Today

Review for Midterm.
First there was logic...

A statement is a true or false.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \[3 = 4 - 1\]?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \(3 = 4 - 1 \) ? Statement!
  \(3 = 5 \) ?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \( 3 = 4 - 1 \) ? Statement!
  \( 3 = 5 \) ? Statement!
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \(3 = 4 - 1\) ? Statement!
  \(3 = 5\) ? Statement!
  \(3\) ?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
  $3 = 5$ ? Statement!
  $3$ ? Not a statement!
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
  $3 = 5$ ? Statement!
  $3$ ? Not a statement!
  $n = 3$ ?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.

Statements?
  \[ 3 = 4 - 1 \] ? Statement!
  \[ 3 = 5 \] ? Statement!
  \[ 3 \] ? Not a statement!
  \[ n = 3 \] ? Not a statement...

Predicate: Statement with free variable(s).

Example:
\[ x = 3 \]
Given a value for \( x \), becomes a statement.

Predicate?
\[ n > 3 \]

Predicate:
\[ P(n) \]

\[ x = y \]

Predicate:
\[ P(x, y) \]

\[ x + y \]
No.

An expression, not a statement.

Quantifiers:
\[ (\forall x) P(x) \]
For every \( x \), \( P(x) \) is true.

\[ (\exists x) P(x) \]
There exists an \( x \), where \( P(x) \) is true.

\[ (\forall n \in \mathbb{N}) n^2 \geq n \]

Any free variables?
No.

So it’s a statement.

\[ (\forall x \in \mathbb{R})(\exists y \in \mathbb{R}) y > x \].

First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
   \(3 = 4 - 1\) ? Statement!
   \(3 = 5\) ? Statement!
   \(3\) ? Not a statement!
   \(n = 3\) ? Not a statement...but a predicate.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
  $3 = 5$ ? Statement!
  $3$ ? Not a statement!
  $n = 3$ ? Not a statement...but a predicate.

Predicate: Statement with free variable(s).
First there was logic...

A statement is a true or false.  
Don’t worry about Gödel. 
Statements?
  \[ 3 = 4 - 1 \] ? Statement!
  \[ 3 = 5 \] ? Statement!
  \[ 3 \] ? Not a statement!
  \[ n = 3 \] ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).
Example: \[ x = 3 \]
First there was logic...

A statement is a true or false. Don’t worry about Gödel.

Statements?

3 = 4 − 1 ? Statement!
3 = 5 ? Statement!
3 ? Not a statement!

n = 3 ? Not a statement...but a predicate.

Predicate: Statement with free variable(s).

Example: x = 3 Given a value for x, becomes a statement.
A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \(3 = 4 - 1\)? Statement!
  \(3 = 5\)? Statement!
  \(3\)? Not a statement!
  \(n = 3\)? Not a statement...but a predicate.

**Predicate**: Statement with free variable(s).
  Example: \(x = 3\)  Given a value for \(x\), becomes a statement.
Predicate?
A statement is a true or false. Don’t worry about Gödel. Statements?

\[ 3 = 4 - 1 \ ? \text{ Statement!} \]
\[ 3 = 5 \ ? \text{ Statement!} \]
\[ 3 \ ? \text{ Not a statement!} \]
\[ n = 3 \ ? \text{ Not a statement... but a predicate.} \]

**Predicate:** Statement with free variable(s).

Example: \[ x = 3 \] Given a value for \( x \), becomes a statement.

Predicate?

\[ n > 3 \ ? \]
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
   \( 3 = 4 - 1 \) ? Statement!
   \( 3 = 5 \) ? Statement!
   \( 3 \) ? Not a statement!
   \( n = 3 \) ? Not a statement...but a predicate.
**Predicate:** Statement with free variable(s).
   Example: \( x = 3 \)    Given a value for \( x \), becomes a statement.
Predicate?
   \( n > 3 \) ? Predicate: \( P(n) \)!
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
    3 = 4 – 1 ? Statement!
    3 = 5 ? Statement!
    3 ? Not a statement!
    n = 3 ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).
    Example: \( x = 3 \)  Given a value for \( x \), becomes a statement.

Predicate?
    \( n > 3 \) ? Predicate: \( P(n) \)!
    \( x = y \)?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \[3 = 4 - 1\] ? Statement!
  \[3 = 5\] ? Statement!
  \[3\] ? Not a statement!
  \[n = 3\] ? Not a statement...but a predicate.

Predicate: Statement with free variable(s).
  Example: \[x = 3\] Given a value for \(x\), becomes a statement.

Predicate?
  \[n > 3\] ? Predicate: \(P(n)\)!
  \[x = y\] ? Predicate: \(P(x, y)\)!
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
  $3 = 5$ ? Statement!
  $3$ ? Not a statement!
  $n = 3$ ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).

Example: $x = 3$ Given a value for $x$, becomes a statement.

Predicate?
  $n > 3$ ? Predicate: $P(n)$!
  $x = y$? Predicate: $P(x,y)$!
  $x + y$?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.

Statements?
   \[3 = 4 - 1\] ? Statement!
   \[3 = 5\] ? Statement!
   \[3\] ? Not a statement!
   \[n = 3\] ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).

Example: \[x = 3\] Given a value for \(x\), becomes a statement.

Predicate?
   \[n > 3\] ? Predicate: \(P(n)\)!
   \[x = y?\] Predicate: \(P(x, y)\)!
   \[x + y\] ? No.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \(3 = 4 - 1\) ? Statement!
  \(3 = 5\) ? Statement!
  \(3\) ? Not a statement!
  \(n = 3\) ? Not a statement...but a predicate.

**Predicate: Statement with free variable(s).**
  Example: \(x = 3\)  Given a value for \(x\), becomes a statement.
Predicate?
  \(n > 3\) ? Predicate: \(P(n)\)!
  \(x = y\) ? Predicate: \(P(x, y)\)!
  \(x + y\) ? No. An expression, not a statement.
A statement is a true or false.
Don’t worry about Gödel.
Statements?
  3 = 4 − 1 ? Statement!
  3 = 5 ? Statement!
  3 ? Not a statement!
  \( n = 3 \) ? Not a statement...but a predicate.
Predicate: Statement with free variable(s).
  Example: \( x = 3 \)  Given a value for \( x \), becomes a statement.
Predicate?
  \( n > 3 \) ? Predicate: \( P(n) \)!
  \( x = y \)? Predicate: \( P(x, y) \)!
  \( x + y \)? No. An expression, not a statement.
Quantifiers:
A statement is a true or false. Don’t worry about Gödel. Statements?

\[3 = 4 - 1\] ? Statement!
\[3 = 5\] ? Statement!
\[3\] ? Not a statement!
\[n = 3\] ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).

Example: \(x = 3\) Given a value for \(x\), becomes a statement.

Predicate?

\[n > 3\] ? Predicate: \(P(n)\)!
\[x = y\] ? Predicate: \(P(x, y)\)!
\[x + y\] ? No. An expression, not a statement.

**Quantifiers:**

\((\forall x) P(x)\).
A statement is a true or false.  
Don’t worry about Gödel.  
Statements?
   \[ 3 = 4 - 1 \] ? Statement!  
   \[ 3 = 5 \] ? Statement!  
   \[ 3 \] ? Not a statement!  
   \[ n = 3 \] ? Not a statement...but a predicate.  

Predicate: Statement with free variable(s).  
Example: \[ x = 3 \]  
   Given a value for \( x \), becomes a statement.  

Predicate?
   \[ n > 3 \] ? Predicate: \( P(n) \)!  
   \[ x = y \] ? Predicate: \( P(x, y) \)!  
   \[ x + y \] ? No. An expression, not a statement.  

Quantifiers:
   \( (\forall x) P(x) \).  
   For every \( x \), \( P(x) \) is true.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
\[3 = 4 - 1\] ? Statement!
\[3 = 5\] ? Statement!
\[3\] ? Not a statement!
\[n = 3\] ? Not a statement...but a predicate.

**Predicate: Statement with free variable(s).**
Example: \(x = 3\)  Given a value for \(x\), becomes a statement.

Predicate?
\[n > 3\] ? Predicate: \(P(n)\)!
\[x = y\] ? Predicate: \(P(x, y)\)!
\[x + y\] ? No. An expression, not a statement.

**Quantifiers:**
\((\forall x) P(x)\). For every \(x\), \(P(x)\) is true.
\((\exists x) P(x)\).
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  \[ 3 = 4 - 1 \] ? Statement!
  \[ 3 = 5 \] ? Statement!
  \[ 3 \] ? Not a statement!
  \[ n = 3 \] ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).
Example: \[ x = 3 \] Given a value for \( x \), becomes a statement.

Predicate?
  \[ n > 3 \] ? Predicate: \( P(n) \)!
  \[ x = y \] ? Predicate: \( P(x, y) \)!
  \[ x + y \] ? No. An expression, not a statement.

**Quantifiers:**
  \((\forall x) P(x)\). For every \( x \), \( P(x) \) is true.
  \((\exists x) P(x)\). There exists an \( x \), where \( P(x) \) is true.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?
  $3 = 4 - 1$ ? Statement!
  $3 = 5$ ? Statement!
  $3$ ? Not a statement!
  $n = 3$ ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).
  Example: $x = 3$  
  Given a value for $x$, becomes a statement.

