included $\overline{18}$ instead of $\overline{89}$

Definition 1.10.

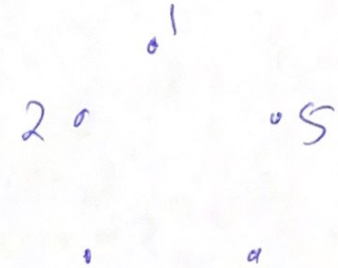
- The **complete graph** on n vertices, with the notation K_n, is the graph with n vertices and an edge between every pair of vertices.
- The **independent graph** on n vertices, with the notation I_n, is the graph with n vertices and no edges.



K_5

how many edges does K_n have?

$\boxed{\binom{n}{2}}$



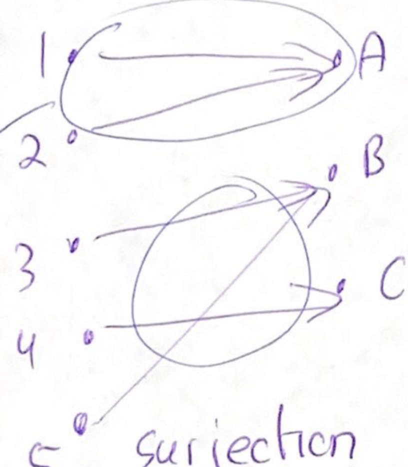
I_5

how many edges does I_n have?

$\boxed{0}$

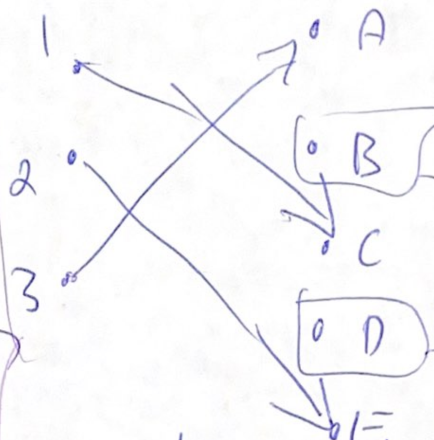
Definition 1.11. A **bijection** between two sets X and Y is a function $f : X \to Y$ that satisfies the following properties:

- it is **surjective**, meaning every element $y \in Y$ has some element $x \in X$ that is mapped to it ($f(x) = y$), and
- it is **injective**, meaning that no two distinct elements of x, x' in X are mapped to the same element y in Y ($x \neq x'$ implies $f(x) \neq f(x')$).



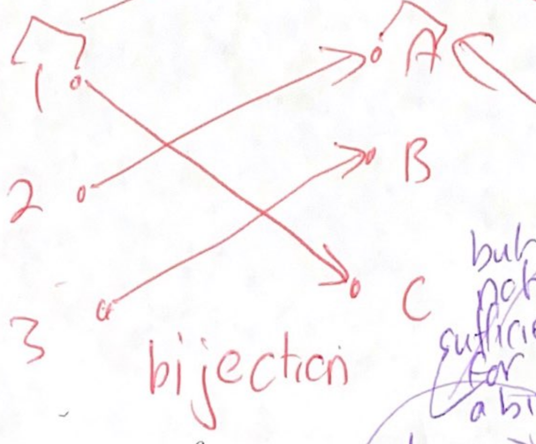
surjection (function that is surjective

not an injection, because two elements, 1 and 2, are mapped to the same element A



not a surjection nothing maps to B and D

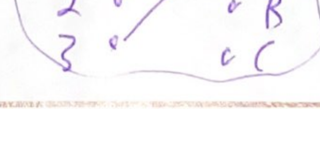
injection each ~~element~~ number goes to a different ~~element~~ letter



bijection

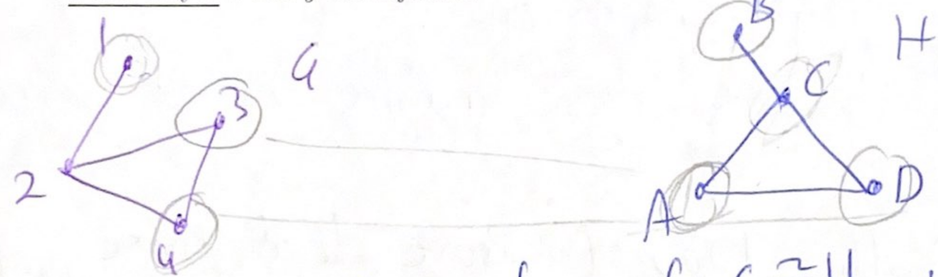
domain and codomain must be the same size to get a bijection between them

but not sufficient for a bijection



Definition 1.12. A **graph isomorphism** f between graphs G and H, usually given by the notation $f : G \cong H$, is a bijection $f : V(G) \to V(H)$ between the vertex sets of G and H that also satisfies the following property:

- for any two vertices x and y in $V(G)$, we have that $f(x)$ and $f(y)$ are adjacent if and only if x and y are adjacent.



graph isomorphism ~~from~~ $f : G \cong H$, what I'm saying is that G and H are structurally the same, you can take G, "peel off" the labels of G, and replace them with the labels of H, and now you have the graph H. The graph isomorphism f is telling you how to replace those labels.

Defining graph isomorphism $f : V(G) \to V(H)$ by

$f(1) = B$ $1 \sim 2$ $f(1) \sim f(2)$, $B \sim C$

$f(2) = C$ $2 \sim 3$ $\Longrightarrow$ $f(2) \sim f(3)$, $C \sim A$

$f(3) = A$ $2 \sim 4$ $f(2) \sim f(4)$, $C \sim D$

$f(4) = D$ $3 \sim 4$ $f(3) \sim f(4)$, $A \sim D$

also need to show 1 not ~ 3 $\Longrightarrow$ $f(1)$ not $\sim f(3)$, B not $\sim A$

1 not ~ 4 $f(1)$ not $\sim f(4)$, B not $\sim D$

so ~~f G~~ f is a graph isomorphism.

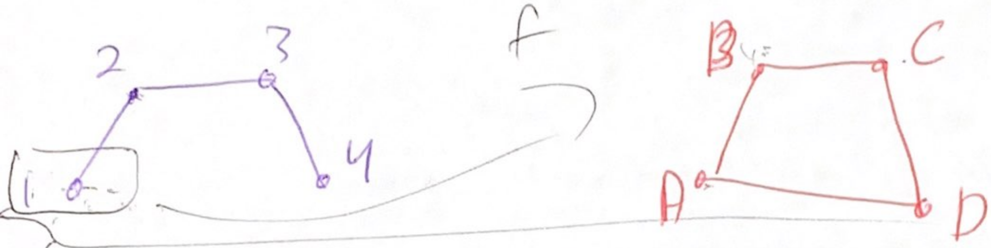
so G and H are isomorphic, but that doesn't mean that every function $f : V(G) \to V(H)$ is also an isomorphism.

Define f' to be

$f'(1) = C$ $f'(4) = D$

$f'(2) = A$

$f'(3) = B$



red graph has a cycle
purple graph has no cycles
$\therefore$ these graphs are not isomorphic

Another Way to Prove that these Graphs are not isomorphic:

vertex labeled 1

purple graph has a vertex of degree 1
red graph: every vertex has degree 2
so, any function f from the vertex set of purple to the vertex set of red will map 1 to a degree 2 vertex.
f(1) is adjacent to two other vertices, but 1 is adjacent to only one vertex, 2,
f(1) will be adjacent to not just f(2) but at least one of f(3) or f(4). So adjacency is not preserved, so f can't be a graph isomorphism. So graphs are not isomorphic.

In general, to show that two graphs are not isomorphic, you show that they differ in structure:

- degrees of vertices are different
- different # of vertices
- different # of edges
- one has a cycle, the other doesn't; one has a longer cycle
- one is connected, the other is not

Exercise 1.13 (Exercise 5.3 from textbook). Draw a graph with 6 vertices having degrees 5, 4, 4, 2, 1, and 1 or explain why such a graph does not exist.

Such a graph does not exist.

$$\boxed{\text{sum of degrees of all vertices in a graph} = 2 \cdot \#\text{ of edges}}$$

so sum of degrees must always be even

$5+4+4+2+1+1$

$= 17$, odd so

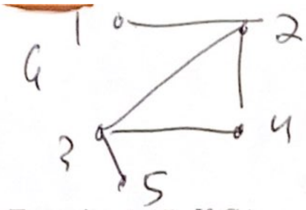
Exercise 1.14. Suppose G is a graph where every vertex has degree 3. If there are 18 edges in G, how many vertices are there?

