Lecture 5: Graphs.

Graphs!

Lecture 5: Graphs.

Graphs!
Euler

Lecture 5: Graphs.

Graphs!
- Euler
- Definitions: model.

Lecture 5: Graphs.

Graphs!
- Euler
- Definitions: model.
- Fact!

Lecture 5: Graphs.

Graphs!
- Euler
- Definitions: model.
- Fact!
- Euler Again!!

Lecture 5: Graphs.

Graphs!
- Euler
- Definitions: model.
- Fact!
- Euler Again!!

Lecture 5: Graphs.

- Graphs!
  - Euler
  - Definitions: model.
  - Fact!
  - Euler Again!!
- Planar graphs.

Lecture 5: Graphs.

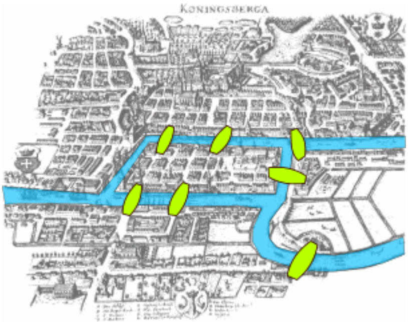
Graphs!
- Euler
- Definitions: model.
- Fact!
- Euler Again!!

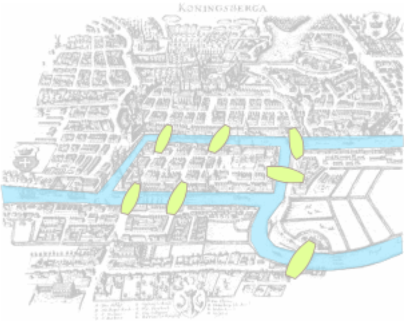
Planar graphs.
- Euler Again!!!!

Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

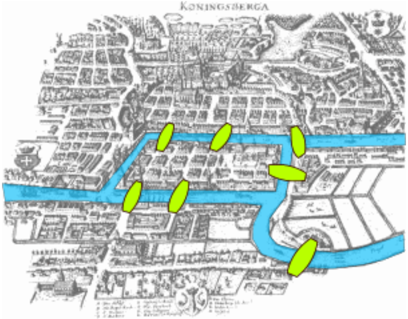


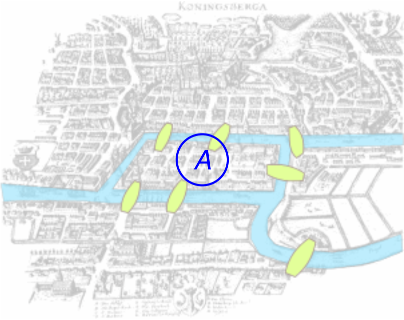


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

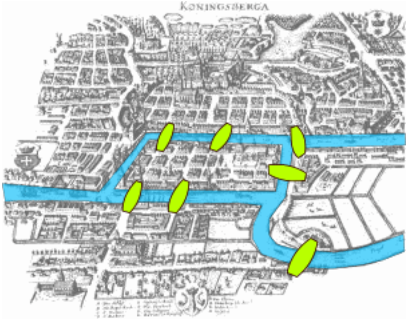


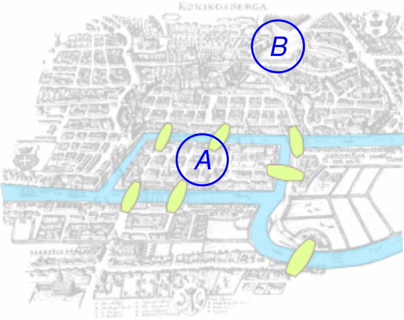


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

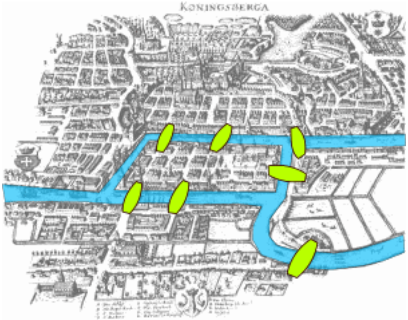


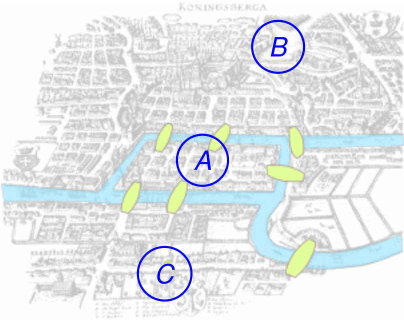


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

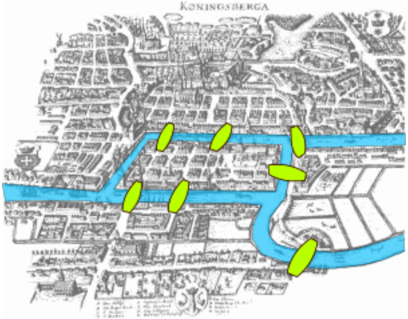


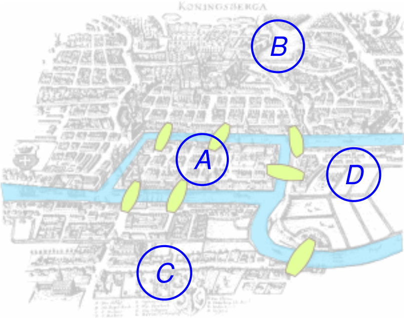


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

"Konigsberg bridges" by Bogdan Giuşcă - License.

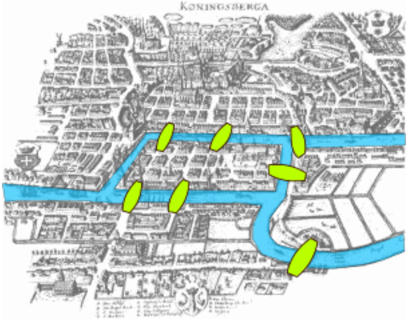


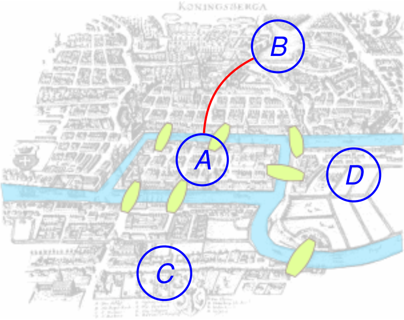


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

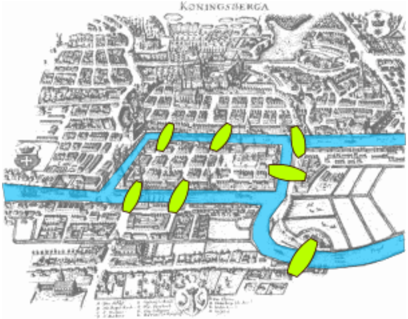


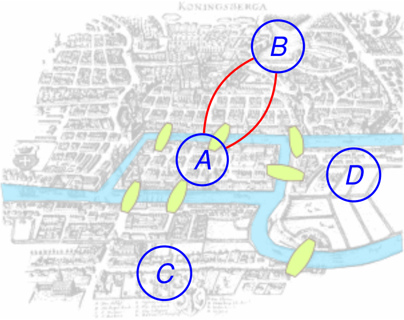


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

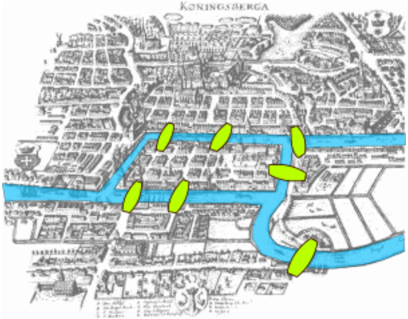


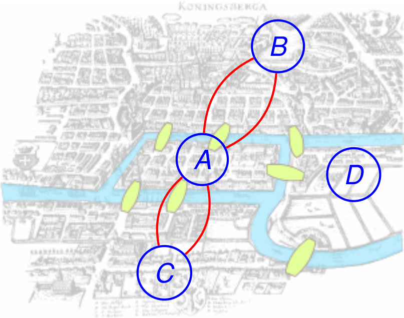


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

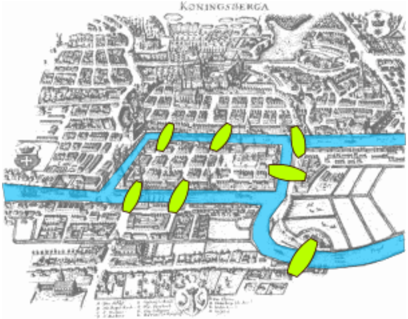


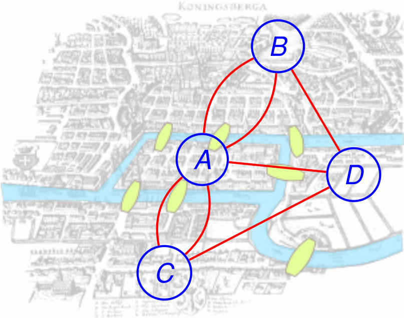


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.

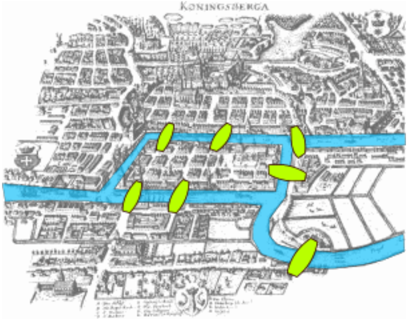


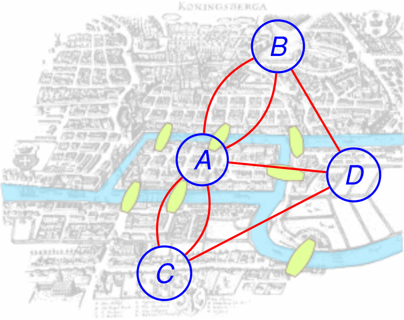


Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

"Konigsberg bridges" by Bogdan Giuşcă - License.



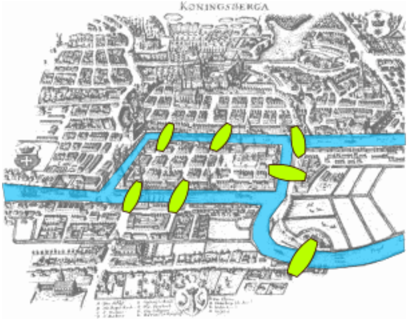


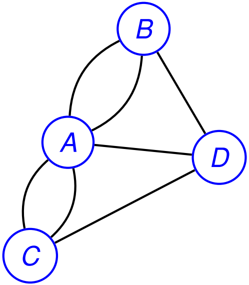
Can you draw a tour in the graph where you visit each edge once?

Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.



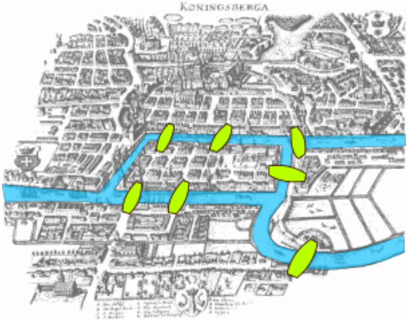


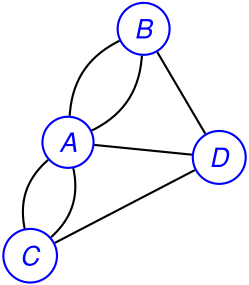
Can you draw a tour in the graph where you visit each edge once?
Yes?

Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

“Konigsberg bridges” by Bogdan Giuşcă - License.



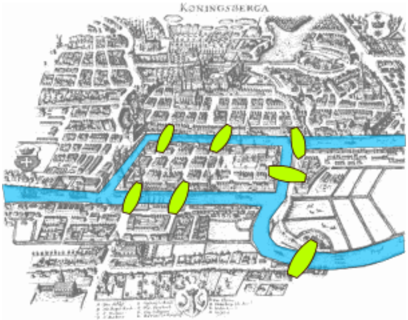


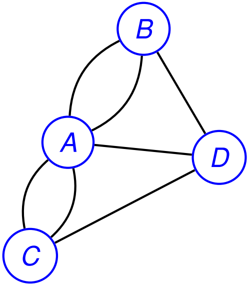
Can you draw a tour in the graph where you visit each edge once?
Yes? No?

Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

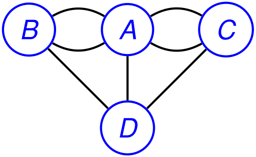
“Konigsberg bridges” by Bogdan Giuşcă - License.





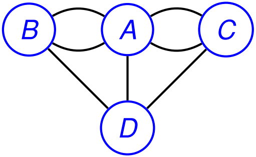
Can you draw a tour in the graph where you visit each edge once?
Yes? No?
We will see!

Graphs: formally.



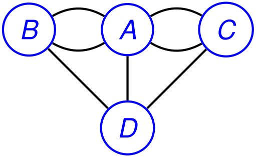
Graph:

Graphs: formally.



Graph: $G = (V, E)$.

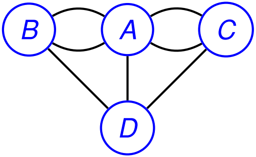
Graphs: formally.



Graph: $G = (V, E)$.

V - set of vertices.

Graphs: formally.

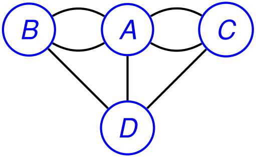


Graph: $G = (V, E)$.

V - set of vertices.

$\{A, B, C, D\}$

Graphs: formally.



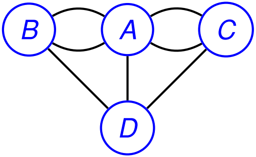
Graph: $G = (V, E)$.

V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ -

Graphs: formally.



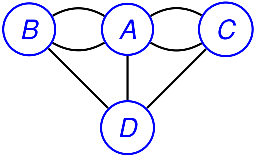
Graph: $G = (V, E)$.

V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

Graphs: formally.



Graph: $G = (V, E)$.

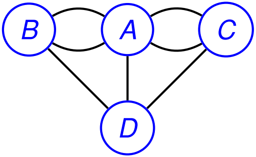
V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

$\{\{A, B\}$

Graphs: formally.



Graph: $G = (V, E)$.

