M106 Summer 2020 Recitation Lectures: Graph Theory

6/26 Lecture

Announcements:

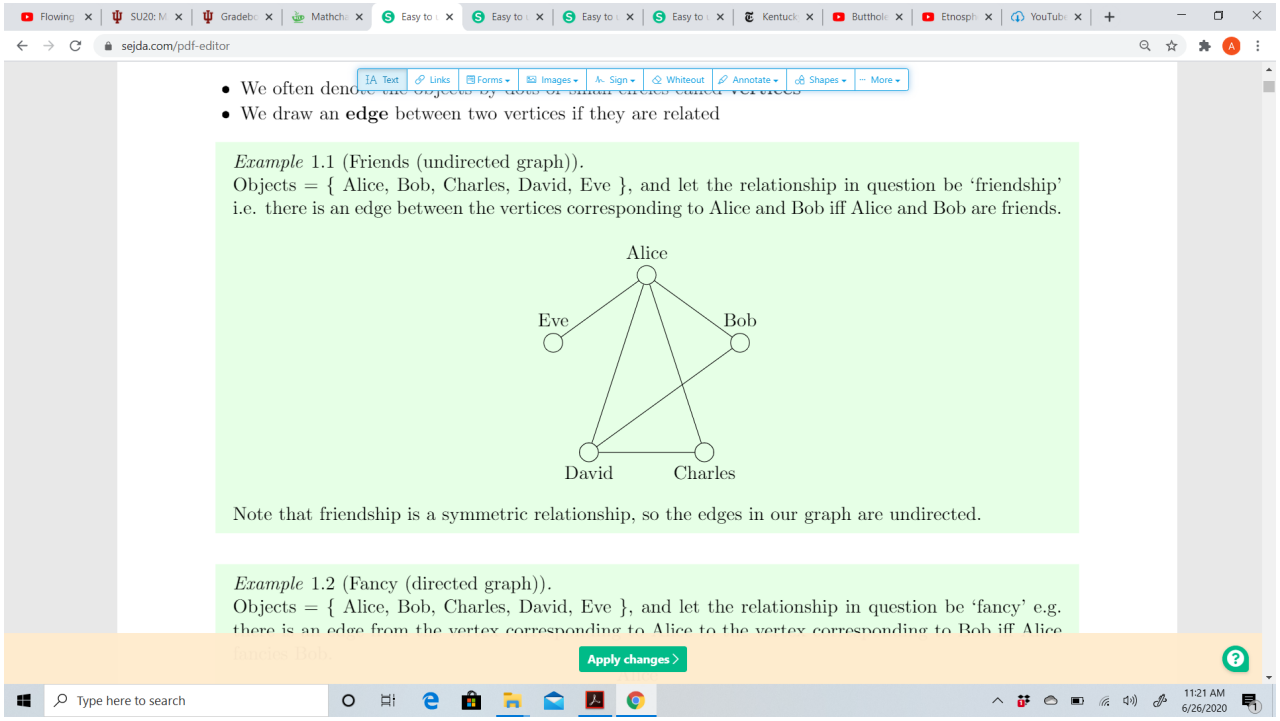
-Technical Submission Issues with the exam
-Exam 2 Grades ETA: Tonight (no later than Sunday)
-New unit, who dis? (Graph Theory)
-Homework 1 due Teusday

Logistics Questions:

Class 1-2: What is a Graph?

What is a graph? I'll tell you what a graph is NOT. It isn't a diagram on a 2-D or 3-D plane. (we'll learn that in the "perspectives" unit)

Instead, a graph is a drawing consisting of nodes and edges; models a network of relations



This unit may be abstract, but this unit has THE MOST "real life applications", and it's not even close!

Computer Science: networks, data structures that can be modeled by graph theory

Linguistics: natural language is modeled by discrete structures

Physics and chemistry: condensed matter physics, models atoms

Biology: Models molecular biology and genomics

networks often used epidemiology to model the spread disease

Neuroscience

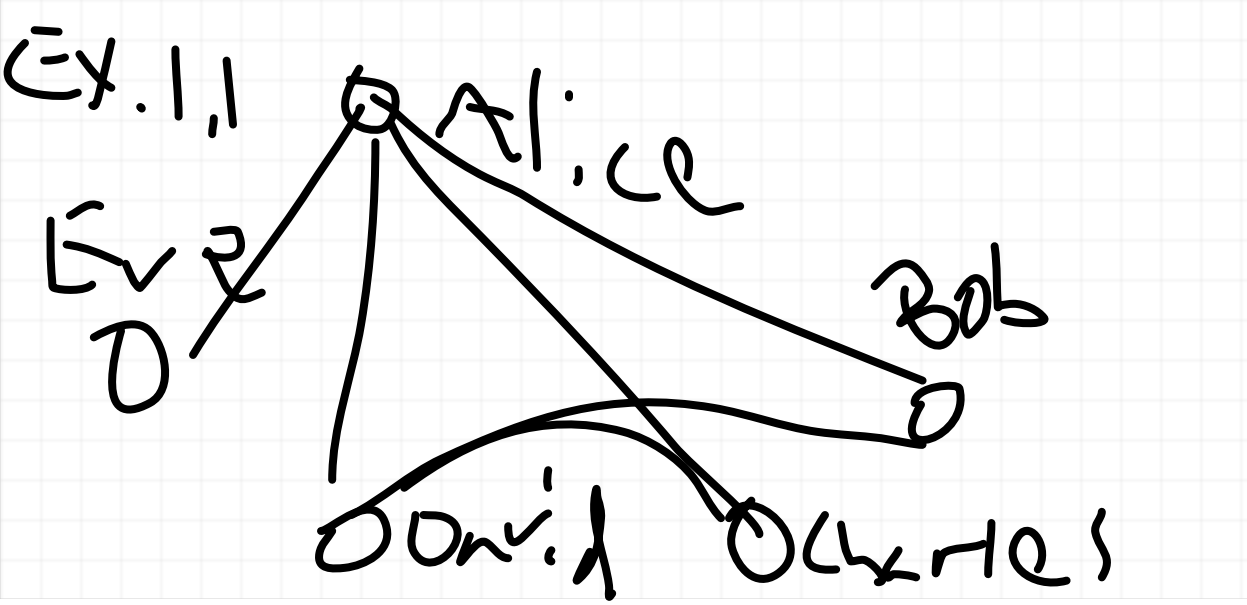
Social sciences: Sociology dealing with social network theory

Mathematics: Abstract math, applied math

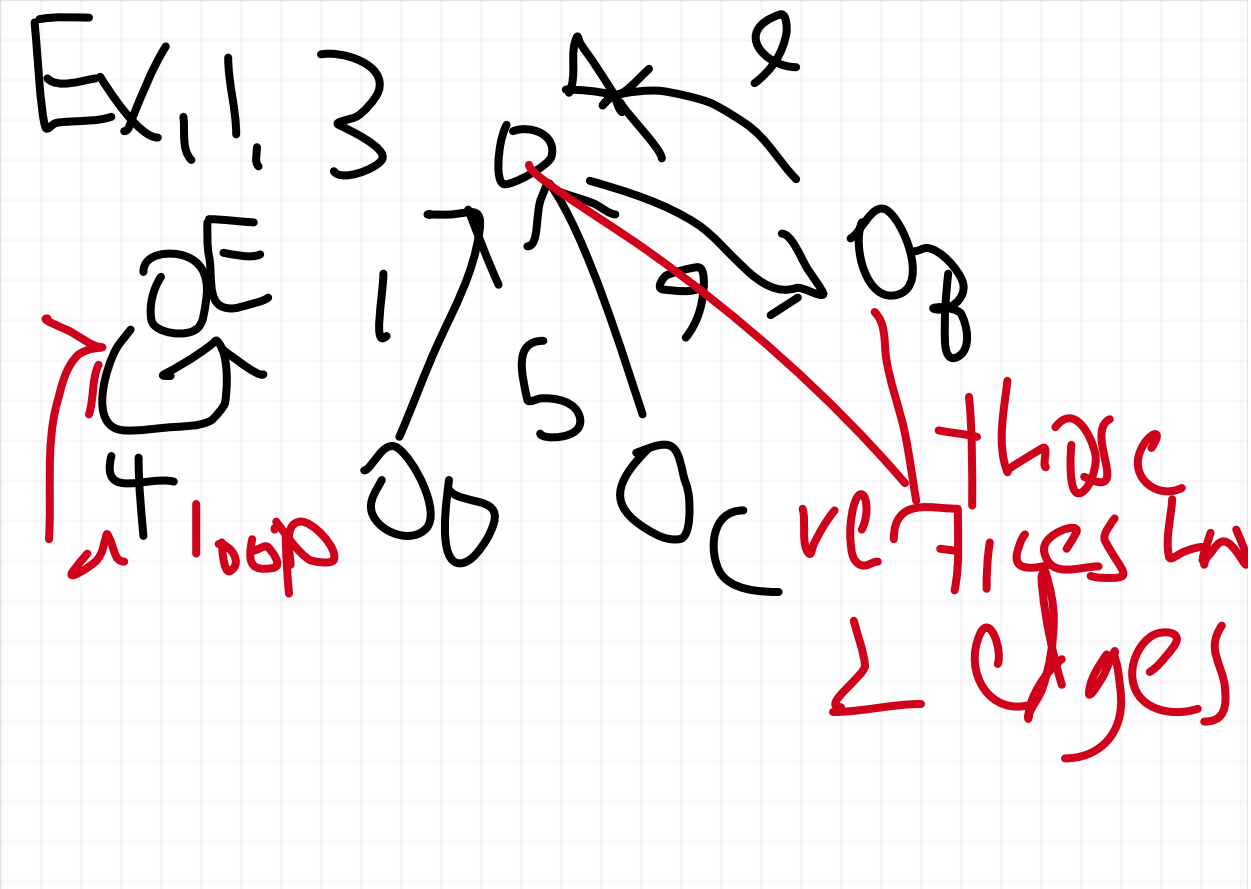
Formal Definition of a graph (Def. 2.1): A **graph** $G = (V,\ E)$ is a collection of vertices $V = \{v_1,\ v_2,\ \ldots, v_n\}$ and **edges** $E = \{(w_1,\ w_2),\ (w_3,\ w_4),\ \ldots (w_m,\ w_{m+1})\}$.

There may be some added structure to a graph:

A graph is **Undirected** if the edges are "undirected" (ie, drawn without arrows).



A graph is **directed** if the edges are directed (if the graph is drawn with arrows).