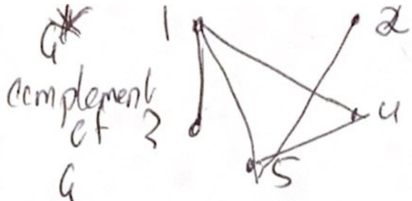
$n = \#$ of vertices

$3 \cdot n = \text{sum of degrees} = 2 \cdot \#\text{ of edges} = 36$

$\Rightarrow 3n = 36$

$\Rightarrow \boxed{n = 12}$





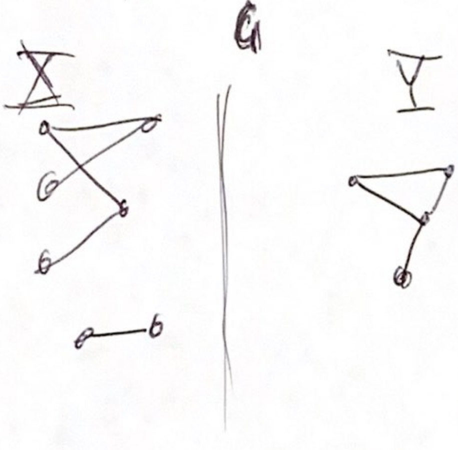
so vertex in X and vertex in Y

2 vertices in X go to vertex in Y and then back

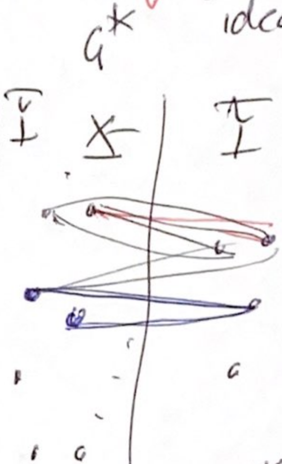
2 vertices in Y, same idea

Exercise 1.15. If G is a graph, its **complement** G^* is the graph that has the same vertices as G, but has an edge between two vertices a and b if and only if there is not an edge between a and b in G. Prove that if G is disconnected, then G^* must be connected.

~~Prove tha~~

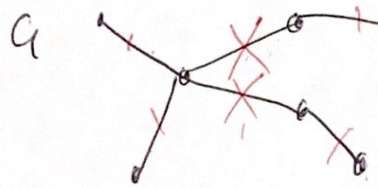


partition G into two nonempty sets, such that X and Y there is no way to get from a vertex in X to a vertex in Y, so there is no edge from a vertex in X to a vertex in Y



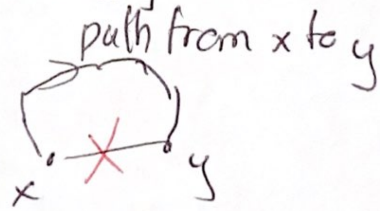
so in G^* ~~every edge~~ there is an edge between any vertex in X and any vertex in Y.

Exercise 1.16. Prove that if G is a tree, then removing any edge will disconnect G.



Show that if removing an edge of G does not disconnect G, then that edge is part of a cycle of G. Since trees can't have cycles, we are done.

Let's say that the edge $\overline{xy}$, when removed, does not disconnect G. So there is a path from x to y:



$$x \to v_1 \to v_2 \to v_3 \to \ldots \to v_n \to y$$

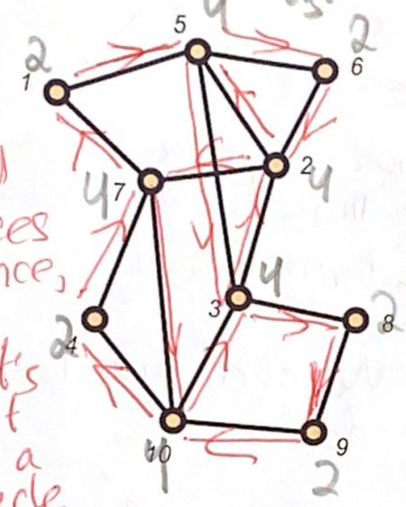
$$x \to v_1 \to v_2 \to v_3 \to \ldots \to v_n \to y \to x$$

