Propositional Logic
**Question:** How do we formalize the definitions and reasoning we use in our proofs?
Where We're Going

- *Propositional Logic* (Today)
  - Reasoning about Boolean values.
- *First-Order Logic* (Wednesday/Friday)
  - Reasoning about properties of multiple objects.
Propositional Logic
A *proposition* is a statement that is either true or false.

In other words, *English sentences can be propositions, but not all are (for example, commands and questions can’t be propositions)*.
Propositional Logic

- **Propositional logic** is a mathematical system for reasoning about propositions and how they relate to one another.
- Every statement in propositional logic consists of *propositional variables* combined via *propositional connectives*.
  - Each variable represents some proposition, such as “You liked it” or “You should have put a ring on it.”
  - Connectives encode how propositions are related, such as “If you liked it, then you should have put a ring on it.”
Propositional Logic as a Boolean Algebra

• In elementary school arithmetic, we learn that two expressions are equivalent, *for specific numbers*:

\[(9 + 5) / 7 = (1/7)(9 + 5)\]

\[(14)/7 = (1/7)(14)\]

\[2 = 2\]

• In high school, we learn algebra, which lets us study the structural patterns of equivalence, *regardless of the specific numbers involved*:

\[(a + b) / c = (1/c)(a + b)\]

• Algebra replaces the numbers with variables so we can focus on analyzing and manipulating the structure.
Propositional Logic as a Boolean Algebra

- Philosophers, mathematicians, and logicians wanted to do the same thing that algebra does for arithmetic, but for the analysis of the structure of arguments not analysis of the structure of numeric calculations.

- We replace individual English sentences that state facts with propositional variables, and replace the “if...then,” “and,” “or,” etc. with operator symbols.

- So we can focus on analyzing and manipulating the structure.
Propositional Variables

• Each proposition will be represented by a *propositional variable*.

• Propositional variables are usually represented as lower-case letters, such as $p, q, r, s$, etc.

• Each variable can take one one of two values: true or false.
Propositional Connectives

• There are seven propositional connectives, many of which will be familiar from programming.

• First, there’s the logical “NOT” operation: \( \neg p \)

• You’d read this out loud as “not \( p \).”

• The fancy name for this operation is **logical negation**.
Propositional Connectives

• There are seven propositional connectives, many of which will be familiar from programming.

• Next, there’s the logical “AND” operation:

  \[ p \land q \]

• You’d read this out loud as “p and q.”

• The fancy name for this operation is *logical conjunction*. 
Propositional Connectives

- There are seven propositional connectives, many of which will be familiar from programming.
- Then, there’s the logical “OR” operation: $p \lor q$
  - You’d read this out loud as “$p$ or $q$.”
  - The fancy name for this operation is logical disjunction. This is an inclusive or.
Truth Tables

• A *truth table* is a table showing the truth value of a propositional logic formula as a function of its inputs.

• Let’s go look at the truth tables for the three connectives we’ve seen so far:

<table>
<thead>
<tr>
<th></th>
<th>∼</th>
<th>∧</th>
<th>∨</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Summary of Important Points

• The $\lor$ connective is an *inclusive* “or.” It's true if at least one of the operands is true.
  • Similar to the $||$ operator in C, C++, Java, etc. and the `or` operator in Python.
• If we need an exclusive “or” operator, we can build it out of what we already have.
• Try this yourself! Take a minute to combine these operators together to form an expression that represents the exclusive or of $p$ and $q$ (something that’s true if and only if exactly one of $p$ and $q$ are true.)
Mathematical Implication
Implication

• We can represent implications using this connective:

   \[ p \rightarrow q \]

• You’d read this out loud as “\( p \) implies \( q \).”

• **Question:** What should the truth table for \( p \rightarrow q \) look like?

• Pull out a sheet of paper, make a guess, and talk things over with your neighbors!
Implication

Dr. Lee: “If you pick a perfect March Madness bracket this year, then I’ll give you an A+ in CS103.”

What if...

• ...you pick a perfect bracket and get an A+?
• ...you pick a bad bracket and get an A+?
• ...you pick a perfect bracket and get a C?
• ...you pick a bad bracket and get a C?
Implication

• ...you pick a **perfect** bracket and get an A+?
• ...you pick a bad bracket and get an A+?
• ...you pick a **perfect** bracket and get a C?
• ...you pick a bad bracket and get a C?