Predicate?
  $n > 3$ ? Predicate: $P(n)!$
  $x = y$? Predicate: $P(x, y)!$
  $x + y$? No. An expression, not a statement.

**Quantifiers:**
  $(\forall x) P(x).$  
  For every $x$, $P(x)$ is true.
  $(\exists x) P(x).$  
  There exists an $x$, where $P(x)$ is true.

$(\forall n \in N), n^2 \geq n$: 
First there was logic...

A statement is a true or false.  Don’t worry about Gödel.

Statements?

- $3 = 4 - 1$ ? Statement!
- $3 = 5$ ? Statement!
- $3$ ? Not a statement!
- $n = 3$ ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).

Example: $x = 3$   
Given a value for $x$, becomes a statement.

Predicate?

- $n > 3$ ? Predicate: $P(n)$!
- $x = y$? Predicate: $P(x, y)$!
- $x + y$? No. An expression, not a statement.

**Quantifiers:**

- $(\forall x) P(x)$.    
  For every $x$, $P(x)$ is true.
- $(\exists x) P(x)$.    
  There exists an $x$, where $P(x)$ is true.

$(\forall n \in N), n^2 \geq n$: Any free variables?
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.
Statements?

\[ 3 = 4 - 1 \]  ? Statement!
\[ 3 = 5 \]  ? Statement!
\[ 3 \]  ? Not a statement!
\[ n = 3 \]  ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).
Example: \( x = 3 \)  Given a value for \( x \), becomes a statement.

Predicate?

\[ n > 3 \]  ? Predicate: \( P(n) \)!
\[ x = y \]  ? Predicate: \( P(x, y) \)!
\[ x + y \]  No. An expression, not a statement.

**Quantifiers:**

\( (\forall x) \ P(x). \)  For every \( x \), \( P(x) \) is true.
\( (\exists x) \ P(x). \)  There exists an \( x \), where \( P(x) \) is true.

\[ (\forall n \in N), n^2 \geq n: \]  Any free variables? No.
First there was logic...

A statement is a true or false.
Don’t worry about Gödel.

Statements?
  \(3 = 4 - 1\) ? Statement!
  \(3 = 5\) ? Statement!
  \(3\) ? Not a statement!
  \(n = 3\) ? Not a statement...but a predicate.

**Predicate:** Statement with free variable(s).

Example: \(x = 3\)  Given a value for \(x\), becomes a statement.

Predicate?
  \(n > 3\) ? Predicate: \(P(n)\)!
  \(x = y\) ? Predicate: \(P(x, y)\)!
  \(x + y\) ? No. An expression, not a statement.

**Quantifiers:**

\((\forall x) P(x)\).  For every \(x\), \(P(x)\) is true.
\((\exists x) P(x)\).  There exists an \(x\), where \(P(x)\) is true.

\((\forall n \in N), n^2 \geq n\): Any free variables? No. So it’s a statement.
\((\forall x \in R)(\exists y \in R)y > x\).
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]
Connecting Statements

\(A \land B, A \lor B, \neg A, A \implies B.\)

Propositional Expressions and Logical Equivalence
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]

Propositional Expressions and Logical Equivalence

\[(A \implies B) \equiv (\neg A \lor B)\]
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
(A \implies B) \equiv (\neg A \lor B) \\
\neg (A \lor B) \equiv (\neg A \land \neg B)
\]
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg(A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg(A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.

\[ (\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x) \]
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
(A \implies B) \equiv (\neg A \lor B) \\
(\neg (A \lor B)) \equiv (\neg A \land \neg B)
\]

Proofs: truth table or manipulation of known formulas.

\[
(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)
\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true.
No matter what \(P\) and \(Q\) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true.
No matter what \(P\) and \(Q\) are!

Or manipulate the formulas.

If you think it’s not true:

Find an example of \(P(x)\) and \(Q(x)\) such that one of the above steps fails.
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
(A \implies B) \equiv (\neg A \lor B) \\
(\neg (A \lor B)) \equiv (\neg A \land \neg B)
\]

Proofs: truth table or manipulation of known formulas.

\[
(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)
\]

If you think it’s true:

Step 1:

No matter what \(P\) and \(Q\) are!

Step 2:

No matter what \(P\) and \(Q\) are!

Or manipulate the formulas.

If you think it’s not true:

Find an example of \(P(x)\) and \(Q(x)\) such that one of the above steps fails.
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg(A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.

\[ (\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x) \]

**If you think it’s true:**

Step 1: Show that when the thing on the left is true, the thing on the right is true.

No matter what \( P(x) \) and \( Q(x) \) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true.

No matter what \( P(x) \) and \( Q(x) \) are!

Or manipulate the formulas.

**If you think it’s not true:**

Find an example of \( P(x) \) and \( Q(x) \) such that one of the above steps fails.
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
(A \implies B) \equiv (\neg A \lor B)
\]

\[
\neg(A \lor B) \equiv (\neg A \land \neg B)
\]

Proofs: truth table or manipulation of known formulas.

\[
(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)
\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
\begin{align*}
(A \implies B) & \equiv (\neg A \lor B) \\
\neg (A \lor B) & \equiv (\neg A \land \neg B)
\end{align*}
\]

Proofs: truth table or manipulation of known formulas.

\[
(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)
\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Step 2:
Connecting Statements

\(A \land B, A \lor B, \neg A, A \implies B.\)

Propositional Expressions and Logical Equivalence

\[(A \implies B) \equiv (\neg A \lor B)\]
\[\neg(A \lor B) \equiv (\neg A \land \neg B)\]

Proofs: truth table or manipulation of known formulas.

\[(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \(P\) and \(Q\) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true.
Connecting Statements

\[ A \land B, \ A \lor B, \ \neg A, \ A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg (A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.

\[ (\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x) \]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true. No matter what \( P \) and \( Q \) are!
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg (A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.

\[ (\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x) \]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true. No matter what \( P \) and \( Q \) are!

Or manipulate the formulas.
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]

Propositional Expressions and Logical Equivalence

\[ (A \implies B) \equiv (\neg A \lor B) \]
\[ \neg (A \lor B) \equiv (\neg A \land \neg B) \]

Proofs: truth table or manipulation of known formulas.

\[ (\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x) \]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true. No matter what \( P \) and \( Q \) are!

Or manipulate the formulas.

If you think it’s not true:
Connecting Statements

\[ A \land B, A \lor B, \neg A, A \implies B. \]

Propositional Expressions and Logical Equivalence

\[
(A \implies B) \equiv (\neg A \lor B)
\]
\[
\neg(A \lor B) \equiv (\neg A \land \neg B)
\]

Proofs: truth table or manipulation of known formulas.

\[
(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)
\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on the right is true. No matter what \( P \) and \( Q \) are!

Step 2: Show that when the thing on the right is true, the thing on the left is true. No matter what \( P \) and \( Q \) are!

Or manipulate the formulas.

If you think it’s not true:

Find an example of \( P(x) \) and \( Q(x) \)
Connecting Statements

\(A \land B, A \lor B, \neg A, A \implies B.\)

Propositional Expressions and Logical Equivalence

\[(A \implies B) \equiv (\neg A \lor B)\]
\[\neg (A \lor B) \equiv (\neg A \land \neg B)\]

Proofs: truth table or manipulation of known formulas.

\[(\forall x \in \mathbb{R})(P(x) \land Q(x)) \equiv (\forall x \in \mathbb{R})P(x) \land (\forall x \in \mathbb{R})Q(x)\]

If you think it’s true:

Step 1: Show that when the thing on the left is true, the thing on
the right is true. No matter what \(P\) and \(Q\) are!

Step 2: Show that when the thing on the right is true, the thing on
the left is true. No matter what \(P\) and \(Q\) are!

Or manipulate the formulas.

If you think it’s not true:

Find an example of \(P(x)\) and \(Q(x)\) such that one of the above
steps fails.
...and then proofs...

**Direct:** $P \implies Q$

---

**Example:**

$a$ is even $\implies a^2$ is even.

**Approach:**

What is even? $a = 2k$ \text{ where } k \in \mathbb{Z}$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! $a^2$ is even.

**Contrapositive:**

$P \implies Q$ or $\neg Q = \implies \neg P$.

**Example:**

$a^2$ is odd $\implies a$ is odd.

**Contrapositive:**

$a$ is even $\implies a^2$ is even.

---

**Contradiction:**

$\neg P \implies \text{false}$

Useful to prove something does not exist:

**Example:**

Rational representation of $\sqrt{2}$ does not exist.

**Example:**

Finite set of primes does not exist.

**Example:**

Rogue couple does not exist.
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.
...and then proofs...

**Direct**: $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even?
...and then proofs...

**Direct:** \( P \implies Q \)

Example: \( a \) is even \( \implies a^2 \) is even.

Approach: What is even? \( a = 2k \)
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2$
Direct: \( P \implies Q \)

Example: \( a \) is even \( \implies \) \( a^2 \) is even.

Approach: What is even? \( a = 2k \)
\[
a^2 = 4k^2 = 2(2k^2)
\]
Direct: $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication!
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.  
Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even.

**Contrapositive:** $P \implies Q$

Example: $a^2$ is odd $\implies a$ is odd.

**Contradiction:** $\neg P \implies \neg Q$

Useful to prove something does not exist: 
Example: rational representation of $\sqrt{2}$ does not exist.