A graph is **weighted** if the edges are "weighted" (if the graph is drawn with numbers next to the edge, representing "weights")

Example 1.3 is also weighted

Other terms:
A **loop** is an edge between itself
A **simple graph** is a graph that contains up to one edge between any two vertices and no "loops"
Example 1.1 is an instance of a simple graph
A **multigraph** is a graph where ther are multiple edges between some pairs of vertices
Example 1.3 is an instance of a multigraph

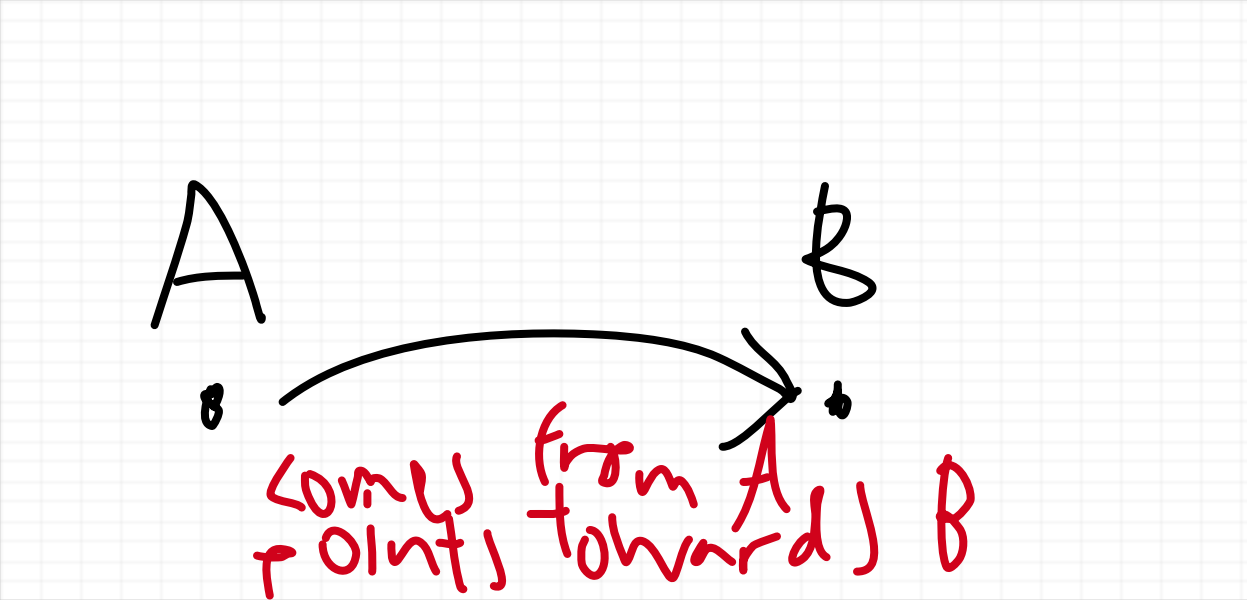
Degree of a vertex (Def. 2.2): The **degree** of a vertex v in a graph G is the number of *connections* between edges that v touches

A **degree sequence** of a graph is the list of all vertex degrees, with repition, in *increasing order*

We denote the degree as $deg(v)$
$deg(v)$ is a function mapping from a vertix to a number that is the number of edges touching v.

NOTE: When counting a loop, we count the loop twice (since there are two connections between that loop)

With a directed graph, we can distinguish between edges *coming from* a vertex and edges *pointing towards* a vertex



In-degree and out-degree (Def. 2.3): If G is a directed, then:

1. The **in-degree** is the numer of edges *pointing towards* v, denoted $deg^+(v)$

2. The **out-degree** is the number of edges *coming from* v, denoted $deg^-(v)$

NOTE: $deg^+(v) + deg^-(v) = deg(v)$

Every edge (as you can see from the diagram above) contributes twice to the degree of some vertex

Handshaking Lemma:

Sum of the degrees of all the vertices=2*(number of edges)

6/29 Lecture

Announcements:

Exam 2 Grades not out: Expect your letter grade (without corrections) later today
Back to back to back homework...unfortunately :(
Graph Theory HW 1 due tomorrow (11:59pm)
Graph Theory HW 2 due Wed.
Graph Theory HW 3 due Thurs.

Questions for Homework 1:

Previously:

Class 1-2:
-Defined what a graph was
-Talked about types of graphs: directed graphs, undirected graphs, weighted graphs, simple graphs and multigraphs

End of Class 2:
-Defined a degree
-Talked about the handshaking lemma

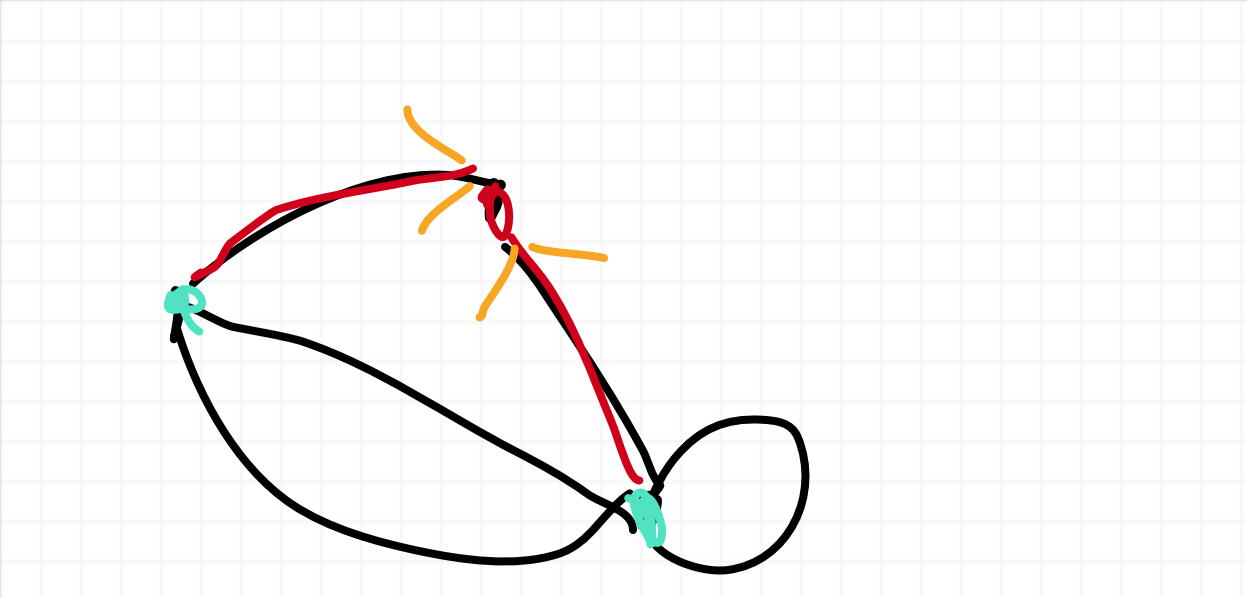
Class 2 (Cont.):

Handshaking Lemma (first version): (Thm. 2.7)

Sum of the degrees of all the vertices=2*(number of edges)

Intuitive Proof:

when we add the degrees all together, we count the edges twice.



QED

Handshaking Lemma (second version): (Thm. 2.8) Every finite undirected has an even number of vertices with odd degree.

Proof: Look at the first version of the handshaking lemma, which tells us the sum of all degrees is an even number.

Sum of all even degrees=even number 1

Sum of all odd degrees=number 2

The first handshaking lemma tells us that

Sum of all odd degrees+sum of all even degrees=Sum of all degrees=even number 3

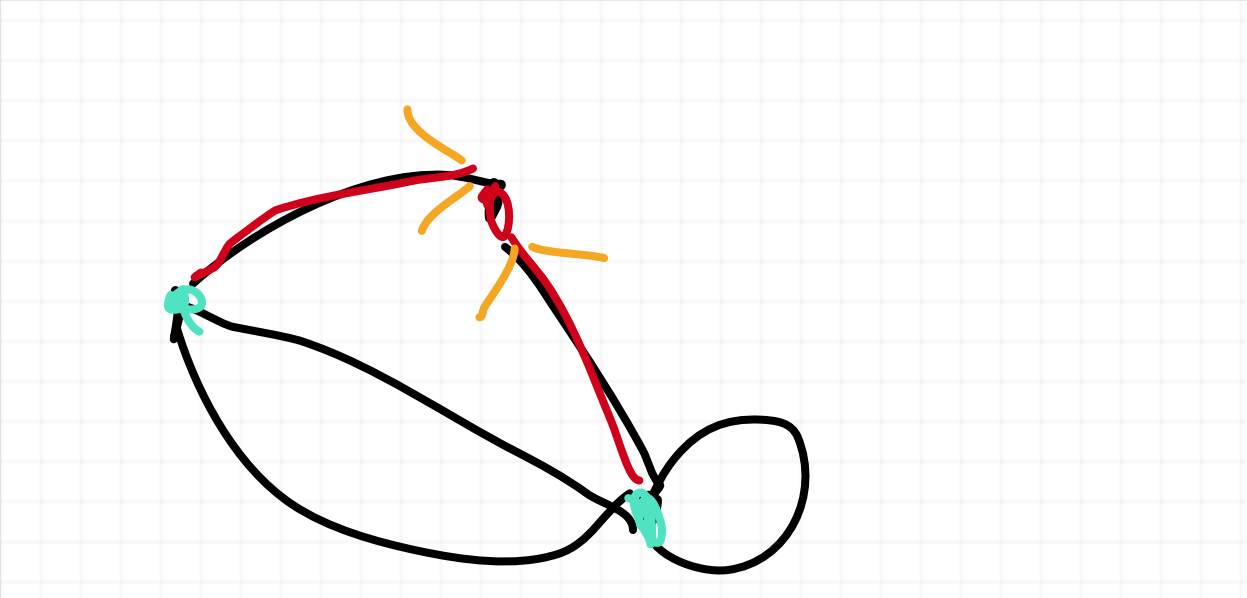
even number 1+number 2=even number 3

number 2 has to be even. QED

Handshaking Lemma (Third Version): (Thm. 2.6)
If G is a directed graph, then
sum of all in-degrees=sum of all out-degrees=the number of edges

Intuitive Proof:

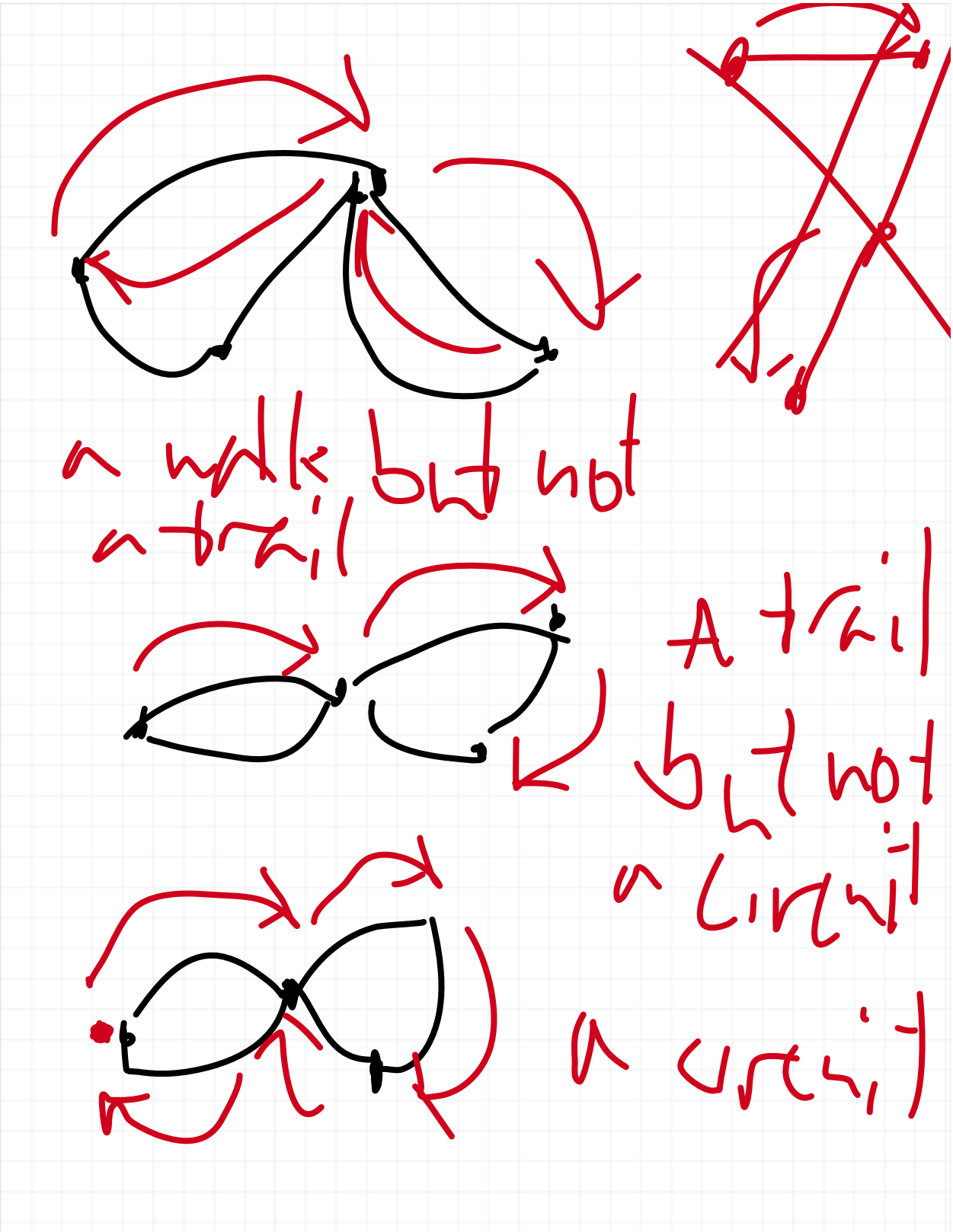
whenever we count an indegree for a vertex, we count an outdegree for a different vertex. (the proof the proof is similar to the proof of Thm. 2.7)



QED

Class 3: Walks, Trails, and Circuits

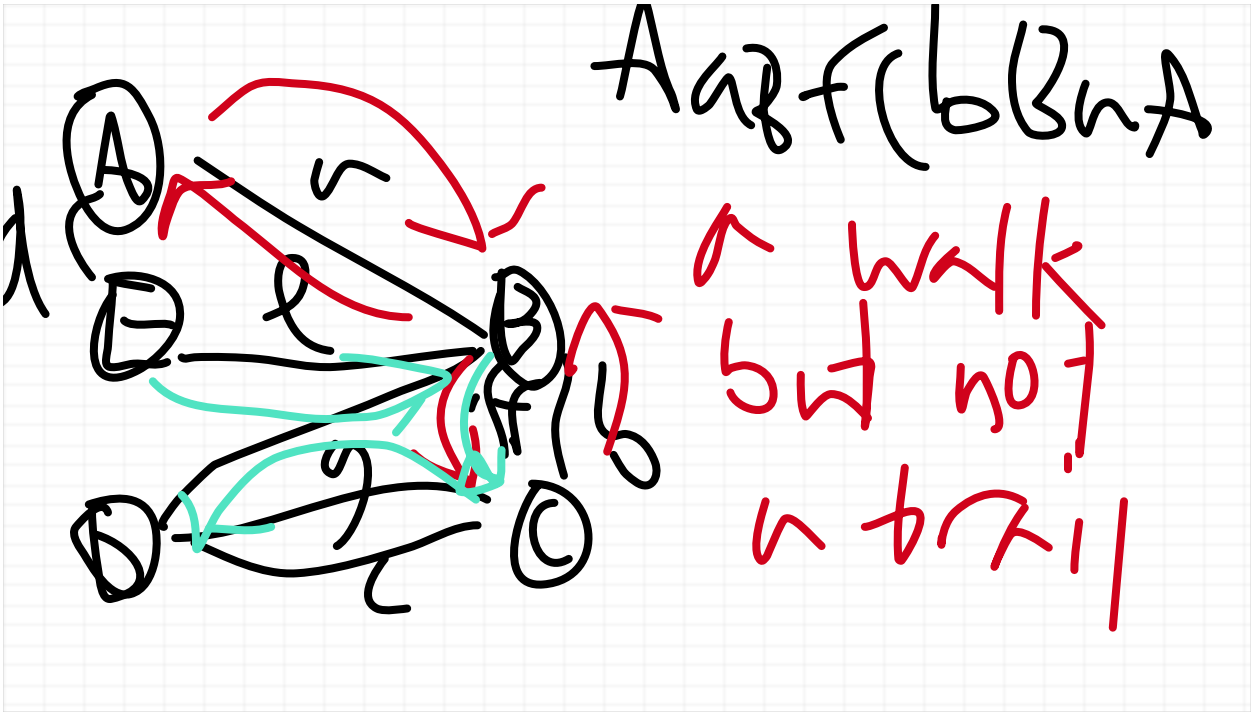
Walks, Trails, and Circuits (Def. 3.2):
A **walk** is a sequence of consecutive edges
A **trail** is a walk without repeated edges
A **circuit** (or a closed trail) is a trail which begins and ends at the same vertex.



NOTE: to articulate a walk, we write the sequence in terms of the vertices it crosses followed

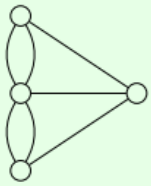
by the edges that it uses (and we label these); in the situation of simple graph, we only need to keep track of the vertices that it crosses (since each pair of vertices has at most one edge)

Ex. 3.4



EeBfCgD is a trail

Example 3.6 (Bridges of Konigsberg).
The bridges of Konigsberg problem is equivalent to finding an Eulerian trail in the following graph:



Bridges of Koinsberg problem (version 1, Thm. 3.1)
It is impossibe to walk across each of the 7 bridges of Koinsberg exactly once.

Eulerian Trails and Eulerian Circuits (Def. 3.5)
A trail for a circuit is **Eulerian** if it uses every edge
An **Eulerian circuit** is a circuit that is also eulerian
A graph containing an Eulerian circuit is called an **Eulerian Graph**

Bridges of Koinsberg problem (version 2, ex. 3.6)
The graph in ex. 3.6 does not contain an eulerian trail.

Proof of Thm 3.1: It suffices to show no Eulerian trail exists (version 2). We note the following is true:

1.

Class 4: Eulerian Graphs and Eulerization

6/30 Lecture

Announcements:

Exam 2 Grades: Uploaded, pending "extra credit"
ETA Wednesday night everyone should get corrections, as well as solutions, and updated
Graph theory homework 1-3 back to back to back

Questions for Homework 1-2:

Previously:

Elaborated on the Handshake Lemma
Defined a **walk**, which is a sequence of consecutive edges
Defined a **trail**, which is a walk with no repeated edges
Defined a **circuit**, which is a trail that begins and ends with the same vertex.

NOTE: A trail either has either a different begin and end vertex OR the vertex is the same

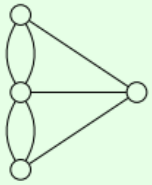
Defined a **Eulerian trail** (resp. circuit) is a trail (resp. circuit) that uses every edge.

Left off talking talking about the bridges of Koinsberg problem:

Class 3 (Cont.):

Bridges of Koinsberg problem (version 1, Thm. 3.1)
It is impossibe to walk across each of the 7 bridges of Koinsberg exactly once.

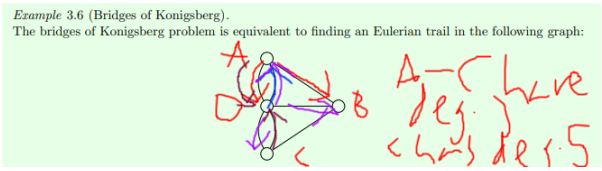
Example 3.6 (Bridges of Konigsberg).
The bridges of Konigsberg problem is equivalent to finding an Eulerian trail in the following graph:



Bridges of Koinsberg problem (version 2, ex. 3.6)
The graph in ex. 3.6 does not contain an eulerian trail.

Proof of Thm 3.1: It suffices to show no Eulerian trail exists (version 2). We note the following is true:

In order for an Eulerian trail to exist, the following has to happen:



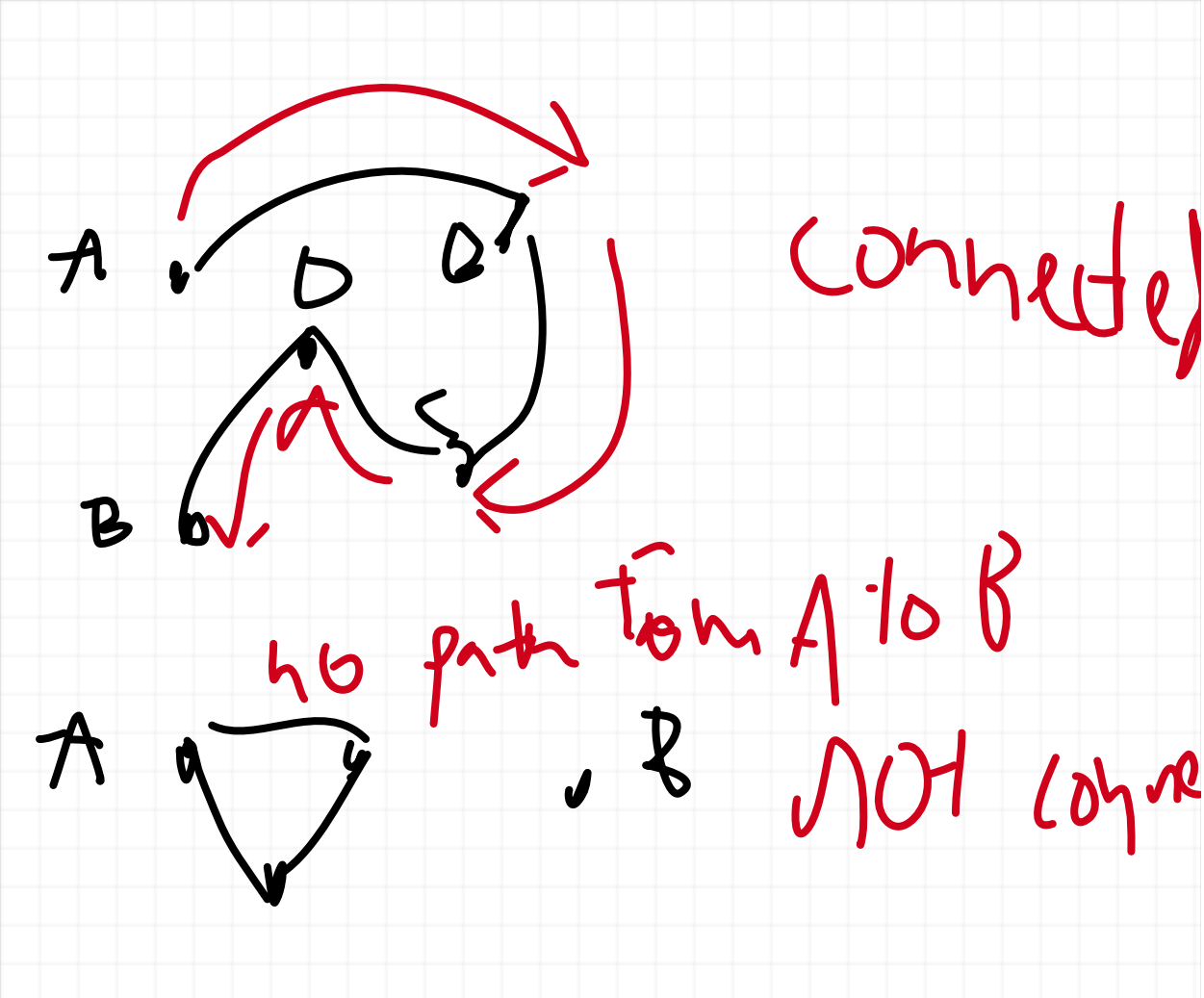
1. A-C has to be passed through at least twice
2. D has to passed through at least three times