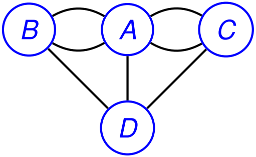
V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

$\{\{A, B\}, \{A, B\}$

Graphs: formally.



Graph: $G = (V, E)$.

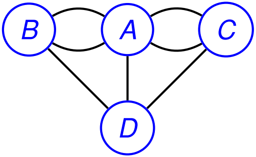
V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

$\{\{A, B\}, \{A, B\}, \{A, C\},$

Graphs: formally.



Graph: $G = (V, E)$.

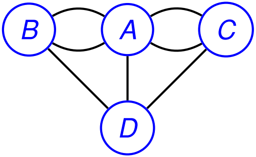
V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

$\{\{A, B\}, \{A, B\}, \{A, C\}, \{B, C\}, \{B, D\}, \{B, D\}, \{C, D\}\}$.

Graphs: formally.



Graph: $G = (V, E)$.

V - set of vertices.

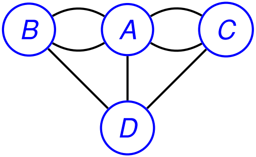
$\{A, B, C, D\}$

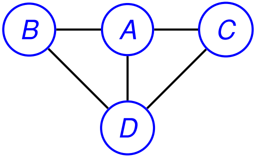
$E \subseteq V \times V$ - set of edges.

$\{\{A, B\}, \{A, B\}, \{A, C\}, \{B, C\}, \{B, D\}, \{B, D\}, \{C, D\}\}$.

For CS 70, usually simple graphs.

Graphs: formally.





Graph: $G = (V, E)$.

V - set of vertices.

$\{A, B, C, D\}$

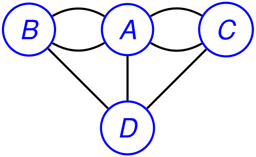
$E \subseteq V \times V$ - set of edges.

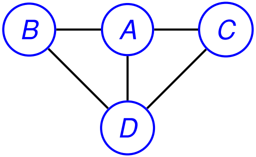
$\{\{A, B\}, \{A, B\}, \{A, C\}, \{B, C\}, \{B, D\}, \{B, D\}, \{C, D\}\}$.

For CS 70, usually simple graphs.

No parallel edges.

Graphs: formally.





Graph: $G = (V, E)$.

V - set of vertices.

$\{A, B, C, D\}$

$E \subseteq V \times V$ - set of edges.

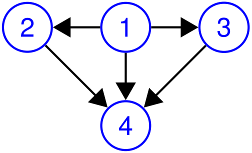
$\{\{A, B\}, \{A, B\}, \{A, C\}, \{B, C\}, \{B, D\}, \{B, D\}, \{C, D\}\}$.

For CS 70, usually simple graphs.

No parallel edges.

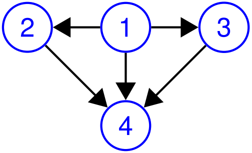
Multigraph above.

Directed Graphs



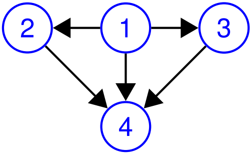
$G = (V, E)$.

Directed Graphs



$G = (V, E)$.
V - set of vertices.

Directed Graphs

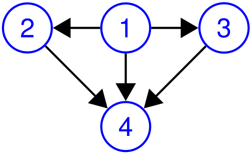


$G = (V, E)$.

V - set of vertices.

$\{1, 2, 3, 4\}$

Directed Graphs



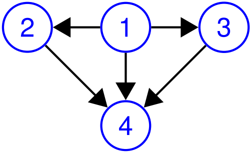
$G = (V, E)$.

V - set of vertices.

$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

Directed Graphs



$G = (V, E)$.

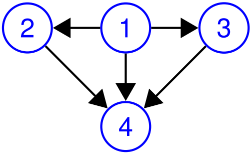
V - set of vertices.

$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

$\{(1, 2),$

Directed Graphs



$G = (V, E)$.

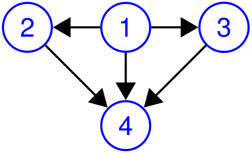
V - set of vertices.

$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

$\{(1, 2), (1, 3),$

Directed Graphs



$G = (V, E)$.

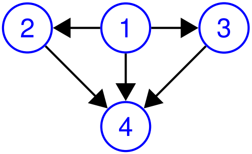
V - set of vertices.

$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

$\{(1, 2), (1, 3), (1, 4),$

Directed Graphs



$G = (V, E)$.

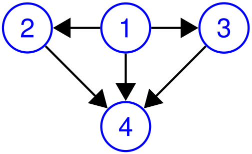
V - set of vertices.

$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

Directed Graphs



$G = (V, E)$.

V - set of vertices.

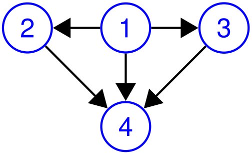
$\{1, 2, 3, 4\}$

E ordered pairs of vertices.

$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.

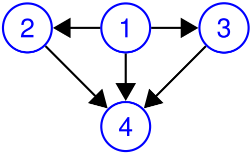
Directed Graphs



$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament:

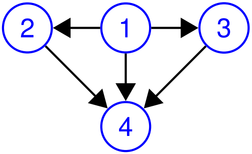
Directed Graphs



$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2,

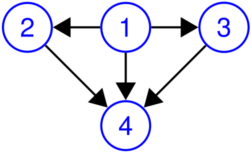
Directed Graphs



$G = (V,E)$.
V - set of vertices.
$\{1,2,3,4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence:

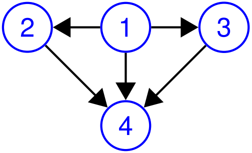
Directed Graphs



$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2,

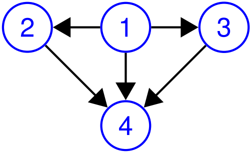
Directed Graphs



$G = (V,E)$.
V - set of vertices.
$\{1,2,3,4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Directed Graphs

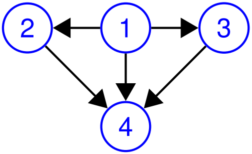


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1, 2), (1, 3), (1, 4), (2, 4), (3, 4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network:

Directed Graphs

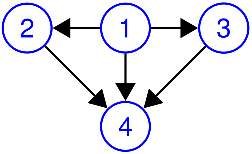


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed?

Directed Graphs



$G = (V, E)$.

V - set of vertices.

$\{1, 2, 3, 4\}$

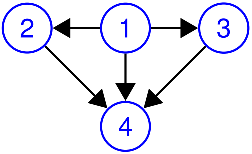
E ordered pairs of vertices.

$\{(1, 2), (1, 3), (1, 4), (2, 4), (3, 4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?

Directed Graphs

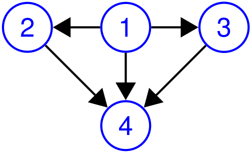


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends.

Directed Graphs

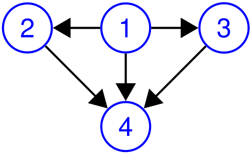


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2), (1,3), (1,4), (2,4), (3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends. Undirected.

Directed Graphs

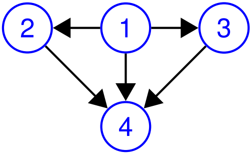


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1, 2), (1, 3), (1, 4), (2, 4), (3, 4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends. Undirected.
Likes.

Directed Graphs

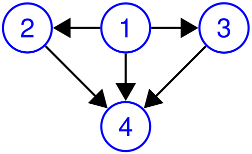


$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2), (1,3), (1,4), (2,4), (3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends. Undirected.
Likes. Directed.

Directed Graphs



$G = (V, E)$.
V - set of vertices.
$\{1, 2, 3, 4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends. Undirected.
Likes. Directed.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

Graph Concepts and Definitions.

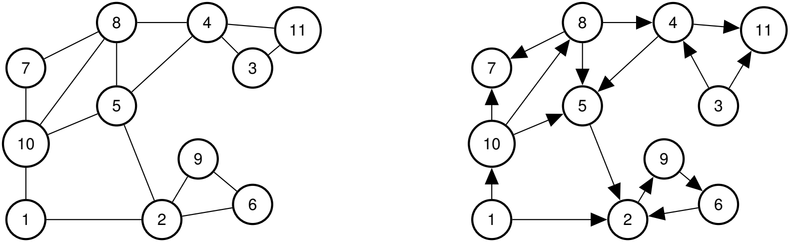
Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree

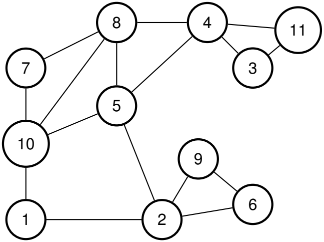


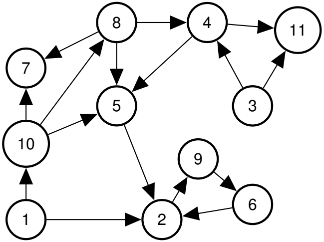
Neighbors of 10?

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



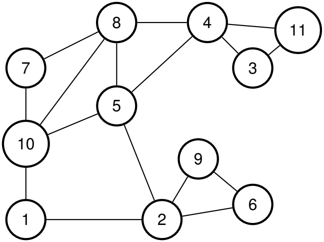


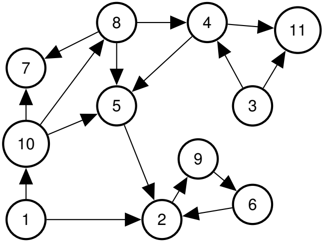
Neighbors of 10? 1,

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



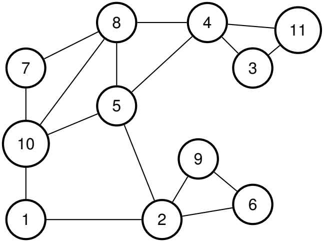


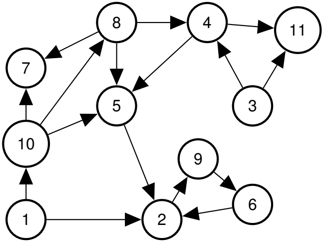
Neighbors of 10? 1,5,

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



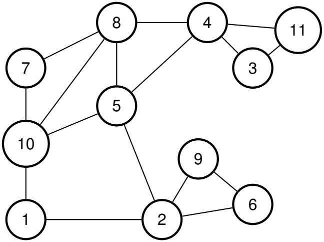


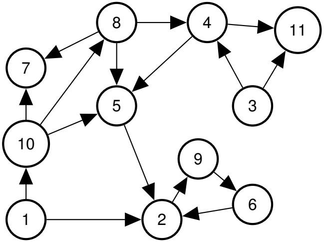
Neighbors of 10? 1,5,7,

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



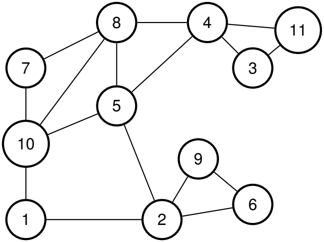


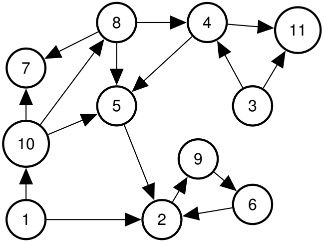
Neighbors of 10? 1,5,7, 8.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





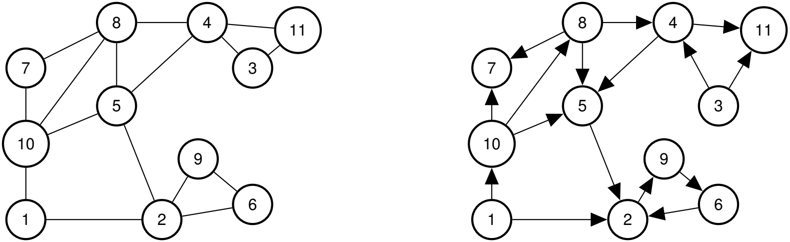
Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



Neighbors of 10? 1,5,7, 8.

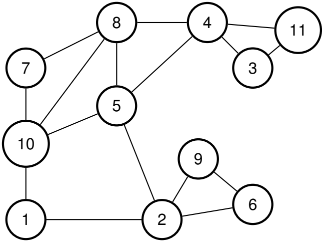
u is neighbor of v if $\{u, v\} \in E$.

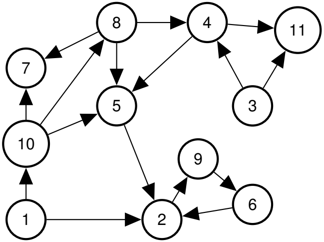
Edge $\{10, 5\}$ is incident to

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

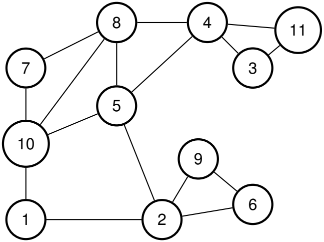
Edge $\{u, v\}$ is incident to u and v.

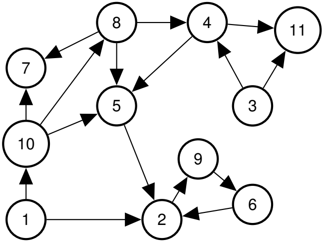
Degree of vertex 1?

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

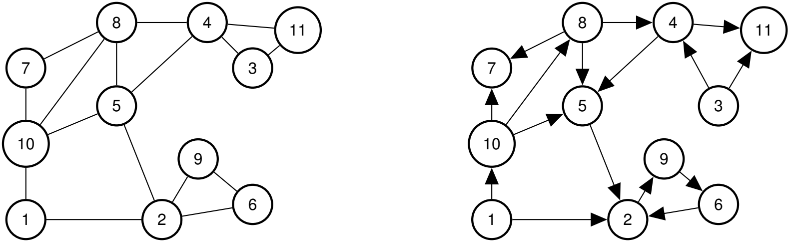
Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

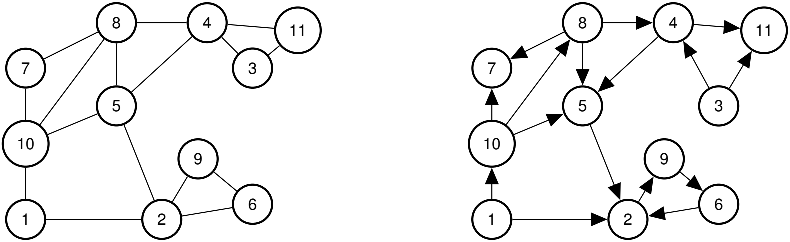
Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

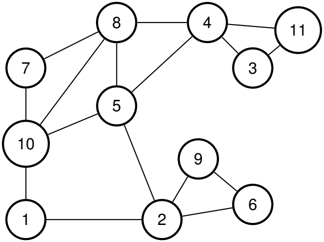
Degree of vertex u is number of incident edges.

