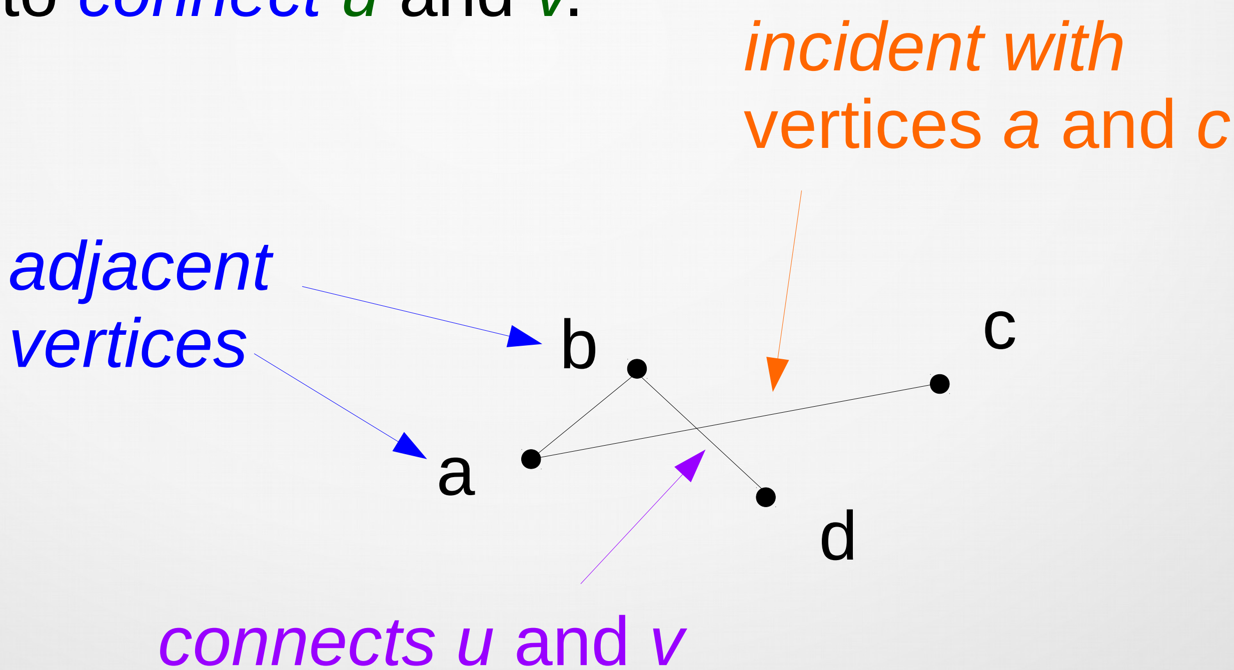
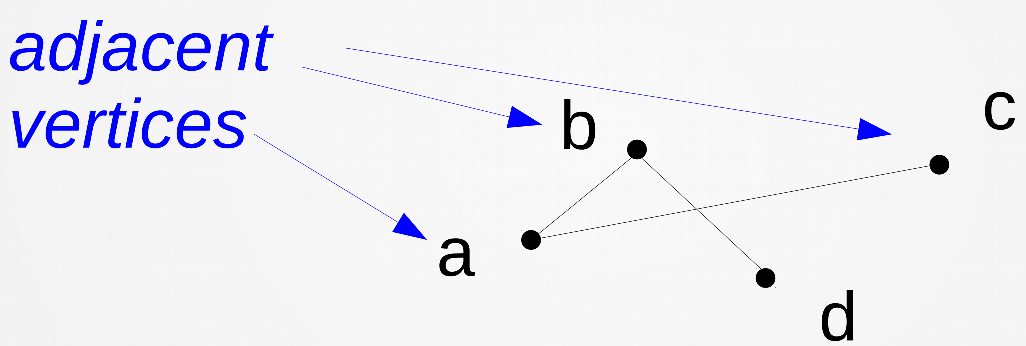
10.2 *Graph terminology and special types of graphs*

[Def] Two vertices *u* and *v* in an undirected graph *G* are called *adjacent* (*neighbors*) in *G* if *u* and *v* are endpoints of an edge *e* of *G*.
Such an edge is called *incident with* vertices *u* and *v*, and is said to *connect* *u* and *v*.



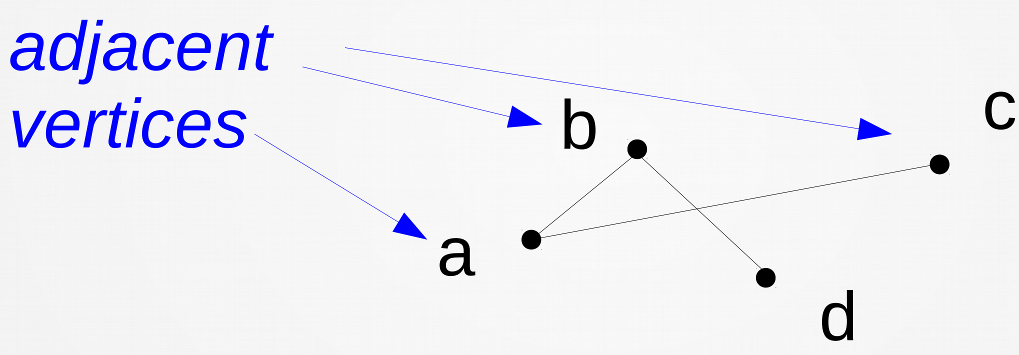
10.2 *Graph terminology and special types of graphs*

[Def] neighborhood of *v*, *N(v)*, is the set of all neighbors of *v*.



10.2 *Graph terminology and special types of graphs*

[Def] neighborhood of *v*, *N(v)*, is the set of all neighbors of *v*.

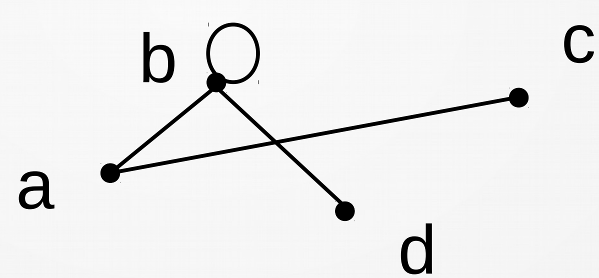


neighborhood of *a*: *N(a) = {b,c}* ,
neighborhood of *b*: *N(b) = {a,d}* ,
neighborhood of *c*: *N(c) = {a}* ,
neighborhood of *d*: *N(d) = {b}* ,

10.2 *Graph terminology and special types of graphs*

[Def] the *degree of a vertex* in undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

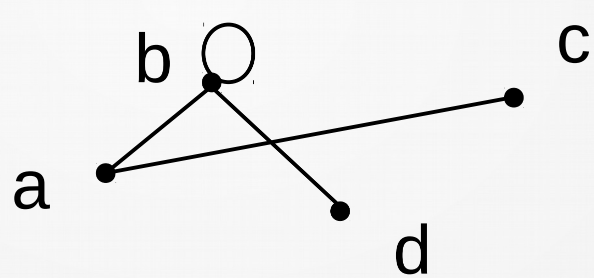
denotaion: deg(v)



10.2 *Graph terminology and special types of graphs*

[Def] the *degree of a vertex* in undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

denotaion: deg(v)

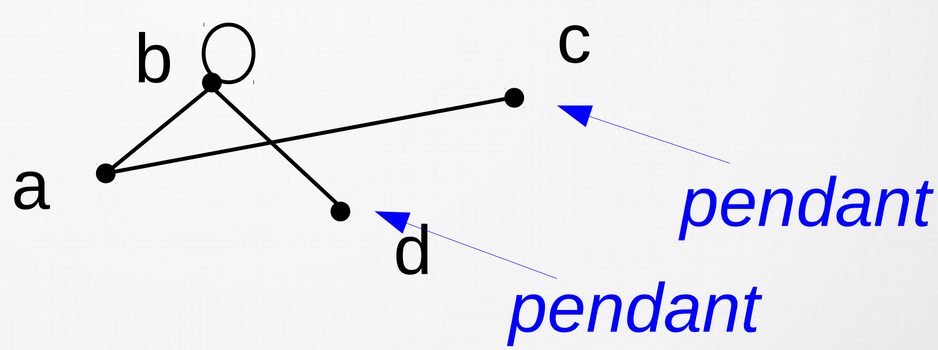


deg(a) = 2
deg(b) = 2+2 = 4
deg(c) = 1
deg(d) = 1

10.2 *Graph terminology and special types of graphs*

[Def] the *degree of a vertex* in undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

denotaion: deg(v)



A vertex of degree 0 is called *isolated*.
A vertex of degree 1 is called *pendant*.

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

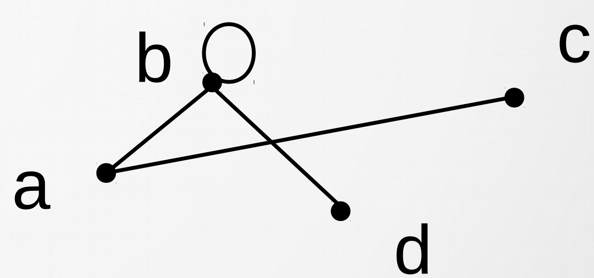
Then $$2m = \sum_{v \in V} deg(v)$$

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$



m = 4

deg(a) = 2, deg(b) = 4, deg(c) = 1, deg(d) = 1

$2 \times 4 = 2 + 4 + 1 + 1$

$8 = 8$

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$

Why is it so?

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let *G = (V,E)* be an undirected graph with *m* edges.

Then $$2m = \sum_{v \in V} deg(v)$$

Why is it so?

- each non-loop edge contributes 2 to the sum of the degrees (1 for each of adjacent vertices)
- each loop contributes 2 to the sum of the degrees (only for one vertex)

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$

[Theorem] An undirected graph has an <u>even</u> number of vertices of <u>odd degree</u>.