Note that all those edges that pass through each vertex are *different;* so the Eulerian trail has to have at least 9 edges; however the bridges of koinsberg graph consists of seven. We conclude that no such trail is possible. QED

Class 4: Eulerian Graphs and Eulerization

There's even more efficient ways than trial and error types of ways in the Bridges of Koinsberg problem to figure out if a graph is Eulerian.

A graph is **connected** is there exists a trail between any two vertices.



Two vertices that have trail between them are said to be **part of the same connected component**.

NOTE: Every Eulerian graph is connected, and even more is true; in fact, there is a powerful classification Eulerian graphs.

Equivalent Characterization of Eulerian Trails and Circuits (Thm. 4.1)

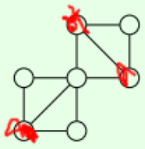
1. An equivalent of an Eulerian graph is one that has the following properties
   *connected
   *has at most two odd degree vertices

NOTE: The Handshake lemma rules out the possibility that there's only one odd vertex, so either an Eulerian graph has no odd degree vertices OR two odd degree vertices

2. A graph that has an Eulerian circuit necessarily is connected and has all even vertices.
3. Conversely, a connected graph that has all even vertices has an Eulerian circuit.

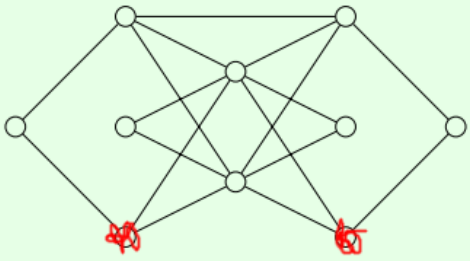
We're going to skip the proof; if you want to see, talk to me during office hours.

Example 4.2 (Eulerian Trails/Circuits).
Does the following graph have an Eulerian trail or an Eulerian circuit?



The degree sequence is $\{2, 2, 2, 3, 3, 3, 5\}$.
Thus the graph doesn't have an Eulerian trail or a Eulerian circuit.

Example 4.3 (Eulerian Trails/Circuits).
Does the following graph have an Eulerian trail or an Eulerian circuit?

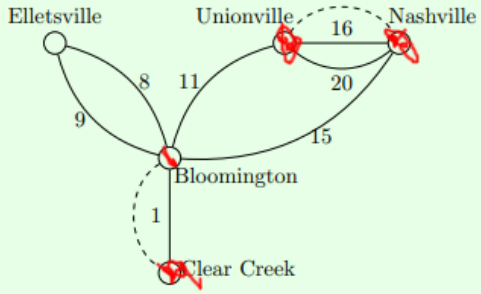


The degree sequence is $\{2, 2, 2, 2, 3, 3, 4, 4, 6, 6\}$.
Thus the graph has an Eulerian trail, but not an Eulerian circuit.

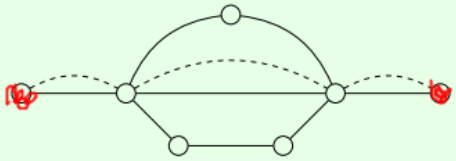
Eulerianization (Def. 4.6): An **Eulerization** is a process of turning non Eurlerian into Eulerian graphs by adding additional edges.

The notes give the following algorithm for Eulerianizing a graph:
1. Pair up the odd degree vertices
2. Choose paths between the pairs
3. Add duplicate edges to the edges along the paths

Example 4.7 (Eulerization - Local Towns (weighted graph)).



Example 4.8 (Eulerization).
Eulerize the following graph by adding as few duplicate edges as possible.



For homework 2, here's an algorithm to Eulerize the graph in a way that adds the *fewest* edges:
1. Pair up the odd degree vertices *closest together* in terms of *edge distance* away
2. Choose the *shortest* paths between those pairs
3. Add duplicate edges to the edges along the paths.

Next Time: We'll finish Eulerization, and we'll talk about class 5-6 stuff (equal and isomorphic graphs).

Class 5-6: Equal and Isomorphic Graphs

7/1 Lecture

Announcements:

Updated exam and grades and corrections and solution ETA: around 6pm tonight
Graph theory homework 1 grade ETA no later than 8-ish tonight
Graph theory homework 2 grade ETA tomorrow around afternoon
Graph theory homework 4 uploaded (due Monday 11:59pm)
Exam 3 Tuesday (usual time)
Yes we do have class Friday (the 3rd)

Questions for Homework 2-3:

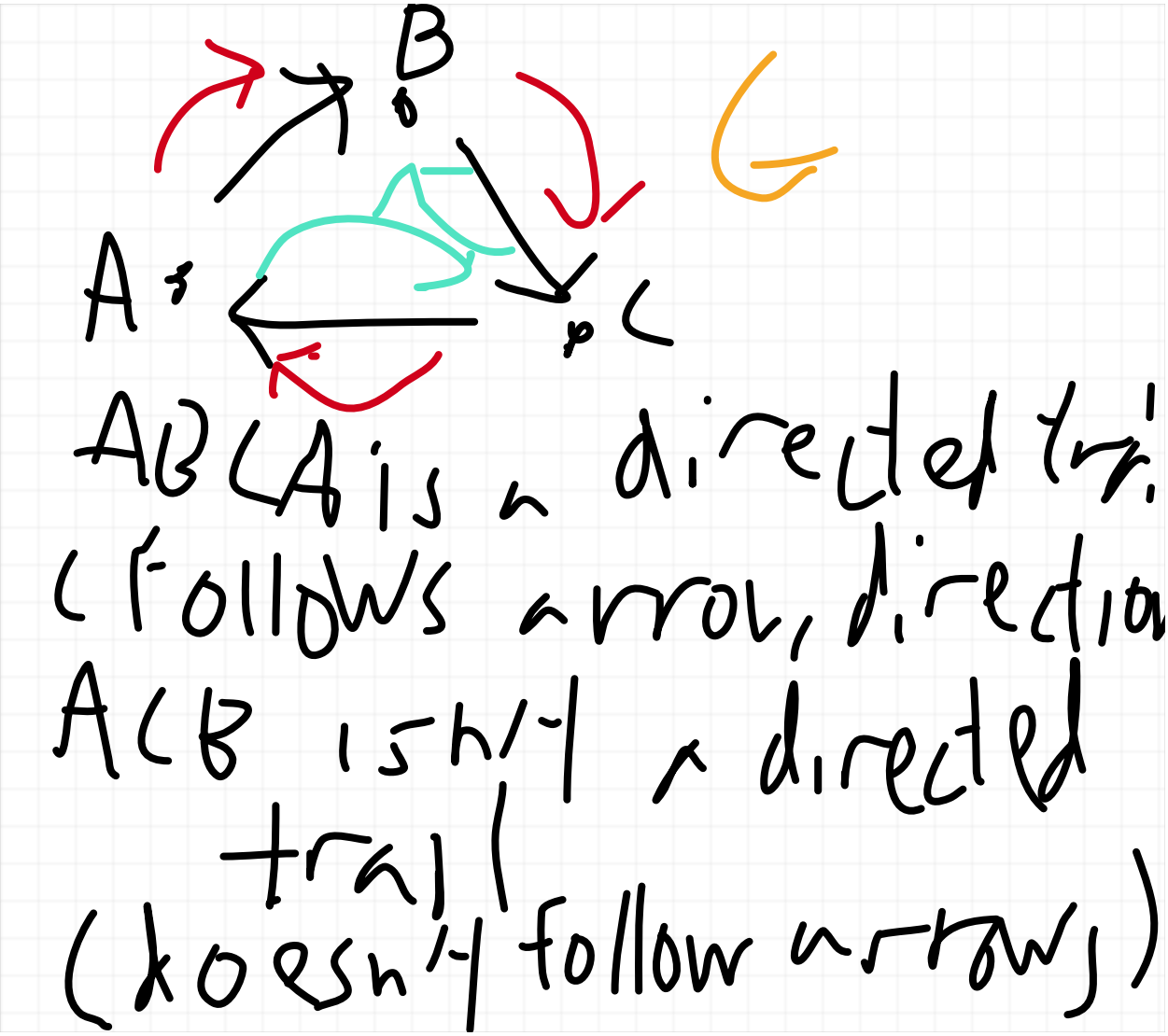
Previously:

-Reframed the Bridges of Koinsberg as an Eulerian problem and proved that the absence of an Eulerian graph
-Provided an equivalent definition of an Eulerian and a graph containing an Eulerian circuit that is easy to verify. (Thm. 4.1)
-Discussed gave an algorithm to Eulerize the graph in way that adds the *fewest* edges

Class 4 (Cont.):

Directed Euler Graphs

A **directed trail** (resp. **directed circuit**) in a *directed graph* is a trail (resp. circuit) that goes through all its edges in a way that follows the direction of the edges



A **directed Eulerian trails** is a directed trail that is also Eulerian as a trail (in the usual sense).

Directed Eulerian circuits and **Directed Eulerian graphs** are defined analogously

The graph G drawn above is also a directed Eulerian graph (and contains a directed Eulerian circuit)

Equivalent of directed Eulerian graphs (Thm. 4.4):
An equivalent of a directed Eulerian graph is one that has the following properties:
1. The graph is connected (if a trail exists between any two vertices)
2. Every vertex v has $deg^+(v) = deg^-(v)$

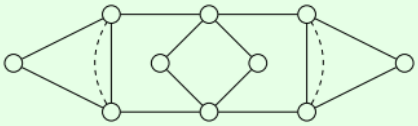
More on Eulerization

The **length** of a trail is the amount of edges that a trail has.
The **distance** between two edges is the length of the shortest trail

In Ex. 4.9, the the dotted trails given have length 1 (since it travels one edge)

For homework 2, here's an algorithm to Eulerize the graph in a way that adds the *fewest* edges:
1. Pair up the odd degree vertices *closest together* in terms of *distance* away (as defined above)
2. Choose the *shortest* paths between those pairs
3. Add duplicate edges to the edges along the paths.