Example: finite set of primes does not exist.

Example: rogue couple does not exist.
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.
...and then proofs...

**Direct:** $P \implies Q$
- Example: $a$ is even $\implies a^2$ is even.
  - Approach: What is even? $a = 2k$
  - $a^2 = 4k^2 = 2(2k^2)$
    - Integers closed under multiplication! So $2k^2$ is even.
  - $a^2$ is even.

**Contrapositive:** $P \iff Q$
...and then proofs...

**Direct**: $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.

**Contrapositive**: $P \implies Q$ or $\neg Q \implies \neg P$. 
...and then proofs...

**Direct**: $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even.

$a^2$ is even.

**Contrapositive**: $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.

Contrapositive: $a$ is even $\implies a^2$ is even.
...and then proofs...

**Direct:** $P \implies Q$
- Example: $a$ is even $\implies a^2$ is even.
  - Approach: What is even? $a = 2k$
    - $a^2 = 4k^2 = 2(2k^2)$
  - Integers closed under multiplication! So $2k^2$ is even.
    - $a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.
- Example: $a^2$ is odd $\implies a$ is odd.
  - Contrapositive: $a$ is even $\implies a^2$ is even.
...and then proofs...

**Direct**: \( P \implies Q \)
- Example: \( a \) is even \( \implies a^2 \) is even.
  - Approach: What is even? \( a = 2k \)
    \[ a^2 = 4k^2 = 2(2k^2) \]
    Integers closed under multiplication! So \( 2k^2 \) is even. \( a^2 \) is even.

**Contrapositive**: \( P \implies Q \) or \( \neg Q \implies \neg P \).
- Example: \( a^2 \) is odd \( \implies a \) is odd.
  - Contrapositive: \( a \) is even \( \implies a^2 \) is even.

**Contradiction**: \( P \)
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

   Approach: What is even? $a = 2k$
   
   $a^2 = 4k^2 = 2(2k^2)$

   Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.

   Contrapositive: $a$ is even $\implies a^2$ is even.

**Contradiction:** $P$

   $\neg P \implies \text{false}$
Direct: \( P \implies Q \)
Example: \( a \) is even \( \implies a^2 \) is even.
Approach: What is even? \( a = 2k \)
\[
a^2 = 4k^2 = 2(2k^2)
\]
Integers closed under multiplication! So \( 2k^2 \) is even.
\( a^2 \) is even.

Contrapositive: \( P \implies Q \) or \( \neg Q \implies \neg P \).
Example: \( a^2 \) is odd \( \implies a \) is odd.
Contrapositive: \( a \) is even \( \implies a^2 \) is even.

Contradiction: \( P \)
\( \neg P \implies \text{false} \)
Useful to prove something does not exist:
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.

Contrapositive: $a$ is even $\implies a^2$ is even.

**Contradiction:** $P$

$\neg P \implies \text{false}$

Useful to prove something does not exist:

Example: rational representation of $\sqrt{2}$
...and then proofs...

**Direct:** \( P \implies Q \)

Example: \( a \) is even \( \implies a^2 \) is even.

Approach: What is even? \( a = 2k \)

\[
a^2 = 4k^2 = 2(2k^2)
\]

Integers closed under multiplication! So \( 2k^2 \) is even.

\( a^2 \) is even.

**Contrapositive:** \( P \implies Q \) or \( \neg Q \implies \neg P \).

Example: \( a^2 \) is odd \( \implies a \) is odd.

Contrapositive: \( a \) is even \( \implies a^2 \) is even.

**Contradiction:** \( P \)

\( \neg P \implies \text{false} \)

Useful to prove something does not exist:

Example: rational representation of \( \sqrt{2} \) does not exist.
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even.

$\implies a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.

Contrapositive: $a$ is even $\implies a^2$ is even.

**Contradiction:** $P$

$\neg P \implies false$

Useful to prove something does not exist:

Example: rational representation of $\sqrt{2}$ does not exist.

Example: finite set of primes
...and then proofs...

**Direct:** $P \implies Q$

Example: $a$ is even $\implies a^2$ is even.

Approach: What is even? $a = 2k$

$a^2 = 4k^2 = 2(2k^2)$

Integers closed under multiplication! So $2k^2$ is even. $a^2$ is even.

**Contrapositive:** $P \implies Q$ or $\neg Q \implies \neg P$.

Example: $a^2$ is odd $\implies a$ is odd.

Contrapositive: $a$ is even $\implies a^2$ is even.

**Contradiction:** $P$

$\neg P \implies \text{false}$

Useful to prove something does not exist:

Example: rational representation of $\sqrt{2}$ does not exist.

Example: finite set of primes does not exist.
...and then proofs...

**Direct:** \( P \implies Q \)

Example: \( a \) is even \( \implies \) \( a^2 \) is even.

Approach: What is even? \( a = 2k \)

\[
a^2 = 4k^2 = 2(2k^2)
\]

Integers closed under multiplication! So \( 2k^2 \) is even. \( a^2 \) is even.

**Contraposition:** \( P \implies Q \) or \( \neg Q \implies \neg P \).

Example: \( a^2 \) is odd \( \implies \) \( a \) is odd.

Contraposition: \( a \) is even \( \implies \) \( a^2 \) is even.

**Contradiction:** \( P \)

\( \neg P \implies \text{false} \)

Useful to prove something does not exist:

Example: rational representation of \( \sqrt{2} \) does not exist.

Example: finite set of primes does not exist.

Example: rogue couple does not exist.
Contradiction in induction:
Contradiction in induction:
   Find a place where induction step doesn’t hold.
Contradiction in induction:
Find a place where induction step doesn’t hold.
Something something Well ordering principle...
...jumping forward..

Contradiction in induction:
   Find a place where induction step doesn’t hold.
   Something something Well ordering principle...

Contradiction in Stable Marriage:
...jumping forward..

Contradiction in induction:
Find a place where induction step doesn’t hold.
Something something Well ordering principle...

Contradiction in Stable Marriage:
First day where no woman improves.
...jumping forward..

Contradiction in induction:
   Find a place where induction step doesn’t hold.
   Something something Well ordering principle...

Contradiction in Stable Marriage:
   First day where no woman improves. Does not exist.
...jumping forward..

Contradiction in induction:
Find a place where induction step doesn’t hold.
Something something Well ordering principle...

Contradiction in Stable Marriage:
First day where no woman improves. Does not exist.

Contradiction in Countability:
...jumping forward..

Contradiction in induction:
  Find a place where induction step doesn’t hold.
  Something something Well ordering principle...

Contradiction in Stable Marriage:
  First day where no woman improves. Does not exist.

Contradiction in Countability:
  Assume there is a list with all the real numbers.
...jumping forward..

Contradiction in induction:
   Find a place where induction step doesn’t hold.
   Something something Well ordering principle...

Contradiction in Stable Marriage:
   First day where no woman improves. Does not exist.

Contradiction in Countability:
   Assume there is a list with all the real numbers. Impossible.
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n). \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1, 8 \mid 3^{2n} - 1. \)
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8 | 3^{2n} - 1 \).

Induction on \( n \).
...and then induction...

\[ P(0) \land ((\forall n) (P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8 \mid 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 \mid 3^2 - 1 \).
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n). \]

**Thm:** For all \( n \geq 1 \), \( 8 \mid 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 \mid 3^2 - 1 \).
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n + 1)) \equiv (\forall n \in N) P(n)). \]

**Thm**: For all \( n \geq 1 \), \( 8 \mid 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 \mid 3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1, 8 \mid 3^{2n} - 1. \)

Induction on \( n. \)

Base: \( 8 \mid 3^2 - 1. \)

Induction Hypothesis: Assume \( P(n): \) True for some \( n. \)

Induction Step: Prove \( P(n+1) \)
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)) \]

**Thm:** For all \( n \geq 1 \), \( 8|3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8|3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).

Induction Step: Prove \( P(n+1) \)

\[ 3^{2n+2} - 1 = \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8 \mid 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 \mid 3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).

Induction Step: Prove \( P(n+1) \)

\[ 3^{2n+2} - 1 = 9(3^{2n}) - 1 \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8 \mid 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 \mid 3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).
\[
(3^{2n} - 1 = 8d)
\]

Induction Step: Prove \( P(n+1) \)
\[ 3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)} \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8|3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8|3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).
\( (3^{2n} - 1 = 8d) \)

Induction Step: Prove \( P(n+1) \)
\[ 3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)} \]
\[ = 9(8d + 1) - 1 \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) P(n). \]

**Thm:** For all \( n \geq 1, 8 \mid 3^{2n} - 1. \)

Induction on \( n. \)

Base: \( 8 \mid 3^2 - 1. \)

Induction Hypothesis: Assume \( P(n): \) True for some \( n. \)

\( (3^{2n} - 1 = 8d) \)

Induction Step: Prove \( P(n + 1) \)

\[ 3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)} \]
\[ = 9(8d + 1) - 1 \]
\[ = 72d + 8 \]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n + 1)) \equiv (\forall n \in N) P(n)). \]

**Thm:** For all \( n \geq 1 \), \( 8 | 3^{2n} - 1 \).

Induction on \( n \).

Base: \( 8 | 3^2 - 1 \).