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$

[Theorem] An undirected graph has an <u>even</u> number of vertices of <u>odd degree</u>.

<u>**Proof**</u>: Let V_1 and V_2 be the set of vertices of even degree and odd degree respectively, in a undirected graph $G = (V,E)$ with m edges. $|E| = m$

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem
Let *G = (V,E)* be an undirected graph with *m* edges.

Then $$2m = \sum_{v \in V} deg(v)$$

[Theorem] An undirected graph has an <u>even</u> number of vertices of <u>odd degree</u>.

<u>Proof</u>: Let V_1 and V_2 be the set of vertices of even degree and odd degree respectively, in a undirected graph *G = (V,E)* with *m* edges. |E| = m

$$2m = \sum_{v \in V} deg(v) = \sum_{v \in V_1} deg(v) + \sum_{v \in V_2} deg(v)$$

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$

[Theorem] An undirected graph has an <u>even</u> number of vertices of <u>odd degree</u>.

<u>**Proof**</u>: Let V_1 and V_2 be the set of vertices of even degree and odd degree respectively, in a undirected graph $G = (V,E)$ with m edges. $|E| = m$

even

$$\underset{\text{even}}{2m} = \sum_{v \in V} deg(v) = \sum_{v \in V_1} deg(v) + \sum_{v \in V_2} deg(v)$$

10.2 *Graph terminology and special types of graphs*

[Theorem] The Handshaking Theorem

Let $G = (V,E)$ be an undirected graph with m edges.

Then $$2m = \sum_{v \in V} deg(v)$$

[Theorem] An undirected graph has an <u>even</u> number of vertices of <u>odd degree</u>.

<u>Proof</u>: Let V_1 and V_2 be the set of vertices of even degree and odd degree respectively, in a undirected graph $G = (V,E)$ with m edges. $|E| = m$

$$\underbrace{2m}_{\text{even}} = \sum_{v \in V} deg(v) = \overbrace{\sum_{v \in V_1} deg(v)}^{\text{even}} + \underbrace{\sum_{v \in V_2} deg(v)}_{\text{must be even}}$$

q.e.d.

10.2 *Graph terminology and special types of graphs*

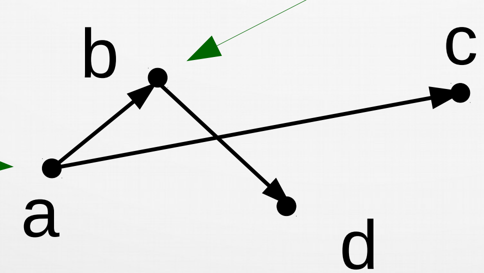
Let's discuss the terminology for directed graphs

10.2 *Graph terminology and special types of graphs*

[Def] When (u,v) is an edge of the directed graph G, *u* is said to be *adjacent to v, and v is adjacent from u*.
u is called *initial vertex* of *(u,v)*, and
v is called *terminal/end* vertex of *(u,v)*.
The initial and terminal vertex of the loop is the same.

for edge (a,b):
terminal vertex,
b is adjacent from a;

edges (a,b), (a,c):
initial vertex,
a is adjacent
to b and to c



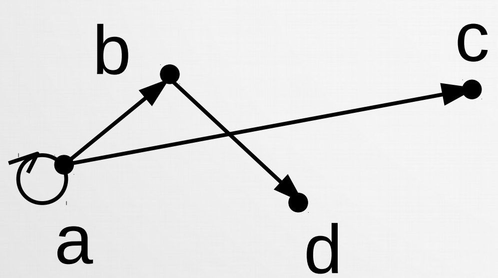
for edge (b,d):
initial vertex,
b is adjacent to d

10.2 *Graph terminology and special types of graphs*

[Def] The *in-degree of a vertex v* is the number of edges with *v* as their terminal vertex. $deg^{-}(v)$

The *out-degree of a vertex v* is the number of edges with *v* as their initial vertex. $deg^{+}(v)$

Loop contributes to both in-degree and out-degree.

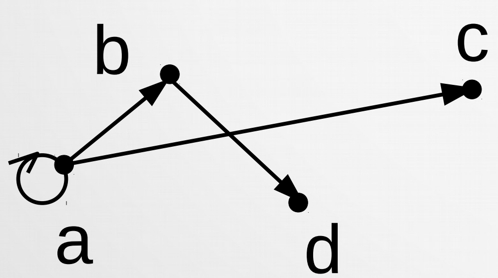


10.2 *Graph terminology and special types of graphs*

[Def] The *in-degree of a vertex v* is the number of edges with *v* as their terminal vertex. $deg^-(v)$

The *out-degree of a vertex v* is the number of edges with *v* as their initial vertex. $deg^+(v)$

Loop contributes to both in-degree and out-degree.

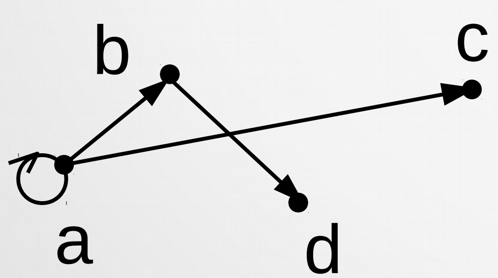


$deg^-(a) = 1$, $deg^+(a) = 3$
$deg^-(b) = 1$, $deg^+(b) = 1$
$deg^-(c) = 1$, $deg^+(c) = 0$
$deg^-(d) = 1$, $deg^+(d) = 0$

10.2 *Graph terminology and special types of graphs*

[Theorem] Let G = (V,E) be a directed graph. Then

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



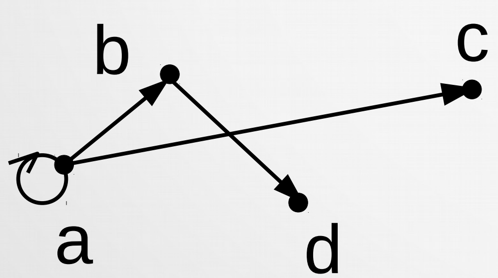
$deg^-(a) = 1$, $deg^+(a) = 3$
$deg^-(b) = 1$, $deg^+(b) = 1$
$deg^-(c) = 1$, $deg^+(c) = 0$
$deg^-(d) = 1$, $deg^+(d) = 0$
4 4
$|E| = 4$

10.2 *Graph terminology and special types of graphs*

[Theorem] Let G = (V,E) be a directed graph. Then

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

Why is it so?



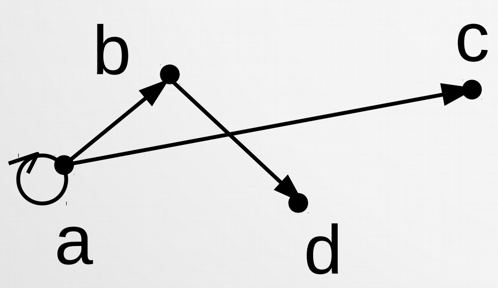
$deg^-(a) = 1$, $deg^+(a) = 3$
$deg^-(b) = 1$, $deg^+(b) = 1$
$deg^-(c) = 1$, $deg^+(c) = 0$
$deg^-(d) = 1$, $deg^+(d) = 0$
4 4
$|E| = 4$

10.2 *Graph terminology and special types of graphs*

[Theorem] Let G = (V,E) be a directed graph. Then

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

Why is it so?

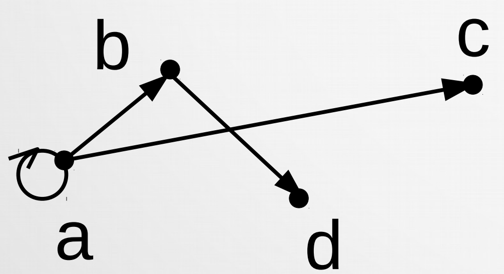


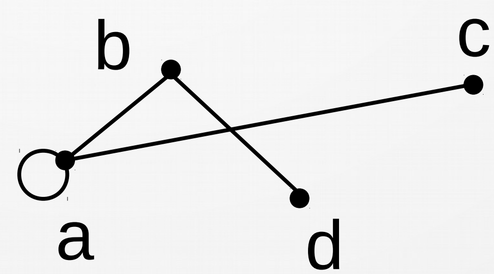
Because each edge has an initial vertex and a terminal vertex, hence contributes to both out-degree and in-degree.

10.2 *Graph terminology and special types of graphs*

We can take a directed graph and convert to to undirected graph by ignoring the directions.
Such a graph is called *underlying undirected graph*.

The directed graph and its underlying undirected graph have the same number of edges.





10.2 *Graph terminology and special types of graphs*

Example: recall *Hollywood graph*.
a) What does the degree of a vertex represent in the *Hollywood graph*?

b) What does the *neighborhood of a vertex* represent?

c) What do *isolated* and *pendant vertices* represent?

10.2 *Graph terminology and special types of graphs*

Example: recall *Hollywood graph*.
a) What does the degree of a vertex represent in the *Hollywood graph*?

the degree of a vertex represents the number of times the actor worked together with other actors on a movie or a TV show

b) What does the *neighborhood of a vertex* represent?

c) What do *isolated* and *pendant vertices* represent?

10.2 *Graph terminology and special types of graphs*

Example: recall *Hollywood graph*.
a) What does the degree of a vertex represent in the *Hollywood graph*?

b) What does the *neighborhood of a vertex* represent?

N(a) represents the list of all actors actor a worked with on a movie or a TV show.

c) What do *isolated* and *pendant vertices* represent?

10.2 *Graph terminology and special types of graphs*

Example: recall *Hollywood graph*.
a) What does the degree of a vertex represent in the *Hollywood graph*?

b) What does the *neighborhood of a vertex* represent?

c) What do *isolated* and *pendant vertices* represent?

Isolated vertices represent actors who didn't work with any other actor (present in the graph) on a movie or a TV show.
Pendant vertices represent only actors with one collaboration only.