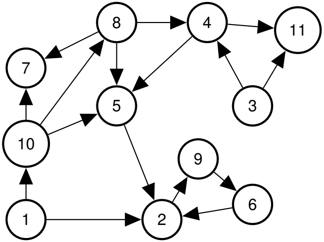
Equals number of neighbors in simple graph.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

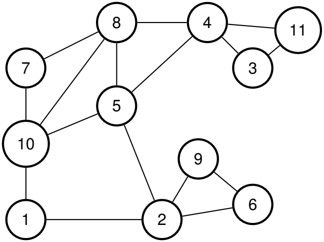
Degree of vertex u is number of incident edges.

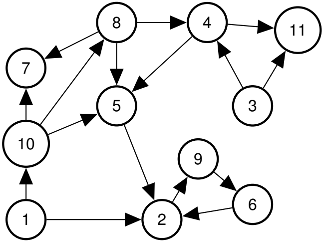
Equals number of neighbors in simple graph.

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

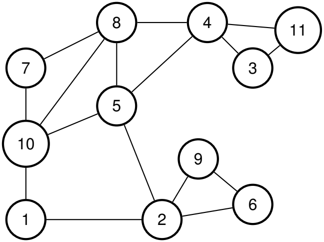
Equals number of neighbors in simple graph.

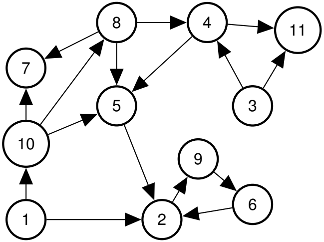
Directed graph?

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Equals number of neighbors in simple graph.

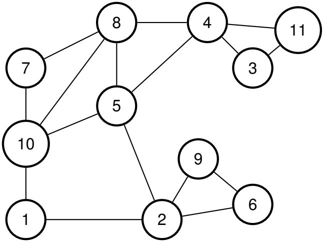
Directed graph?

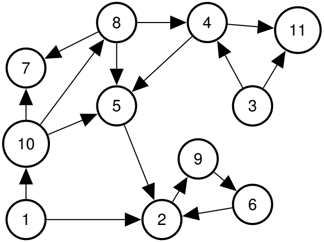
In-degree of 10?

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Equals number of neighbors in simple graph.

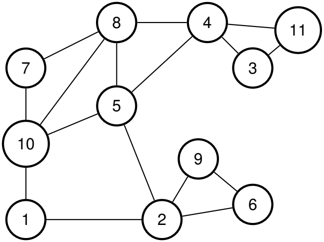
Directed graph?

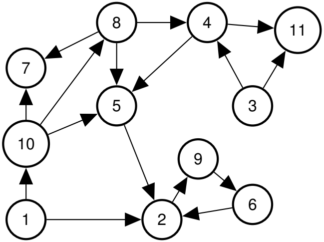
In-degree of 10? 1

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Equals number of neighbors in simple graph.

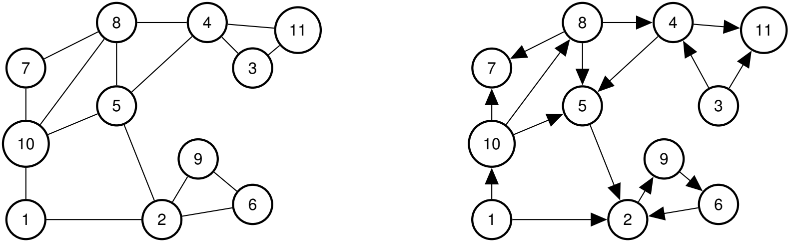
Directed graph?

In-degree of 10? 1   Out-degree of 10?

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree



Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Equals number of neighbors in simple graph.

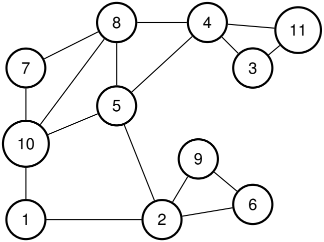
Directed graph?

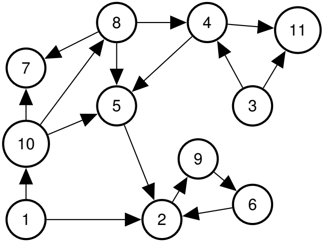
In-degree of 10? 1   Out-degree of 10? 3

Graph Concepts and Definitions.

Graph: $G = (V, E)$

neighbors, adjacent, degree, incident, in-degree, out-degree





Neighbors of 10? 1,5,7, 8.

u is neighbor of v if $\{u, v\} \in E$.

Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.

Edge $\{u, v\}$ is incident to u and v.

Degree of vertex 1? 2

Degree of vertex u is number of incident edges.

Equals number of neighbors in simple graph.

Directed graph?

In-degree of 10? 1   Out-degree of 10? 3

Quick Proof.

The sum of the vertex degrees is equal to

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)!

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

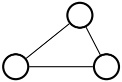
Not (A)! Triangle.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



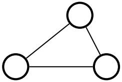
Not (B)!

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



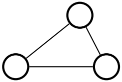
Not (B)! Triangle.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



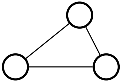
Not (B)! Triangle.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

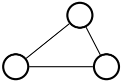
What?

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

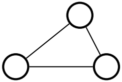
What? For triangle number of edges is 3, the sum of degrees is 6.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

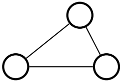
Could it always be...

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

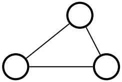
Could it always be...$2|E|$? ..

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

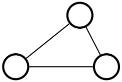
Could it always be...$2|E|$? ..or $2|V|$?

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

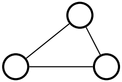
How many incidences does each edge contribute?

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

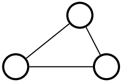
How many incidences does each edge contribute? 2.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

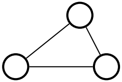
How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

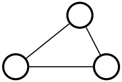
How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!
What is degree v?

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

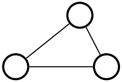
How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!
What is degree v? incidences contributed to v!

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

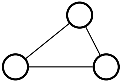
How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!
What is degree v? incidences contributed to v!
sum of degrees is total incidences

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

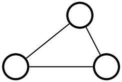
How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!
What is degree v? incidences contributed to v!
sum of degrees is total incidences ... or $2|E|$.

Quick Proof.

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

Not (A)! Triangle.



Not (B)! Triangle.

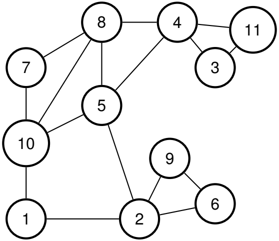
What? For triangle number of edges is 3, the sum of degrees is 6.

Could it always be...$2|E|$? ..or $2|V|$?

How many incidences does each edge contribute? 2.
$2|E|$ incidences are contributed in total!
What is degree v? incidences contributed to v!
sum of degrees is total incidences ... or $2|E|$.

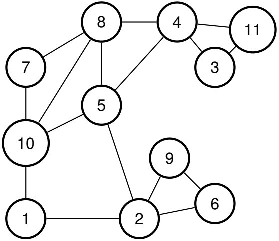
Thm: Sum of vertex degress is $2|E|$.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

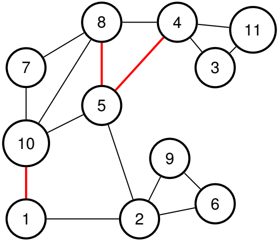
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path?

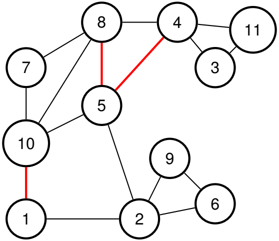
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$?

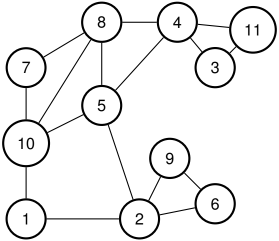
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!

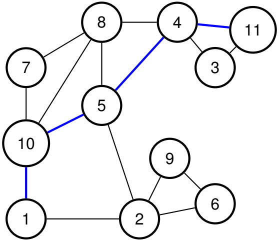
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1, 10\}$, $\{8, 5\}$, $\{4, 5\}$? No!
Path?

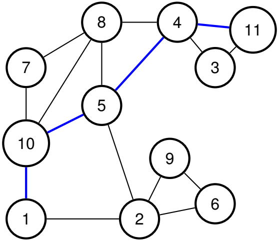
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$?

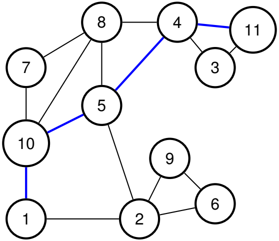
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Paths, walks, cycles, tour.

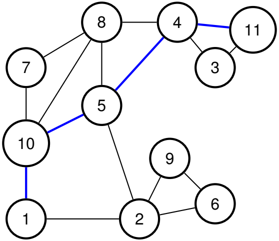


A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Paths, walks, cycles, tour.



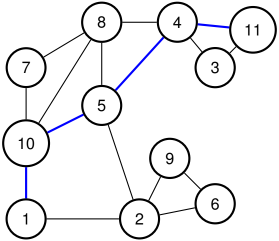
A path in a graph is a sequence of edges.

Path? $\{1, 10\}$, $\{8, 5\}$, $\{4, 5\}$? No!
Path? $\{1, 10\}$, $\{10, 5\}$, $\{5, 4\}$, $\{4, 11\}$? Yes!

Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Quick Check!

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

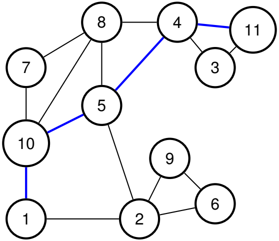
Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!

Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.

Quick Check! Length of path?

Paths, walks, cycles, tour.

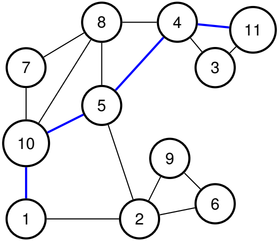


A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Quick Check! Length of path? k vertices

Paths, walks, cycles, tour.

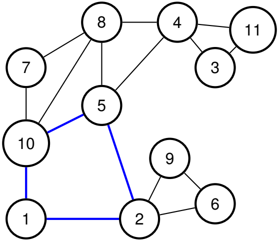


A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!

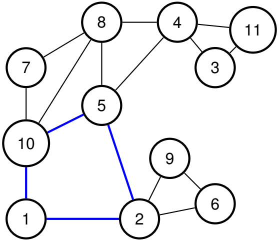
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.

Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

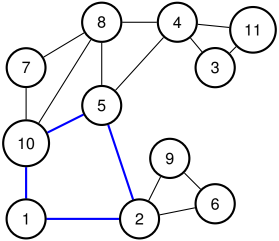
Path? $\{1,10\}, \{8,5\}, \{4,5\}$? No!
Path? $\{1,10\}, \{10,5\}, \{5,4\}, \{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.

Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle?

Paths, walks, cycles, tour.

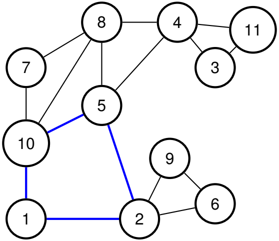


A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!

Paths, walks, cycles, tour.

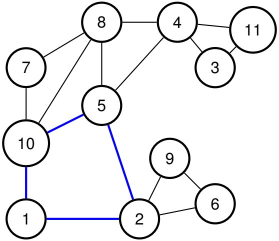


A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple.

Paths, walks, cycles, tour.



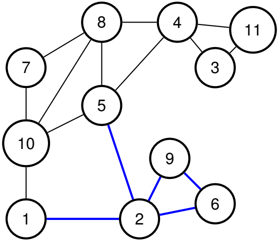
A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!

Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.

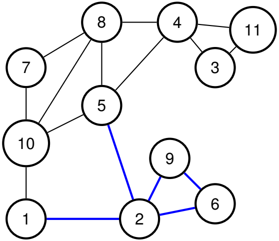
Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!

Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.

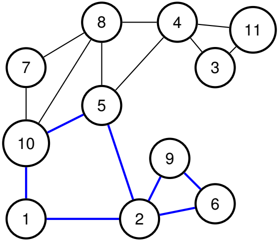
Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!

Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

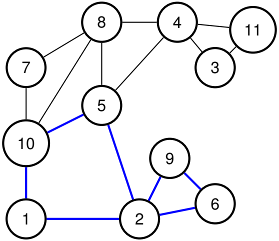
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Tour is walk that starts and ends at the same node.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

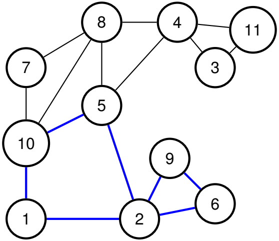
Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Tour is walk that starts and ends at the same node.

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\ldots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

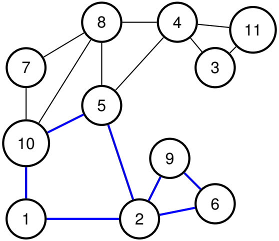
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Tour is walk that starts and ends at the same node.

Quick Check!

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1,v_2),(v_2,v_3),\dots(v_{k-1},v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

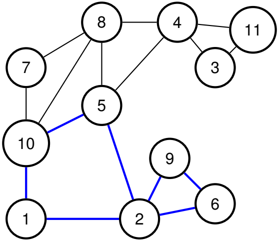
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

Tour is walk that starts and ends at the same node.

Quick Check!
Path is to Walk as Cycle is to ??

Paths, walks, cycles, tour.



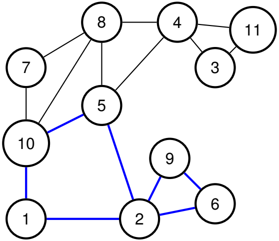
A path in a graph is a sequence of edges.

Path? $\{1,10\}, \{8,5\}, \{4,5\}$? No!
Path? $\{1,10\}, \{10,5\}, \{5,4\}, \{4,11\}$? Yes!

Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!
Walk is sequence of edges with possible repeated vertex or edge.
Tour is walk that starts and ends at the same node.

Quick Check!
Path is to Walk as Cycle is to ?? Tour!

Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.

Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!

Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.

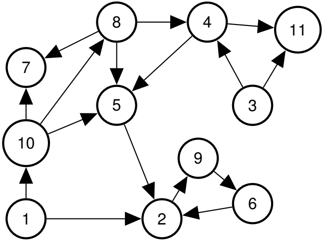
Cycle: Path with $v_1 = v_k$. Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!

Walk is sequence of edges with possible repeated vertex or edge.

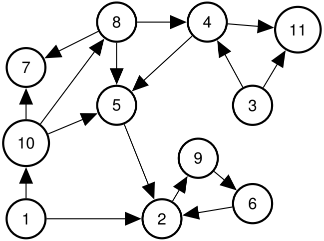
Tour is walk that starts and ends at the same node.

Quick Check!
Path is to Walk as Cycle is to ?? Tour!

Directed Paths.

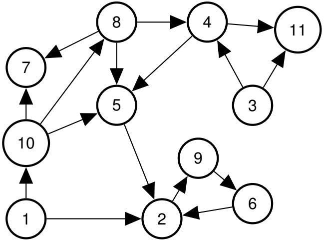


Directed Paths.



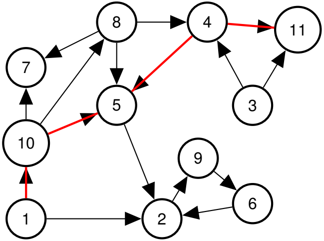
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Directed Paths.



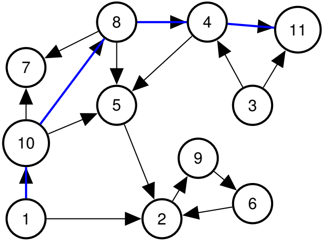
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Directed Paths.



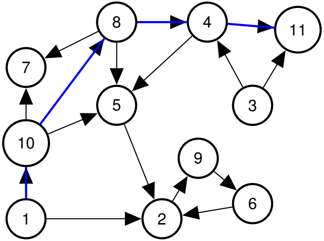
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Directed Paths.



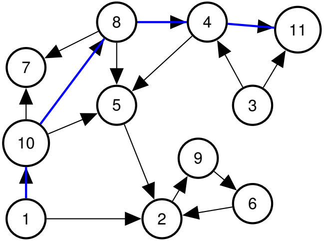
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Directed Paths.



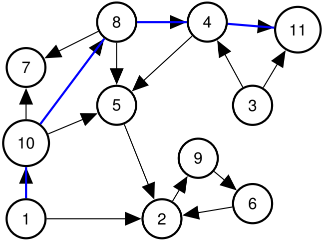
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Paths,

Directed Paths.



Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Paths, walks,

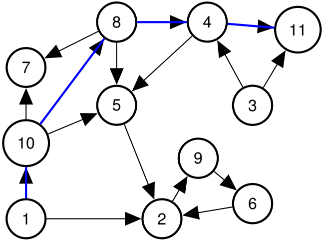
Directed Paths.



Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

Paths, walks, cycles,

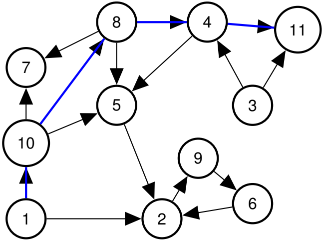
Directed Paths.



Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.

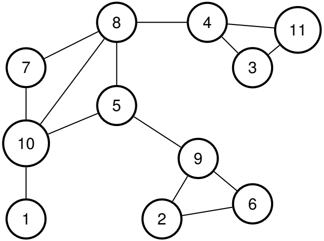
Paths, walks, cycles, tours

Directed Paths.



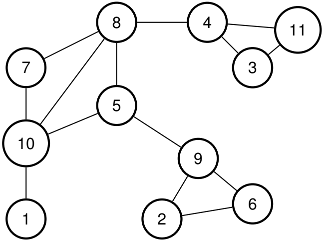
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Paths, walks, cycles, tours ... are analagous to undirected now.

Connectivity: undirected graph.



u and *v* are connected if there is a path between *u* and *v*.

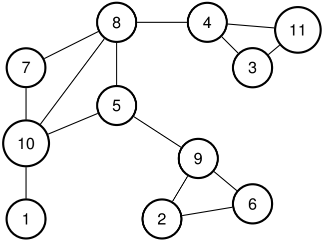
Connectivity: undirected graph.



u and v are connected if there is a path between u and v.

A connected graph is a graph where all pairs of vertices are connected.

Connectivity: undirected graph.

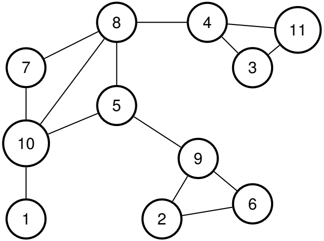


u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.

Connectivity: undirected graph.

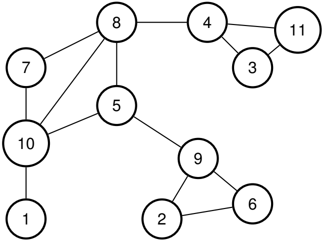


u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected?

Connectivity: undirected graph.

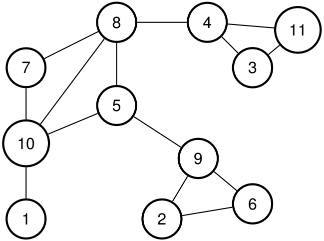


u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes?

Connectivity: undirected graph.

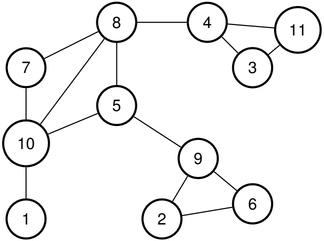


u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Connectivity: undirected graph.



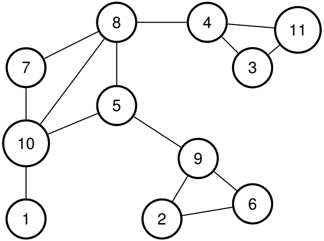
u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Proof:

Connectivity: undirected graph.



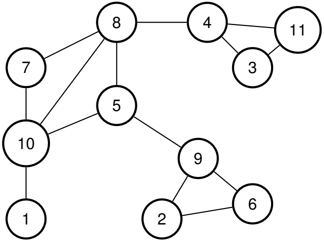
u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from *u* to *x* and then from *x* to *v*.

Connectivity: undirected graph.



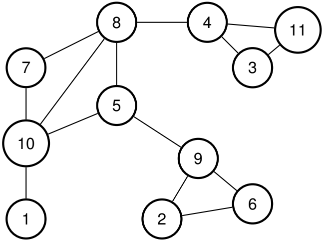
u and *v* are connected if there is a path between *u* and *v*.

A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from *u* to *x* and then from *x* to *v*. □

Connectivity: undirected graph.



u and *v* are connected if there is a path between *u* and *v*.

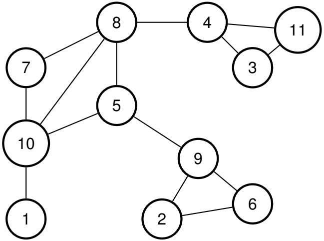
A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from *u* to *x* and then from *x* to *v*. □

May not be simple!

Connectivity: undirected graph.



u and v are connected if there is a path between u and v.

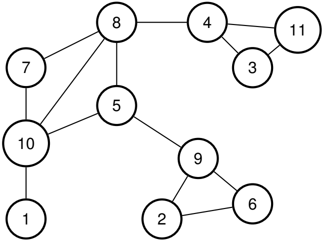
A connected graph is a graph where all pairs of vertices are connected.

If one vertex x is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from u to x and then from x to v. □

May not be simple!
Either modify definition to walk.

Connectivity: undirected graph.



u and v are connected if there is a path between u and v.

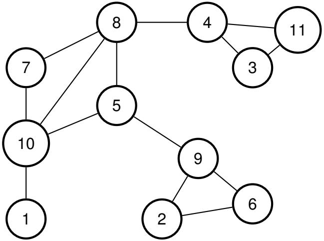
A connected graph is a graph where all pairs of vertices are connected.

If one vertex x is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from u to x and then from x to v. □

May not be simple!
Either modify definition to walk.
Or cut out cycles.

Connectivity: undirected graph.



u and *v* are connected if there is a path between *u* and *v*.

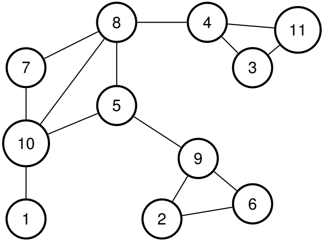
A connected graph is a graph where all pairs of vertices are connected.

If one vertex *x* is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from *u* to *x* and then from *x* to *v*. □

May not be simple!
Either modify definition to walk.
Or cut out cycles. .

Connectivity: undirected graph.



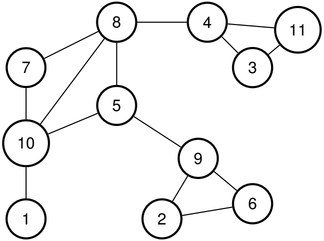
u and v are connected if there is a path between u and v.

A connected graph is a graph where all pairs of vertices are connected.

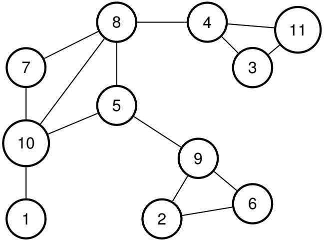
If one vertex x is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from u to x and then from x to v. □

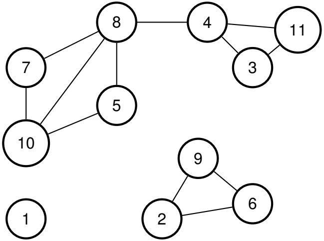
May not be simple!
Either modify definition to walk.
Or cut out cycles. .



Is graph above connected?

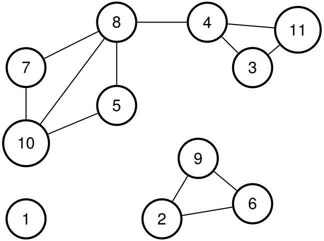


Is graph above connected? Yes!



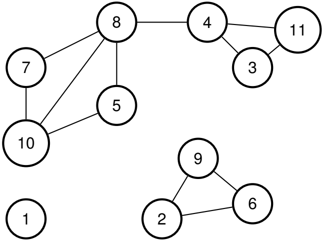
Is graph above connected? Yes!

How about now?



Is graph above connected? Yes!

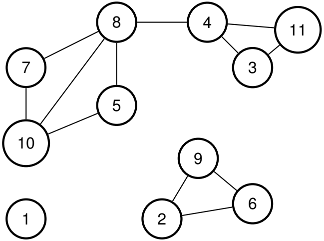
How about now? No!



Is graph above connected? Yes!

How about now? No!

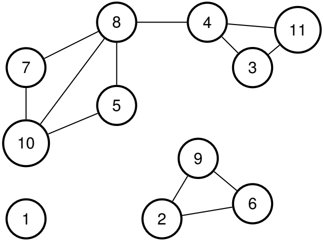
Connected Components?



Is graph above connected? Yes!

How about now? No!

Connected Components? $\{1\}, \{10, 7, 5, 8, 4, 3, 11\}, \{2, 9, 6\}$.

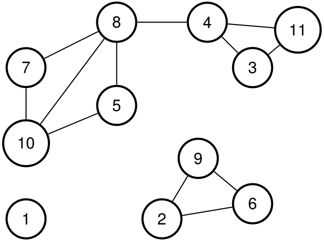


Is graph above connected? Yes!

How about now? No!

Connected Components? $\{1\}, \{10, 7, 5, 8, 4, 3, 11\}, \{2, 9, 6\}$.

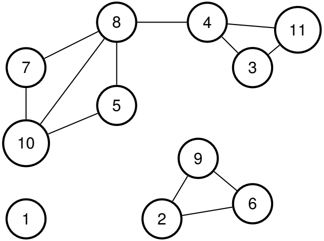
Connected component - maximal set of connected vertices.



Is graph above connected? Yes!

How about now? No!

Connected Components? $\{1\}, \{10, 7, 5, 8, 4, 3, 11\}, \{2, 9, 6\}$.
Connected component - maximal set of connected vertices.
Quick Check: Is $\{10, 7, 5\}$ a connected component?



Is graph above connected? Yes!

How about now? No!

Connected Components? $\{1\}, \{10, 7, 5, 8, 4, 3, 11\}, \{2, 9, 6\}$.

Connected component - maximal set of connected vertices.

Quick Check: Is $\{10, 7, 5\}$ a connected component? No.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. $\square$

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

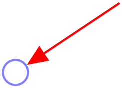
Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. $\square$

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



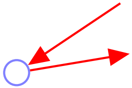
When you enter,

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



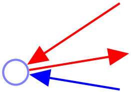
When you enter, you can leave.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



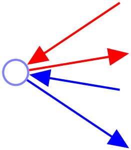
When you enter, you can leave.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex *v* on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore *v* has even degree. □



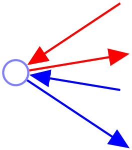
When you enter, you can leave.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



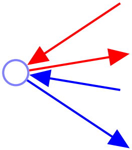
When you enter, you can leave.
For starting node,

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



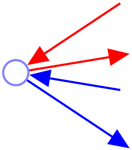
When you enter, you can leave.
For starting node, tour leaves first

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\Longrightarrow$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



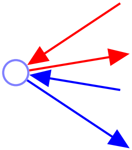
When you enter, you can leave.
For starting node, tour leaves first ....then enters at end.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



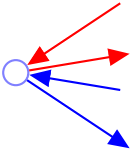
When you enter, you can leave.
For starting node, tour leaves first ....then enters at end.

Finally..back to Euler!

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



When you enter, you can leave.
For starting node, tour leaves first ....then enters at end.
Not The Hotel California.

Finding a tour!

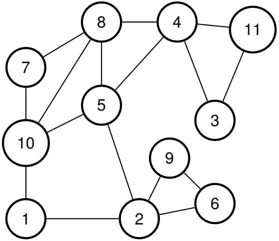
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm.

Finding a tour!

Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.

