# Elements of Graph Theory

**Quick review of Chapters 9.1… 9.5, 9.7 (studied in Mt1348/2008) = all basic concepts must be known**

**New topics**

- we will mostly skip shortest paths (Chapter 9.6), as that was covered in Data Structures
- Graph colouring (Chapter 9.8)
- Trees (Chapter 3.1, 3.2)

# Applications of Graphs

Applications of Graphs: Potentially anything (graphs can represent relations, relations can describe the extension of any predicate).

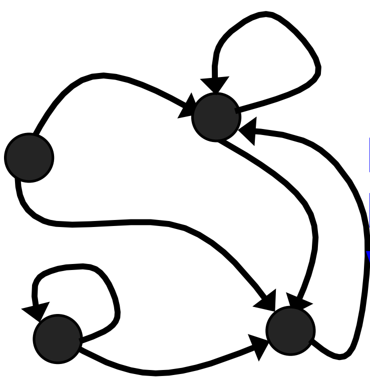
Applications in networking, scheduling, flow optimization, circuit design, path planning.

More applications: Geneology analysis, computer game -playing, program compilation, object-oriented design, …

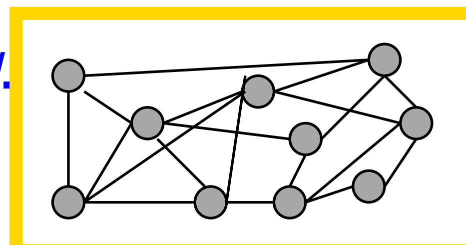
# Simple Graphs

Simple Graphs: Correspond to symmetric, irreflexive binary relations *R*.

- A *simple graph* *G*=(*V*,*E*) consists of:
  - a set *V* of *vertices* or *nodes* (*V* corresponds to the universe of the relation *R*),
  - a set *E* of *edges* / *arcs* / *links*: unordered pairs of [distinct] elements $u,v \in V$, such that *uRv*.
- A *directed graph* (*V*,*E*) consists of a set of vertices *V* and a binary relation (need not be symmetric) *E* on *V*.



*u*, *v* are *adjacent* / *neighbors* / *connected*.
Edge *e* is *incident with* vertices *u* and *v*.
Edge *e connects u* and *v*.
Vertices *u* and *v* are *endpoints* of edge *e*.



*Visual Representation of a Simple Graph*

# Degree of a Vertex

- Let *G* be an undirected graph, $v \in V$ a vertex.
- The *degree* of *v*, deg(*v*), is its number of incident edges. (Except that any self-loops are counted twice.)
- A vertex with degree 0 is called *isolated*.
- A vertex of degree 1 is called *pendant*.

Handshaking Theorem: Let *G* be an undirected (simple, multi-, or pseudo-) graph with vertex set *V* and edge set *E*. Then

$$\sum_{v \in V} \deg(v) = 2|E|$$

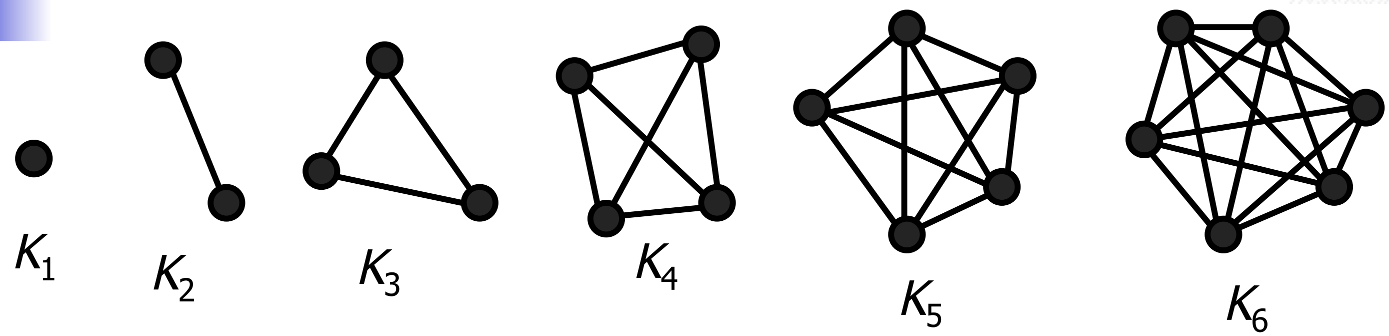
**Corollary:** Any undirected graph has an even number of vertices of odd degree.

# Directed Degree

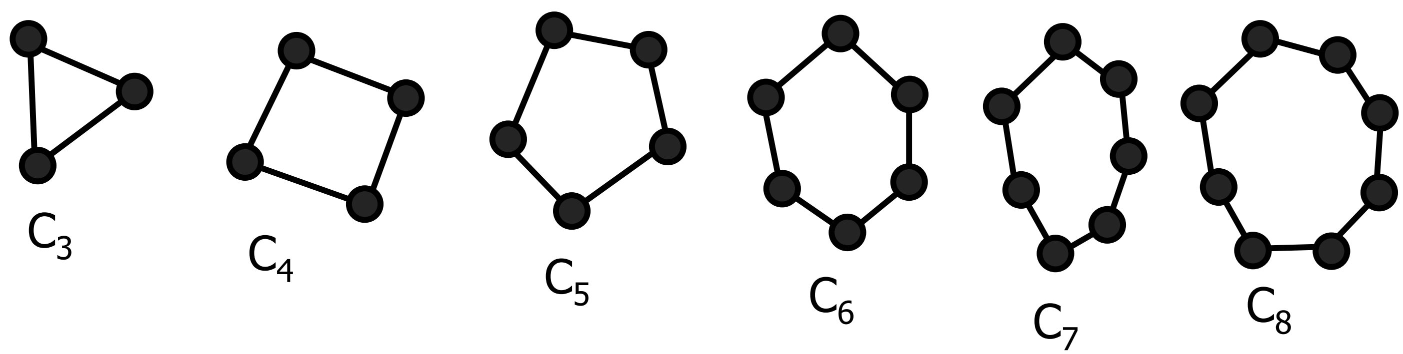
- Let *G* be a directed graph, *v* a vertex of *G*.
  - The *in-degree* of *v*, $\deg^-(v)$, is the number of edges going to *v*.
  - The *out-degree* of *v*, $\deg^+(v)$, is the number of edges coming from *v*.
  - The *degree* of *v*, $\deg(v) :\equiv \deg^-(v) + \deg^+(v)$, is the sum of *v*'s in-degree and out-degree.
- Directed Handshaking Theorem:

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = \frac{1}{2} \sum_{v \in V} \deg(v) = |E|$$

# Special Graph Structures

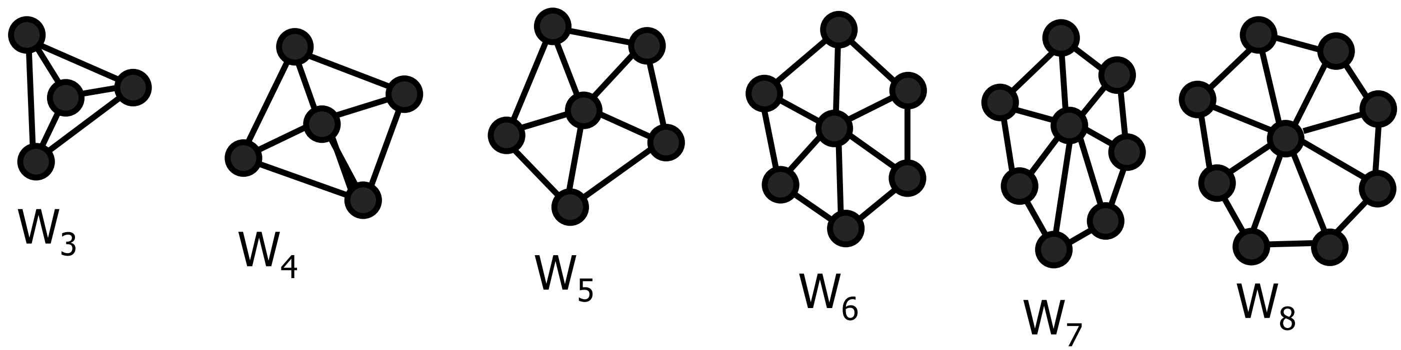



Complete graphs $K_n$

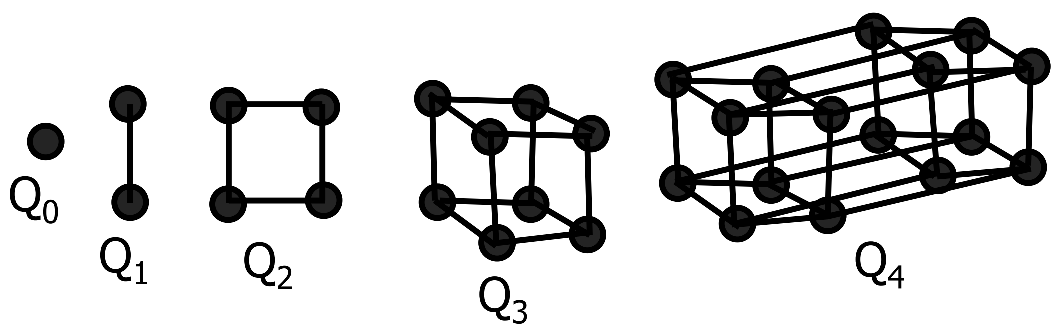



Cycles $C_n$

# Special Graph Structures




Wheels $W_n$

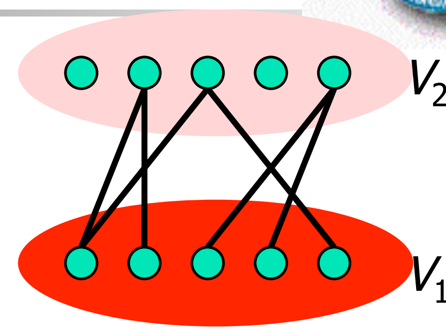



$n$-Cubes $Q_n$

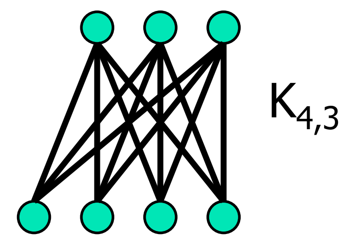
*Number of vertices:* $2^n$. *Number of edges?*

# Bipartite Graphs

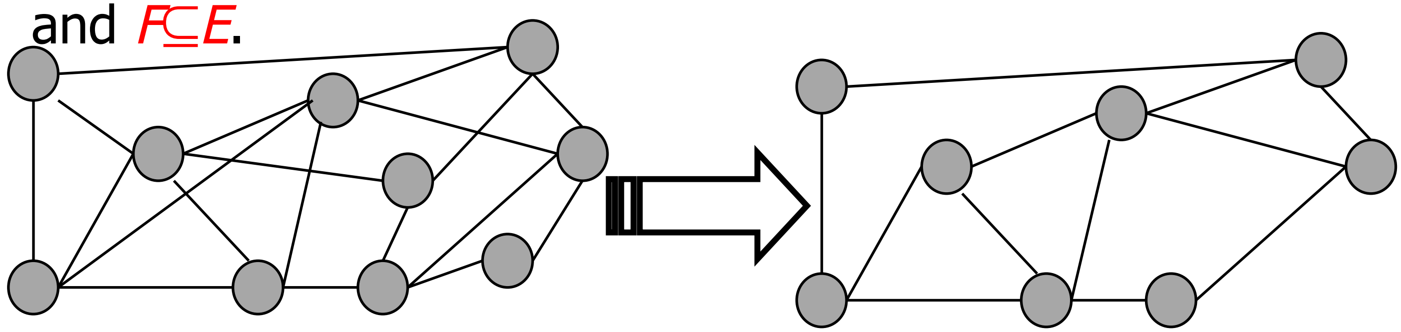
- A graph $G=(V,E)$ is *bipartite* (two-part) iff $V = V_1 \cap V_2$ where $V_1 \cup V_2=\emptyset$ and $\forall e\in E$: $\exists v_1\in V_1, v_2\in V_2$: $e=\{v_1,v_2\}$.