Example 4.9 (Eulerization).
Eulerize the following graph by adding as few duplicate edges as possible.

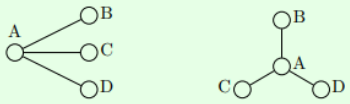


Class 5-6: Equal and Isomorphic Graphs

A **labeling** is where you lable the vertices. A **labeled graph** is with the vertices labled.

Equivalent Graphs (Def. 5.1) We say that two (labeled graphs) are **equal** if they have the same set of vertices and the same set of edges with respect to the labeling

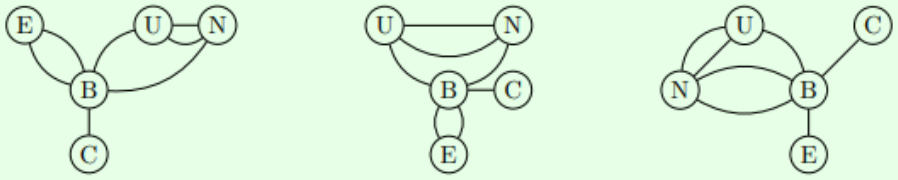
Example 5.2 (Equal Graphs).
The following two graphs are equal.



It is clear that they have the same vertex set (i.e. $\{A, B, C, D\}$), and the same edge set.

Vertex sets are the same; both graphs contain the following edges: $AB,\ AC,\ AD$

Example 5.4 (Equal Graphs).
The first and second graphs are equal, but the third is not.



First note that in the first two graphs we have that $\deg(E) = 2$, but in the third $\deg(E) = 1$. Thus the third graph is not equal to the first or the second.

We now show that the first and the second graphs are equal.

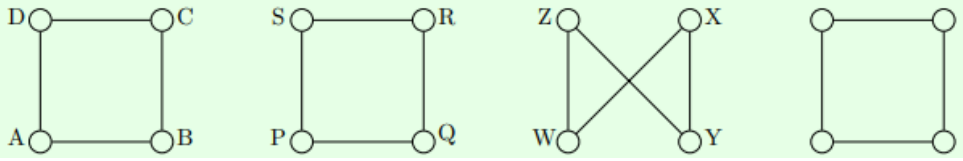
Vertex	Edges (First Graph)	Edges (Second Graph)
E	B, B	B, B
U	B, N, N	B, N, N
N	U, U, B	U, U, B
B	E, E, U, N, C	E, E, U, N, C
C	B	B

Isomorphic Graphs (Def. 5.5): We say that two graphs are **isomorphic** if there is a labeling or relabeling of vertices which make them equal.

NOTE: Graph Isomorphisms don't originally have to be labeled (equality does), we can assign the labeling

to show that two graphs are isomorphic, you must:
1. find a specific labeling/relabling of the graphs that works
2. Show that with this new labeling, the graphs are equal.

Example 5.6 (Isomorphic Graphs).
All of these graphs are isomorphic:



For the first two graphs, we can relable S, R, P, Q as E, C, A, B (respectively) and that gives us an isomorphism.

For the third graph, relable Z, X, W, Y as D,B,A,C

Z, Y, X, W matches D, C, B, A

For the fourth graph, we can lable it Z, Y, X, W, and it's isomorphic to the third.

Next Class: Give some structural properties isomorphisms that will help with Homework 3.

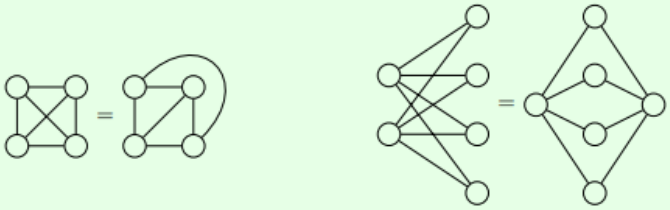
Class 7: Planar Graphs

Also on Homework 3.

Planar Graphs (Def. 7.1): A **planar graph** is a graph can be drawn with no edge crossings.

More specifically, a planar graph is one that can be equivalently/isomorphically drawn with no edge crossings.

Example 7.2 (Planar Graphs).
The following graphs are planar.



7/2 Lecture

Announcements:

Updated Exam grades, corrections, and solutions uploaded
Graph Theory Hoemwork 4-5 uploaded and BOTH due Monday (Extend Office hours)
Exam 3 Tuesday

Questions for Homework 3:

Previously:

-Finish the last of class 4 (finally!)
-Defined equivalent and isomorphic graphs (class 5):
Two **equivalent graphs** are labeled graphs that have the same vertex and edge sets

Two **isomorphic graphs** are graphs that can be labeled/relabled so that they are equal.
-Began to define planar graphs (class 7):
A **planar graph** is a graph that can be drawn no edge crossing

Class 5-6 (Cont.):

-last lecture we talked a lot when a graph is isomorphic, but how do we tell if two graphs aren't isomorphic?

-We find a structural properties that are different, i.e., structural inconsistencies

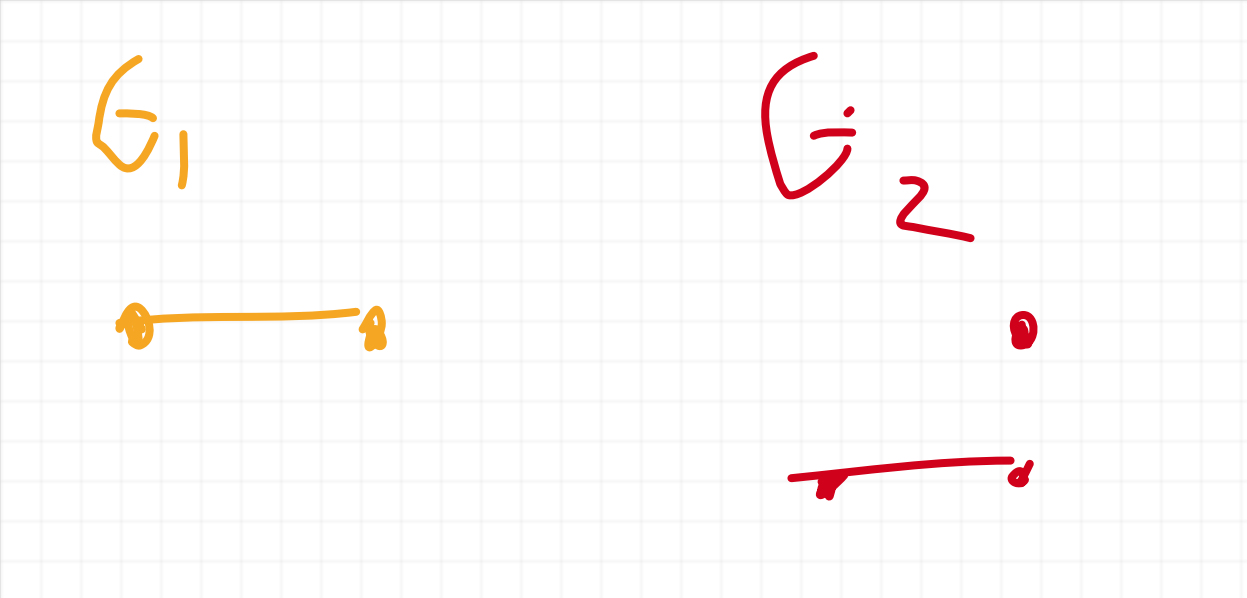
Structural properties of isomorphic graphs (Thm. 5.11):
Two isomorphic graphs have the following properties:
1. number of vertices are the same
2. number of edges are the same
3. The degree sequence is the same
4. number of connected components are the same

Additionally (not in the notes), we have:
5. Existence subgraphs isomorphic to each other

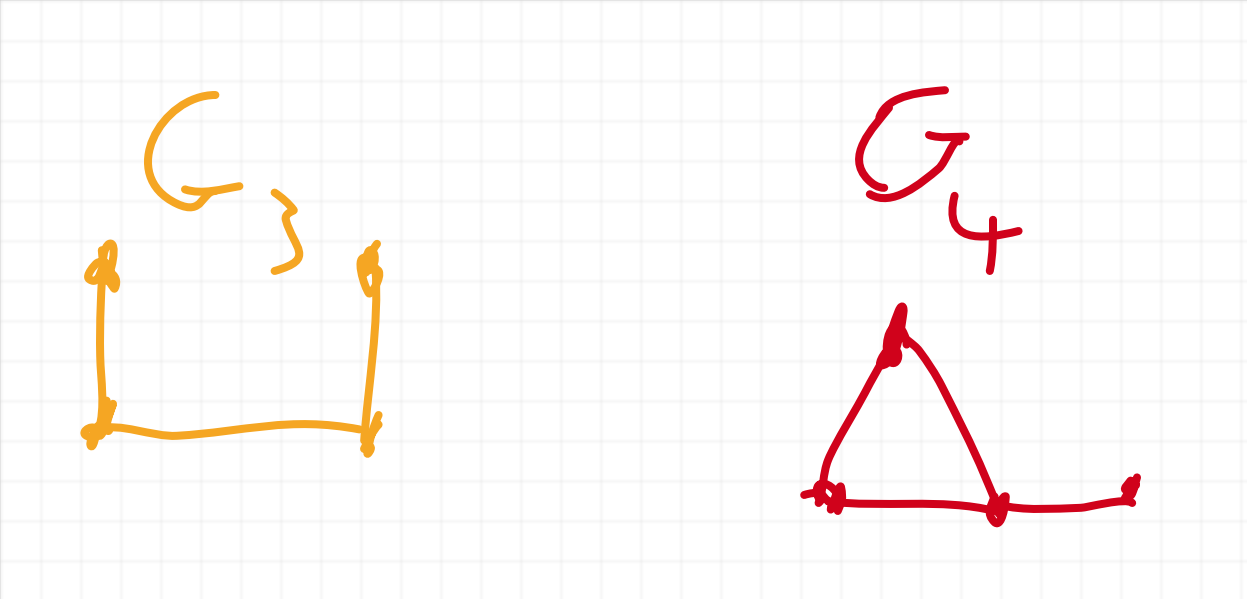
Corollary to Theorem 5.11: Any two grahs that have any of the above structural properties *different* are *not* isomorphic.

NOTE: Not everything as to how graph is drawn is necessarily preserved between isomorphisms (for example, edge crossing)

Another NOTE: We haven't really "connected components" and we haven't "subgraph" (we will a bit later)

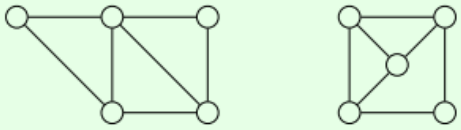


with G_1 and G_2 they have a different number of vertices, so they're not isomorphic



Note that here, the vertices are the same for G_3 and G_4 but the number edges are different, so they're also not isomorphic.