Finding a tour!

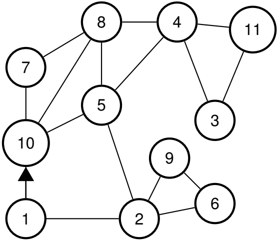
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges

Finding a tour!

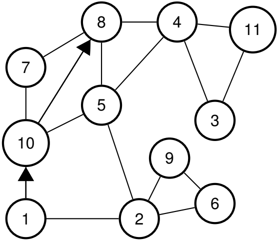
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges

Finding a tour!

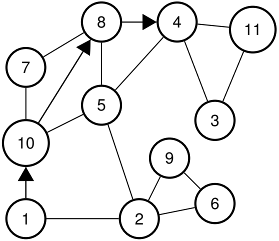
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges

Finding a tour!

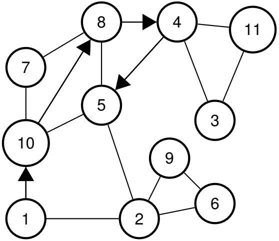
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges

Finding a tour!

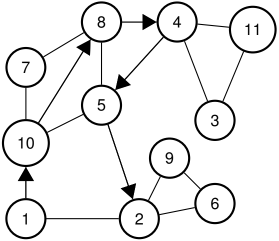
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges

Finding a tour!

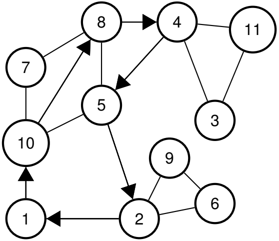
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges

Finding a tour!

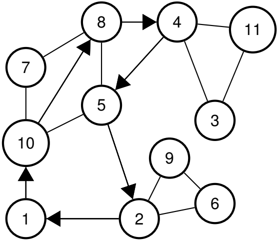
Proof of if: Even + connected $\Longrightarrow$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.

Finding a tour!

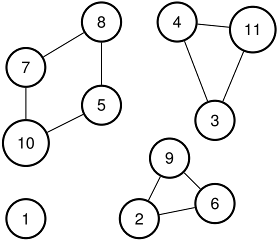
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.

Finding a tour!

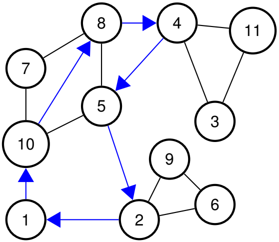
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.

Finding a tour!

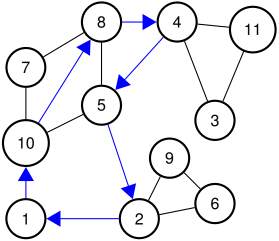
Proof of if: Even + connected $\Longrightarrow$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.

Finding a tour!

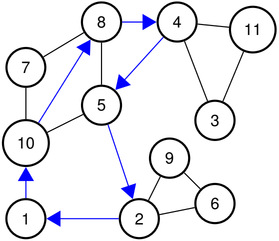
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why?

Finding a tour!

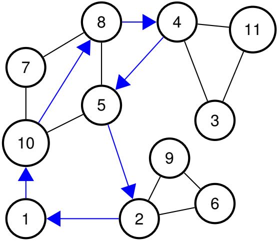
Proof of if: Even + connected $\Longrightarrow$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.

Finding a tour!

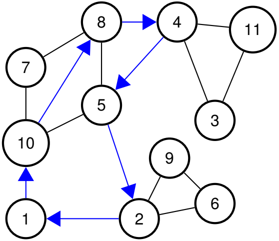
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.

Finding a tour!

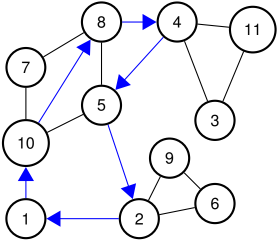
Proof of if: Even + connected $\Longrightarrow$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$,

Finding a tour!

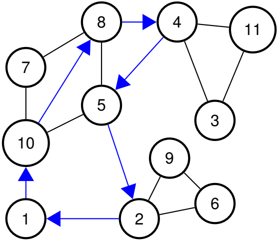
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
 Each is touched by C.
  Why? G was connected.
 Let v_i be (first) node in G_i touched by C.
 Example: $v_1 = 1$, $v_2 = 10$,

Finding a tour!

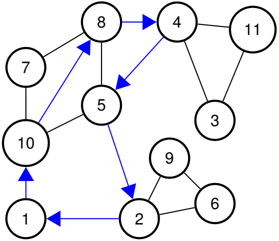
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$,

Finding a tour!

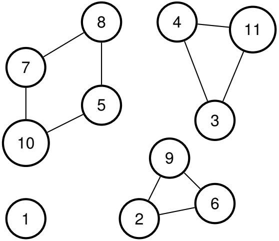
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.

Finding a tour!

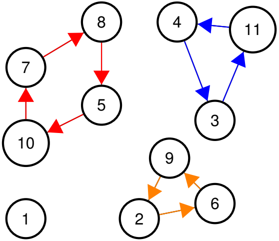
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i

Finding a tour!

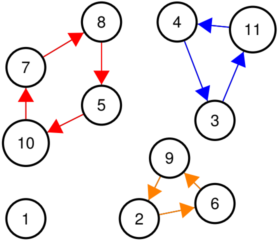
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i

Finding a tour!

Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.

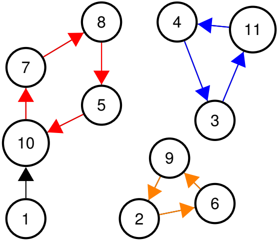


1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.

Finding a tour!

Proof of if: Even + connected $\implies$ Eulerian Tour.

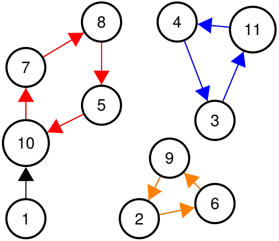
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10

Finding a tour!

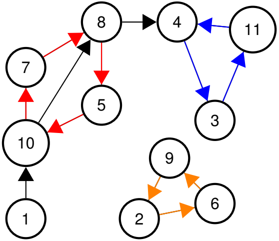
Proof of if: Even + connected $\Longrightarrow$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on “unused” edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
Each is touched by C.
Why? G was connected.
Let v_i be (first) node in G_i touched by C.
Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
1,10,7,8,5,10

Finding a tour!

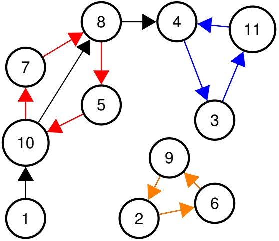
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \dots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \dots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4

Finding a tour!

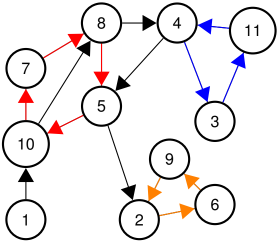
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
   Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
   Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4,3,11,4

Finding a tour!

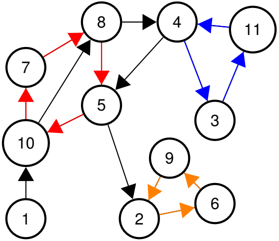
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
   Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4,3,11,4 5,2

Finding a tour!

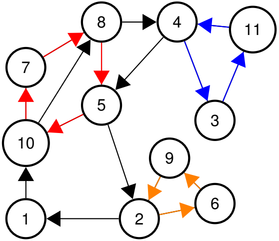
Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4,3,11,4 5,2,6,9,2

Finding a tour!

Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4,3,11,4 5,2,6,9,2 and to 1!

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.
Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.
Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. $\square$

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. $\square$

3. Find tour T_i of G_i

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.
Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Visits every edge once:
Visits edges in C

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Visits every edge once:
Visits edges in C exactly once.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\Longrightarrow$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Visits every edge once:
Visits edges in C exactly once.
By induction for all edges in each G_i.

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.

Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
Why is there a v_i in C?
G was connected $\implies$
a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Visits every edge once:
Visits edges in C exactly once.
By induction for all edges in each G_i. □

Break time!

Well admin time!

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don’t have to do homework.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!

Should do homework. No need to write up.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!
    Should do homework. No need to write up.
Homework Option: have to do homework.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

The truth:

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

The truth: mostly test,

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

- Test Option: don't have to do homework. Yes!!
  - Should do homework. No need to write up.
- Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!

- Variance mostly in exams for A/B range.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!
    Should do homework. No need to write up.
Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!
  Variance mostly in exams for A/B range.
   most homework students get near perfect scores on homework.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!
Should do homework. No need to write up.
Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!
Variance mostly in exams for A/B range.
most homework students get near perfect scores on homework.

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!
Should do homework. No need to write up.
Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!
Variance mostly in exams for A/B range.
most homework students get near perfect scores on homework.

How will I do?

Break time!

Well admin time!

Must choose homework option or test only: soon after recieving hw 1 scores.

Test Option: don't have to do homework. Yes!!
    Should do homework. No need to write up.
Homework Option: have to do homework. Bummer!

The truth: mostly test, both options!
  Variance mostly in exams for A/B range.
    most homework students get near perfect scores on homework.

How will I do?

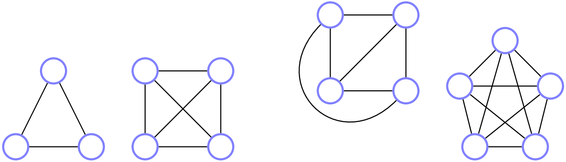
Mostly up to you.

Planar graphs.

A graph that can be drawn in the plane without edge crossings.

Planar graphs.

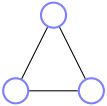
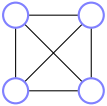
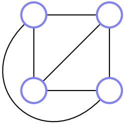
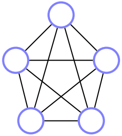
A graph that can be drawn in the plane without edge crossings.



Planar?

Planar graphs.

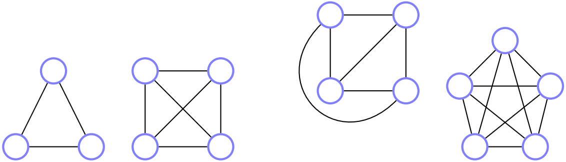
A graph that can be drawn in the plane without edge crossings.

Planar? Yes for Triangle.

Planar graphs.

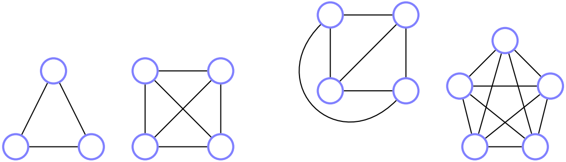
A graph that can be drawn in the plane without edge crossings.



Planar? Yes for Triangle.
Four node complete?

Planar graphs.

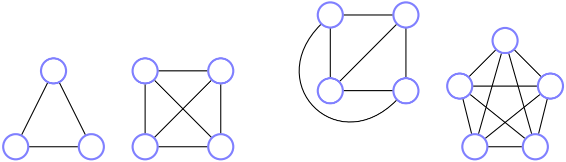
A graph that can be drawn in the plane without edge crossings.



Planar? Yes for Triangle.
Four node complete? Yes.

Planar graphs.

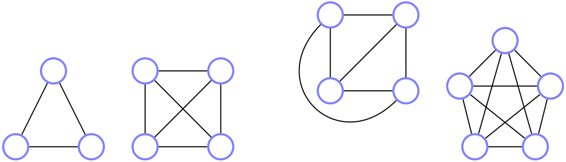
A graph that can be drawn in the plane without edge crossings.



Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5?

Planar graphs.

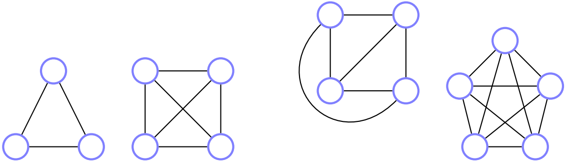
A graph that can be drawn in the plane without edge crossings.



Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No!

Planar graphs.

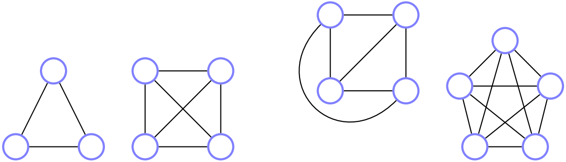
A graph that can be drawn in the plane without edge crossings.



Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why?

Planar graphs.

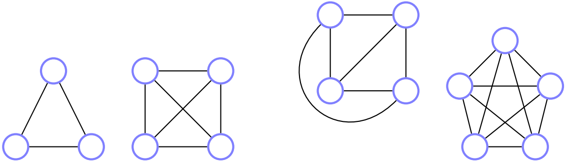
A graph that can be drawn in the plane without edge crossings.



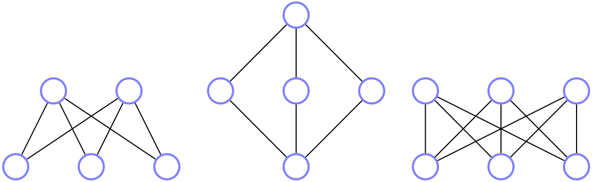
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.

Planar graphs.

A graph that can be drawn in the plane without edge crossings.

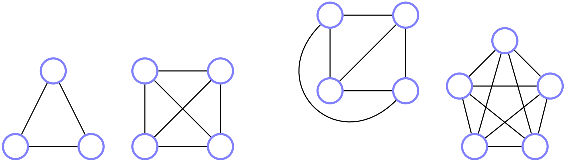


Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.

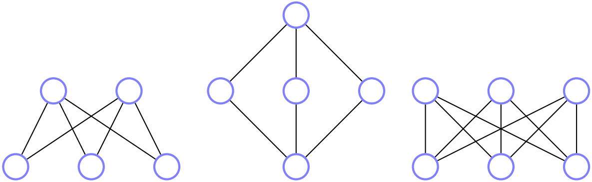


Planar graphs.

A graph that can be drawn in the plane without edge crossings.



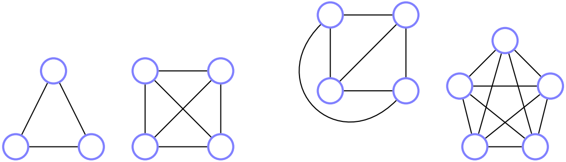
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.