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

<u>Solution</u>: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

Solution: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

2 vertices:

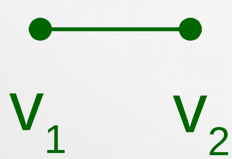
$\deg(v_1) = 0$, $\deg(v_2) = 0$

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

Solution: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

2 vertices:



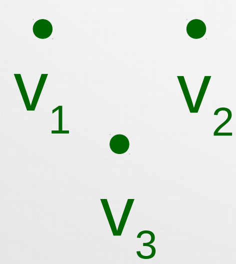
$\deg(v_1) = 1, \ \deg(v_2) = 1$

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

<u>Solution</u>: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

3 vertices:



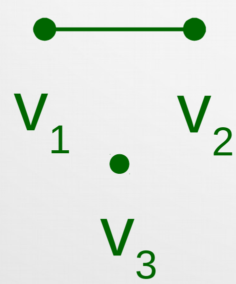
$\deg(v_1) = 0$, $\deg(v_2) = 0$, $\deg(v_3) = 0$,

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

<u>Solution</u>: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

3 vertices:



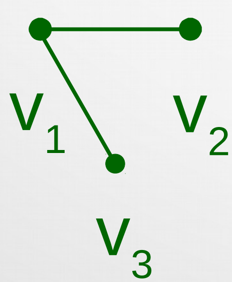
$\deg(v_1) = 1$, $\deg(v_2) = 1$, $\deg(v_3) = 0$,

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

<u>Solution</u>: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

3 vertices:



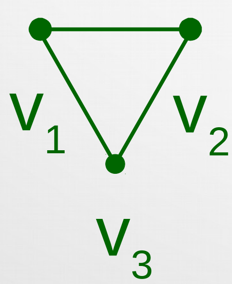
$\deg(v_1) = 2$, $\deg(v_2) = 1$, $\deg(v_3) = 1$,

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.

<u>Solution</u>: recall that *simple graphs* are undirected graphs that have no loops, no multiple edges.

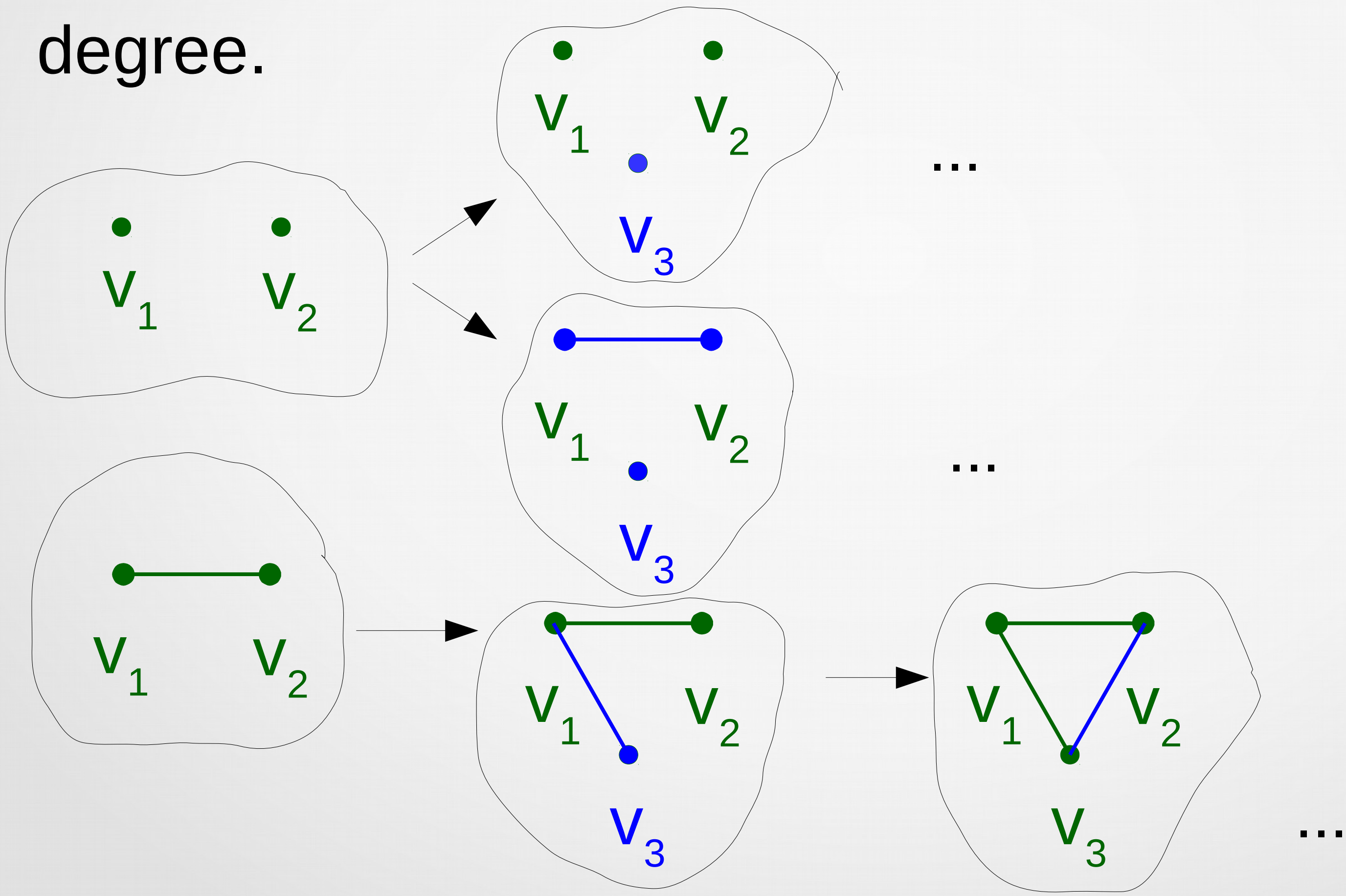
3 vertices:



$\deg(v_1) = 2$, $\deg(v_2) = 2$, $\deg(v_3) = 2$,

10.2 *Graph terminology and special types of graphs*

Example: show that in a *simple graph* with at least two vertices there must be two vertices that have the same degree.



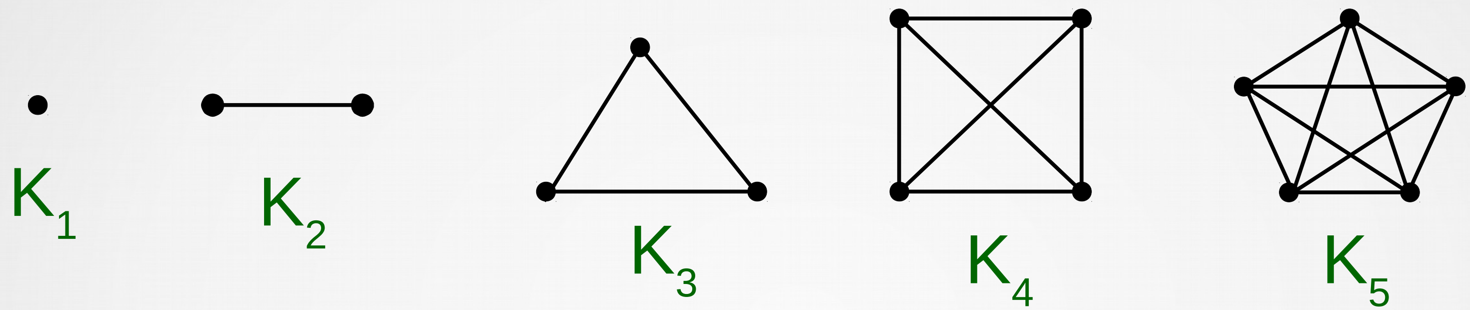
10.2 *Graph terminology and special types of graphs*

Special Simple Graphs

- complete graphs (K_n, for $n \in \mathbf{Z}^+$)
- cycles (C_n, for $n \in \mathbf{Z}^+$ and $n \geq 3$)
- wheels (W_n, for $n \in \mathbf{Z}^+$ and $n \geq 3$)
- *n*-cubes (Q_n, for $n \in \mathbf{Z}^+$)

10.2 *Graph terminology and special types of graphs*

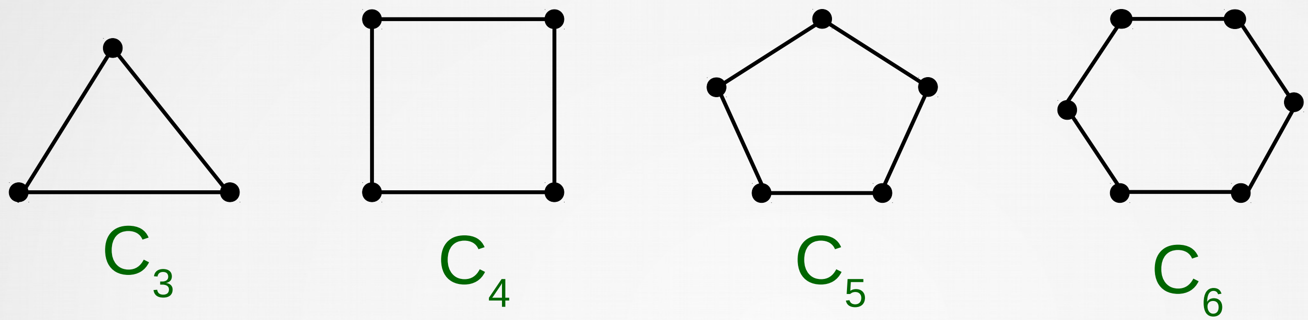
Special Simple Graphs: Complete graphs



A complete graph K_n on n vertices is a simple graph that contains exactly one edge between each pair of distinct vertices.
n = 1, 2, 3, 4, ...