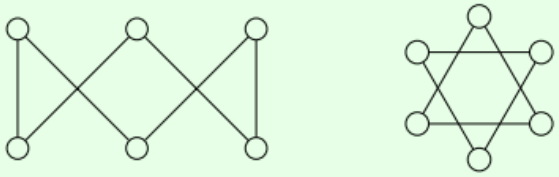
Example 5.12 (Non-Isomorphic Graphs - (Degree Sequence)).
These two graphs are not isomorphic.



Degree sequence of the left graph: $\{2, 2, 3, 3, 4\}$
Degree sequence of the right graph: $\{2, 3, 3, 3, 3\}$

Note that the edges and vertex numbers are the same, but the structural difference of example 5.12 is the degree sequence.

Example 5.14 (Non-Isomorphic Graphs - (Connected Components)).
These two graphs are not isomorphic.

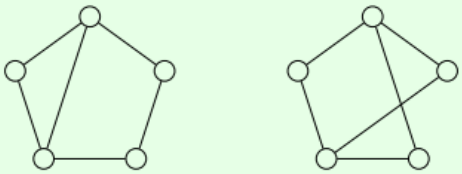


Degree sequence of the left graph: $\{2, 2, 2, 2, 2, 2\}$
Degree sequence of the right graph: $\{2, 2, 2, 2, 2, 2\}$
Both graphs have the same degree sequences, but that isn't sufficient to conclude that they are equal.

Note that the left graph is connected, while the right graph has two connected components.

Note that in example 5.14, the number of edges, vertices, and degree sequences are the same, but they're not isomorphic, and that's to do with the number of connected components being different (we'll define more on that soon)

Example 6.1 (Non-Isomorphic Graphs - (3-cycle)).
The following two graphs are not isomorphic.



One way to see this is to note that the left graph has a triangle (a cycle of length 3), while the right graph doesn't.

We have the same vertices, edges, degree sequence, and even the same "level of connectedness", but there's a difference in the graph to the left having a circuit length 3, but the graph to the right not.

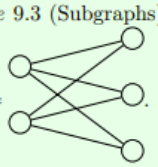
The difference (more generally) is if that one has a subgraph not isorphic to any other subgraph of the the other graph, then the graphs aren't isomorphic.

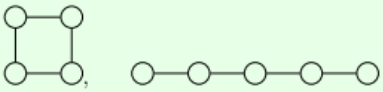
Class 7-9: Special Types of Graphs

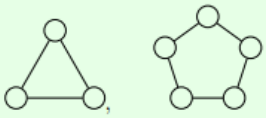
Subgraphs (Def. 9.1): A graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if V' is a subset of V and E' is a subset of E.

Intuitively, a subgraph is graph "inside of another graph"

Example 9.3 (Subgraphs).

Let $G =$  .

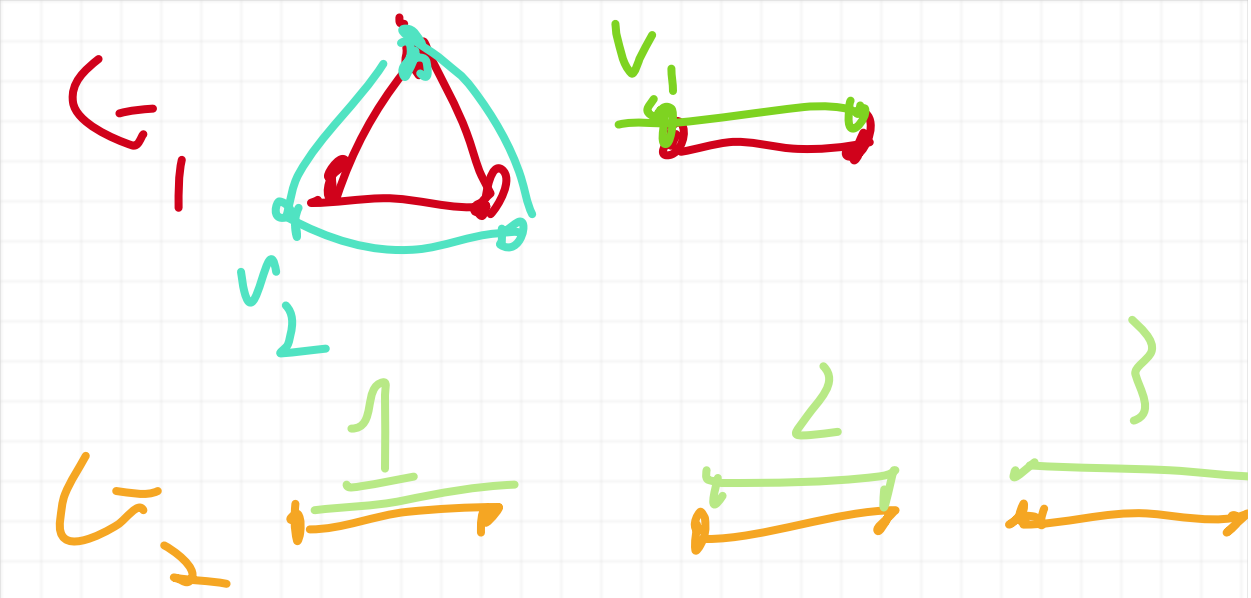
The the following are subgraphs of G:  ,

The following are NOT subgraphs of G:  ,

Now we can make sense of Example 6.1 (above)

Recall the definition of a connected graph:
A graph is **connected** when any two vertices have a trail

A subgraph is said to be a **(connected) component** of a graph if it consists of every vertex that has a trail with a specific vertex v

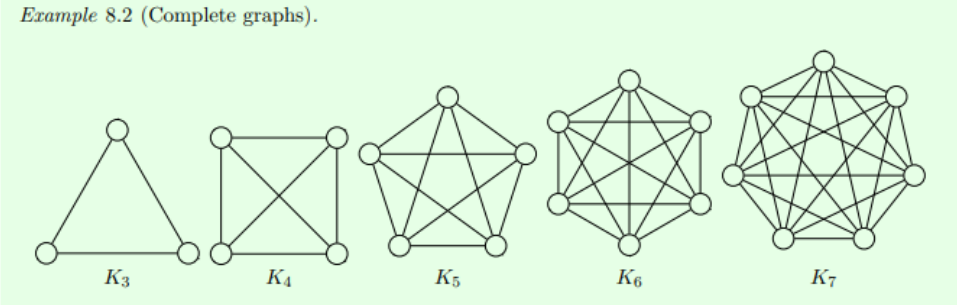


NOTE: In general, a connected graph has one component (every vertex by definition has a trail between each other)
G_1 has two connected components
G_2 has three connected components

The stuctural difference in example 5.14 is that they have different connected components.

Special Types of Graphs (Def. 8.1, 8.3, and 8.5):

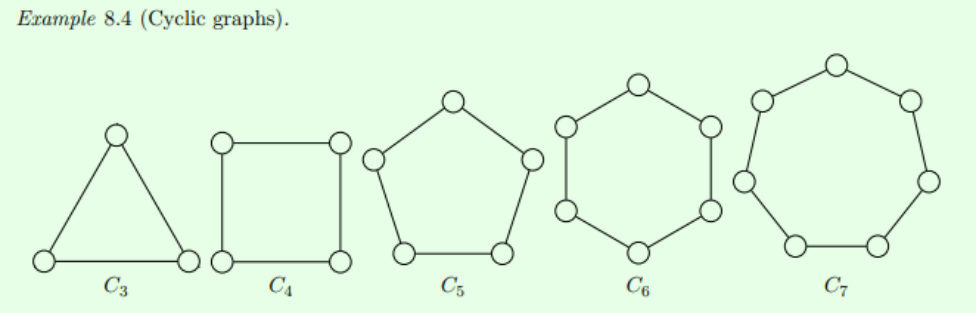
A **complete graph** is a graph where every pair of vertices is connected by an edge; we denote a complete on n-vertices by K_n



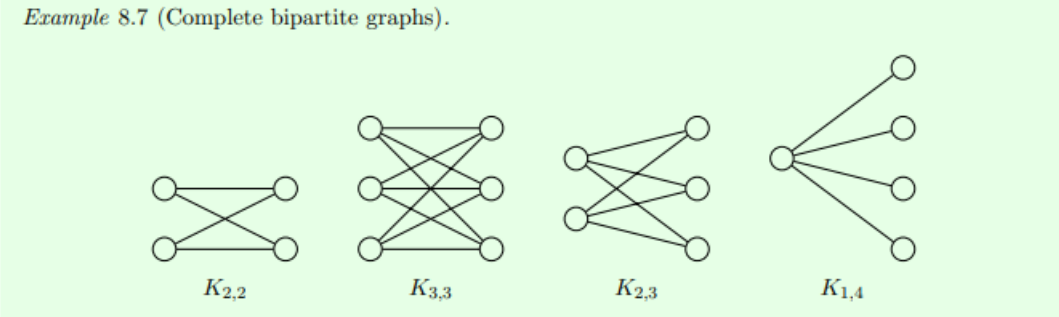
A **cyclic graph** is a graph which consists of exactly one circuit with no repeating vertices, or a "cycle"

NOTE: cyclic graphs are an example of a Euler graph

We denote the ciyclic on n-vertices by C_n



A graph G is called **bipartite** if its vertices can be partitioned into two disjoint sets, so that each edge of G has one end in one set, and the other end in another set
(bipartitite graph is equivalently defined if it has a two coloring)
A **complete bipartite graph** is a bipartite graph where every vertex in one partition of the partition sets has an edge with every vertex in the other partition set, and vice versa



It's denoted $K_{n,m}$, when the graph has a n vertex partition with a m an vertex partition

HINT on Homework 3:

NOTE: K_5 and $K_{3,3}$ are both nonplanar (we prove K_5 in the notes)

Any strict subgraph of K_5 and $K_{3,3}$ that has less edges is planar, so in other words, the bottom two graphs in question 4 of homework are planar for this reason.

I'll give a more elaborate on a canvas announcement later today.