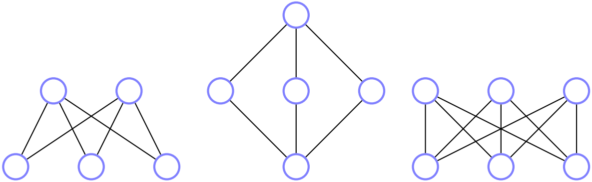
Two to three nodes, bipartite?

Planar graphs.

A graph that can be drawn in the plane without edge crossings.



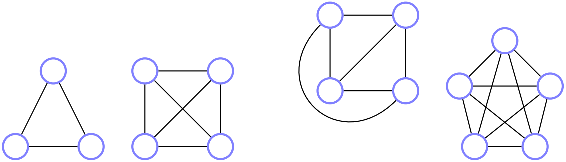
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.



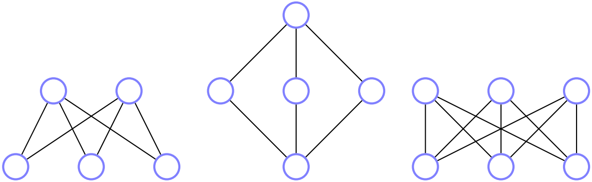
Two to three nodes, bipartite? Yes.

Planar graphs.

A graph that can be drawn in the plane without edge crossings.



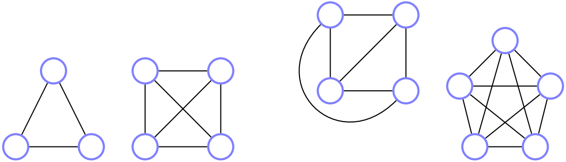
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.



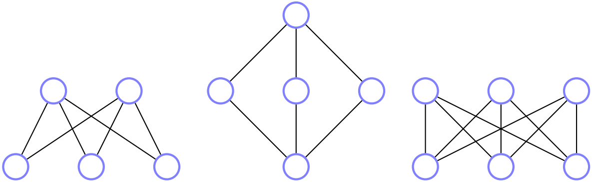
Two to three nodes, bipartite? Yes.
Three to three nodes, complete/bipartite or $K_{3,3}$.

Planar graphs.

A graph that can be drawn in the plane without edge crossings.



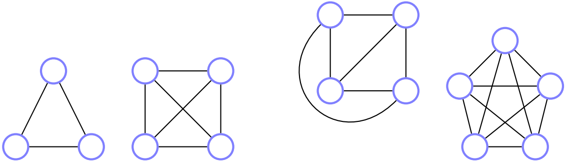
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.



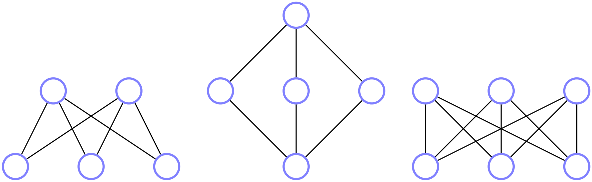
Two to three nodes, bipartite? Yes.
Three to three nodes, complete/bipartite or $K_{3,3}$. No.

Planar graphs.

A graph that can be drawn in the plane without edge crossings.



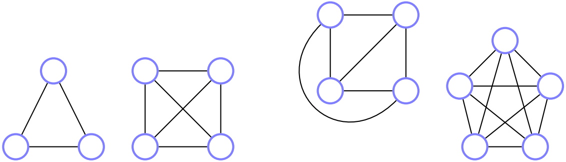
Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.



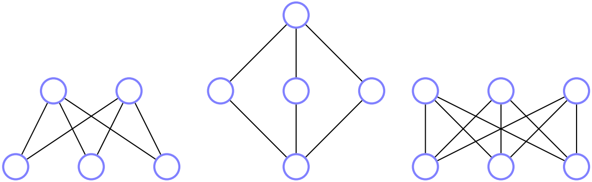
Two to three nodes, bipartite? Yes.
Three to three nodes, complete/bipartite or $K_{3,3}$. No. Why?

Planar graphs.

A graph that can be drawn in the plane without edge crossings.

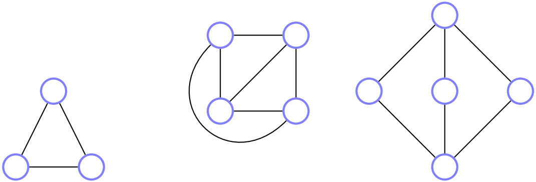


Planar? Yes for Triangle.
Four node complete? Yes.
Five node complete or K_5? No! Why? Later.

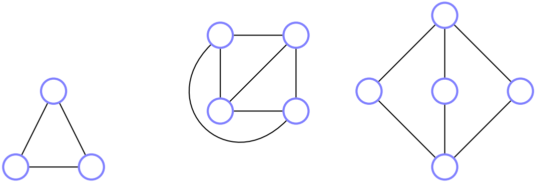


Two to three nodes, bipartite? Yes.
Three to three nodes, complete/bipartite or $K_{3,3}$. No. Why? Later.

Euler's Formula.

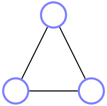
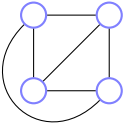
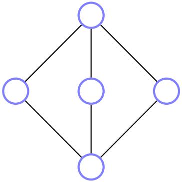


Euler's Formula.



Faces: connected regions of the plane.

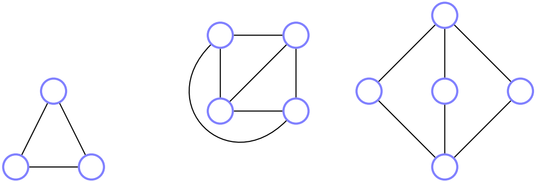
Euler's Formula.

Faces: connected regions of the plane.

How many faces for

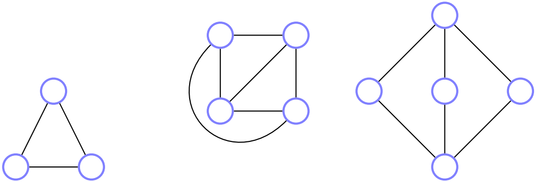
Euler's Formula.



Faces: connected regions of the plane.

How many faces for
triangle?

Euler's Formula.

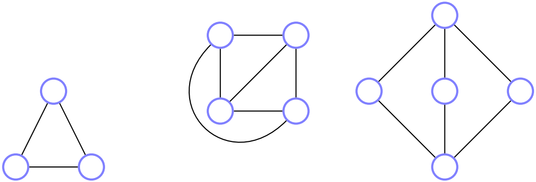


Faces: connected regions of the plane.

How many faces for

triangle? 2

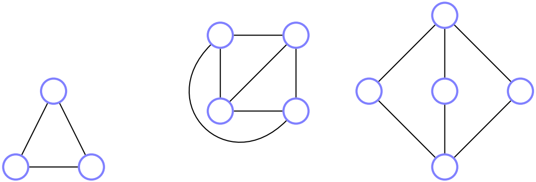
Euler's Formula.



Faces: connected regions of the plane.

How many faces for
  triangle? 2
  complete on four vertices or K_4?

Euler's Formula.



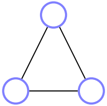
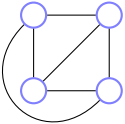
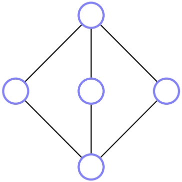
Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

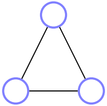
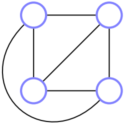
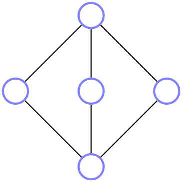
Euler's Formula.

Faces: connected regions of the plane.

How many faces for
- triangle? 2
- complete on four vertices or K_4? 4
- bipartite, complete two/three or $K_{2,3}$?

Euler's Formula.

Faces: connected regions of the plane.

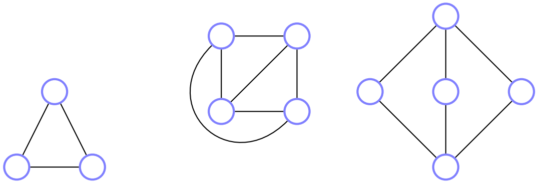
How many faces for

triangle? 2

complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

Euler's Formula.



Faces: connected regions of the plane.

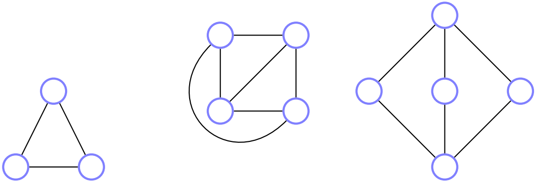
How many faces for

triangle? 2

complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

Euler's Formula.

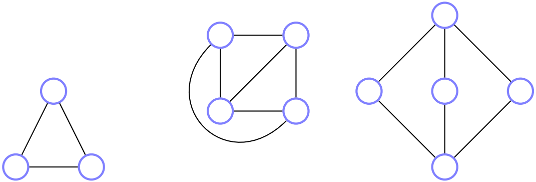


Faces: connected regions of the plane.

How many faces for
    triangle? 2
    complete on four vertices or K_4? 4
    bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula.



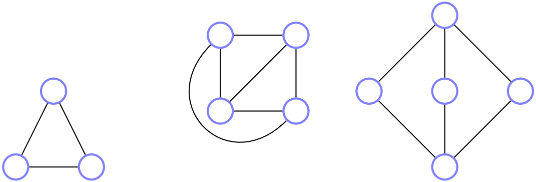
Faces: connected regions of the plane.

How many faces for
triangle? 2
complete on four vertices or K_4? 4
bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

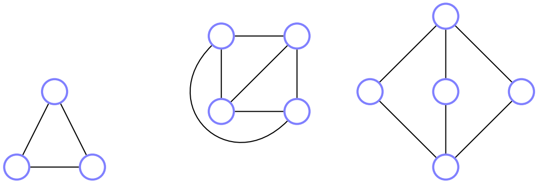
complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Euler's Formula.



Faces: connected regions of the plane.

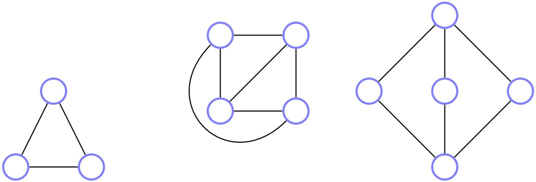
How many faces for
triangle? 2
complete on four vertices or K_4? 4
bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v+f=e+2$.

Triangle:

Euler's Formula.



Faces: connected regions of the plane.

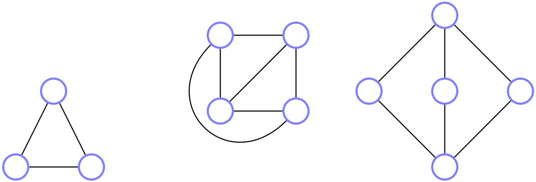
How many faces for
    triangle? 2
    complete on four vertices or K_4? 4
    bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

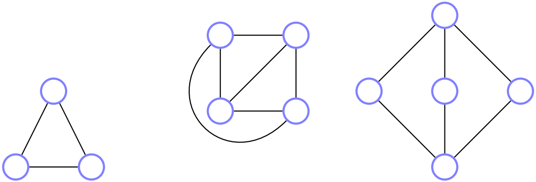
v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v+f=e+2$.

Triangle: $3+2=3+2$!

K_4:

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

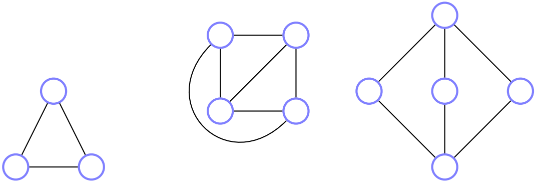
bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!

Euler's Formula.



Faces: connected regions of the plane.

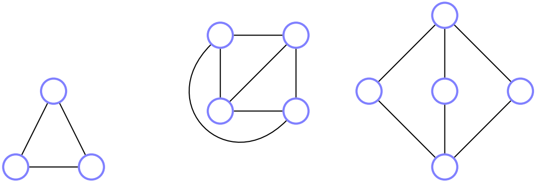
How many faces for
 triangle? 2
 complete on four vertices or K_4? 4
 bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$:

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

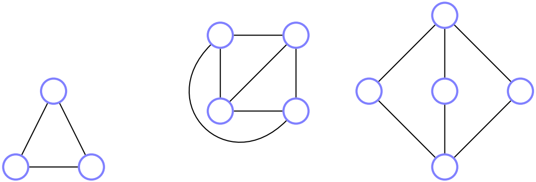
bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$: $5 + 3 = 6 + 2$!

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

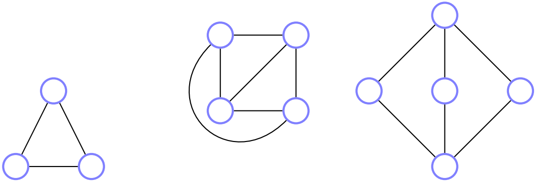
triangle? 2
complete on four vertices or K_4? 4
bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$: $5 + 3 = 6 + 2$!

Euler's Formula.



Faces: connected regions of the plane.

How many faces for
    triangle? 2
    complete on four vertices or K_4? 4
    bipartite, complete two/three or $K_{2,3}$? 3

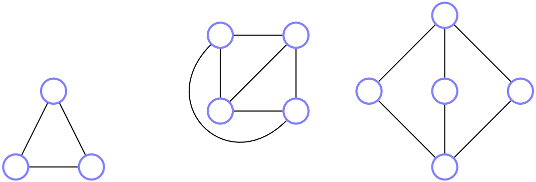
v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$: $5 + 3 = 6 + 2$!

Examples = 3!

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

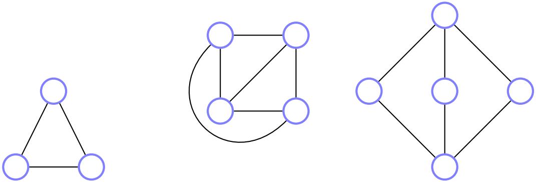
v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$: $5 + 3 = 6 + 2$!

Examples = 3! Proven!

Euler's Formula.



Faces: connected regions of the plane.

How many faces for

triangle? 2

complete on four vertices or K_4? 4

bipartite, complete two/three or $K_{2,3}$? 3

v is number of vertices, e is number of edges, f is number of faces.

Euler's Formula: Connected planar graph has $v + f = e + 2$.

Triangle: $3 + 2 = 3 + 2$!
K_4: $4 + 4 = 6 + 2$!
$K_{2,3}$: $5 + 3 = 6 + 2$!