10.2 *Graph terminology and special types of graphs*

Special Simple Graphs: Cycles

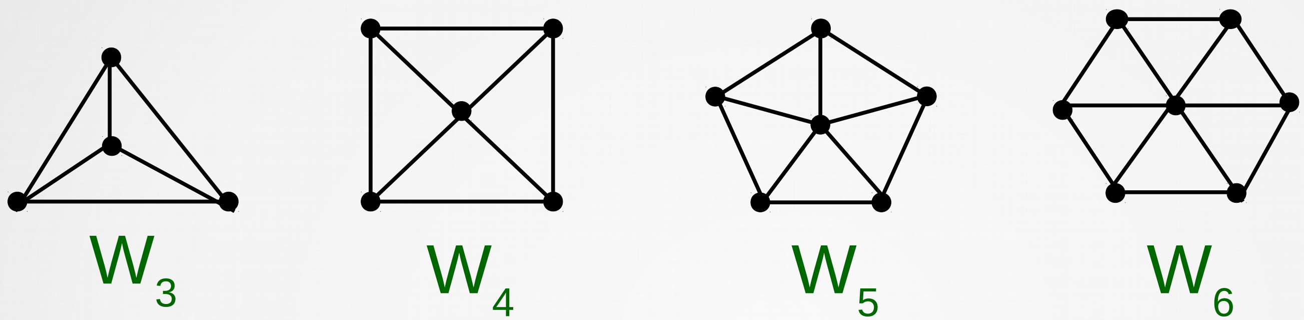


A cycle C_n on n vertices consists on n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1,v_2\}, \{v_2,v_3\},\ldots,$ and $\{v_{n-1},v_n\}$.

$n = 3, 4, 5, \ldots$

10.2 *Graph terminology and special types of graphs*

Special Simple Graphs: Wheels

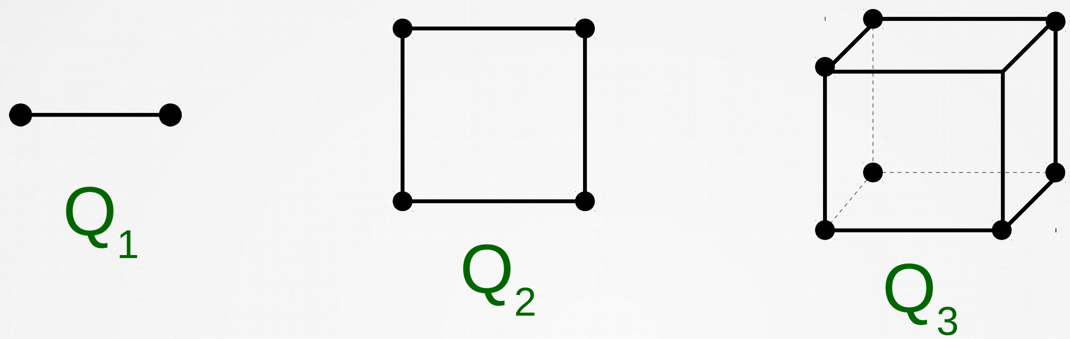


A wheel W_n can be obtained from cycle C_n by adding an additional vertex and connecting this new vertex with each of the *n* vertices in C_n.

$n = 3, 4, 5, ...$

10.2 *Graph terminology and special types of graphs*

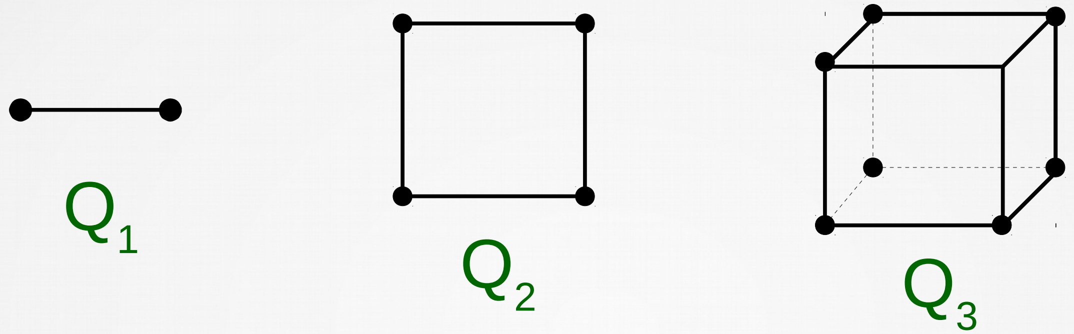
Special Simple Graphs: *n*-Cubes



Graph of Q_n has 2^n vertices

10.2 *Graph terminology and special types of graphs*

Special Simple Graphs: *n*-Cubes



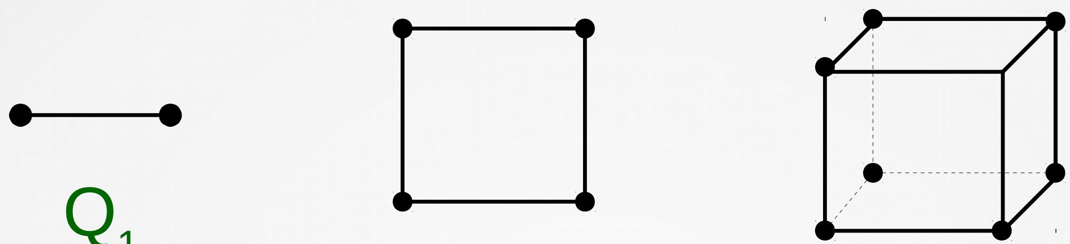
Graph of Q_n has 2^n vertices

n-Cubes can represent *bit strings* of length *n*

$n = 1, 2, 3, 4, \ldots$

10.2 *Graph terminology and special types of graphs*

Special Simple Graphs: *n*-Cubes



Q_1 Q_2 Q_3

Graph of Q_n has 2^n vertices

n-Cubes can represent *bit strings* of length *n*

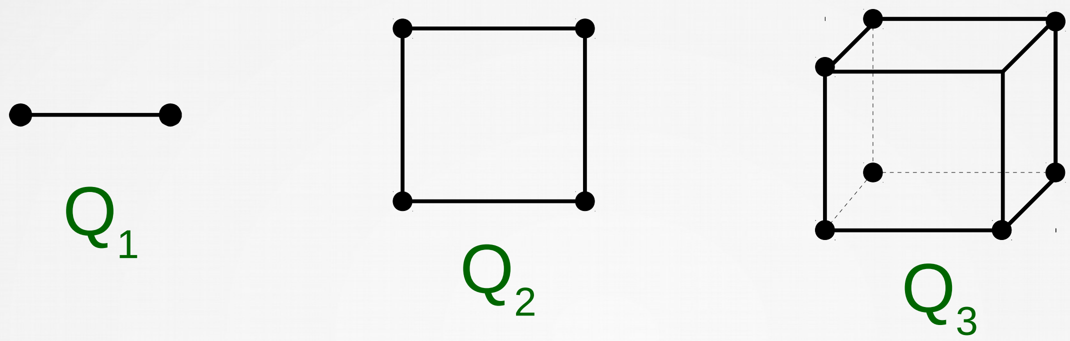
n = 1, 2, 3, 4, ...

0 1

Q_1

10.2 *Graph terminology and special types of graphs*

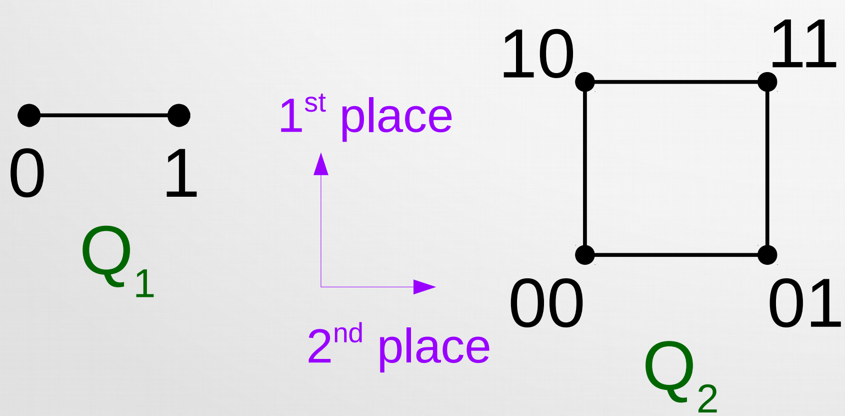
Special Simple Graphs: *n*-Cubes



Graph of Q_n has 2^n vertices

n-Cubes can represent *bit strings* of length *n*

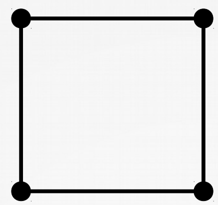
$n = 1, 2, 3, 4, \ldots$



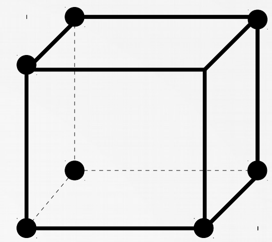
10.2 *Graph terminology and special types of graphs*