For $m,n\in\mathbf{N}$, the *complete bipartite graph* $K_{m,n}$ is a bipartite graph where $|V_1| = m$, $|V_2| = n$, and $E = \{\{v_1,v_2\}|v_1\in V_1 \wedge v_2\in V_2\}$.




A subgraph of a graph $G=(V,E)$ is a graph $H=(W,F)$ where $W\subseteq V$ and $F\subseteq E$.



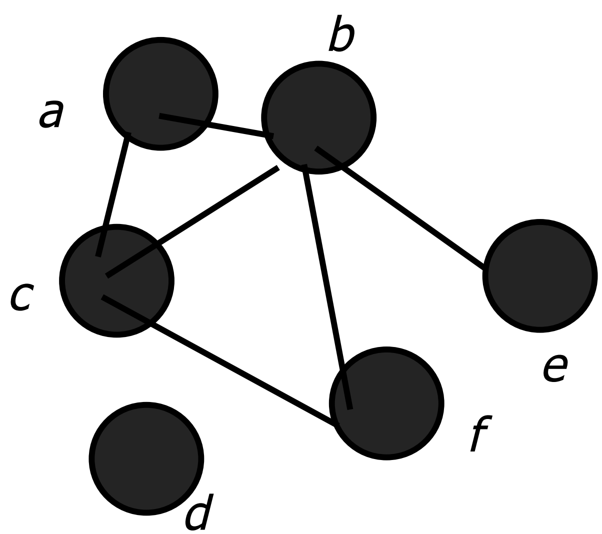
*G* *H*

# §9.3: Graph Representations & Isomorphism

- Graph representations:
  - Adjacency lists.
  - Adjacency matrices.
  - Incidence matrices.
- Graph isomorphism:
  - Two graphs are isomorphic iff they are identical except for their node names.

# Adjacency Lists

Adjacency Lists: A table with 1 row per vertex, listing its adjacent vertices.




| *Vertex* | *Adjacent Vertices* |
|---|---|
| *a* | *b, c* |
| *b* | *a, c, e, f* |
| *c* | *a, b, f* |
| *d* | |
| *e* | *b* |
| *f* | *c, b* |

Directed Adjacency Lists: 1 row per node, listing the terminal nodes of each edge incident from that node.

# Adjacency Matrices

- A way to represent simple graphs
  - possibly with self-loops.
- Matrix $\mathbf{A}=[a_{ij}]$, where $a_{ij}$ is 1 if $\{v_i, v_j\}$ is an edge of $G$, and is 0 otherwise.
- Can extend to pseudographs by letting each matrix elements be the number of links (possibly >1) between the nodes.

# Graph Isomorphism

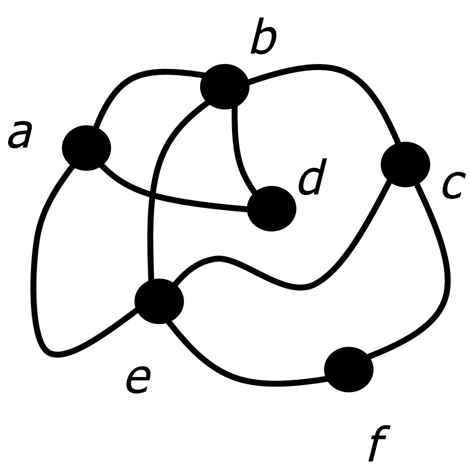
- Formal definition:
  - Simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ are *isomorphic* iff $\exists$ a bijection $f: V_1 \rightarrow V_2$ such that $\forall\ a,b \in V_1$, $a$ and $b$ are adjacent in $G_1$ iff $f(a)$ and $f(b)$ are adjacent in $G_2$.
  - $f$ is the "renaming" function between the two node sets that makes the two graphs identical.
  - This definition can easily be extended to other types of graphs.

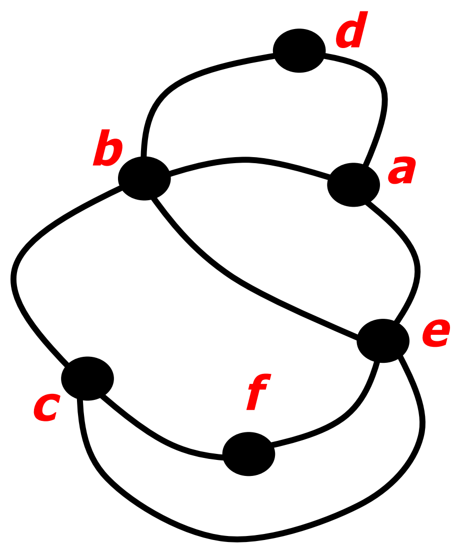
*Necessary* but not *sufficient* conditions for $G1=(V1, E1)$ to be isomorphic to $G2=(V2, E2)$:

- We must have that $|V1|=|V2|$, and $|E1|=|E2|$.
- The number of vertices with degree $n$ is the same in both graphs.
- For every proper subgraph $g$ of one graph, there is a proper subgraph of the other graph that is isomorphic to $g$.