Induction Hypothesis: Assume \( P(n) \): True for some \( n \).
\[ (3^{2n} - 1 = 8d) \]

Induction Step: Prove \( P(n + 1) \)
\[
3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)}
\]
\[
= 9(8d + 1) - 1
\]
\[
= 72d + 8
\]
\[
= 8(9d + 1)
\]
...and then induction...

\[ P(0) \land ((\forall n)(P(n) \implies P(n+1)) \equiv (\forall n \in \mathbb{N}) \ P(n)). \]

**Thm:** For all \( n \geq 1, 8 \mid 3^{2n} - 1. \)

Induction on \( n. \)

Base: \( 8 \mid 3^2 - 1. \)

Induction Hypothesis: Assume \( P(n): \text{ True for some } n. \)

\( (3^{2n} - 1 = 8d) \)

Induction Step: Prove \( P(n+1) \)

\[
3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)}
\]
\[
= 9(8d + 1) - 1
\]
\[
= 72d + 8
\]
\[
= 8(9d + 1)
\]

Divisible by 8.
Thm: For all $n \geq 1$, $8 \mid 3^{2n} - 1$.

Induction on $n$.

Base: $8 \mid 3^2 - 1$.

Induction Hypothesis: Assume $P(n)$: True for some $n$. 
\[(3^{2n} - 1 = 8d)\]

Induction Step: Prove $P(n+1)$
\[
3^{2n+2} - 1 = 9(3^{2n}) - 1 \quad \text{(by induction hypothesis)}
\]
\[
= 9(8d + 1) - 1
\]
\[
= 72d + 8
\]
\[
= 8(9d + 1)
\]

Divisible by 8.
Graphs

\[ G = (V, E) \]

- **Vertex set**: \( V \)
- **Edge set**: \( E \subseteq V \times V \)
- Directed graph: ordered pair of vertices
- **Adjacent**: Two vertices are connected by an edge
- **Incident**: An edge that shares a vertex
- **Degree**: Total number of edges connected to a vertex
  - **In-degree**: Number of edges incident to a vertex
  - **Out-degree**: Number of edges pointing away from a vertex

**Theorem**

\[ \text{Sum of degrees} = 2 |E| \]

An edge is incident to two vertices.

**Pair of Vertices are Connected**

- If there is a path between them.

**Connected Component**

- Maximal set of connected vertices

**Connected Graph**

- One connected component
Graphs

\[ G = (V, E) \]
\[ V - \text{set of vertices.} \]
Graphs...

\( G = (V, E) \)
- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices. 
Adjacent, Incident, Degree.
In-degree, Out-degree.

Thm: \( \sum \text{of degrees} = 2 \mid E \mid \).

Edge is incident to 2 vertices.
Degree of vertices is total incidences.

Pair of Vertices are Connected: If there is a path between them.

Connected Component: maximal set of connected vertices.
Connected Graph: one connected component.
Graphs

\[ G = (V, E) \]
- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.
Adjacent, Incident, Degree.
In-degree, Out-degree.

Thm: Sum of degrees is 2 times the number of edges. 
Edge is incident to 2 vertices.
Degree of vertices is total incidences.
Pair of Vertices are Connected: If there is a path between them.
Connected Component: maximal set of connected vertices.
Connected Graph: one connected component.
Graphs...

\[ G = (V, E) \]

- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

- Adjacent, Incident, Degree.
- In-degree, Out-degree.

Thm: Sum of degrees is \( |E| \).

- Edge is incident to 2 vertices.
- Degree of vertices is total incidences.

Pair of Vertices are Connected: If there is a path between them.

Connected Component: maximal set of connected vertices.

Connected Graph: one connected component.
...Graphs...

\[ G = (V, E) \]
\[ V \text{ - set of vertices.} \]
\[ E \subseteq V \times V \text{ - set of edges.} \]

Directed: ordered pair of vertices.

Directed: ordered pair of vertices.

Thm: Sum of degrees is 2 | E |

Edge is incident to 2 vertices.

Degree of vertices is total incidences.

Pair of Vertices are Connected: If there is a path between them.

Connected Component: maximal set of connected vertices.

Connected Graph: one connected component.
Graphs

\[ G = (V, E) \]

- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

Thm: Sum of degrees is 2\(|E|\).

Edge is incident to 2 vertices.

Degree of vertices is total incidences.

Pair of Vertices are Connected: If there is a path between them.

Connected Component: maximal set of connected vertices.

Connected Graph: one connected component.
Graphs

\[ G = (V, E) \]
\[ V - \text{set of vertices.} \]
\[ E \subseteq V \times V - \text{set of edges.} \]

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

Theorem:

\[ \text{Sum of degrees is } 2 |E|. \]

Edge is incident to 2 vertices.

Pair of Vertices are Connected:

If there is a path between them.

Connected Component: maximal set of connected vertices.

Connected Graph: one connected component.
Graphs...

\[ G = (V, E) \]
\[ V \] - set of vertices.
\[ E \subseteq V \times V \] - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.
In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
Graphs

\[ G = (V, E) \]
- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.
- Adjacent, Incident, Degree.
- In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
- Edge is incident to 2 vertices.
Graphs

\[ G = (V, E) \]

- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).

- Edge is incident to 2 vertices.
- Degree of vertices is total incidences.
Graphs...

\[ G = (V, E) \]

\( V \) - set of vertices.
\( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
Edge is incident to 2 vertices.
Degree of vertices is total incidences.
Graphs...

\[ G = (V, E) \]
\[ V \text{ - set of vertices.} \]
\[ E \subseteq V \times V \text{ - set of edges.} \]

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.
In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
Edge is incident to 2 vertices.
Degree of vertices is total incidences.

Pair of Vertices are Connected:
Graphs

\[ G = (V, E) \]
- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.
- In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
- Edge is incident to 2 vertices.
- Degree of vertices is total incidences.

Pair of Vertices are Connected:
- If there is a path between them.
Graphs

\[ G = (V, E) \]

- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.
In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
Edge is incident to 2 vertices.
Degree of vertices is total incidences.

Pair of Vertices are Connected:
If there is a path between them.
Graphs

\[ G = (V, E) \]
\[ V \text{ - set of vertices.} \]
\[ E \subseteq V \times V \text{ - set of edges.} \]

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.
In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).
Edge is incident to 2 vertices.
Degree of vertices is total incidences.

Pair of Vertices are Connected:
If there is a path between them.

Connected Component: maximal set of connected vertices.
Graphs

\[ G = (V, E) \]
\[ V - \text{set of vertices.} \]
\[ E \subseteq V \times V - \text{set of edges.} \]

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).

Edge is incident to 2 vertices.

Degree of vertices is total incidences.

Pair of Vertices are Connected:

If there is a path between them.

Connected Component: maximal set of connected vertices.
Graphs

\[ G = (V, E) \]

- \( V \) - set of vertices.
- \( E \subseteq V \times V \) - set of edges.

Directed: ordered pair of vertices.

Adjacent, Incident, Degree.

In-degree, Out-degree.

**Thm:** Sum of degrees is \( 2|E| \).

- Edge is incident to 2 vertices.
- Degree of vertices is total incidences.

Pair of Vertices are Connected:

- If there is a path between them.

Connected Component: maximal set of connected vertices.

Connected Graph: one connected component.
**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.
**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**

- Take a walk using each edge at most once.
- Property: return to starting point.
- Why? Even degree.
- Remove the walk from the graph
- Recurse on connected components.
- Put together.
- Property: walk visits every component.
- Proof Idea: Original graph connected.
**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
Thm: Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

Algorithm:
- Take a walk using each edge at most once.
- Property: return to starting point.
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
- **Property:** return to starting point.
  - Why? Even degree.
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
- **Property:** return to starting point.
  - Why? Even degree. Always have an option
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- **Take a walk using each edge at most once.**
- **Property:** return to starting point.
  - Why? Even degree. Always have an option
**Graph Algorithm: Eulerian Tour**

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.

**Property:** return to starting point.
- Why? Even degree. Always have an option

Remove the walk from the graph
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- **Take a walk using each edge at most once.**
- **Property:** return to starting point.
  - Why? Even degree. Always have an option

Remove the walk from the graph
Recurse on connected components.
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
- **Property:** return to starting point.
  - Why? Even degree. Always have an option

  Remove the walk from the graph
  Recurse on connected components.
  Put together.
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
- **Property:** return to starting point.
  - Why? Even degree. Always have an option

Remove the walk from the graph
Recurse on connected components.
Put together.
- **Property:** walk visits every component.
Graph Algorithm: Eulerian Tour

**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
  - **Property:** return to starting point.
    - Why? Even degree. Always have an option

Remove the walk from the graph
Recurse on connected components.
Put together.
  - **Property:** walk visits every component.
    - Proof Idea: Original graph connected.
**Thm:** Every connected graph where every vertex has even degree has an Eulerian Tour; a tour which visits every edge exactly once.

**Algorithm:**
- Take a walk using each edge at most once.
- **Property:** return to starting point.
  - Why? Even degree. Always have an option

Remove the walk from the graph
Recurse on connected components.
Put together.
- **Property:** walk visits every component.
  - Proof Idea: Original graph connected.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.

Notice that the last one, has one three colors.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.

Notice that the last one, has one three colors. Fewer colors than number of vertices.
Graph Coloring.