Special Simple Graphs: *n*-Cubes

Q_1



Q_2



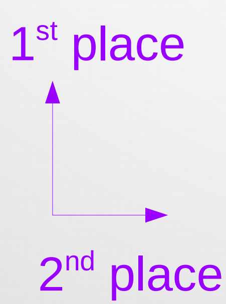
Q_3

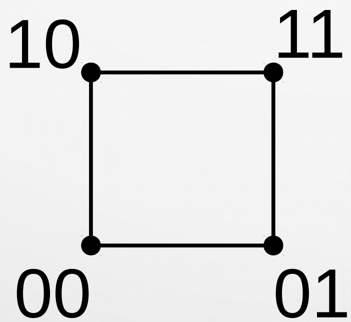
Graph of Q_n has 2^n vertices

n-Cubes can represent *bit strings* of length *n*

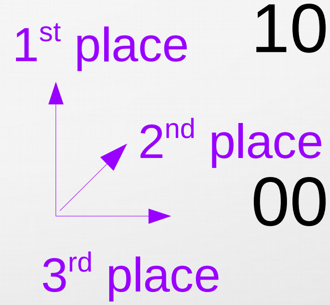
$n = 1, 2, 3, 4, ...$

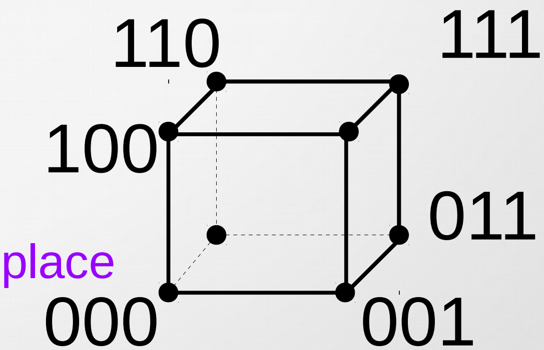
Q_1





Q_2

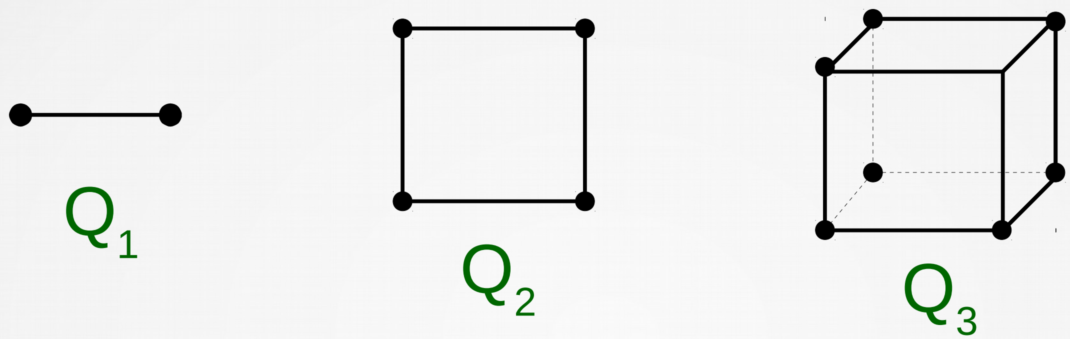




Q_3

10.2 *Graph terminology and special types of graphs*

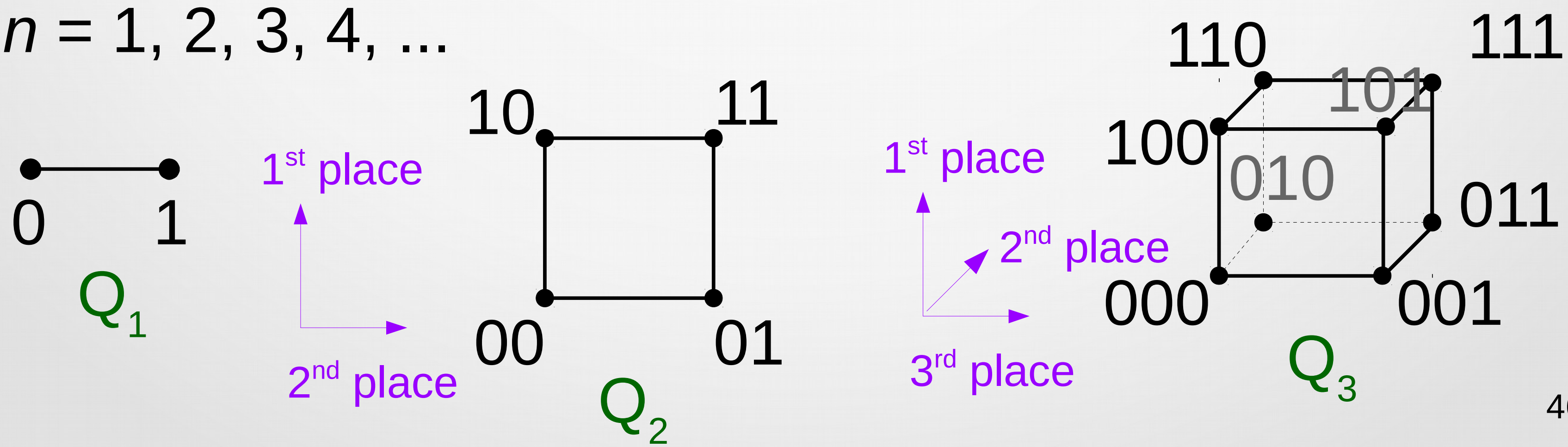
Special Simple Graphs: *n*-Cubes



Graph of Q_n has 2^n vertices

n-Cubes can represent *bit strings* of length *n*

n = 1, 2, 3, 4, ...

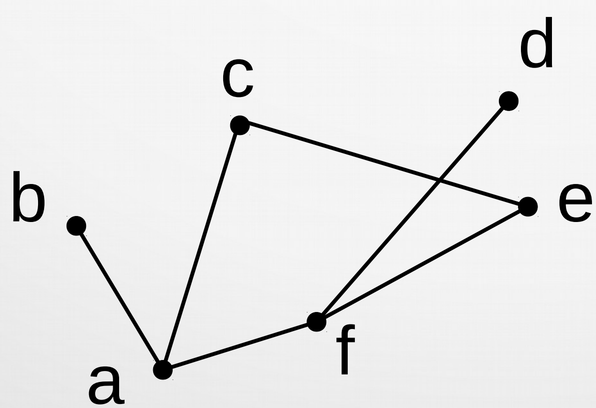


10.2 *Graph terminology and special types of graphs*

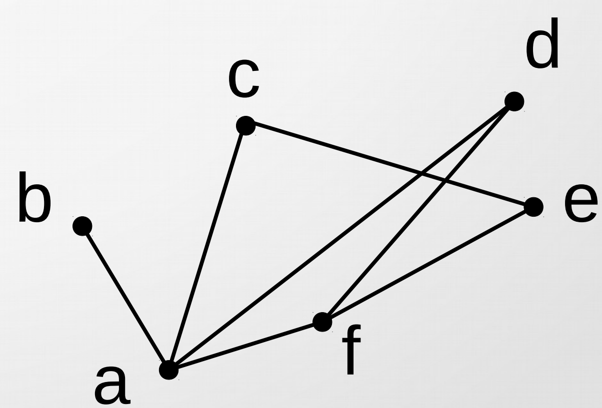
Bipartite graphs

[Def] a simple graph is called *bipartite* if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (i.e. no same-set vertices connections). We call the pair (V_1, V_2) a *bipartition* of V.

Example:



bipartite graph



not a bipartite graph

10.2 *Graph terminology and special types of graphs*

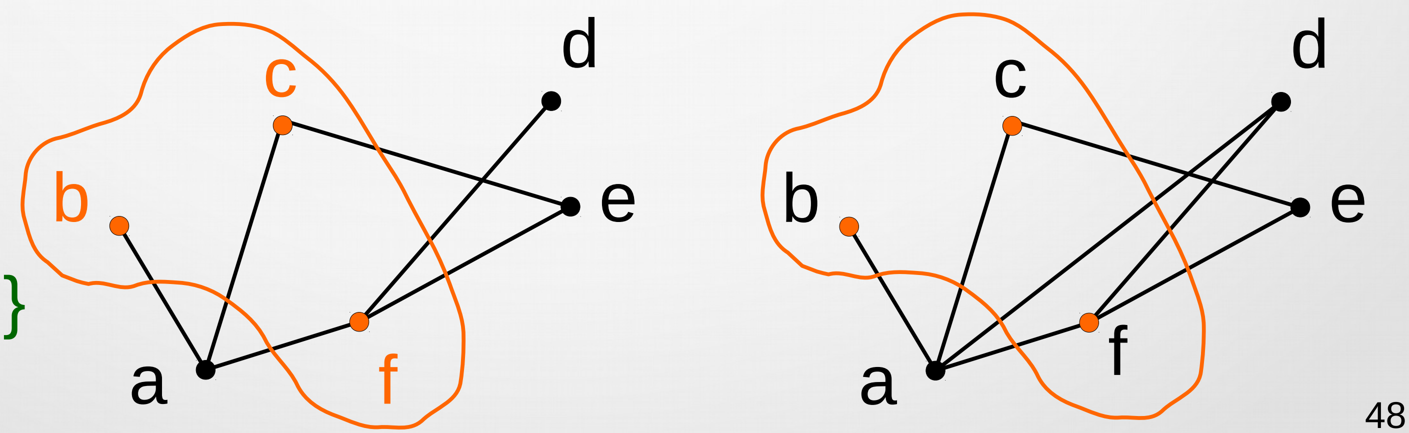
Bipartite graphs

[Def] a simple graph is called *bipartite* if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (i.e. no same-set vertices connections). We call the pair (V_1, V_2) a *bipartition* of V.

Example:

$V_1 = \{b,c,f\}$

$V_2 = \{a,d,e\}$



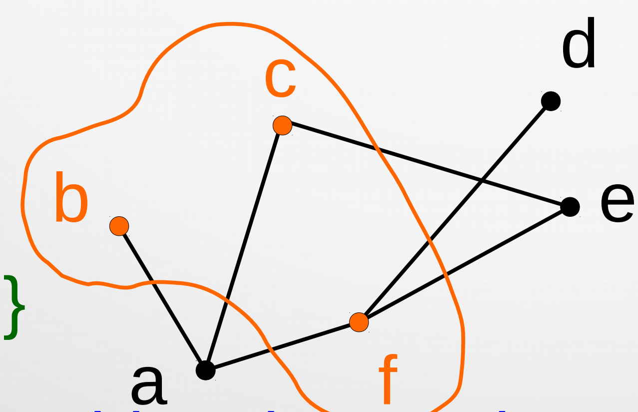
10.2 *Graph terminology and special types of graphs*

Bipartite graphs

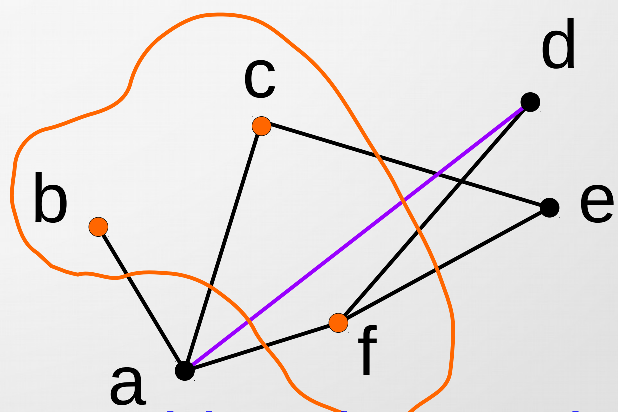
[Def] a simple graph is called *bipartite* if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (i.e. no same-set vertices connections).
We call the pair (V_1, V_2) a *bipartition* of V.