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
### Implication

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Implication

- ...you pick a perfect bracket and get an A+?
- ...you pick a bad bracket and get an A+?
- ...you pick a perfect bracket and get a C?
- ...you pick a bad bracket and get a C?

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>q</td>
<td>p → q</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

| T | T | T | T |
Implication

- ...you pick a perfect bracket and get an A+?
- ...you pick a bad bracket and get an A+?
- ...you pick a perfect bracket and get a C?
- ...you pick a bad bracket and get a C?

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Implication

- ...you pick a **perfect** bracket and get an A+?
- ...you pick a **bad** bracket and get an A+?
- ...you pick a **perfect** bracket and get a C?
- ...you pick a bad bracket and get a C?

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$q$</td>
<td>$p \rightarrow q$</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>
• ...you pick a perfect bracket and get an A+?
• ...you pick a bad bracket and get an A+?
• ...you pick a perfect bracket and get a C?
• ...you pick a bad bracket and get a C?

Implication

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Implication

- ...you pick a **perfect** bracket and get an A+?
- ...you pick a bad bracket and get an A+?
- ...you pick a **perfect** bracket and get a C?
- ...you pick a bad bracket and get a C?

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Implication

- ...you pick a perfect bracket and get an A+?
- ...you pick a bad bracket and get an A+?
- ...you pick a perfect bracket and get a C?
- ...you pick a bad bracket and get a C?

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Implication

- ...you pick a perfect bracket and get an A+?
- ...you pick a bad bracket and get an A+?
- ...you pick a perfect bracket and get a C?
- ...you pick a bad bracket and get a C?

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$q$</td>
<td>$p \rightarrow q$</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>$p$</td>
<td>$q$</td>
<td>$p \rightarrow q$</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>------------------</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
An implication is false only when the antecedent is true and the consequent is false.

Every formula is either true or false, so these other entries have to be true.
**Important observation:**
The statement $p \rightarrow q$ is true whenever $p \land \neg q$ is false.
An implication with a false antecedent is called *vacuously true*. 

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Please commit this table to memory. We’re going to need it, extensively, over the next couple of weeks.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>$F$</td>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
</tbody>
</table>
Fun Fact: The Contrapositive Revisited
The Biconditional Connective
The Biconditional Connective

- On Friday, we saw that “$p$ if and only if $q$” means both that $p \rightarrow q$ and $q \rightarrow p$.
- We can write this in propositional logic using the biconditional connective:

$$p \leftrightarrow q$$

- This connective’s truth table has the same meaning as “$p$ implies $q$ and $q$ implies $p$.”
- Based on that, what should its truth table look like?
- Take a guess, and talk it over with your neighbor!
Biconditionals

- The **biconditional** connective $p \leftrightarrow q$ is read “$p$ if and only if $q$.”
- Here's its truth table:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \leftrightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Biconditionals

- The *biconditional* connective $p \leftrightarrow q$ is read “$p$ if and only if $q$.”
- Here's its truth table:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \leftrightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

One interpretation of $\leftrightarrow$ is to think of it as equality: the two propositions must have equal truth values.
True and False

- There are two more “connectives” to speak of: true and false.
  - The symbol $\top$ is a value that is always true.
  - The symbol $\bot$ is value that is always false.
- These are often called connectives, though they don't connect anything.
  - (Or rather, they connect zero things.)
Proof by Contradiction

• Suppose you want to prove \( p \) is true using a proof by contradiction.

• The setup looks like this:
  • Assume \( p \) is false.
  • Derive something that we know is false.
  • Conclude that \( p \) is true.

• In propositional logic:
  \[
  (\neg p \rightarrow \bot) \rightarrow p
  \]
Operator Precedence

• How do we parse this statement?

\[ \neg x \rightarrow y \lor z \rightarrow x \lor y \land z \]

• Operator precedence for propositional logic:

\[ \neg, \land, \lor, \rightarrow, \leftrightarrow \]

• All operators are right-associative.

