# ~~Ehs~~, Exam II Study Guide

## Chs. 5, 12, 13

Side Note: nothing on interval graphs, intersection graphs

**Basic Terminology:**
- graph; vertex; edge
- degree; adjacency
- walk; path; cycle; trail; circuit
- isomorphism
- connectedness; connected component

**Subgraphs:**
- what are they?
- induced subgraphs; spanning subgraphs

**Special Types of Graphs**
- complete; independent
- bipartite; complete bipartite
- tree; forest

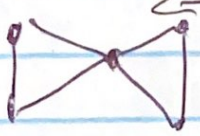
**Eulerian Graphs:**
- definition; Eulerian ~~trail~~ circuit
- how do we tell if a graph is Eulerian?

Thm: Eul. $\Leftrightarrow$ connected & everybody has even degree

**Hamiltonian Graphs:**
- definition; Hamiltonian cycle
- how do we tell if a graph is Hamiltonian?
  - to show Hamiltonian: ~~del~~ construct a Hamiltonian cycle
  - to show not Hamiltonian: show there's a bottleneck

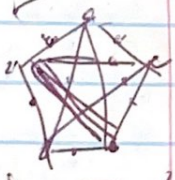
Alternative; use Thm about # of neighbors, if applicable

e.g.



## Planar Graphs

- definition
- how to tell if a graph is planar:
  - might not work →
    - to show planar: draw it with no edge crossings
    - to show non-planar: ~~formula~~ inequality $m \leq 3n-6$
  - only surefire ways, Kuratowski's Theorem
    - elementary subdivision; homeomorphism
    - planar $\iff$ no subgraph homeomorphic to $K_{3,3}$ or $K_5$



Strategy:

look for 5 vertices of degree 4 (or more)

look for homeomorphic to $K_5$

look for 6 vertices of degree 3 or more for homeomorphic to $K_{3,3}$

## Coloring:

- (proper) k-coloring; chromatic number $\chi(G)$
- how to find $\chi(G)$?
  - finding an upper bound: find a proper k-coloring → $\chi(G) \leq k$
  - finding a lower bound:
    - $\chi(G) = 2 \iff$ no odd cycles
    - clique number $\omega(G) \leq \chi(G)$

## Prüfer Codes:

- what is the correspondence? labeled trees, on n vertices ↔ strings on n letters of length n-2
  - how to create the Prüfer code of a tree
  - how to reconstruct the labeled tree from the code

## Isomorphic Graphs

- definitions: isomorphism, isomorphic graphs
- how to check if two graphs are isomorphic?
  - to show isomorphic: construct an isomorphism
  - to show non-isomorphic: show they differ on a structural issue, e.g. # of vertices; # of edges, # of cycles/paths of a certain length; # of connected components; # of vertices of a certain degree, ...

## Finding a Minimum Weight Spanning Tree:

- weighted graph; spanning tree; (min. weight spanning tree)
- Kruskal's Algorithm; Prim's Algorithm } ← remember to show your work

## Finding a Shortest Path

- directed edge; directed graph
  - how this affected walks
- definition of shortest path
- Dijkstra's Algorithm } ← remember to show your work

## Network Flows

- network; source; sink
- network flow; value of a flow
- finding the max flow
  - step of Ford-Fulkerson Algorithm } ← remember to show your work