# Isomorphism Example

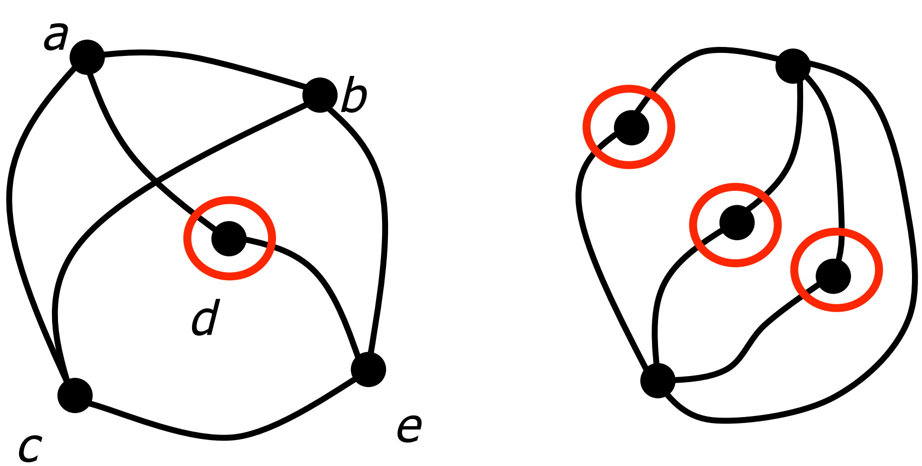
If isomorphic, label the 2nd graph to show the isomorphism, else identify difference.



# Are These Isomorphic?

- If isomorphic, label the 2nd graph to show the isomorphism, else identify difference.




- *Same # of vertices*
- *Same # of edges*
- *Different # of vertices of degree 2! (1 vs 3)*

# §9.4: Connectivity

- In an undirected graph, a *path of length n from u to v* is a sequence of adjacent edges going from vertex *u* to vertex *v*.
- A path is a *circuit* if $u=v$.
- A path *traverses* the vertices along it.
- A path is *simple* if it contains no edge more than once.

- Paths in Directed Graphs: Same as in undirected graphs, but the path must go in the direction of the arrows.

An undirected graph is *connected* iff there is a path between every pair of distinct vertices in the graph.
There is a *simple* path between any pair of vertices in a connected undirected graph.
*Connected component*: connected subgraph
A *cut vertex* or *cut edge* separates 1 connected component into 2 if removed

# Directed Connectedness

- A directed graph is *strongly connected* iff there is a directed path from *a* to *b* for any two vertices *a* and *b*.
- It is *weakly connected* iff the underlying *undirected* graph (*i.e.*, with edge directions removed) is connected.
- Note *strongly* implies *weakly* but not vice-versa.

Note that connectedness, and the existence of a circuit or simple circuit of length *k* are graph invariants with respect to isomorphism.

Counting different paths: the number of different paths from a vertex *i* to a vertex *j* is the (i, j) entry in $\mathbf{A}^r$, where A is the adjacency matrix of the graph

proof by induction on *r*

# §9.5: Euler & Hamilton Paths

- An ***Euler circuit*** in a graph *G* is a simple circuit containing every edge of *G*.
- An ***Euler path*** in *G* is a simple path containing every edge of *G*.
- A ***Hamilton circuit*** is a circuit that traverses each vertex in *G* exactly once.
- A ***Hamilton path*** is a path that traverses each vertex in *G* exactly once.

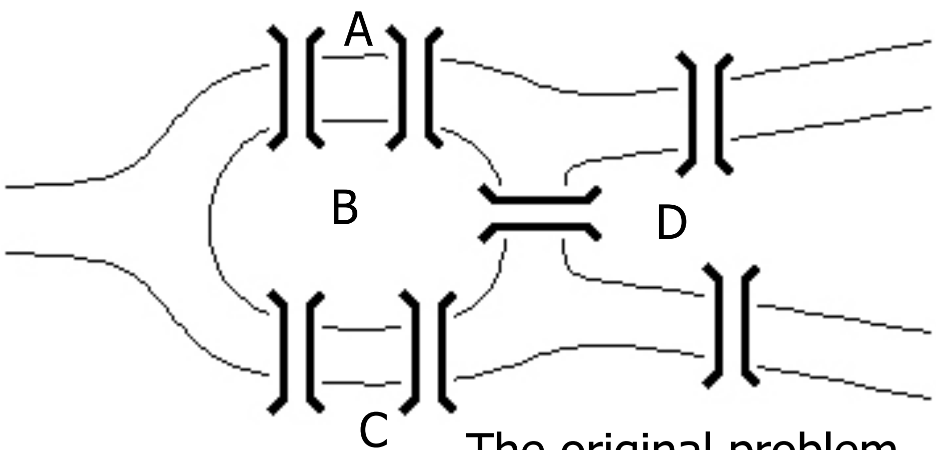
# Euler and Hamiltonian Tours

**Chinese postman problem**

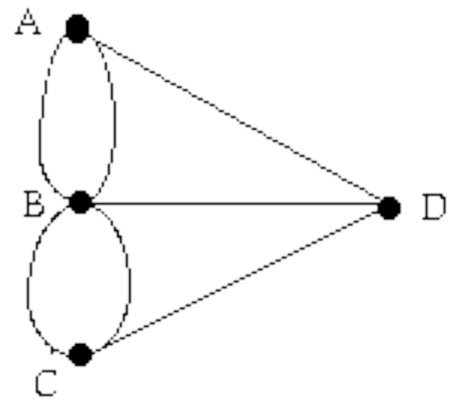
- find a shortest tour in a non-Euler graph
- some edges will be traversed twice
- corresponds to finding the cheapest set of paths connecting matching vertices of odd degree
  - the number of odd-degree vertices is even
  - the paths are edge-disjoint

# Bridges of Königsberg Problem

Can we walk through town, crossing each bridge exactly once, and return to start?




The original problem




Equivalent multigraph

**Theorem:** A connected multigraph has an Euler circuit iff each vertex has even degree.

**Proof:**

(→) The circuit contributes 2 to degree of each node.