• We can use parentheses to disambiguate.
Operator Precedence

- How do we parse this statement?
  \[ \neg x \rightarrow y \lor z \rightarrow x \lor y \land z \]

- Operator precedence for propositional logic:

- All operators are right-associative.
- We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?
  \((\neg x) \rightarrow y \lor z \rightarrow x \lor y \land z\)

• Operator precedence for propositional logic:

  \text{\textbf{\textcolor{red}{\rightarrow}}} \quad \text{\textbf{\textcolor{blue}{\land}}} \quad \text{\textbf{\textcolor{green}{\lor}}} \quad \text{\textbf{\textcolor{magenta}{\neg}}} \quad \text{\textbf{\textcolor{cyan}{\leftrightarrow}}}

• All operators are right-associative.

• We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?

\((\neg x) \to y \lor z \to x \lor y \land z\)

• Operator precedence for propositional logic:

\[\begin{array}{c}
\neg \\
\land \\
\lor \\
\to \\
\leftrightarrow
\end{array}\]

• All operators are right-associative.

• We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?

\[(\neg x) \rightarrow y \lor z \rightarrow x \lor (y \land z)\]

• Operator precedence for propositional logic:

\[\neg \land \lor \rightarrow \leftrightarrow\]

• All operators are right-associative.
• We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?

\[(\neg x) \to y \lor z \to x \lor (y \land z)\]

• Operator precedence for propositional logic:

\[
\begin{align*}
\neg & \quad \land & \quad \lor & \quad \to & \quad \leftrightarrow \\
& & & & 
\end{align*}
\]

• All operators are right-associative.

• We can use parentheses to disambiguate.
Operator Precedence

- How do we parse this statement?
  
  \((\neg x) \rightarrow (y \lor z) \rightarrow (x \lor (y \land z))\)

- Operator precedence for propositional logic:

  - \(\neg\)
  - \(\land\)
  - \(\lor\)
  - \(\rightarrow\)
  - \(\leftrightarrow\)

- All operators are right-associative.
- We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?
  $$(\neg x) \rightarrow (y \lor z) \rightarrow (x \lor (y \land z))$$

• Operator precedence for propositional logic:

  \[
  \rightarrow \quad \land \quad \lor \quad \neg
  \]

• All operators are right-associative.
• We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?

   \((\neg x) \rightarrow ((y \lor z) \rightarrow (x \lor (y \land z)))\)

• Operator precedence for propositional logic:

  \[
  \neg \quad \land \quad \lor \quad \rightarrow \quad \leftrightarrow
  \]

• All operators are right-associative.
• We can use parentheses to disambiguate.
Operator Precedence

• How do we parse this statement?

\[ (\neg x) \rightarrow ((y \lor z) \rightarrow (x \lor (y \land z))) \]

• Operator precedence for propositional logic:

\[ \begin{align*}
\neg \\
\land \\
\lor \\
\rightarrow \\
\leftrightarrow
\end{align*} \]

• All operators are right-associative.
• We can use parentheses to disambiguate.
Operator Precedence

• The main points to remember:
  • \( \neg \) binds to whatever immediately follows it.
  • \( \land \) and \( \lor \) bind more tightly than \( \rightarrow \).
  • We will commonly write expressions like \( p \land q \rightarrow r \) without adding parentheses.

• *For more complex expressions, let’s agree to use parentheses!*
# The Big Table

<table>
<thead>
<tr>
<th>Connective</th>
<th>Read Aloud As</th>
<th>C++ Version</th>
<th>Fancy Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>¬</td>
<td>“not”</td>
<td>!</td>
<td>Negation</td>
</tr>
<tr>
<td>∧</td>
<td>“and”</td>
<td>&amp;&amp;</td>
<td>Conjunction</td>
</tr>
<tr>
<td>∨</td>
<td>“or”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>→</td>
<td>“implies”</td>
<td>see PS2!</td>
<td>Implication</td>
</tr>
<tr>
<td>↔</td>
<td>“if and only if”</td>
<td>see PS2!</td>
<td>Biconditional</td>
</tr>
<tr>
<td>T</td>
<td>“true”</td>
<td>true</td>
<td>Truth</td>
</tr>
<tr>
<td>⊥</td>
<td>“false”</td>
<td>false</td>
<td>Falsity</td>
</tr>
</tbody>
</table>
Recap So Far

• A *propositional variable* is a variable that is either true or false.