Given $G = (V, E)$, a coloring of a $G$ assigns colors to vertices $V$ where for each edge the endpoints have different colors.

Notice that the last one, has one three colors. Fewer colors than number of vertices. Fewer colors than max degree node.
Planar graphs and maps.

Planar graph coloring $\equiv$ map coloring.
Planar graphs and maps.

Planar graph coloring $\equiv$ map coloring.

Four color theorem is about planar graphs!
**Theorem:** Every planar graph can be colored with six colors.
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
From Euler’s Formula:
**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: $e \leq 3v - 6$ for any planar graph.

From Euler’s Formula: $v + f = e + 2$. 

Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.

From Euler’s Formula: \( v + f = e + 2 \). 
\[ 3f \leq 2e \]
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
   From Euler’s Formula: \( v + f = e + 2 \).
   \( 3f \leq 2e \)

Total degree: \( 2e \)
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
From Euler’s Formula: \( v + f = e + 2 \).
\[ 3f \leq 2e \]
Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \)
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: $e \leq 3v - 6$ for any planar graph.
    From Euler’s Formula: $v + f = e + 2$.
    $3f \leq 2e$

Total degree: $2e$
Average degree: $\leq \frac{2e}{v} \leq \frac{2(3v-6)}{v}$
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
   From Euler’s Formula: \( v + f = e + 2 \).
   \[ 3f \leq 2e \]

Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v} \).
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: $e \leq 3v - 6$ for any planar graph.
   From Euler’s Formula: $v + f = e + 2$.
   $3f \leq 2e$

Total degree: $2e$
Average degree: $\leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v}$.

There exists a vertex with degree $< 6$. 


Theorem: Every planar graph can be colored with six colors.

Proof:
Recall: $e \leq 3v - 6$ for any planar graph.
    From Euler’s Formula: $v + f = e + 2$.
    $3f \leq 2e$

Total degree: $2e$
Average degree: $\leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v}$.

There exists a vertex with degree $< 6$ or at most 5.
**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
   - From Euler’s Formula: \( v + f = e + 2 \).
   - \( 3f \leq 2e \)

Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v} \).

There exists a vertex with degree \(< 6\) or at most 5.
   - Remove vertex \( v \) of degree at most 5.
Six color theorem.

**Theorem:** Every planar graph can be colored with six colors.

**Proof:**
Recall: \( e \leq 3v - 6 \) for any planar graph.
From Euler’s Formula: \( v + f = e + 2 \).

\[ 3f \leq 2e \]

Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v} \).

There exists a vertex with degree \(< 6\) or at most \(5\).

Remove vertex \(v\) of degree at most \(5\).
Inductively color remaining graph.
Theorem: Every planar graph can be colored with six colors.

Proof:
Recall: \( e \leq 3v - 6 \) for any planar graph.
   From Euler’s Formula: \( v + f = e + 2 \).
   \( 3f \leq 2e \)

Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v} \).

There exists a vertex with degree < 6 or at most 5.
   Remove vertex \( v \) of degree at most 5.
   Inductively color remaining graph.
   Color is available for \( v \) since only five neighbors...
Theorem: Every planar graph can be colored with six colors.

Proof:
Recall: \( e \leq 3v - 6 \) for any planar graph.

From Euler’s Formula: \( v + f = e + 2 \).
\[ 3f \leq 2e \]

Total degree: \( 2e \)
Average degree: \( \leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v} \).

There exists a vertex with degree < 6 or at most 5.

Remove vertex \( v \) of degree at most 5.
Inductively color remaining graph.
Color is available for \( v \) since only five neighbors...
and only five colors are used.
Theorem: Every planar graph can be colored with six colors.

Proof:
Recall: $e \leq 3v - 6$ for any planar graph.
From Euler’s Formula: $v + f = e + 2$.
$3f \leq 2e$

Total degree: $2e$
Average degree: $\leq \frac{2e}{v} \leq \frac{2(3v-6)}{v} \leq 6 - \frac{12}{v}$.

There exists a vertex with degree $< 6$ or at most 5.

Remove vertex $v$ of degree at most 5.
Inductively color remaining graph.
Color is available for $v$ since only five neighbors... and only five colors are used.
Theorem: Every planar graph can be colored with five colors.
Theorem: Every planar graph can be colored with five colors.

Proof:
Theorem: Every planar graph can be colored with five colors.

Proof: Not Today!
Four Color Theorem
Four Color Theorem

**Theorem:** Any planar graph can be colored with four colors.
Four Color Theorem

**Theorem:** Any planar graph can be colored with four colors.

**Proof:**
Theorem: Any planar graph can be colored with four colors.
Proof: Not Today!
Four Color Theorem

**Theorem:** Any planar graph can be colored with four colors.

**Proof:** Not Today!
Graph Types: Complete Graph.

- $K_n$, $|V| = n$ every edge present.
- Degree of vertex $|V| - 1$.
- Very connected. Lots of edges: $n(n - 1)/2$. Wow.
Graph Types: Complete Graph.

$K_n, \ |V| = n$
Graph Types: Complete Graph.

$K_n$, $|V| = n$

every edge present.
Graph Types: Complete Graph.

$K_n$, $|V| = n$

every edge present.
degree of vertex?
Graph Types: Complete Graph.

\[ K_n, \ |V| = n \]

every edge present.

degree of vertex? \(|V| - 1\).
Graph Types: Complete Graph.

$K_n, |V| = n$

every edge present.
degree of vertex? $|V| - 1$.

Very connected.
Graph Types: Complete Graph.

$K_n, |V| = n$

- every edge present.
- degree of vertex? $|V| - 1$.

Very connected.
Lots of edges:
Graph Types: Complete Graph.

$K_n$, $|V| = n$

- every edge present.
- degree of vertex? $|V| - 1$.

Very connected.

Lots of edges: $n(n-1)/2$. 
Graph Types: Complete Graph.

\[ K_n, \ |V| = n \]

- every edge present.
- degree of vertex? \(|V| - 1\).

Very connected.
Lots of edges: \( n(n - 1)/2 \).
Wow.
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.
- An acyclic graph where any edge addition creates a cycle.
- To tree or not to tree!

Property: Can remove a single node and break into components of size at most $|V|/2$. 
Definitions:

A connected graph without a cycle.
Trees.

Definitions:

A connected graph without a cycle.
A connected graph with $|V| - 1$ edges.
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.

Property: Can remove a single node and break into components of size at most $|V|/2$. 
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.
- An acyclic graph where any edge addition creates a cycle.
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.
- An acyclic graph where any edge addition creates a cycle.
Trees.

Definitions:

A connected graph without a cycle.
A connected graph with $|V| - 1$ edges.
A connected graph where any edge removal disconnects it.
An acyclic graph where any edge addition creates a cycle.

To tree or not to tree!
Trees.

Definitions:

A connected graph without a cycle.
A connected graph with $|V| - 1$ edges.
A connected graph where any edge removal disconnects it.
An acyclic graph where any edge addition creates a cycle.

To tree or not to tree!

Minimally connected, minimum number of edges to connect.
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.
- An acyclic graph where any edge addition creates a cycle.

To tree or not to tree!

Minimally connected, minimum number of edges to connect.

Property:
Trees.

Definitions:

- A connected graph without a cycle.
- A connected graph with $|V| - 1$ edges.
- A connected graph where any edge removal disconnects it.
- An acyclic graph where any edge addition creates a cycle.

To tree or not to tree!

Minimally connected, minimum number of edges to connect.

Property:
- Can remove a single node and break into components of size at most $|V|/2$. 
Hypercube

Hypercubes.

Wait what?

I thought it was $n^2 - 1$.

Oh...

$2^n = |V|...

Also represents bit-strings nicely.

$G = (V, E)$

$|V| = \{0, 1\}^n$

$|E| = \{(x, y) | x$ and $y$ differ in exactly one bit position.\}$

0

1

00

10

01

11

000

010

001

011

100

110

101

111
Hypercubes. Really connected.
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what?
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what? I thought it was $n2^{n-1}$.
Hypercube

Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what? I thought it was $n2^{n-1}$. Oh...
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|...
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...
Also represents bit-strings nicely.
Hypercube

Hypercubes. Really connected. $O(|V| \log |V|)$ edges!
Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...
Also represents bit-strings nicely.
Hypercube

Hypercubes. Really connected. $O(|V| \log |V|)$ edges!

Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...

Also represents bit-strings nicely.

$G = (V, E)$
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!

Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...

Also represents bit-strings nicely.

$$G = (V, E)$$

$$|V| = \{0, 1\}^n,$$
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!

Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...

Also represents bit-strings nicely.

$$G = (V, E)$$

$|V| = \{0, 1\}^n$,

$|E| = \{(x, y)|x \text{ and } y \text{ differ in exactly one bit position.}\}$
Hypercubes. Really connected. $O(|V| \log |V|)$ edges!

Wait what? I thought it was $n2^{n-1}$. Oh... $2^n = |V|$...

Also represents bit-strings nicely.

$G = (V, E)$

$|V| = \{0, 1\}^n$,

$|E| = \{(x, y)|x$ and $y$ differ in exactly one bit position.$\}$
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n-1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$.
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n-1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$.
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n-1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges ($0x, 1x$).
A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n - 1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$.
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n-1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$. 
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n-1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$. 
Recursive Definition.