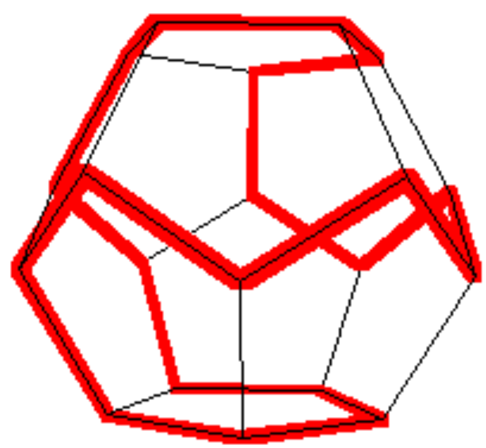
(←) By construction using algorithm on p. 580-581

# Euler Path Problem

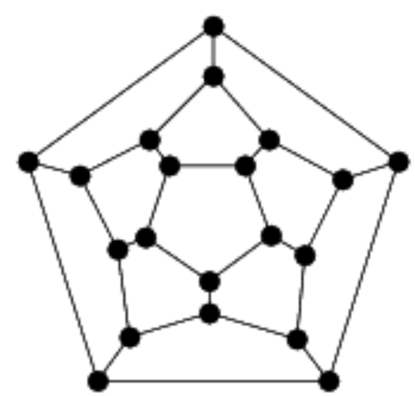
- **Theorem:** A connected multigraph has an Euler path (but not an Euler circuit) iff it has exactly 2 vertices of odd degree.
  - One is the start, the other is the end.
- **Euler tour in a directed graph**
  - in-degrees must match out-degrees in all nodes
- Euler Circuit Algorithm
  - Begin with any arbitrary node.
  - Construct a simple path from it till you get back to start.
  - Repeat for each remaining subgraph, splicing results back into original cycle.

# Round-the-World Puzzle

- Can we traverse all the vertices of a dodecahedron, visiting each once?`



Dodecahedron puzzle



Equivalent graph

# Hamiltonian Path Theorems

- **Dirac's theorem**: If (but not only if) $G$ is connected, simple, has $n \geq 3$ vertices, and $\forall v \deg(v) \geq n/2$, then $G$ has a Hamilton circuit.
  - **Ore's corollary:** If $G$ is connected, simple, has $n \geq 3$ nodes, and $\deg(u)+\deg(v) \geq n$ for every pair $u,v$ of non-adjacent nodes, then $G$ has a Hamilton circuit.

# Hamiltonian Tours - Applications

**Traveling salesmen problem**

- in a weighted graph, find the shortest tour visiting every vertex
- we can solve it if we can solve the problem of finding the shortest Hamiltonian path in complete graphs

**Gray codes**

- find a sequence of codewords such that each binary string is used, but adjacent codewords are close to each other (differ by 1 bit only)
- all binary strings of length **n** = vertices of **n-**dimensional hypercube
- edges of the hypercube = vertices that differ by 1 bit
- our problem = find a Hamiltonian circuit in hypercubes
- Gray codes – one particular solution
  - can be defined recursively (as hypercubes are)

# Planar Graphs

***Planar graphs*** are graphs that can be drawn in the plane without edges having to cross.

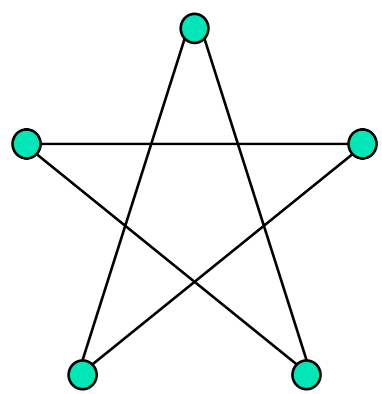
Understanding planar graph is important:

- Any graph representation of maps/ topographical information is planar.
  - graph algorithms often specialized to planar graphs (e.g. traveling salesperson)
- Circuits usually represented by planar graphs

# Planar Graphs
# -Common Misunderstanding

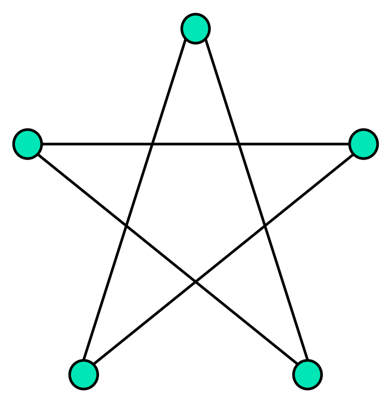
Just because a graph is drawn with edges crossing doesn't mean its not planar.

Q: Why can't we conclude that the following is non-planar?



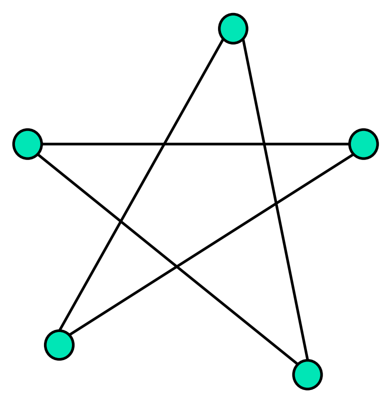
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



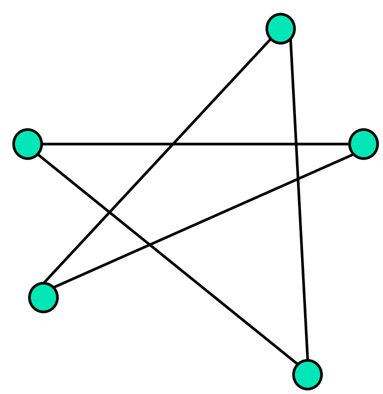
# Planar Graphs -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



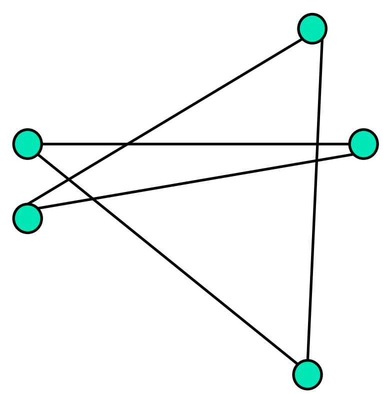
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



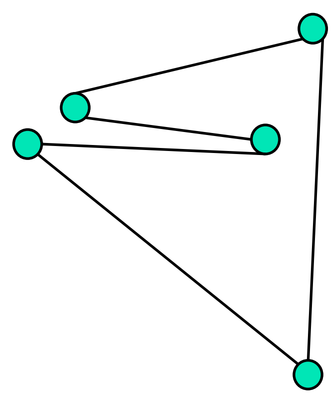
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