7/6 Lecture

Announcements:

Homework 4 and 5 due tomorrow
Exam 3 Wednesday
ADDITIONAL OFFICE HOURS: Teus 2pm-3pm (usual office hours today at that time)
Homework 1-3 solutions (tomorrow afternoon at the latest)
Homework 4-5 grades (I can get 4-5 grades early with corrections)

Lecture Plan (for the next 4 days):
-Monday (7/6):
  -Planar graph concepts (faces) (class 7)
  -tree graphs (class 9)
  -colourings (class 10)
  -Hamiltonian Graphs & Circuits (class 12)
  -office hours (2pm-3pm)

-Tues (7/7):
  -Finish Hamiltonian Graphs (class 12-13)
  -Finish on colourings I didn't cover (class 10-11)
  -Review for the exam (talk about what to expect and what not to expect)
  -office hours (2pm-3pm)

-Wed (7/8):
  -Exam 3

-Thurs (7/9)
 -Voting Theory (expect lectures uploaded soon)

Questions for Homework 4-5:

Previously:

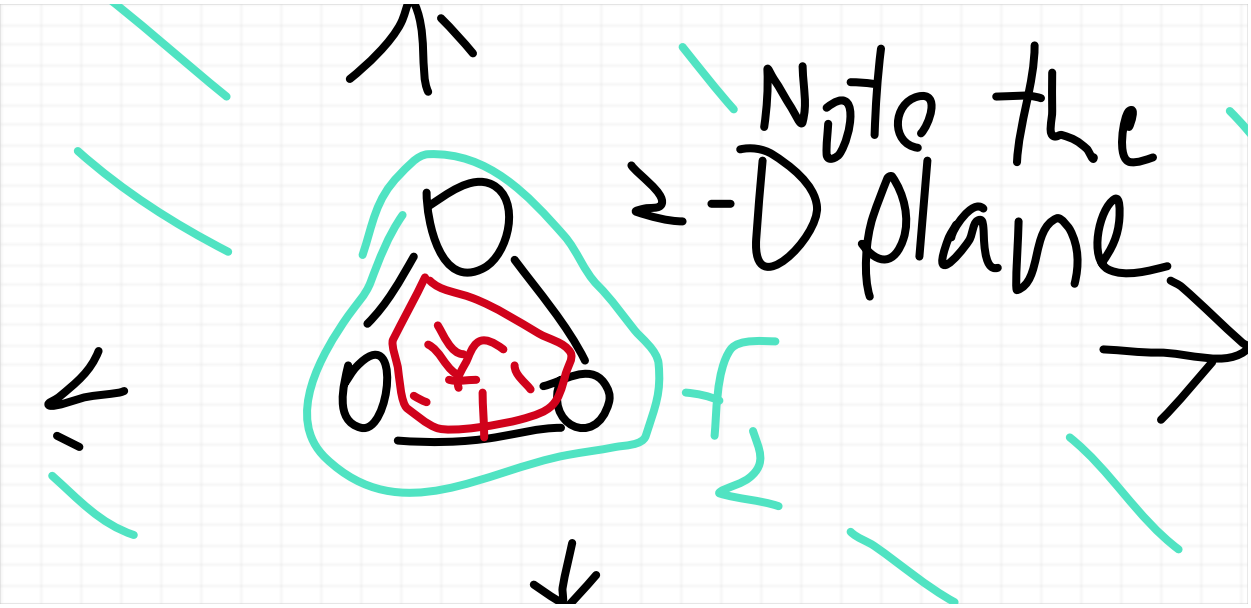
-Talked about identifying nonisomorphic graphs.
-Talk about **subgraph**

-Talked about special types of graphs.
  -**complete graphs**: K_n
  -**cyclic graphs**: C_n
  -**bipartite graphs**
    *__complete bipartite graphs__: $K_{n,m}$

Class 7-9 (Cont.):

Finish up Planar Graphs (Faces):

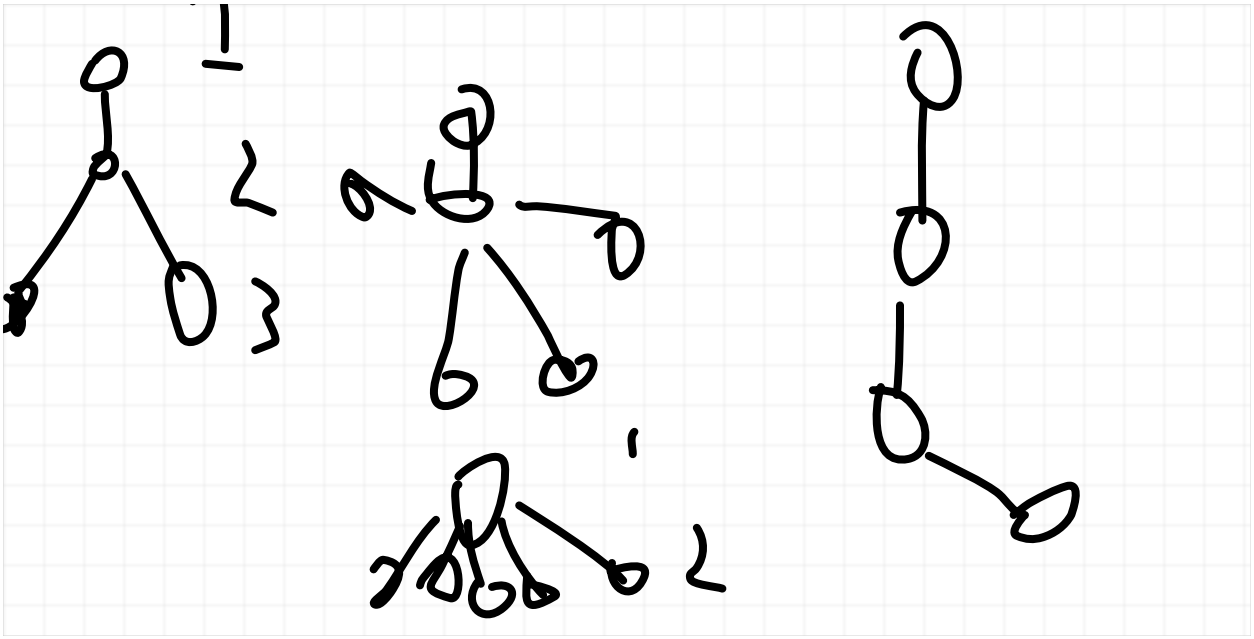
Planar graphs have the following features (as mentioned in Def. 7.3):
vertices (as before); we denote the number of vertices by v
edges (as before); we denote the number of edges by e
faces are the regions that are created by the boundaries established by the edges of a planar graph (we denote f as the number of faces)



NOTE:
1. When identifying faces, don't forget the **outer face**, which is the "outside region" of the graph that extends infinitely. (as a result of this outer face, every planar graph contains at least 1 face)
2. Faces around regions are meaningless when a graph is drawn with crossings
3. Faces are an additional feature of a graphs preserved under isomorphism (provided the two isomorphic graphs are both drawn without crossings)

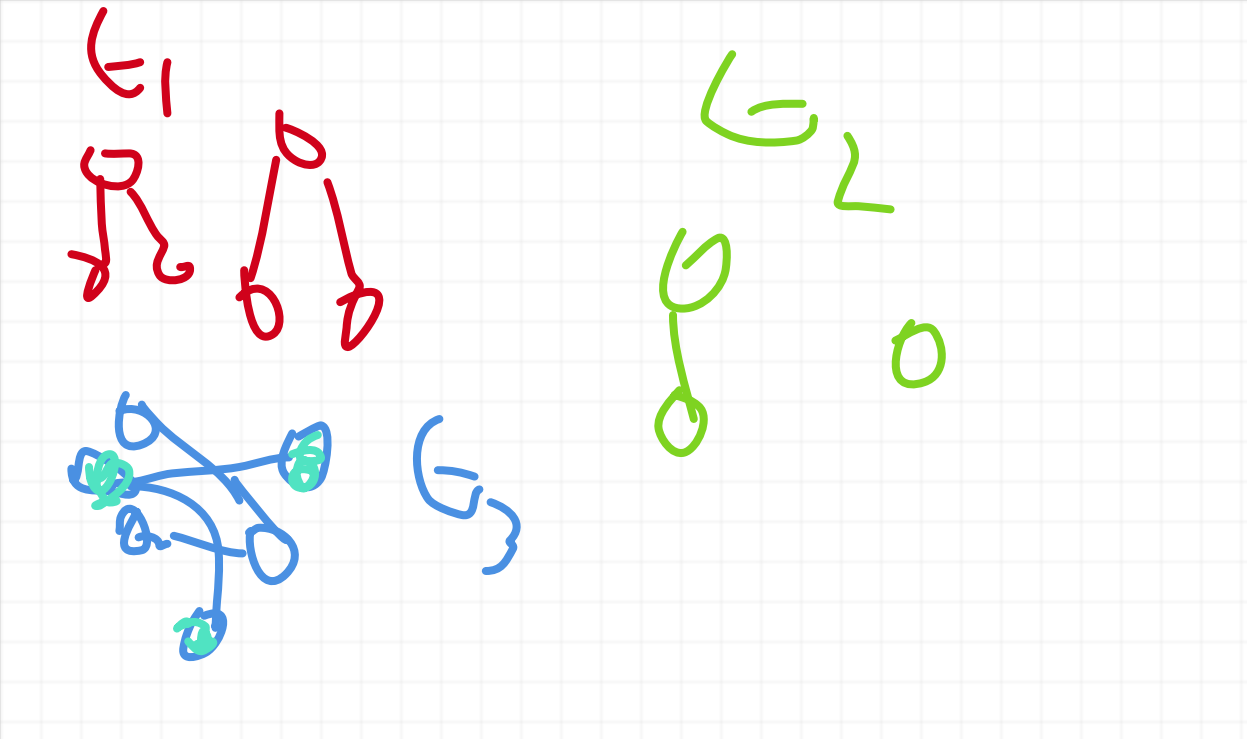
Tree Graphs (Class 9):

Tree Graph (Def. 9.5)**:** A **tree** (or a **tree graph**) is a connected graph with no cyclic subgraph.

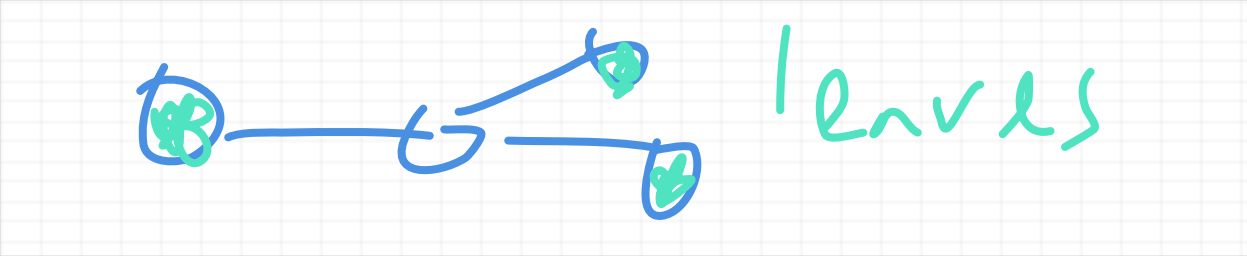


Auxillery Definitions (Def. 9.7, 9.8, 12.3):