A 0-dimensional hypercube is a node labelled with the empty string of bits.

An $n$-dimensional hypercube consists of a 0-subcube (1-subcube) which is a $n - 1$-dimensional hypercube with nodes labelled $0x$ ($1x$) with the additional edges $(0x, 1x)$. 
Hypercube: properties

Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.
Dense cuts: Cutting off \( k \) nodes needs \( \geq k \) edges.

FYI:
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.
FYI: Also cuts represent boolean functions.
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.
Hypercube: properties

Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.
Hypercube: properties

Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
Hypercube: properties

Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
Get from 000100 to 101000.
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
Get from 000100 to 101000.
  000100 $\rightarrow$ 100100 $\rightarrow$ 101100 $\rightarrow$ 101000

Correct bits in string, moves along path in hypercube!

Good communication network!
Hypercube: properties

Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
Get from 000100 to 101000.

\[
000100 \rightarrow 100100 \rightarrow 101100 \rightarrow 101000
\]

Correct bits in string, moves along path in hypercube!
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
Get from 000100 to 101000.

000100 $\rightarrow$ 100100 $\rightarrow$ 101100 $\rightarrow$ 101000

Correct bits in string, moves along path in hypercube!
Dense cuts: Cutting off $k$ nodes needs $\geq k$ edges.

FYI: Also cuts represent boolean functions. One side of the cut takes value 0. The other side takes value 1.

Nice Paths between nodes.
   Get from 000100 to 101000.
      000100 $\rightarrow$ 100100 $\rightarrow$ 101100 $\rightarrow$ 101000
   Correct bits in string, moves along path in hypercube!

Good communication network!
Bipartite graphs

There is a cut with all the edges.

Cycles have length 4 or more edges.
Bipartite graphs

There is a cut with all the edges.
Bipartite graphs

There is a cut with all the edges.

Cycles have length 4 or more edges.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.
Stable Marriage: a study in definitions and WOP.

\( n \)-men, \( n \)-women.

Each person has completely ordered preference list
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.
Stable Marriage: a study in definitions and WOP.

\[ n \text{-men, } n \text{-women.} \]

Each person has completely ordered preference list that contains every person of opposite gender.

**Pairing/Marching.**
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs?
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs? $n$. 
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are **partners** in pairing.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people exactly once.
How many pairs? $n$.
People in pair are partners in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once. How many pairs? $n$.
People in pair are partners in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are **partners** in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners

**Stable Pairing.**
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_j, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are **partners** in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners

**Stable Pairing.**
Pairing with no rogue couples.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_j, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are *partners* in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners

**Stable Pairing.**
Pairing with no rogue couples.

Does stable pairing exist?
Stable Marriage: a study in definitions and WOP.

\( n \)-men, \( n \)-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs \((m_i, w_j)\) containing all people \emph{exactly} once.
How many pairs? \( n \).
People in pair are \textbf{partners} in pairing.

**Rogue Couple in a pairing.**
A \( m_j \) and \( w_k \) who like each other more than their partners

**Stable Pairing.**
Pairing with no rogue couples.

Does stable pairing exist?
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are **partners** in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners

**Stable Pairing.**
Pairing with no rogue couples.

Does stable pairing exist?
Yes for matching.
Stable Marriage: a study in definitions and WOP.

$n$-men, $n$-women.

Each person has completely ordered preference list contains every person of opposite gender.

**Pairing/Marching.**
Set of pairs $(m_i, w_j)$ containing all people *exactly* once.
How many pairs? $n$.
People in pair are **partners** in pairing.

**Rogue Couple in a pairing.**
A $m_j$ and $w_k$ who like each other more than their partners

**Stable Pairing.**
Pairing with no rogue couples.

Does stable pairing exist?
Yes for matching.
No, for roommates problem.
Traditional Marriage Algorithm:

Each Day:

Every man proposes to his favorite woman from the ones that haven't already rejected him.

Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:

- Man crosses off woman who rejected him.
- Woman's current proposer is "on string."

Key Property: Improvement Lemma:

Every day, if man on string for woman, \[ \Rightarrow \] any future man on string is better.
(proof by contradiction)

Stability:

No rogue couple.

\[ \text{rogue couple} (M, W) \Rightarrow M \text{ proposed to } W \Rightarrow W \text{ ended up with someone she liked better than } M. \]

Not rogue couple!
Traditional Marriage Algorithm:

Each Day:

Every man proposes to his favorite woman from the ones that haven’t already rejected him.

Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:

Man crosses off woman who rejected him.

Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:

Every day, if man on string for woman, then any future man on string is better.

(proof by contradiction)

Stability:

No rogue couple.

rogue couple \((M,W)\) = \(M\) proposed to \(W\) = \(W\) ended up with someone she liked better than \(M\).

Not rogue couple!
Traditional Marriage Algorithm:

Each Day:
   Every man proposes to his favorite woman from the ones that haven’t already rejected him.
Traditional Marriage Algorithm:

Each Day:
Every man proposes to his favorite woman from the ones that haven’t already rejected him.
Every woman rejects all but best man who proposes.
Traditional Marriage Algorithm:

Each Day:

Every man proposes to his favorite woman from the ones that haven’t already rejected him.
Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man **crosses off** woman who rejected him.
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man **crosses off** woman who rejected him.
- Woman’s current proposer is “**on string**.”
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man **crosses off** woman who rejected him.
- Woman’s current proposer is “**on string.**”

Key Property: Improvement Lemma:
- Every day, if man on string for woman,
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
- Every day, if man on string for woman, any future man on string is better.
Traditional Marriage Algorithm:

**Each Day:**
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

**Useful Algorithmic Definitions:**
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

**Key Property: Improvement Lemma:**
- Every day, if man on string for woman,
  \[
  \Rightarrow \text{ any future man on string is better. (proof by contradiction)}
  \]
Traditional Marriage Algorithm:

**Each Day:**
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

**Useful Algorithmic Definitions:**
- Man **crosses off** woman who rejected him.
- Woman’s current proposer is “on string.”

**Key Property: Improvement Lemma:**
- Every day, if man on string for woman,
  - any future man on string is better. (proof by contradiction)

**Stability:**
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
- Every day, if man on string for woman,
  \[ \implies \] any future man on string is better. (proof by contradiction)

Stability: No rogue couple.
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
- Every day, if man on string for woman,
  \[ \implies \text{any future man on string is better.} \] (proof by contradiction)

Stability: No rogue couple.
- rogue couple (M,W)
Traditional Marriage Algorithm:

Each Day:
Every man proposes to his favorite woman from the ones that haven’t already rejected him.
Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
Man crosses off woman who rejected him.
Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
Every day, if man on string for woman,
\[ \implies \text{any future man on string is better.} \] (proof by contradiction)

Stability: No rogue couple.
rogue couple (M,W)
\[ \implies M \text{ proposed to } W \]
Traditional Marriage Algorithm:

Each Day:
  Every man proposes to his favorite woman from the ones that haven’t already rejected him.
  Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
  Man crosses off woman who rejected him.
  Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
  Every day, if man on string for woman,
  \[\implies\] any future man on string is better. (proof by contradiction)

Stability: No rogue couple.
  rogue couple (M,W)
  \[\implies\] M proposed to W
  \[\implies\] W ended up with someone she liked better than M.
Traditional Marriage Algorithm:

Each Day:
- Every man proposes to his favorite woman from the ones that haven’t already rejected him.
- Every woman rejects all but best man who proposes.

Useful Algorithmic Definitions:
- Man crosses off woman who rejected him.
- Woman’s current proposer is “on string.”

Key Property: Improvement Lemma:
- Every day, if man on string for woman,
  \[ \implies \] any future man on string is better. (proof by contradiction)

Stability: No rogue couple.
- rogue couple (M,W)
  \[ \implies \] M proposed to W
  \[ \implies \] W ended up with someone she liked better than \( M \).
- Not rogue couple!
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.

Thm: TMA produces male optimal pairing, S.

Proof by contradiction: Let M be the first man to propose to someone worse than optimal partner W. TMA: M asked W. And then got replaced by M'. W prefers M'. How much doesn't M like W? Better than his match in optimal pairing? Impossible. Worse than his match in the optimal pairing? Then M wasn't the first!!

Thm: Woman pessimal. Man optimal =⇒ Woman pessimal. Woman optimal =⇒ Man pessimal.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
Not necessarily first in list.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.
Optimality/Pessimal

Optimal partner if best partner in any *stable* pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.

**Proof by contradiction:** Let $M$ be the first man to propose to someone worse than optimal partner $W$.

TMA: $M$ asked $W$. And then got replaced by $M'$!

$W$ prefers $M'$.

How much doesn $M'$ like $W$?
Better than his match in optimal pairing?
Impossible.

Worse than his match in the optimal pairing?
Then $M$ wasn't the first!!

**Thm:** Woman pessimal.
Man optimal $\Rightarrow$ Woman pessimal.
Woman optimal $\Rightarrow$ Man pessimal.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$. 
**Proof by contradiction:** Let $M$ be the first man to propose to someone worse than optimal partner $W$. 
TMA: $M$ asked $W$. And then got replaced by $M'$!
$W$ prefers $M'$. 
How much doesn $M'$ like $W$?
Better than his match in optimal pairing? Impossible.
Worse than his match in the optimal pairing? Then $M$ wasn't the first!!