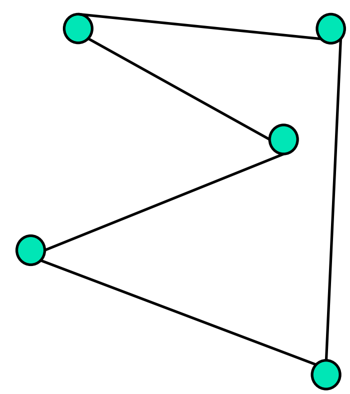
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



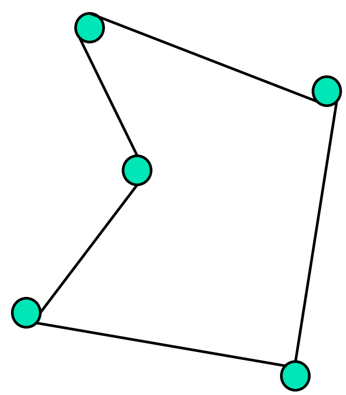
## Planar Graphs
## -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



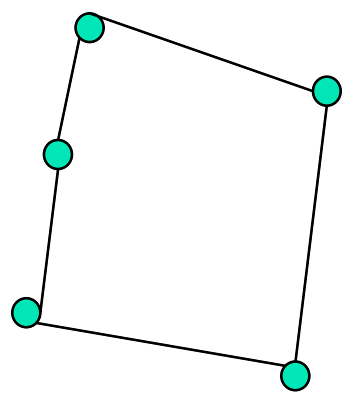
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



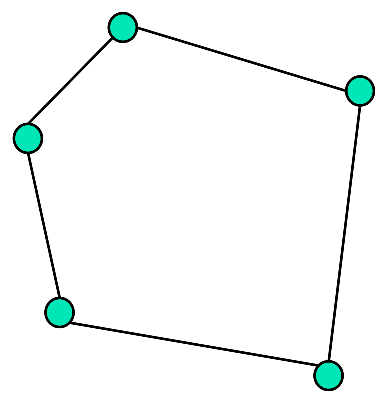
# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:



# Planar Graphs
# -Common Misunderstanding

A: Because it is isomorphic to a graph which *is* planar:

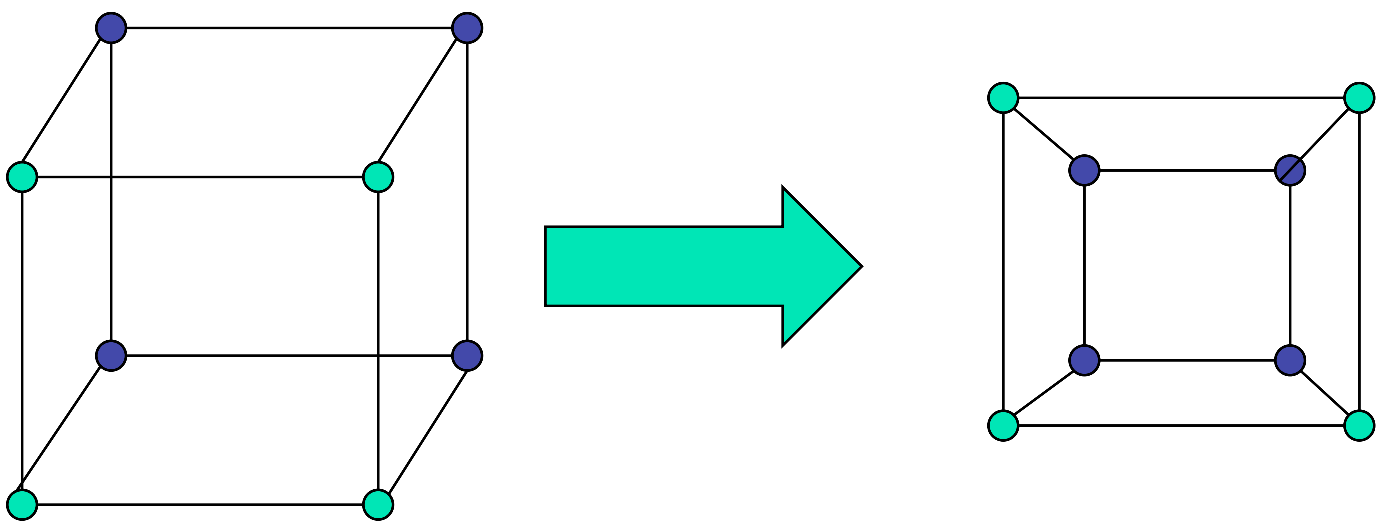


# Proving Planarity

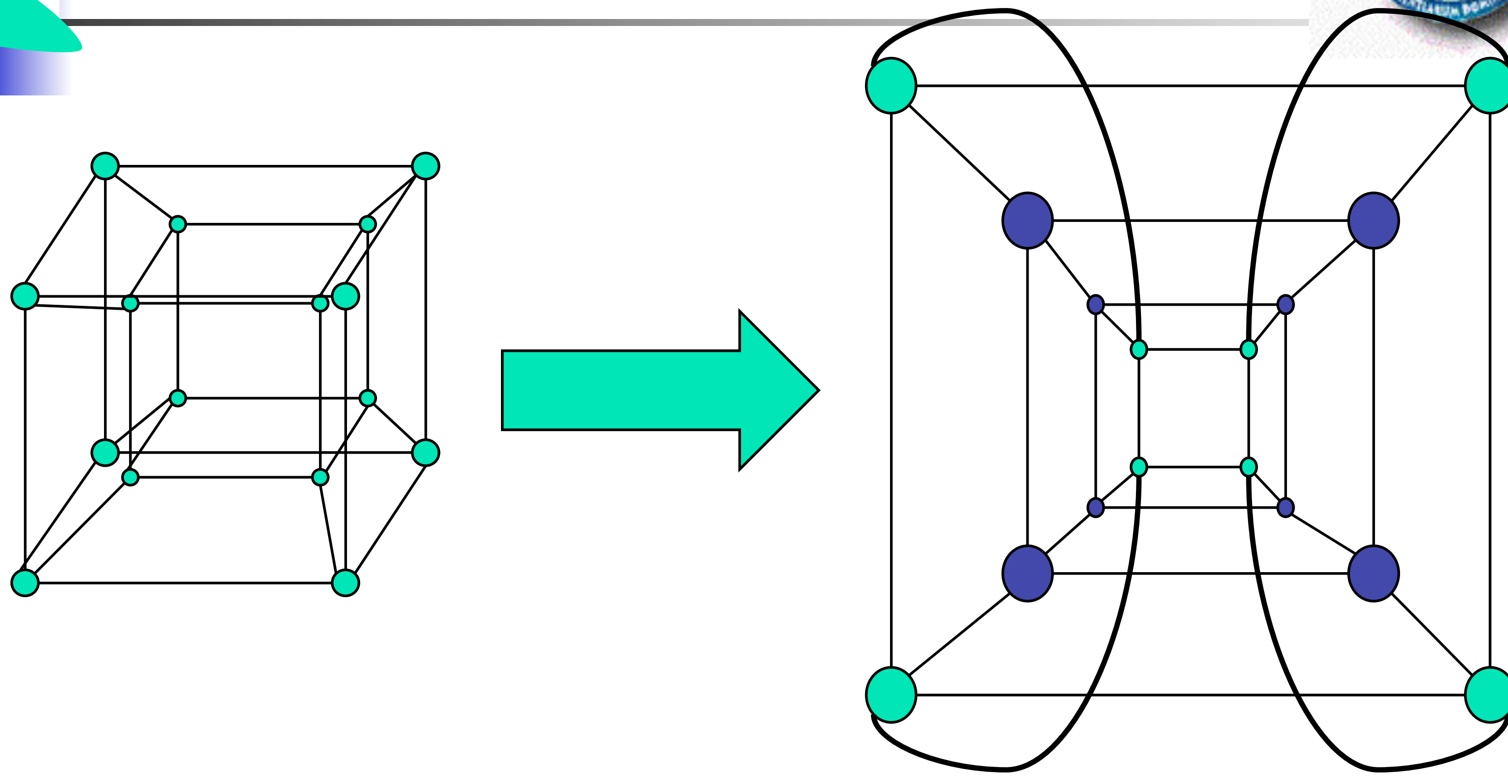
To prove that a graph is planar amounts to redrawing the edges in a way that no edges will cross. May need to move vertices around and the edges may have to be drawn in a very indirect fashion.

E.G. show that the 3-cube is planar:

# Proving Planarity
# 3-Cube



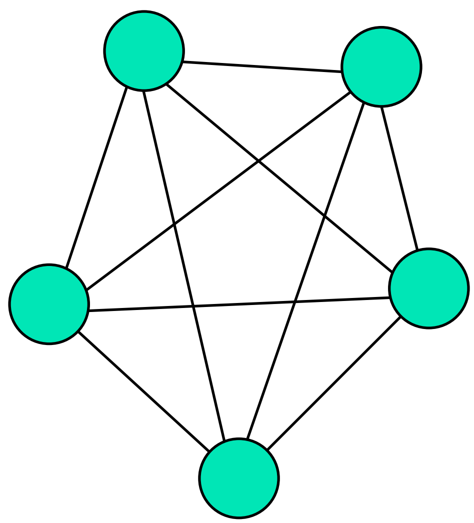
# Proving Planarity?
# 4-Cube

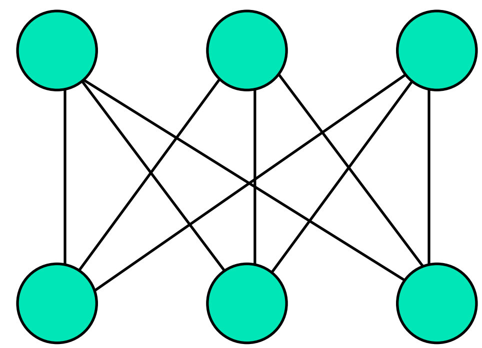


Seemingly not planar, but how would one prove this!

# The smallest graphs that are not planar

- $K_5$, $K_{3,3}$





## Disproving Planarity: Kuratowski / Wagner

- A graph is planar if and only if it does not contain the $K_5$ and the $K_{3,3}$ as a *homeomorphic subgraph* / as a *minor.*
- Minor: H is a minor of G, if H can be obtained from G by a series of 0 or more deletions of vertices, deletions of edges, and contraction of edges.
- Does not yield fast recognition algorithm!

# Euler's theorem

- Let G be a connected plane graph with $v$ vertices, $e$ edges, and $f$ faces.

  Then $v + f - e = 2$.
- Proof. By induction.
  - True if $e$=0.
  - If G has a circuit, then delete an edge and …
  - If G has a vertex $w$ of degree 1, then delete $w$ and …
- …

# Euler's theorem Corollaries

- If G is a connected plane graph with no parallel edges and no self-loops, with **$v \geq 3$** , then **$e \leq 3v-6$**.
  - Every face `has' at least three edges; each edge `is on' two faces, or twice on the same face.
- Every plane graph with no parallel edges and no self -loops has a vertex of degree at most 5.
  - **$K_5$** is not a planar graph
- If **G** is a planar simple graph with **$v \geq 3$** and no cycles of length **3**, then **$e \leq 2v-4$**
  - **$K_{3,3}$** is not planar

# Proof of Euler's theorem Corollaries

Suppose that G has at most m edges, consider some plane drawing of G, with f faces. Consider the number of pairs (e, F) where e is one of the edges bounding the face F.

For each edge e, there are at most 2 faces that it bounds. So the total number of these edge face pairs has to be less than 2e. On the other hand, because G is a simple graph, each face is bounded by at least 3 edges. Therefore, the total number of edge-face pairs is greater than or equal to 3f. So $3f \leq 2e$

By the Euler Polyhedron Formula, $v-e+f=2$, so, $3v-3e+3f=6$. Since $3f \leq 2e$, $3f = 6 - 3v+3e \leq 2e$. Therefore $e \leq 3v - 6$.