Example:

$V_1 = \{b,c,f\}$
$V_2 = \{a,d,e\}$



bipartite graph

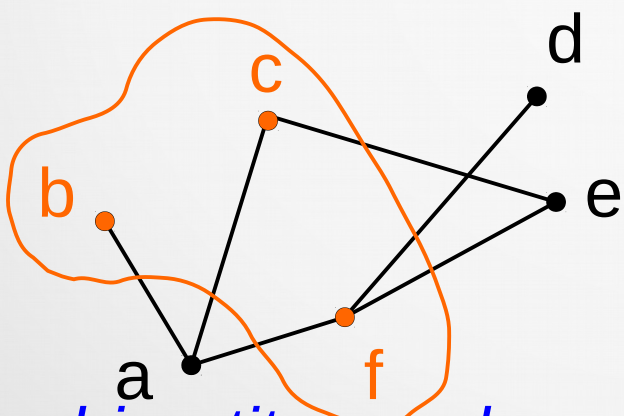


not a bipartite graph

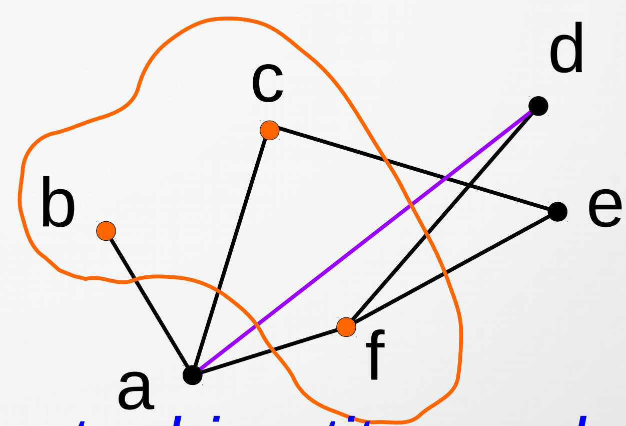
10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertics have the same color.



bipartite graph



not a bipartite graph

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices have the same color.

Proof: $(\rightarrow)$: Assume that G = (V,E) is a bipartite graph, so there exist $V_1 \subset V$ and $V_2 \subset V$, such that $V_1 \cap V_2 = \varnothing$, $V = V_1 \cup V_2$, and every edge in E connects a vertex from V_1 to vertex from V_2.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices have the same color.

Proof: $(\rightarrow)$: Assume that G = (V,E) is a bipartite graph, so there exist $V_1 \subset V$ and $V_2 \subset V$, such that $V_1 \cap V_2 = \varnothing$, $V = V_1 \cup V_2$, and every edge in E connects a vertex from V_1 to vertex from V_2.
Let's color all vertices from V_1 with one color and all vertices from V_2 with other color. No adjacent vertices will have the same color.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices have the same color.

Proof: ($\leftarrow$): Assume that it is possible to assign colors to the vertices of the graph using only two colors so that no two adjacent vertices have the same color.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices have the same color.

Proof: (←): Assume that it is possible to assign colors to the vertices of the graph using only two colors so that no two adjacent vertices have the same color.
Let V_1 be the set of vertices of color 1, and
V_2 be the set of vertices of color 1, then

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices have the same color.

Proof: (←): Assume that it is possible to assign colors to the vertices of the graph using only two colors so that no two adjacent vertices have the same color.
Let V_1 be the set of vertices of color 1, and
V_2 be the set of vertices of color 1, then
$V_1 \cap V_2 = \varnothing$, $V = V_1 \cup V_2$, and every edge in E connects a vertex from V_1 to vertex from V_2, therefore, G is bipartite.

q.e.d.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

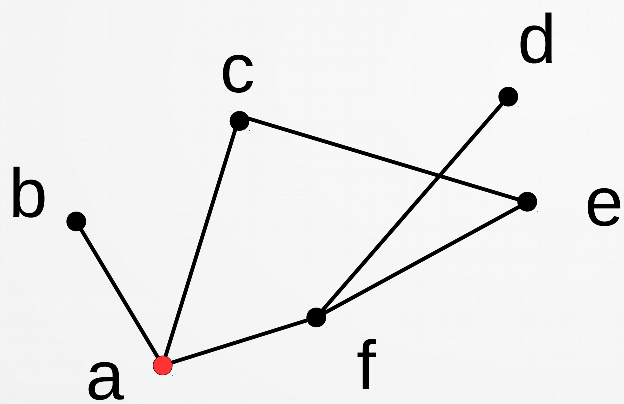
[Theorem] a simple graph is *bipartite* if and only if (iff) it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertics have the same color.

Theorem can be used to determine whether the graph is bipartite, and find the partition of bipartite graphs.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

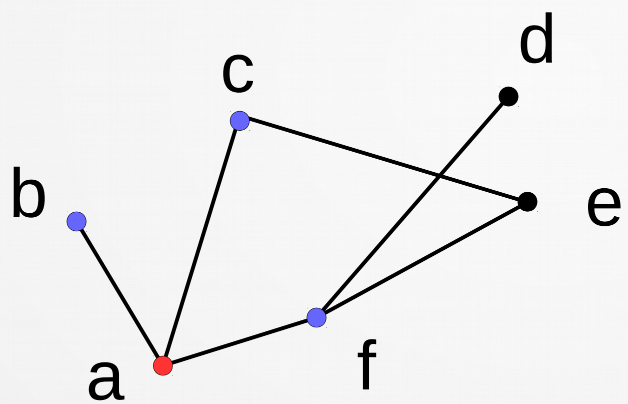
Start with any vertex (with edge). Assign one color to it, say red.



10.2 *Graph terminology and special types of graphs*

Bipartite graphs

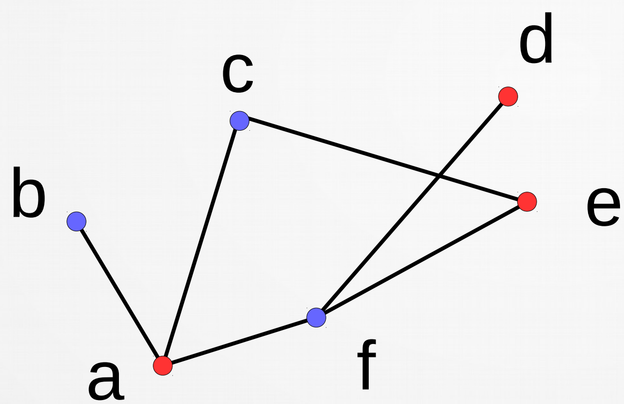
Assign other color, say blue, to all vertices that are adjacent to it.



10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Then assign the red color to all vertices that are adjacent to vertices of blue color.

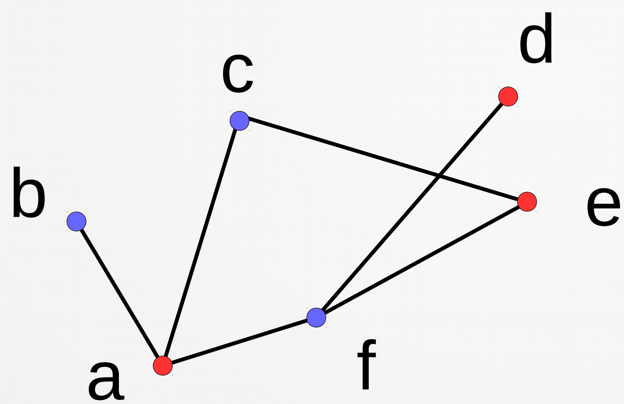


10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Done!

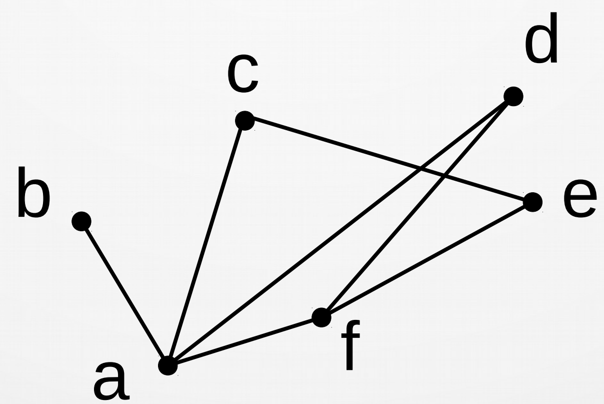
$V_1 = \{a,d,e\}$, $V_2 = \{b,c,f\}$



10.2 *Graph terminology and special types of graphs*

Bipartite graphs

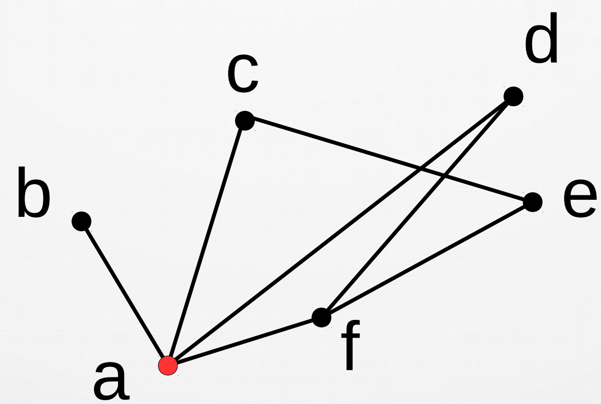
Let's try the other one:



10.2 *Graph terminology and special types of graphs*

Bipartite graphs

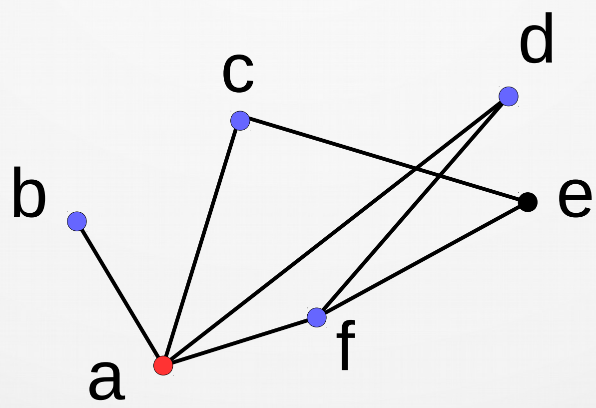
Let's try the other one: start with marking a red



10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Let's try the other one: continue by with marking vertices adjacent to a blue

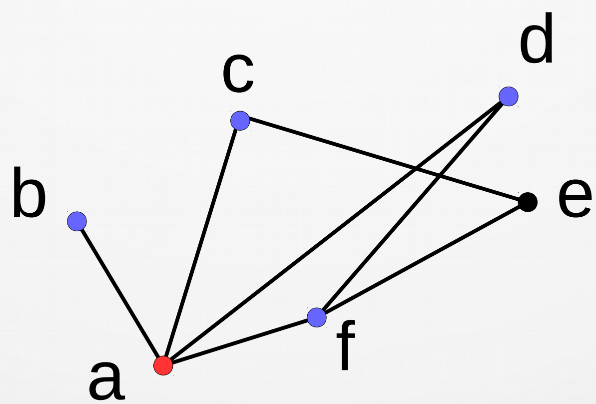


Problem: d and f are adjacent!!!

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Let's try the other one: continue by with marking vertices adjacent to a blue

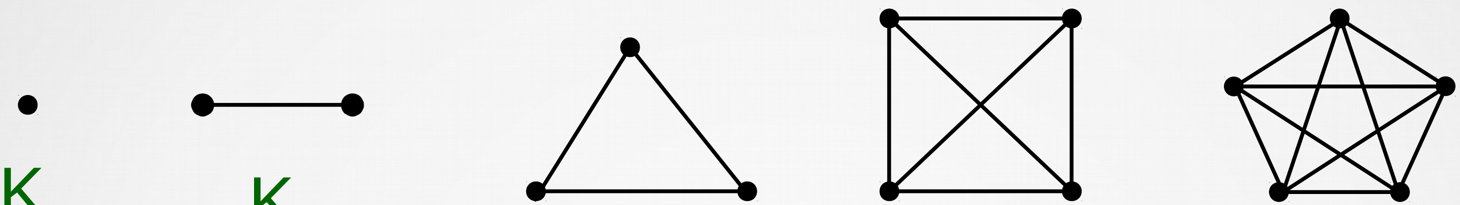


not a bipartite graph

Problem: d and f are adjacent!!!

10.2 *Graph terminology and special types of graphs*

Complete bipartite graphs



K_1 K_2 K_3 K_4 K_5

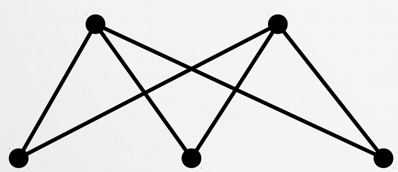
A complete graph K_n on *n* vertices is a simple graph that contains exactly one edge between each pair of distinct vertices. $n = 1, 2, 3, 4, \ldots$

A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets of m and n vertices, respectively with an edge between two vertices iff one vertex is in the first subset, and the other is in the second.

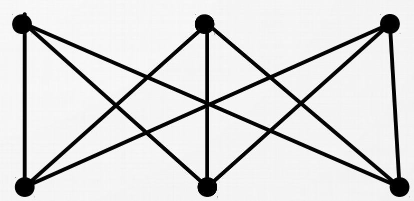
10.2 *Graph terminology and special types of graphs*

Complete bipartite graphs

A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets of m and n vertices, respectively with an edge between two vertices iff one vertex is in the first subset, and the other is in the second.



$K_{2,3}$



$K_{3,3}$

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Bipartite graphs can be used to model many types of applications that involve matching the elements of one set to elements of another.

Example: Suppose there are m employees in a group and n different jobs than need to be done, where $m \geq n$. Each employee is trained to do one or more of these jobs. We want to assign *one employee to each job*.

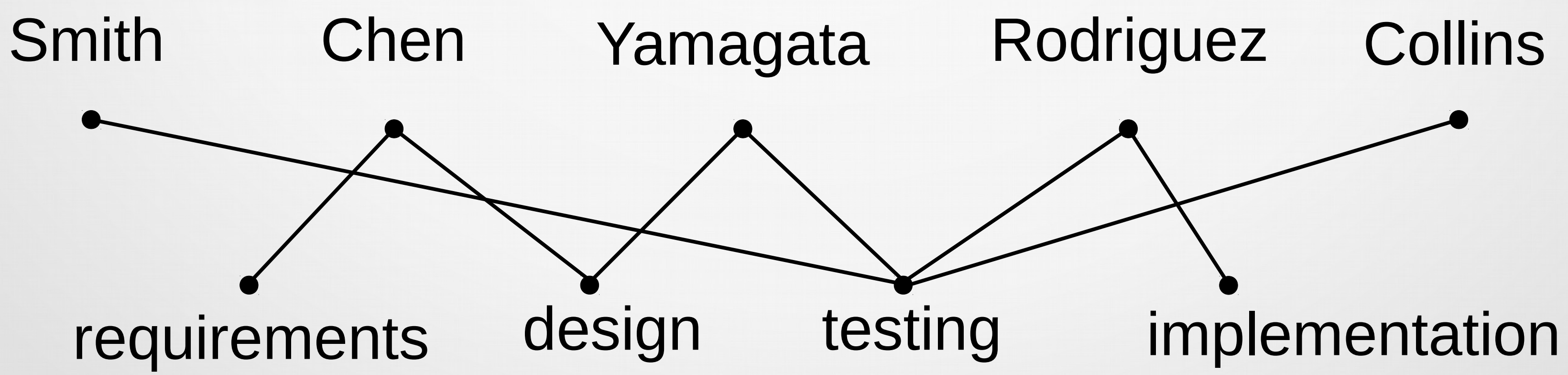
Use bipartite graphs to model the situation!

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

m employees
n different jobs, where $m \geq n$

Vertices: employees and jobs
Edges: connect employees with jobs he/she can do

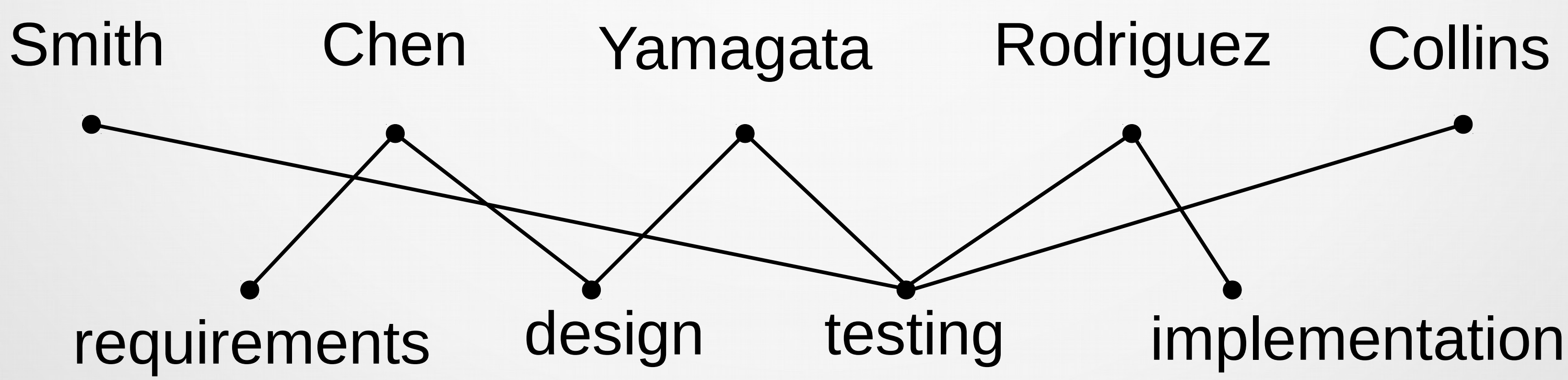


10.2 *Graph terminology and special types of graphs*

Bipartite graphs

m employees
n different jobs, where $m \geq n$

Vertices: employees and jobs
Edges: connect employees with jobs he/she can do



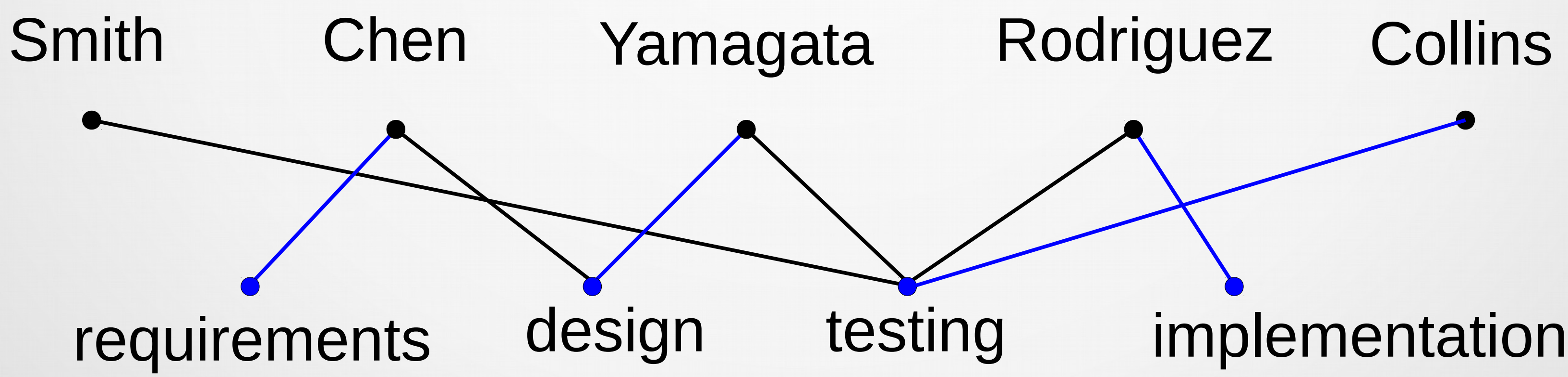
A *matching* M is a set of edges from simple graph G, such that for any {s,t} and {u,v} $\in$ M, s,t,u, and v are different vertices.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