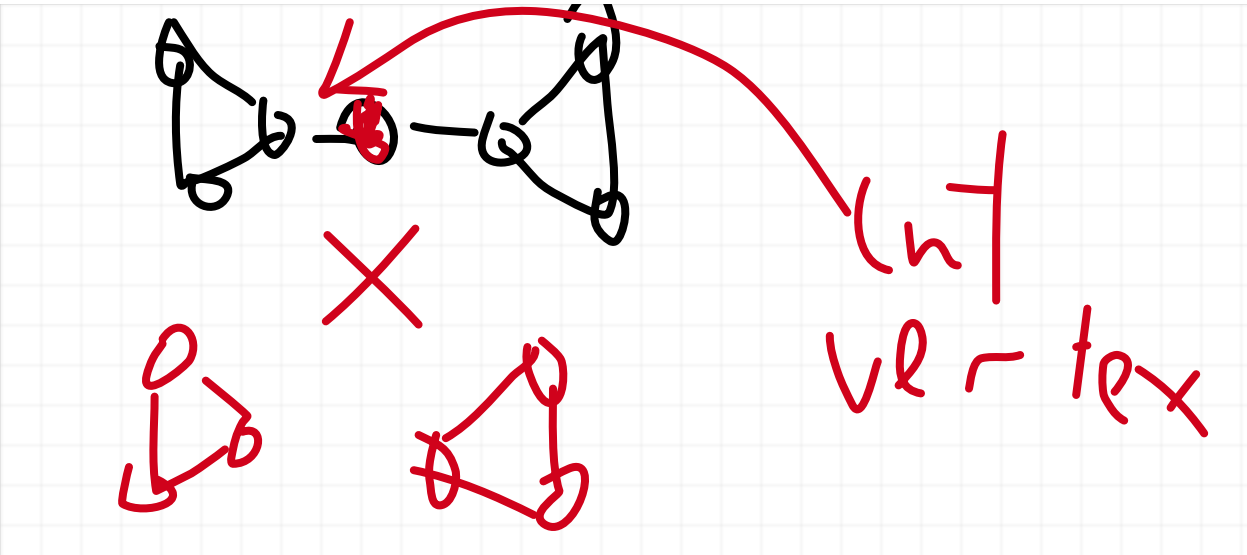
-A **forest** is a graph that consists of a collection of trees (as components)



-A **leaf** a vertex of degree 1



-A **cut vertex** is a vertex such that if deleted (along with the edges joining) causes the number of connected graphs to increase
(in other words it separates a component into two components)



Equivalent Definitions of a tree graph:

-G is acyclic and a cylce is formed if any new edge is added
-G is connected, but would become disconnected if a single edge is removed from G
-G is connected and every nonleaf vertex (any vertex with degree 2 or more) is a cut vertex
-Any two vertices have a trail and a *unique* trail.
-G is connected and has $n-1$ edges (hint for HW 4 2d)
-G has no simple cycles and has $n-1$ edges.

Class 10-11: Graph Colouring

Colouring (overview):

(Def 10.1 and 10.5):

-A **proper colourong** of a graph G is an assignment of colours to the vertices so that no two vertices share the same colour
-We call a proper colouring with k colours a k-**colouring**
-The **chromatic number** (often denoted $k(G)$ for a graph G) is the minimum number of graphs necessary to colour the graph

Class 12-13: Hamiltonian Circuits

7/7 Lecture

Announcements:

Homework 2 graded (Homework 3 ETA this afternoon for real); Homework 1-2 solutions are up, homework 3 solution ETA this afternoon
EXTRA Office hours TODAY 2pm-3pm (and by appointment)
Exam 3 tomorrow
Voting Theory lectures are up

Questions for Homework 4-5:

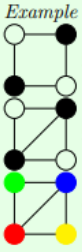
Previously:

-Talked about **faces**
-Talked about **tree graphs** and their properties
-Left off at colouring

Class 10-11: Graph Colouring

-(Def. 10.1) A **proper colourong** of a graph G is an assignment of colours to the vertices so that no two vertices share the same colour

Example 10.2.



is a proper colouring.

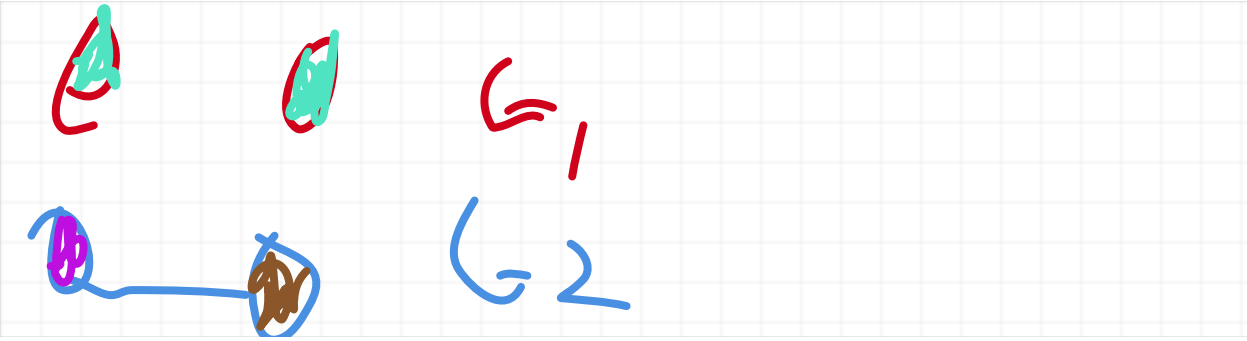
is not a proper colouring.

is a proper colouring.

-(Def. 10.1) We call a proper colouring with k colours a k-**colouring**

Bipartite graphs can defined much easier in terms of colorings

An equivalent of a bipartite graph is one that is 2-colourable



G_1 is 1-colorable (and hence bipartite)
G_2 is bipartite as well

-(Def. 10.5) The **chromatic number** (often denoted $k(G)$ for a graph G) is the minimum number of colours necessary to colour the graph

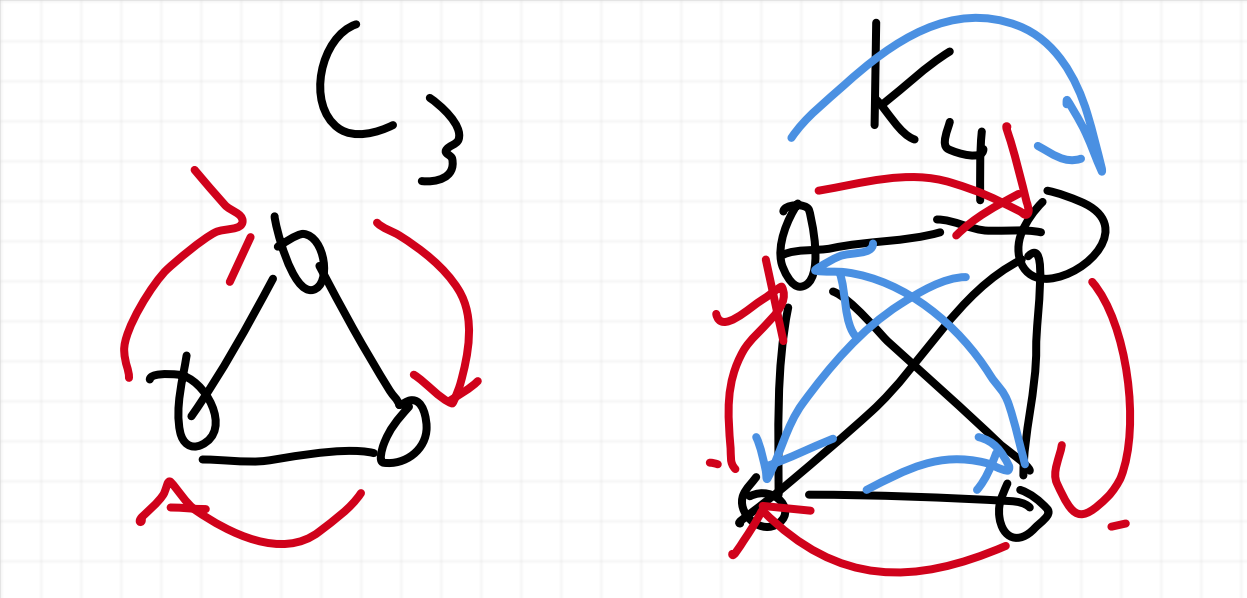
NOTE:
1. Chromatic is preserved under isomorphism (so if there's two graphs with different chromatic numbers, then they're not isomorphic).
2. Bipartite graphs have chromatic 1 OR 2.

K_n has chromatic number n (an equivalent of a complete is a simple graph with n vertices and chromatic number n
C_n, for an even number n, tree graphs, and $K_{n,m}$ have chromatic number 2.
The chomatic number of *any planar graph* has chromatic number less than or equal to 4. (the famous 4-colour theorem, which stated in Thm. 4.11)

Greedy Colouring Algorithm

Class 12-13: Hamiltonian Circuits

(Def 12.1)
A **Hamiltonian circuit** is a circuit which visits every vertex EXACTLY ONCE
A **Hamiltonian graph** is a graph which has a hamiltonian circuit



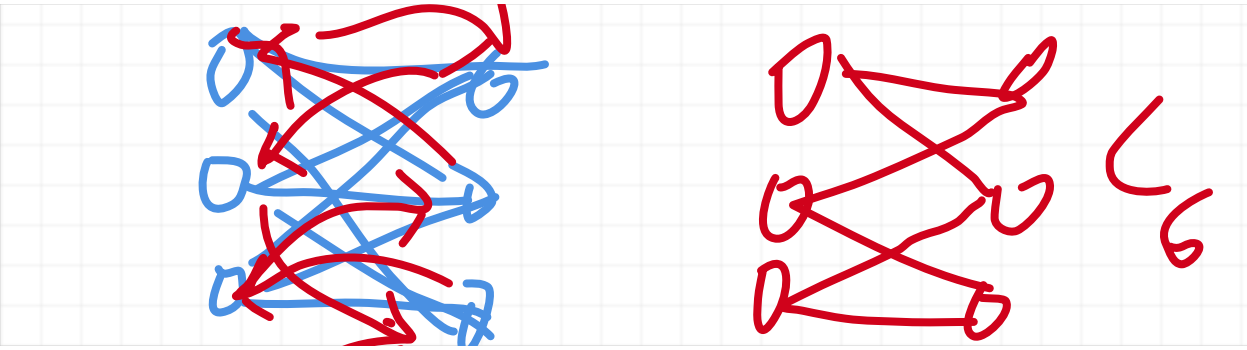
NOTE: Unlike Eulerian graphs, there's no easy "equivalent" definition by looking at the degrees

Examples of Hamiltonian Graphs:

1. C_n is a hamiltonian graph (in fact, an equivalent definition of Hamiltonian graph is a simple graph with n vertices that contains C_n)

2. K_n is Hamiltonian (because is contains C_n)

3. $K_{n,n}$ is Hamiltonian



C_{2n} ends up in $K_{n,n}$.

Examples of non-Hamiltonina Graphs:

-Tree graphs (contain no cycles at all!)
-non-connected graphs (since Hamiltonian are necessarily connected)

To find if a graph is hamiltonian, you want to look for a circuit around every vertex that does so only once (find the C_n that is inside any graph with n vertices)

To find if a graph is non-hamiltonian, you want a structural that we know a Hamiltonian graph doesn't contain.

Possible Properties of Non-Hamiltonian (Thm. 12.5):
-Any graph with a cut vertex (defined in Class 12)
-Bipartite with different size partition sets ($K_{n,m}$ for $n \neq m$ is not Hamiltonian for instance)
-Any graph with a leaf (a degree one vertex)

ANSWER HOMEWORK 5 QUESTION 5