• The *propositional connectives* are
  • Negation: \( \neg p \)
  • Conjunction: \( p \land q \)
  • Disjunction: \( p \lor q \)
  • Implication: \( p \rightarrow q \)
  • Biconditional: \( p \leftrightarrow q \)
  • True: \( \top \)
  • False: \( \bot \)
Translating into Propositional Logic
Some Sample Propositions

\(a\): I will be in the path of totality.

\(b\): I will see a total solar eclipse.
Some Sample Propositions

\[ a: \text{I will be in the path of totality.} \]
\[ b: \text{I will see a total solar eclipse.} \]

“\text{I won't see a total solar eclipse if I'm not in the path of totality.}”
Some Sample Propositions

\[ \neg a \rightarrow \neg b \]

\(a\): I will be in the path of totality.

\(b\): I will see a total solar eclipse.

“I won't see a total solar eclipse if I'm not in the path of totality.”
“p if q” translates to $q \rightarrow p$

It does not translate to $p \rightarrow q$
Some Sample Propositions

\( a \): I will be in the path of totality.
\( b \): I will see a total solar eclipse.
\( c \): There is a total solar eclipse today.
Some Sample Propositions

$a$: I will be in the path of totality.

$b$: I will see a total solar eclipse.

$c$: There is a total solar eclipse today.

“If I will be in the path of totality, but there's no solar eclipse today, I won't see a total solar eclipse.”
Some Sample Propositions

\[ a: \text{I will be in the path of totality.} \]
\[ b: \text{I will see a total solar eclipse.} \]
\[ c: \text{There is a total solar eclipse today.} \]

“\text{If I will be in the path of totality, but there's no solar eclipse today, I won't see a total solar eclipse.}”

\[ a \land \neg c \rightarrow \neg b \]
“p, but q” translates to

\( p \land q \)
The Takeaway Point

• When translating into or out of propositional logic, be very careful not to get tripped up by nuances of the English language.
  • In fact, this is one of the reasons we have a symbolic notation in the first place!

• Many prepositional phrases lead to counterintuitive translations; make sure to double-check yourself!
Propositional Equivalences
Quick Question:

What would I have to show you to convince you that the statement $p \land q$ is false?
Quick Question:
What would I have to show you to convince you that the statement \( p \lor q \) is false?
de Morgan's Laws

• Using truth tables, we concluded that 
  \[ \neg(p \land q) \]
  is equivalent to 
  \[ \neg p \lor \neg q \]

• We also saw that 
  \[ \neg(p \lor q) \]
  is equivalent to 
  \[ \neg p \land \neg q \]

• These two equivalences are called **De Morgan's Laws**.
de Morgan's Laws in Code

• **Pro tip:** Don't write this:

```java
if (!(p() && q())) {
    /* … */
}
```

• Write this instead:

```java
if (!p() || !q()) {
    /* … */
}
```

• (This even short-circuits correctly!)
An Important Equivalence

• Earlier, we talked about the truth table for $p \rightarrow q$. We chose it so that

$$p \rightarrow q \text{ is equivalent to } \neg(p \land \neg q)$$

• Later on, this equivalence will be incredibly useful:

$$\neg(p \rightarrow q) \text{ is equivalent to } p \land \neg q$$
Another Important Equivalence

- Here's a useful equivalence. Start with $p \rightarrow q$ is equivalent to $\neg (p \land \neg q)$

- By de Morgan's laws:
  
  $p \rightarrow q$ is equivalent to $\neg (p \land \neg q)$
  
  is equivalent to $\neg p \lor \neg \neg q$
  
  is equivalent to $\neg p \lor q$

- Thus $p \rightarrow q$ is equivalent to $\neg p \lor q$
Another Important Equivalence

• Here's a useful equivalence. Start with

\[ p \rightarrow q \text{ is equivalent to } \neg (p \land \neg q) \]

• By de Morgan's laws:

\[ p \rightarrow q \text{ is equivalent to } \neg p \lor \neg \neg q \]

\[ p \rightarrow q \text{ is equivalent to } \neg p \lor q \]

• Thus \( p \rightarrow q \) is equivalent to \( \neg p \lor q \)

If \( p \) is false, then \( \neg p \lor q \) is true. If \( p \) is true, then \( q \) has to be true for the whole expression to be true.
Next Time

- **First-Order Logic**
  - Reasoning about groups of objects.
- **First-Order Translations**
  - Expressing yourself in symbolic math!