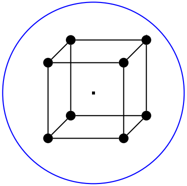
Examples = 3! Proven! Not!!!!

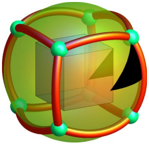
Euler and Polyhedron.

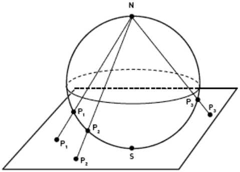
Greeks knew formula for polyhedron.

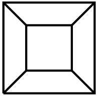
Euler and Polyhedron.

Greeks knew formula for polyhedron.



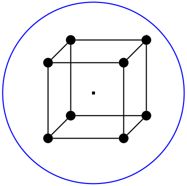


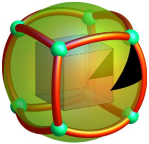


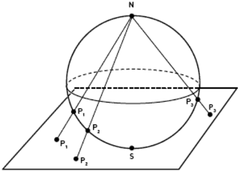


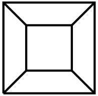
Euler and Polyhedron.

Greeks knew formula for polyhedron.





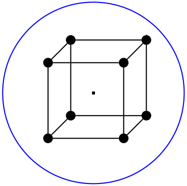


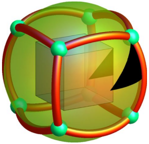


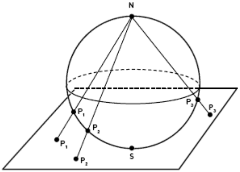
Faces?

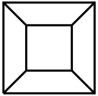
Euler and Polyhedron.

Greeks knew formula for polyhedron.





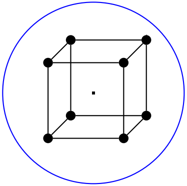


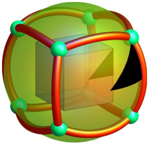


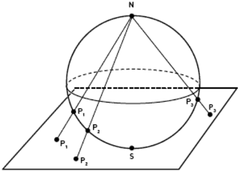
Faces? 6. Edges?

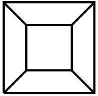
Euler and Polyhedron.

Greeks knew formula for polyhedron.





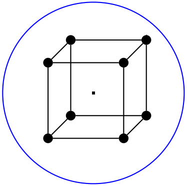


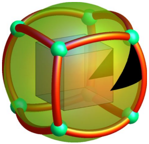


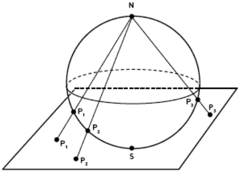
Faces? 6.   Edges? 12.

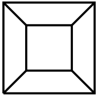
Euler and Polyhedron.

Greeks knew formula for polyhedron.





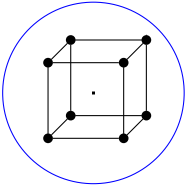


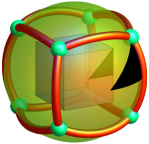


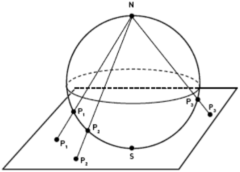
Faces? 6. Edges? 12. Vertices?

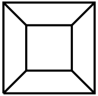
Euler and Polyhedron.

Greeks knew formula for polyhedron.





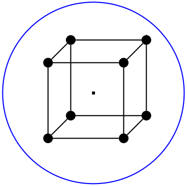


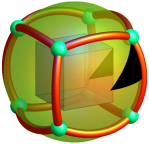


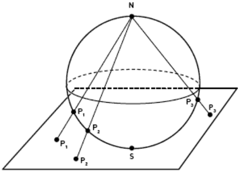
Faces? 6. Edges? 12. Vertices? 8.

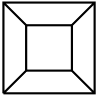
Euler and Polyhedron.

Greeks knew formula for polyhedron.







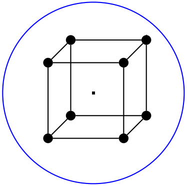


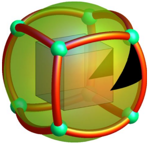
Faces? 6. Edges? 12. Vertices? 8.

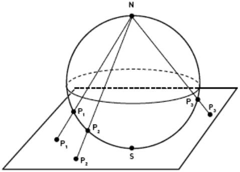
Euler: Connected planar graph: $v + f = e + 2$.

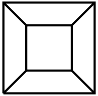
Euler and Polyhedron.

Greeks knew formula for polyhedron.







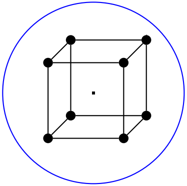


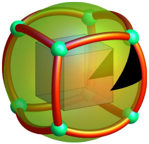
Faces? 6. Edges? 12. Vertices? 8.

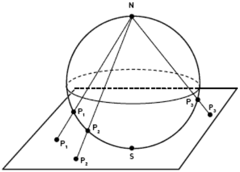
Euler: Connected planar graph: $v + f = e + 2$.

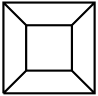
Euler and Polyhedron.

Greeks knew formula for polyhedron.









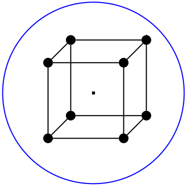
Faces? 6. Edges? 12. Vertices? 8.

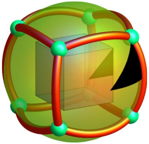
Euler: Connected planar graph: $v + f = e + 2$.

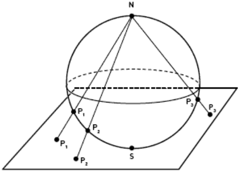
$8 + 6 = 12 + 2$.

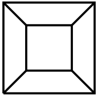
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6. Edges? 12. Vertices? 8.

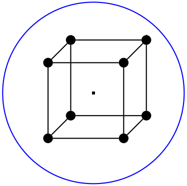
Euler: Connected planar graph: $v + f = e + 2$.

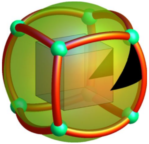
$8 + 6 = 12 + 2$.

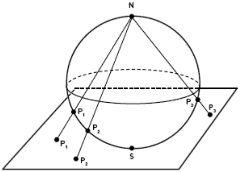
Greeks couldn't prove it.

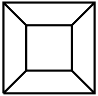
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6. Edges? 12. Vertices? 8.

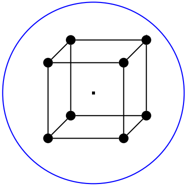
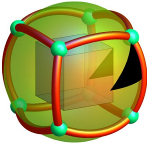
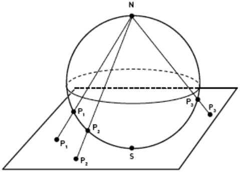
Euler: Connected planar graph: $v + f = e + 2$.

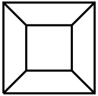
$8 + 6 = 12 + 2$.

Greeks couldn't prove it. Induction?

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

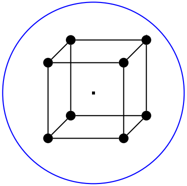
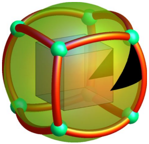
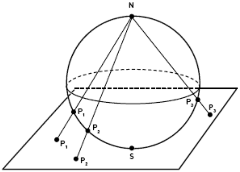
Euler: Connected planar graph: $v + f = e + 2$.

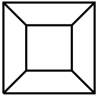
$8 + 6 = 12 + 2$.

Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

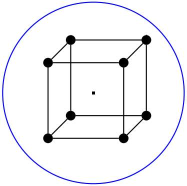
Euler: Connected planar graph: $v + f = e + 2$.

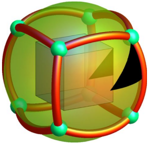
$8 + 6 = 12 + 2$.

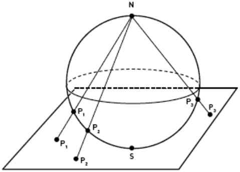
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

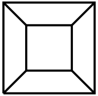
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6.   Edges? 12.   Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

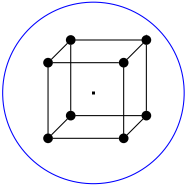
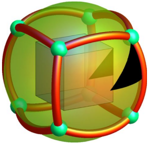
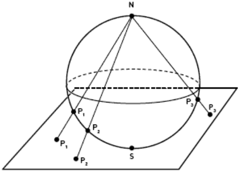
$8 + 6 = 12 + 2$.

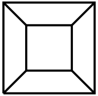
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

Polyhedron without holes

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v+f=e+2$.

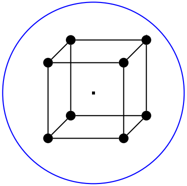
$8+6=12+2$.

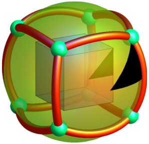
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

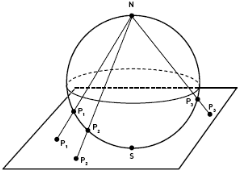
Polyhedron without holes $\equiv$

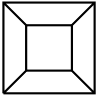
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

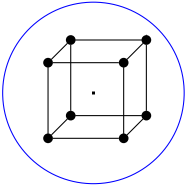
$8 + 6 = 12 + 2$.

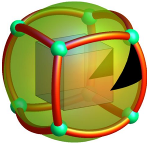
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

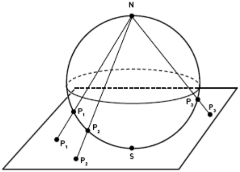
Polyhedron without holes $\equiv$ Planar graphs.

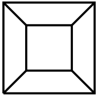
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

$8 + 6 = 12 + 2$.

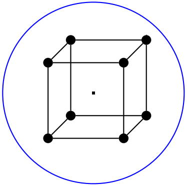
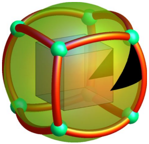
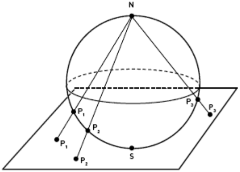
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

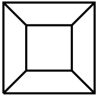
Polyhedron without holes $\equiv$ Planar graphs.

Surround by sphere.

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

$8 + 6 = 12 + 2$.

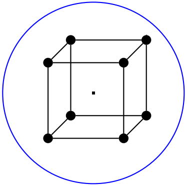
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

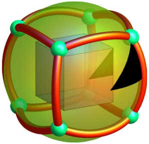
Polyhedron without holes $\equiv$ Planar graphs.

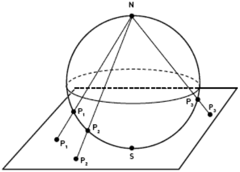
Surround by sphere.
Project from point inside polytope onto sphere.

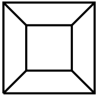
Euler and Polyhedron.

Greeks knew formula for polyhedron.









Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

$8 + 6 = 12 + 2$.

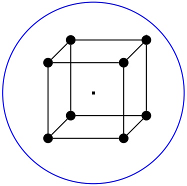
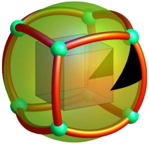
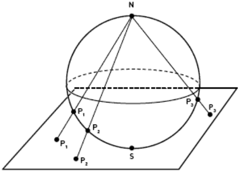
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

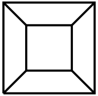
Polyhedron without holes $\equiv$ Planar graphs.

Surround by sphere.
Project from point inside polytope onto sphere.
Sphere

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

$8 + 6 = 12 + 2$.

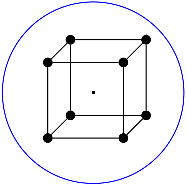
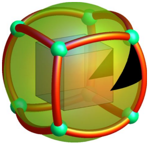
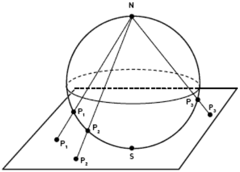
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

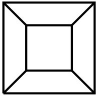
Polyhedron without holes $\equiv$ Planar graphs.

Surround by sphere.
Project from point inside polytope onto sphere.
Sphere $\equiv$ Plane!

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.

$8 + 6 = 12 + 2$.

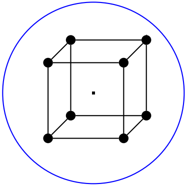
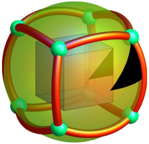
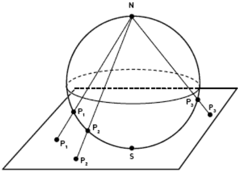
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

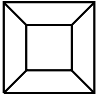
Polyhedron without holes $\equiv$ Planar graphs.

Surround by sphere.
Project from point inside polytope onto sphere.
Sphere $\equiv$ Plane! Topologically.

Euler and Polyhedron.

Greeks knew formula for polyhedron.



Faces? 6. Edges? 12. Vertices? 8.

Euler: Connected planar graph: $v + f = e + 2$.
$8 + 6 = 12 + 2$.

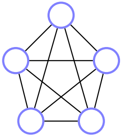
Greeks couldn't prove it. Induction? Remove vertice for polyhedron?

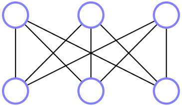
Polyhedron without holes $\equiv$ Planar graphs.

Surround by sphere.
Project from point inside polytope onto sphere.
Sphere $\equiv$ Plane! Topologically.

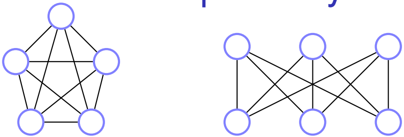
Euler proved formula thousands of years later!

Euler and non-planarity of K_5 and $K_{3,3}$



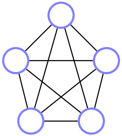


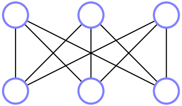
Euler and non-planarity of K_5 and $K_{3,3}$



Euler: $v + f = e + 2$ for connected planar graph.

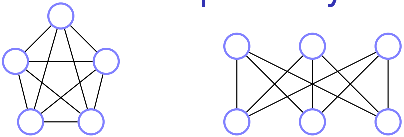
Euler and non-planarity of K_5 and $K_{3,3}$





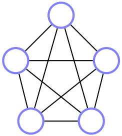
Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.

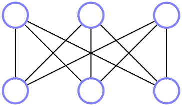
Euler and non-planarity of K_5 and $K_{3,3}$



Euler: $v+f=e+2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

Euler and non-planarity of K_5 and $K_{3,3}$

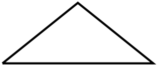


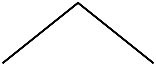


Euler: $v + f = e + 2$ for connected planar graph.