If G has no triangles, each face must be bounded by 4 or more edges. Thus $2f \leq e$ and $4 - 2v+2e \leq e$, therefore $e \leq 2v - 4$.

# Euler's theorem Corollaries

Every simple, planar graph has a vertex of degree less than 6.

Proof: $$\sum_{w \in V} \deg(w) = 2e \leq 2(3v-6) = 6v-12$$

Thus the average degree <=(6v-12)/v = 6 – 12/v < 6.

So at least one of the vertices has degree less than 6.

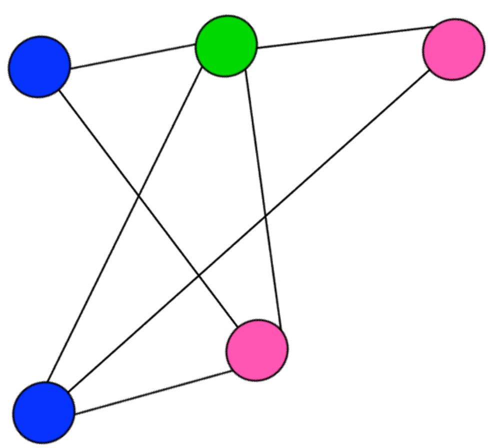
# Graph Colorings

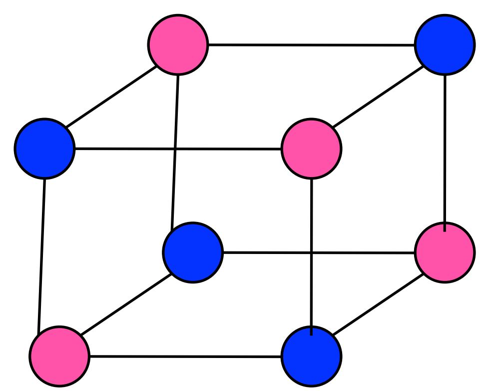
**Vertex coloring of a graph**

- assign a color to each vertex so that adjacent vertices are of different colors
- i.e. find **c: V → N** such that **(u,v) ∈E→ c(u) ≠ c(v)**

**Chromatic number χ of a graph G**

- the least amount of colors needed to color the graph





# Graph Colorings

So, what is a chromatic number for

- **$K_n$**?
- **$C_n$**?
- **$K_{m,n}$**?

Bipartite Graphs

The chromatic number of a graph G is 2 if and only if G is a bipartite graph

Planar Graphs?

# The four Color Theorem

**The four color theorem:** **The chromatic number of every simple planar graph is at most four**

We can prove that six colors are enough

For general graphs?

only exponential algorithms known

even finding approximation is difficult

# Six color Theorem

Proof of the six color theorem: by induction on n, the number of vertices of the graph.

Basis Step: If G has fewer than seven vertices then the result is obvious.

Inductive step: Let n>=7. We assume that all simple graphs with n-1 vertices are 6 colorable. Because of planarity and Euler's theorem we know that G has a vertex v with degree less than 6. Remove v from G and all adjacent edges to v. The remaining subgraph has n-1 vertices and by the induction hypothesis it can be properly colored by 6 colors. Since v has at most 5 adjacent vertices in G, then v can be colored with a color different from all of its neighbours. This ends the proof.

# Graph Colorings - Applications

**Scheduling exams**

- many exams, each student have specified which exams he/she has to take
- how many different exam slots are needed? (a student cannot be at two different exams at the same time)

*Vertices:* courses
*Edges:* if there is a student taking both courses
*Exam slots:* colors

**Frequency assignments**

- TV channels, mobile networks

# §10.1: Introduction to Trees

- A *tree* is a connected undirected graph that contains no circuits.
  - **Theorem:** There is a unique simple path between any two of its nodes.
- A (not-necessarily-connected) undirected graph without simple circuits is called a *forest*.
  - You can think of it as a set of trees having disjoint sets of nodes.
- A *leaf* node in a tree or forest is any pendant or isolated vertex. An *internal* node is any non-leaf vertex (thus it has degree ≥).

# Trees as Models

- Can use trees to model the following:
  - Saturated hydrocarbons
  - Organizational structures
  - Computer file systems
- In each case, would you use a rooted or a non-rooted tree?

# Some Tree Theorems

- Any tree with $n$ nodes has $e = n-1$ edges.
  - **Proof:** Consider removing leaves.
- A full $m$-ary tree with $i$ internal nodes has $n=mi+1$ nodes, and $\ell=(m-1)i+1$ leaves.
  - **Proof:** There are $mi$ children of internal nodes, plus the root. And, $\ell = n-i = (m-1)i+1$. □
- Thus, when $m$ is known and the tree is full, we can compute all four of the values $e$, $i$, $n$, and $\ell$, given any one of them.

# Some More Tree Theorems

- **Definition:** The *level* of a node is the length of the simple path from the root to the node.
  - The *height* of a tree is maximum node level.
  - A rooted *m*-ary tree with height *h* is called *balanced* if all leaves are at levels $h$ or $h-1$.
- **Theorem:** There are at most $m^h$ leaves in an *m*-ary tree of height *h*.
  - **Corollary:** An *m*-ary tree with $\ell$ leaves has height $h \geq \lceil \log_m \ell \rceil$. If *m* is full and balanced then $h = \lceil \log_m \ell \rceil$.

# §10.2: Applications of Trees

- Binary search trees
  - A simple data structure for sorted lists
- Decision trees
  - Minimum comparisons in sorting algorithms
- Prefix codes
  - Huffman coding
- Game trees

# §10.3: Tree Traversal

- Universal address systems
- Traversal algorithms
  - Depth-first traversal:
    - Preorder traversal
    - Inorder traversal
    - Postorder traversal
  - Breadth-first traversal
- Infix/prefix/postfix notation