putting $\overline{xy}$ back into G turns path into cycle

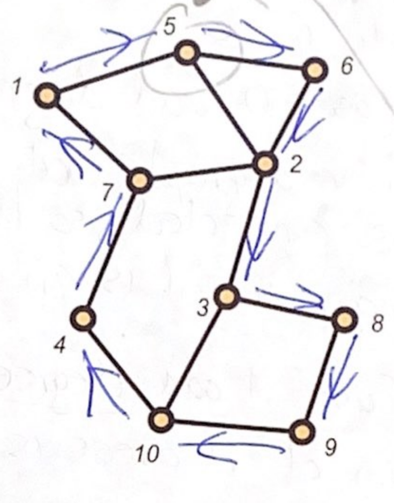
Eulerian trail, but no Eulerian cycle

not Hamiltonian

Eulerian circuit: we visited some vertices more than once, that's ok, because it's a circuit and not a cycle



G



H

Hamiltonian cycle: do not have to use every edge

vertex with odd degree, so this graph is not Eulerian

2 Section 5.3: Eulerian and Hamiltonian Graphs

Definition 2.1.

- An **Eulerian circuit** is a circuit that uses each edge exactly once. A graph is **Eulerian** if it contains no isolated vertices and has an Eulerian circuit.
- A **Hamiltonian cycle** is a cycle that visits every vertex in the graph. A graph G is **Hamiltonian** if it has a Hamiltonian cycle.

Theorem 2.2. *A graph G is Eulerian if and only if it is connected and every vertex has even degree.*

Eulerian circuits are Eulerian trails

Exercise 2.3 (Exercise 5.11 from the textbook). An **Eulerian trail** is defined in the same manner as an Eulerian circuit, except that we drop the condition that $x_0 = x_t$ (that is, the starting vertex and ending vertex do not have to be the same). Prove that a graph has an Eulerian trail if and only if it is connected and has at most 2 vertices of odd degree.

Eulerian circuit $\Rightarrow$ Eulerian trail

0 vertices odd degree $\Rightarrow$ ≤ 2 vertices odd degree

prove

have Eulerian trail $\Rightarrow$ connected and has at most 2 odd-degree vertices

AND

connected and has at most 2 odd-degree vertices $\Rightarrow$ Eulerian trail connected and ≤ 2 odd $\Longrightarrow$ Eulerian trail

$\Longleftarrow$ look at every vertex besides start and end vertices

pair up all the edges, so this must have even degree according to them appearing consecutively in Eulerian trail

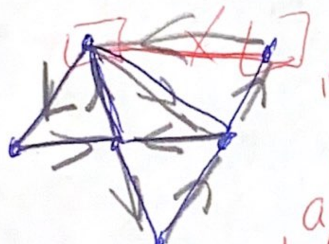
Break into Cases:

Case: 0 odd degree:

We have a graph that has all vertices of even degree and it's connected, so by Thm it has an Eulerian circuit. Eulerian circuit is an Eulerian trail.

Case: 1 odd degree: No such graph exists. (sum of degrees is odd)

Case: 2 odd degree:



if two odd degree vertices are not adjacent, add in an edge between them;

transformed into an Eulerian graph, it has an Eulerian circuit

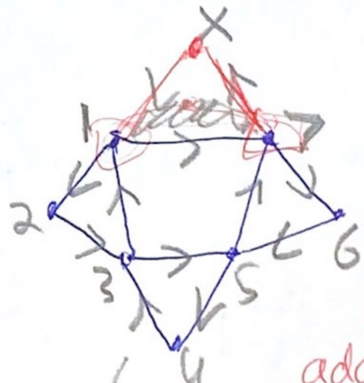
· consider the circuit when the edge we added is the last edge. Throw that edge away, we have our Eulerian trail.

$1 \to 2 \to 3 \to 1$

stop here instead

$\underbrace{1 \to 2 \to 3}$

Eulerian trail of original



if two odd degree vertices are adjacent add in extra vertex and ~~two edges, one~~ edges between added vertex and odd degree vertices.

so we have Eulerian circuit

$1 \to 2 \to 3 \to 1 \to 7 \to 6 \to 5$
$\to 4 \to 3 \to 5 \to 7$
$\to x \to 1$

stop here, we have our Eulerian trail.