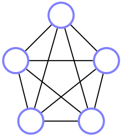
We consider simple graphs where $v \geq 3$.

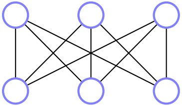
Consider Face edge Adjacencies.





Euler and non-planarity of K_5 and $K_{3,3}$

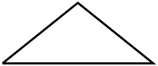


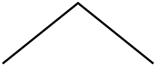


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

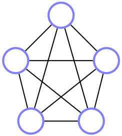
Consider Face edge Adjacencies.

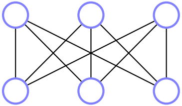




Each face is adjacent to at least three edges.

Euler and non-planarity of K_5 and $K_{3,3}$

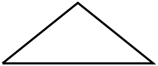


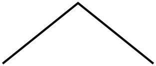


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.

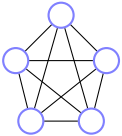


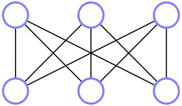


Each face is adjacent to at least three edges.

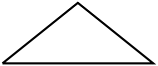
$\geq 3f$ face-edge adjacencies.

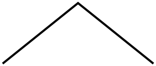
Euler and non-planarity of K_5 and $K_{3,3}$





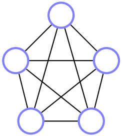
Euler: $v+f=e+2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

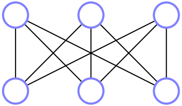




Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.

Euler and non-planarity of K_5 and $K_{3,3}$

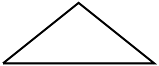


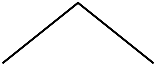


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





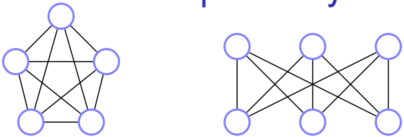
Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

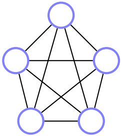
Euler and non-planarity of K_5 and $K_{3,3}$

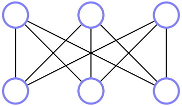


Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$

Euler and non-planarity of K_5 and $K_{3,3}$

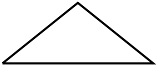


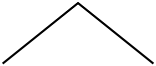


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

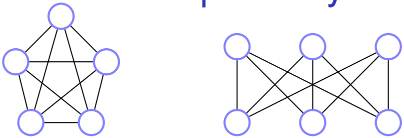
$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for

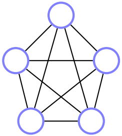
Euler and non-planarity of K_5 and $K_{3,3}$

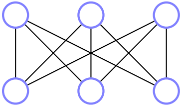


Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$ for any

Euler and non-planarity of K_5 and $K_{3,3}$

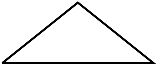


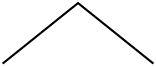


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

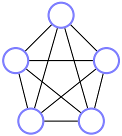
$\geq 3f$ face-edge adjacencies.

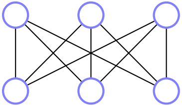
Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for any planar

Euler and non-planarity of K_5 and $K_{3,3}$

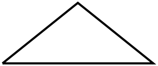


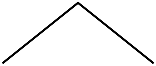


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

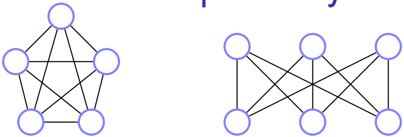
$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for any planar graph.

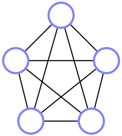
Euler and non-planarity of K_5 and $K_{3,3}$

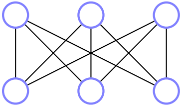


Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Euler and non-planarity of K_5 and $K_{3,3}$

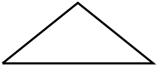


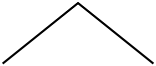


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

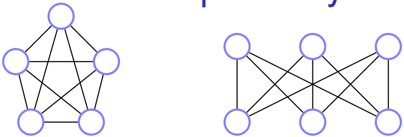
Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

$\Longrightarrow$ $3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler:

Euler and non-planarity of K_5 and $K_{3,3}$

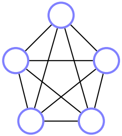
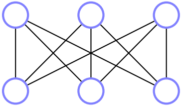


Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2$

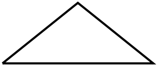
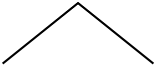
Euler and non-planarity of K_5 and $K_{3,3}$

Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

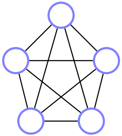
Each edge is adjacent to (at most) two faces.

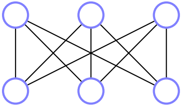
$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

Euler and non-planarity of K_5 and $K_{3,3}$

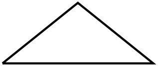


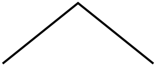


Euler: $v+f=e+2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

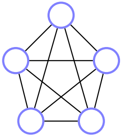
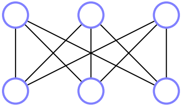
$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v+\frac{2}{3}e \geq e+2 \implies e \leq 3v-6$

K_5

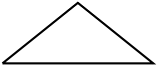
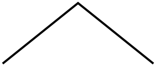
Euler and non-planarity of K_5 and $K_{3,3}$

Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

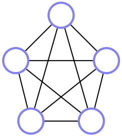
$\leq 2e$ face-edge adjacencies.

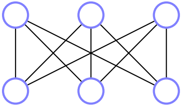
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

K_5 Edges?

Euler and non-planarity of K_5 and $K_{3,3}$

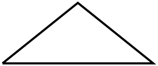


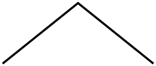


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

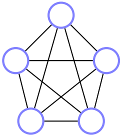
$\leq 2e$ face-edge adjacencies.

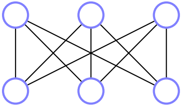
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

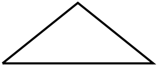
K_5 Edges? $4+3+2+1$

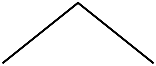
Euler and non-planarity of K_5 and $K_{3,3}$





Euler: $v+f=e+2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.



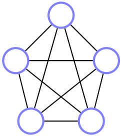


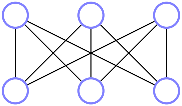
Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v+\frac{2}{3}e \geq e+2 \implies e \leq 3v-6$

K_5 Edges? $4+3+2+1=10$.

Euler and non-planarity of K_5 and $K_{3,3}$

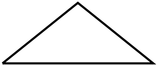


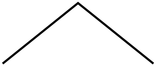


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

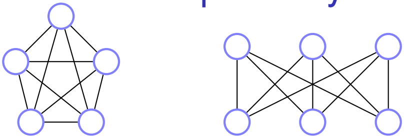
$\leq 2e$ face-edge adjacencies.

$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

K_5 Edges? $4 + 3 + 2 + 1 = 10$. Vertices?

Euler and non-planarity of K_5 and $K_{3,3}$



Euler: $v+f=e+2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.

Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

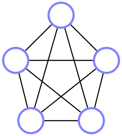
$\leq 2e$ face-edge adjacencies.

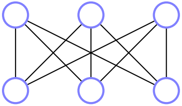
$\Longrightarrow$ $3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v+\frac{2}{3}e \geq e+2 \implies e \leq 3v-6$

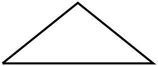
K_5 Edges? $4+3+2+1=10$. Vertices? 5.

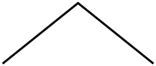
Euler and non-planarity of K_5 and $K_{3,3}$





Euler: $v + f = e + 2$ for connected planar graph.
We consider simple graphs where $v \geq 3$.
Consider Face edge Adjacencies.



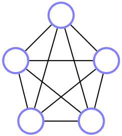


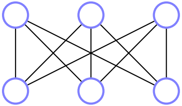
Each face is adjacent to at least three edges.
$\geq 3f$ face-edge adjacencies.
Each edge is adjacent to (at most) two faces.
$\leq 2e$ face-edge adjacencies.
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

K_5 Edges? $4+3+2+1 = 10$. Vertices? 5.
$10 \not\leq 3(5) - 6 = 9$.

Euler and non-planarity of K_5 and $K_{3,3}$

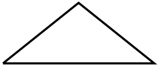


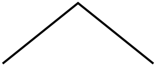


Euler: $v + f = e + 2$ for connected planar graph.

We consider simple graphs where $v \geq 3$.

Consider Face edge Adjacencies.





Each face is adjacent to at least three edges.

$\geq 3f$ face-edge adjacencies.

Each edge is adjacent to (at most) two faces.

$\leq 2e$ face-edge adjacencies.

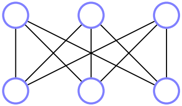
$\implies 3f \leq 2e$ for any planar graph. Or $f \leq \frac{2}{3}e$.

Plug into Euler: $v + \frac{2}{3}e \geq e + 2 \implies e \leq 3v - 6$

K_5 Edges? $4 + 3 + 2 + 1 = 10$. Vertices? 5.

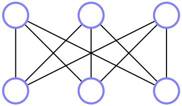
$10 \not\leq 3(5) - 6 = 9$. $\implies$ K_5 is not planar.

Proving non-planarity for $K_{3,3}$



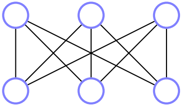
Euler's formula $\implies$ $3f \leq 2e$

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

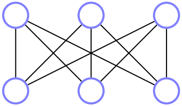
Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$?

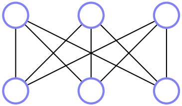
Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges?

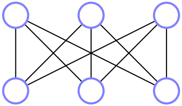
Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9.

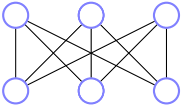
Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

Proving non-planarity for $K_{3,3}$

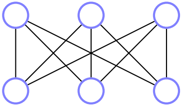


Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$$9 \leq 3(6) - 6?$$

Proving non-planarity for $K_{3,3}$

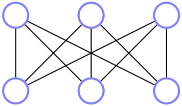


Euler's formula $\Longrightarrow$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proving non-planarity for $K_{3,3}$

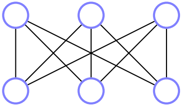


Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proving non-planarity for $K_{3,3}$



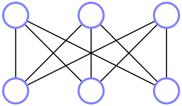
Euler's formula $\Longrightarrow$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work.

Proving non-planarity for $K_{3,3}$



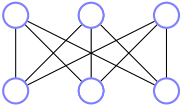
Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

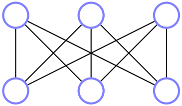
$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

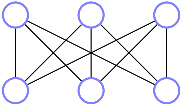
$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Proving non-planarity for $K_{3,3}$



Euler's formula $\Longrightarrow$ $3f \leq 2e$ for any planar graph.

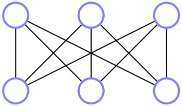
$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

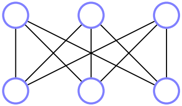
$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length;

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

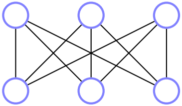
$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

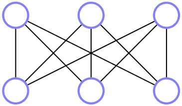
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

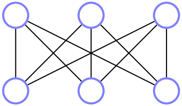
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \le 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \le 3(6) - 6$? Sure!

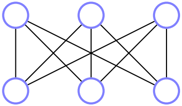
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \le 2e$ for any

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

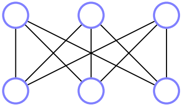
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

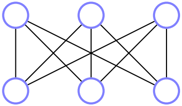
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite planar graph.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

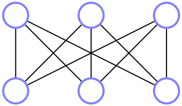
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite planar graph.
Euler: $v + \frac{1}{2}e \geq e + 2$

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

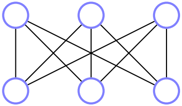
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite planar graph.
Euler: $v + \frac{1}{2}e \geq e + 2 \implies e \leq 2v - 4$ for bipartite planar graph

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

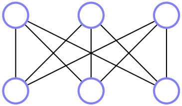
Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite planar graph.
Euler: $v + \frac{1}{2}e \geq e + 2 \implies e \leq 2v - 4$ for bipartite planar graph

Proving non-planarity for $K_{3,3}$



Euler's formula $\Longrightarrow$ $3f \le 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \le 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

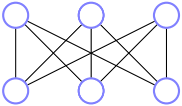
But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \le 2e$ for any bipartite planar graph.
Euler: $v + \frac{1}{2}e \geq e + 2 \Longrightarrow e \le 2v - 4$ for bipartite planar graph

$9 \not\le 2(6) - 4$.

Proving non-planarity for $K_{3,3}$



Euler's formula $\implies$ $3f \leq 2e$ for any planar graph.

$K_{3,3}$? Edges? 9. Vertices. 6.

$9 \leq 3(6) - 6$? Sure!

Proof doesn't work. Let's fix this.

But no cycles that are triangles. Face is of length ≥ 4.

Because all cycles are even length; bipartite or edges only go between two groups.

.... $4f \leq 2e$ for any bipartite planar graph.
Euler: $v + \frac{1}{2}e \geq e + 2 \implies e \leq 2v - 4$ for bipartite planar graph

$9 \not\leq 2(6) - 4$. $\implies$ $K_{3,3}$ is not planar!

Oh my goodness..what have we done!

Graphs.

Oh my goodness..what have we done!

Graphs.
  Basics.

Oh my goodness..what have we done!

Graphs.
 Basics.
Connectivity.

Oh my goodness..what have we done!

Graphs.
  Basics.
Connectivity.
Algorithm for Eulerian Tour.

Oh my goodness..what have we done!

Graphs.
  Basics.
Connectivity.
Algorithm for Eulerian Tour.

Oh my goodness..what have we done!

Graphs.
  Basics.
Connectivity.
Algorithm for Eulerian Tour.

Planar Graphs.

Oh my goodness..what have we done!

Graphs.
  Basics.
Connectivity.
Algorithm for Eulerian Tour.

Planar Graphs.

  Euler's formula.

Oh my goodness..what have we done!

Graphs.
 Basics.
Connectivity.
Algorithm for Eulerian Tour.

Planar Graphs.

 Euler's formula.

 Non-planarity of K_5 and $K_{3,3}$.

Oh my goodness..what have we done!

Graphs.
Basics.
Connectivity.
Algorithm for Eulerian Tour.

Planar Graphs.

Euler's formula.

Non-planarity of K_5 and $K_{3,3}$.

Yay!