**Thm:** woman pessimal.
Man optimal $\Rightarrow$ Woman pessimal.
Woman optimal $\Rightarrow$ Man pessimal.
Optimality/Pessimal

Optimal partner if best partner in any **stable** pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$. 
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.
Proof by contradiction:
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

Thm: TMA produces male optimal pairing, $S$.
   Proof by contradiction:
   Let $M$ be the first man to propose to someone worse than optimal partner $W$. 


Optimality/Pessimal

Optimal partner if best partner in any stable pairing. Not necessarily first in list. Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

Thm: TMA produces male optimal pairing, S.
Proof by contradiction:
Let $M$ be the first man to propose to someone worse than optimal partner $W$.
TMA: $M$ asked $W$. And then got replaced by $M'$!
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.  
Not necessarily first in list.  
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

Thm: TMA produces male optimal pairing, S.  
Proof by contradiction:  
Let $M$ be the first man to propose to someone worse than optimal partner $W$.  
TMA: $M$ asked $W$. And then got replaced by $M'$!  
$W$ prefers $M'$.  

Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.
   Proof by contradiction:
   Let $M$ be the first man to propose to someone worse than optimal partner $W$.
   TMA: $M$ asked $W$. And then got replaced by $M'$!
   $W$ prefers $M'$.
   How much doesn $M'$ like $W$?
Optimality/Pessimal

Optimal partner if best partner in any stable pairing. Not necessarily first in list. Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.
Proof by contradiction:
Let $M$ be the first man to propose to someone worse than optimal partner $W$.
TMA: $M$ asked $W$. And then got replaced by $M'$!
$W$ prefers $M'$.
How much doesn $M'$ like $W$?
Better than his match in optimal pairing?
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, \( S \).
   Proof by contradiction:
   \( M \) be the first man to propose to someone worse than optimal partner \( W \).
   TMA: \( M \) asked \( W \). And then got replaced by \( M' \)!
   \( W \) prefers \( M' \).
   How much doesn \( M' \) like \( W \)?
   Better than his match in optimal pairing? Impossible.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.
   Proof by contradiction:
   Let $M$ be the first man to propose to someone worse than optimal partner $W$.
   TMA: $M$ asked $W$. And then got replaced by $M'$!
   $W$ prefers $M'$.
   How much doesn $M'$ like $W$?
   Better than his match in optimal pairing? Impossible.
   Worse than his match in the optimal pairing?
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
   Not necessarily first in list.
   Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, S.
   Proof by contradiction:
   Let $M$ be the first man to propose to someone worse than optimal partner $W$.
   TMA: $M$ asked $W$. And then got replaced by $M'$!
   $W$ prefers $M'$.
   How much doesn $M'$ like $W$?
   Better than his match in optimal pairing? Impossible.
   Worse than his match in the optimal pairing?
      Then $M$ wasn’t the first!!
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.
Proof by contradiction:
Let $M$ be the first man to propose to someone worse than optimal partner $W$.
TMA: $M$ asked $W$. And then got replaced by $M'$!
$W$ prefers $M'$.
How much doesn $M'$ like $W$?
Better than his match in optimal pairing? Impossible.
Worse than his match in the optimal pairing?
Then $M$ wasn’t the first!!
Optimality/Pessimal

Optimal partner if best partner in any stable pairing. Not necessarily first in list. Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, S.
Proof by contradiction:
Let $M$ be the first man to propose to someone worse than optimal partner $W$.

TMA: $M$ asked $W$. And then got replaced by $M'$!

$W$ prefers $M'$.

How much doesn $M'$ like $W$?
Better than his match in optimal pairing? Impossible.
Worse than his match in the optimal pairing?
Then $M$ wasn’t the first!!

**Thm:** Woman pessimal.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing.
Not necessarily first in list.
Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.

Proof by contradiction:
Let $M$ be the first man to propose to someone worse than optimal partner $W$.

TMA: $M$ asked $W$. And then got replaced by $M'$!
$W$ prefers $M'$.
How much doesn $M'$ like $W$?
Better than his match in optimal pairing? Impossible.
Worse than his match in the optimal pairing?
Then $M$ wasn’t the first!!

**Thm:** woman pessimal.

Man optimal $\implies$ Woman pessimal.
Optimality/Pessimal

Optimal partner if best partner in any stable pairing. Not necessarily first in list. Possibly no stable pairing with that partner.

Man-optimal pairing is pairing where every man gets optimal partner.

**Thm:** TMA produces male optimal pairing, $S$.

Proof by contradiction:

Let $M$ be the first man to propose to someone worse than optimal partner $W$.

TMA: $M$ asked $W$. And then got replaced by $M'$!

$W$ prefers $M'$.

How much doesn't $M'$ like $W$?
Better than his match in optimal pairing? Impossible.
Worse than his match in the optimal pairing?

Then $M$ wasn’t the first!!

**Thm:** woman pessimal.

Man optimal $\implies$ Woman pessimal.
Woman optimal $\implies$ Man pessimal.
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...

Why?

There is a list!!

Some things are not countable, like the reals, or the set of all subsets of the naturals...

Why?

Diagonalization: Assume there is a list. Can construct a diagonal element $x$. $x$ is not in the list! Contradiction.
And then countability

More than one infinities
And then countability

More than one infinities
Some things are countable
More than one infinities

Some things are countable, like the natural numbers
More than one infinities

Some things are countable, like the natural numbers, or the rationals...
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...
   Why?
And then countability

**More than one infinities**

Some things are countable, like the natural numbers, or the rationals...
  Why? There is a list!!
And then countability

**More than one infinities**

Some things are countable, like the natural numbers, or the rationals...
   Why? There is a list!!
Some things are not countable
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...

Why? There is a list!!
Some things are not countable, like the reals
And then countability

**More than one infinities**

Some things are countable, like the natural numbers, or the rationals...

  Why? There is a list!!

Some things are not countable, like the reals, or the set of all subsets of the naturals...
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...

    Why? There is a list!!

Some things are not countable, like the reals, or the set of all subsets of the naturals...

    Why?
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...
   Why? There is a list!!
Some things are not countable, like the reals, or the set of all subsets of the naturals...
   Why? Diagonalization:
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...
  Why? There is a list!!
Some things are not countable, like the reals, or the set of all subsets of the naturals...
  Why? **Diagonalization:** Well, assume there is a list.
And then countability

More than one infinities
Some things are countable, like the natural numbers, or the rationals...
  Why? There is a list!!
Some things are not countable, like the reals, or the set of all subsets of the naturals...
  Why? **Diagonalization:** Well, assume there is a list. Can construct a diagonal element \( x \).
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...

- Why? There is a list!!

Some things are not countable, like the reals, or the set of all subsets of the naturals...

- Why? **Diagonalization:** Well, assume there is a list. Can construct a diagonal element \( x \). \( x \) is not in the list!
And then countability

More than one infinities

Some things are countable, like the natural numbers, or the rationals...
   Why? There is a list!!
Some things are not countable, like the reals, or the set of all subsets of the naturals...
   Why? **Diagonalization:** Well, assume there is a list. Can construct a diagonal element $x$. $x$ is not in the list! Contradiction.
The HALT problem:
Is there a program that can tell you if another (generic) program halts on an input?

NO!
Why?
Self reference!
Who cares?
Using the same trick I can show that a bunch of problems are undecidable!
Like: Will this program P even print "Hello World"?
Or "Is there an input for this program P that will give an attacker admin access?"
The HALT problem:
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?
HALTING

The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

NO!
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

**NO!**

Why?
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

NO!
Why? Self reference!
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

NO!

Why? Self reference!

Who cares?
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

**NO!**

Why? Self reference!

Who cares? Using the same trick I can show that a bunch of problems are undecidable!
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

**NO!**

Why? Self reference!

Who cares? Using the same trick I can show that a bunch of problems are undecidable!

  - Like: Will this program $P$ even print "Hello World"?
The HALT problem: Is there a program that can tell you if another (generic) program halts on an input?

**NO!**

Why? Self reference!

Who cares? Using the same trick I can show that a bunch of problems are undecidable!

- Like: Will this program $P$ even print "Hello World"?
- Or "Is there an input for this program $P$ that will give an attacker admin access?"
Counting!

Sample $k$ items out of $n$.

<table>
<thead>
<tr>
<th></th>
<th>With Replacement</th>
<th>Without Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order matters</td>
<td>$n^k$</td>
<td>$\frac{n!}{(n-k)!}$</td>
</tr>
<tr>
<td>Order doesn’t matter</td>
<td>$\binom{n+k-1}{n-1}$</td>
<td>$\binom{n}{k}$</td>
</tr>
</tbody>
</table>
Confusion yesterday: 10 hats.
7 days.
I can wear the same hat on different days (replacement).
I don't care which day I wore what (order doesn't matter).
Why is this stars and bars?
How many stars?
One for each day.
So 7
How many bars?
One fewer than the hats.
So 9

|⋆ | ⋆ |
| ⋆ | ⋆ |
| ⋆ | ⋆ |
| ⋆ | ⋆ |
Didn't wear hats 1 and 2. Wore hat 3 for 1 day, hat 4 for 2 days, hat 5 days. Didn't wear hats 6 and 7. Hat 8 for 3 days. Didn't wear hats 9 and 10.
Stars and bars!