m employees
n different jobs, where $m \geq n$

Vertices: employees and jobs
Edges: connect employees with jobs he/she can do



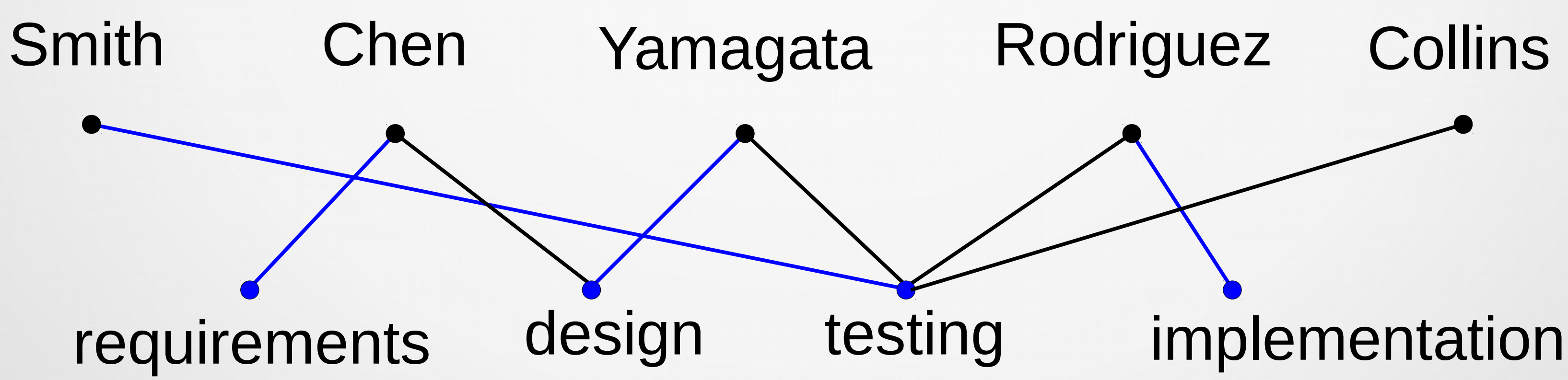
A *matching* M is a set of edges from simple graph G, such that for any {s,t} and {u,v} $\in$ M, s,t,u, and v are different vertices.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

m employees
n different jobs, where $m \geq n$

Vertices: employees and jobs
Edges: connect employees with jobs he/she can do



A *matching* M is a set of edges from simple graph G, such that for any {s,t} and {u,v} ∈ M, s,t,u, and v are different vertices.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

A matching M is a set of edges from simple graph G, such that for any {s,t} and {u,v} $\in$ M, s,t,u, and v are different vertices.

A vertex that is an endpoint of an edge in matching M is said to be *matched* in M; otherwise it is said to be *unmatched*.

A *maximum matching* is a matching with the largest number of edges.

A *complete matching* M from V_1 to V_2 is the one where $|M| = |V_1|$.

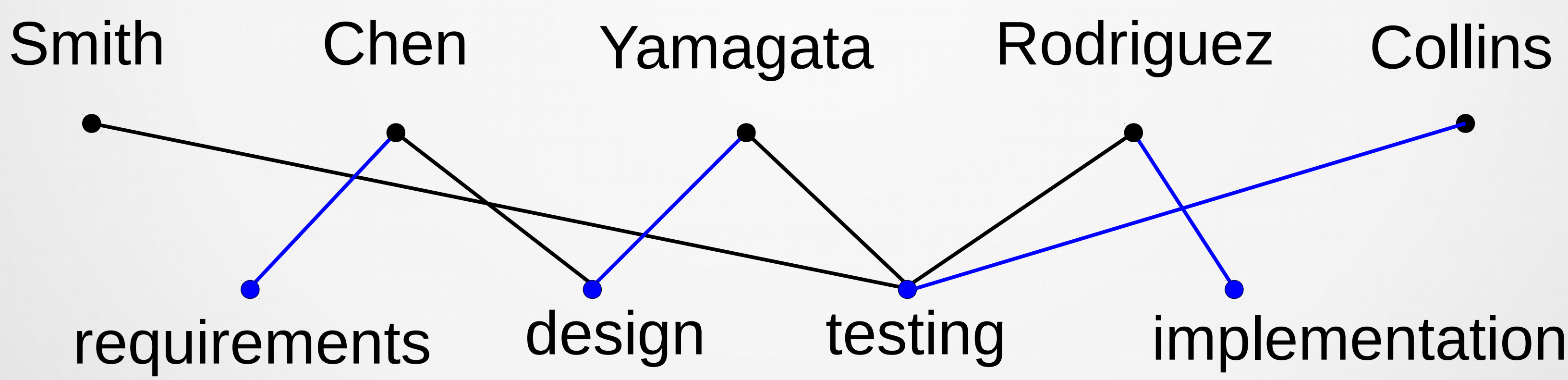
10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Goal: one employee to each job

V_1 = jobs

V_2 = employees



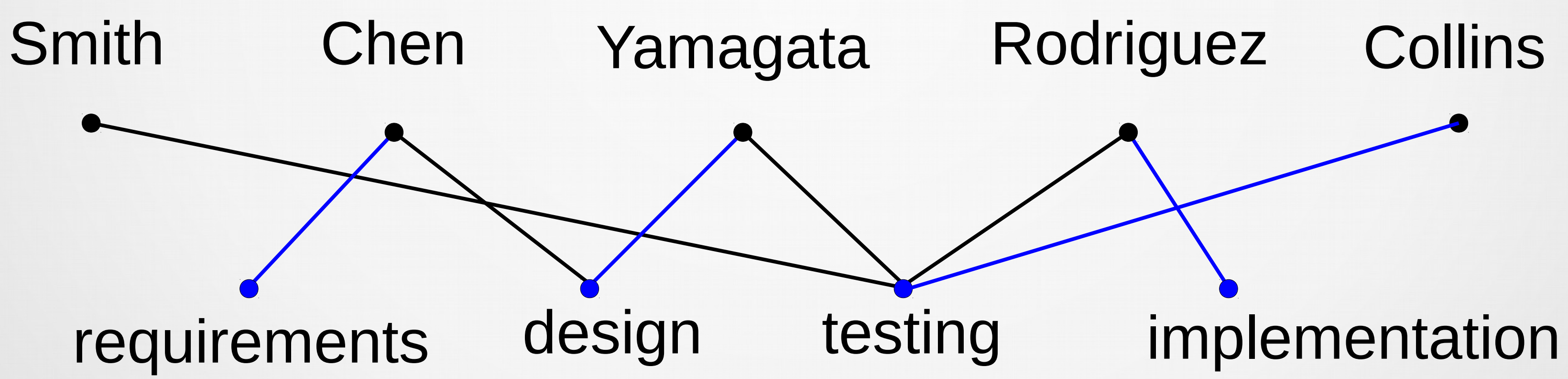
This is a complete matching, because $|V_1| = 4$ and we see 4 blue edges.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Goal: one employee to each job, and all employees should get a job

V_1 = employees $\qquad\qquad$ V_2 = jobs



This is not a complete matching, because $|V_1| = 5$ and we see only 4 blue edges.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Example: marriages on an Island

Suppose that there are m men and n women on an island. Each person has a list of members of the opposite gender acceptable as a spouse.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Example: marriages on an Island

Suppose that there are m men and n women on an island. Each person has a list of members of the opposite gender acceptable as a spouse.

We construct a bipartite graph $G = (V,E)$, where V_1 is the set of men, and V_2 is the set of women. Edges are between people that find each other acceptable as a spouse.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Example: marriages on an Island

Suppose that there are m men and n women on an island. Each person has a list of members of the opposite gender acceptable as a spouse.

We construct a bipartite graph $G = (V,E)$, where V_1 is the set of men, and V_2 is the set of women. Edges are between people that find each other acceptable as a spouse.

A *maximum matching* is the largest possible set of couples to merry.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Example: marriages on an Island

Suppose that there are m men and n women on an island. Each person has a list of members of the opposite gender acceptable as a spouse.

We construct a bipartite graph $G = (V,E)$, where V_1 is the set of men, and V_2 is the set of women. Edges are between people that find each other acceptable as a spouse.

A *maximum matching* is the largest possible set of couples to merry.

A *complete matching* of set V_1 is a set of all couples where every man is can merry, but possible not all women.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Necessary and sufficient conditions for complete matchings:

[Theorem] Hall's marriage theorem
The bipartite graph $G = (V,E)$ with bipartition (V_1,V_2) has a *complete matching* from V_1 to V_2 if and only if (iff) $|N(A)| \geq |A|$ for all subsets A of V_1.

No proof.

10.2 *Graph terminology and special types of graphs*

Bipartite graphs

Example: marriages on an Island

Suppose that there are m men and n women on an island. Each person has a list of members of the opposite gender acceptable as a spouse.

We construct a bipartite graph $G = (V,E)$, where V_1 is the set of men, and V_2 is the set of women. Edges are between people that find each other acceptable as a spouse.

A *maximum matching* is the largest possible set of couples to merry.

A *complete matching* of set V_1 is a set of all couples where every man is can merry, but possible not all women.