Confusion yesterday: 10 hats.
Confusion yesterday: 10 hats. 7 days.
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement).
Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?
Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars?
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day.
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day. So 7
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?
How many stars? One for each day. So 7
How many bars?
Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day. So 7

How many bars? One fewer than the hats.
Stars and bars!

Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day. So 7

How many bars? One fewer than the hats. So 9
Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day. So 7

How many bars? One fewer than the hats. So 9

||*|**|***||***||
Confusion yesterday: 10 hats. 7 days. I can wear the same hat on different days (replacement). I don’t care which day I wore what (order doesn’t matter).

Why is this stars and bars?

How many stars? One for each day. So 7

How many bars? One fewer than the hats. So 9

Didn’t wear hats 1 and 2. Wore hat 3 for 1 day, hat 4 for 2 days, hat 5 days. Didn’t wear hats 6 and 7. Hat 8 for 3 days. Didn’t wear hats 9 and 10.
Combinatorial Proofs.

Easy ones:
\[ \binom{n}{k} = \binom{n}{n-k} \]

Harder ones:
\[ \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \]

What's the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What's the thing on the right? Each subset either has, or doesn't have 1. How many subsets of size \( k \) have 1? \( k-1 \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k-1} \) How many subsets of size \( k \) don't have 1? \( k \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k} \)

Add them up. (Sum rule)
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What's the thing on the left? Number of subsets of size \( k \) of \{1, 2, ..., n+1\}.

What's the thing on the right? Each subset either has, or doesn't have 1. How many subsets of size \( k \) have 1? \( k \) elements left to pick, from \{2, ..., n+1\}. \( \binom{n}{k-1} \) How many subsets of size \( k \) don't have 1? \( k \) elements left to pick, from \{2, ..., n+1\}. \( \binom{n}{k} \) Add them up. (Sum rule)
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left?
Combinatorial Proofs.

Easy ones: \((\binom{n}{k}) = (\binom{n}{n-k})\)

Harder ones: \((\binom{n+1}{k}) = (\binom{n}{k}) + (\binom{n}{k-1})\)

What’s the thing on the left? Number of subsets of size \(k\) of \(\{1,2,\ldots,n+1\}\).
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right?
Combinatorial Proofs.

Easy ones: $\binom{n}{k} = \binom{n}{n-k}$

Harder ones: $\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1}$

What’s the thing on the left? Number of subsets of size $k$ of \{1, 2, ..., $n+1$\}.

What’s the thing on the right? Each subset either has, or doesn’t have 1.
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1?
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k - 1 \) elements left to pick
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k-1 \) elements left to pick, from \( \{2, \ldots, n+1\} \).
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k - 1 \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k-1} \)
Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k - 1 \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k-1} \)

How many subsets of size \( k \) don’t have 1?
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have \( 1 \).

How many subsets of size \( k \) have \( 1 \)? \( k - 1 \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k-1} \)

How many subsets of size \( k \) don’t have \( 1 \)? \( k \) elements left to pick.
Combinatorial Proofs.

Easy ones: $\binom{n}{k} = \binom{n}{n-k}$

Harder ones: $\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1}$

What’s the thing on the left? Number of subsets of size $k$ of $\{1, 2, \ldots, n+1\}$.

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size $k$ have 1? $k - 1$ elements left to pick, from $\{2, \ldots, n+1\}$. $\binom{n}{k-1}$

How many subsets of size $k$ don’t have 1? $k$ elements left to pick, from $\{2, \ldots, n+1\}$. 
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \{1, 2, \ldots, n+1\}.

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k-1 \) elements left to pick, from \{2, \ldots, n+1\}. \( \binom{n}{k-1} \)

How many subsets of size \( k \) don’t have 1? \( k \) elements left to pick, from \{2, \ldots, n+1\}. \( \binom{n}{k} \)
Combinatorial Proofs.

Easy ones: \( \binom{n}{k} = \binom{n}{n-k} \)

Harder ones: \( \binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \)

What’s the thing on the left? Number of subsets of size \( k \) of \( \{1, 2, \ldots, n+1\} \).

What’s the thing on the right? Each subset either has, or doesn’t have 1.

How many subsets of size \( k \) have 1? \( k-1 \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k-1} \)

How many subsets of size \( k \) don’t have 1? \( k \) elements left to pick, from \( \{2, \ldots, n+1\} \). \( \binom{n}{k} \)

Add them up. (Sum rule)
Midterm format

Time: 110 minutes.
Midterm format

Time: 110 minutes.

Some short answers.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well:

Know material fast, correct.
Know material medium, slower, less correct.
Know material not so well, Uh oh.
Some longer questions.
Proofs, properties.
Not so much calculation.

Remember that a problem from hw and/or discussions is in the midterm! (identical or almost identical)
So study those!
Midterm format

Time: 110 minutes.

Some short answers.
   Get at ideas that you learned.
   If something is taking too long maybe there is a trick!
   Know material well: fast,
Midterm format

Time: 110 minutes.

Some short answers.
   Get at ideas that you learned.
   If something is taking too long maybe there is a trick!
   Know material well: fast, correct.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium:
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium: slower,
Midterm format

Time: 110 minutes.

Some short answers.
Get at ideas that you learned.
If something is taking too long maybe there is a trick!
Know material well: fast, correct.
Know material medium: slower, less correct.
Midterm format

Time: 110 minutes.

Some short answers.
   Get at ideas that you learned.
   If something is taking too long maybe there is a trick!
   Know material well:       fast, correct.
   Know material medium:     slower, less correct.
   Know material not so well:
Midterm format

Time: 110 minutes.

Some short answers.
Get at ideas that you learned.
If something is taking too long maybe there is a trick!
Know material well: fast, correct.
Know material medium: slower, less correct.
Know material not so well: Uh oh.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well:  fast, correct.
  Know material medium:  slower, less correct.
  Know material not so well:  Uh oh.

Some longer questions.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium: slower, less correct.
  Know material not so well: Uh oh.

Some longer questions.
  Proofs,
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium: slower, less correct.
  Know material not so well: Uh oh.

Some longer questions.
  Proofs, properties.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium: slower, less correct.
  Know material not so well: Uh oh.

Some longer questions.
  Proofs, properties.
  Not so much calculation.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well:   fast, correct.
  Know material medium: slower, less correct.
  Know material not so well:  Uh oh.

Some longer questions.
  Proofs, properties.
  Not so much calculation.
Midterm format

Time: 110 minutes.

Some short answers.
  Get at ideas that you learned.
  If something is taking too long maybe there is a trick!
  Know material well: fast, correct.
  Know material medium: slower, less correct.
  Know material not so well: Uh oh.

Some longer questions.
  Proofs, properties.
  Not so much calculation.

Remember that a problem from hw and/or discussions is in the midterm! (identical or almost identical)
Midterm format

Time: 110 minutes.

Some short answers.
Get at ideas that you learned.
If something is taking too long maybe there is a trick!
Know material well: fast, correct.
Know material medium: slower, less correct.
Know material not so well: Uh oh.

Some longer questions.
Proofs, properties.
Not so much calculation.

Remember that a problem from hw and/or discussions is in the midterm! (identical or almost identical)

So study those!
FAQ

Will this proof from the notes that I don't like be in the midterm?
No.

The why should I study it?
Understanding a complex proof is a useful skill. Also, big proofs are usually a bunch of little proofs put together. And every proof is a new trick. And we like tricks!
FAQ

- Will this proof from the notes that I don’t like be in the midterm?

  No.
  Understanding a complex proof is a useful skill. Also, big proofs are usually a bunch of little proofs put together. And every proof is a new trick. And we like tricks!
FAQ

► Will this proof from the notes that I don’t like be in the midterm?

No.
FAQ

- Will this proof from the notes that I don’t like be in the midterm?
  No.
- The why should I study it?
FAQ

- Will this proof from the notes that I don’t like be in the midterm?
  No.

- The why should I study it?
  Understanding a complex proof is a useful skill.
FAQ

Will this proof from the notes that I don’t like be in the midterm?
No.

The why should I study it?
Understanding a complex proof is a useful skill.
Also, big proofs are usually a bunch of little proofs put together.
FAQ

- Will this proof from the notes that I don’t like be in the midterm?
  No.

- The why should I study it?
  Understanding a complex proof is a useful skill.
  Also, big proofs are usually a bunch of little proofs put together.
  And every proof is a new trick.
FAQ

▶ Will this proof from the notes that I don’t like be in the midterm?
   No.

▶ The why should I study it?
   Understanding a complex proof is a useful skill.

   Also, big proofs are usually a bunch of little proofs put together. And every proof is a new trick. And we like tricks!
Wrapup.
Wrapup.

If you sent us an email about Midterm conflicts
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by know.

Other issues....
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by know.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by know.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!
If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
  email us.
  Private message on piazza.

Good (sort of last minute)
Studying!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by know.

Other issues....
  email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!
If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts Other arrangements. Should have received an email from us. You should know what to do by now.

Other issues.... email us. Private message on piazza.

Good (sort of last minute) Studying!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by now.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!!!!
Wrapup.

If you sent us an email about Midterm conflicts
Other arrangements.
Should have received an email from us.
You should know what to do by know.

Other issues....
email us.
Private message on piazza.

Good (sort of last minute)
Studying!!!!!!!!!!!!!!!!!!!