First-Order Logic

Part One
Recap from Last Time
Recap So Far

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

• The *propositional connectives* are as follows:
  • Negation: $\neg p$
  • Conjunction: $p \land q$
  • Disjunction: $p \lor q$
  • Implication: $p \rightarrow q$
  • Biconditional: $p \leftrightarrow q$
  • True: $\top$
  • False: $\bot$
Take out a sheet of paper!
What's the truth table for the $\rightarrow$ connective?
What's the negation of $p \rightarrow q$?
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:

```c
if (!(p() && q())) {
    /* ... */
}
```

• Write this instead:

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

• Later on, this equivalence will be incredibly useful:
  \[
  \neg (p \rightarrow q) \quad \text{is equivalent to} \quad p \land \neg q
  \]
Another Important Equivalence

• Here's a useful equivalence. Start with
  \[ p \to q \quad \text{is equivalent to} \quad \neg(p \land \neg q) \]
• By de Morgan's laws:
  \[ p \to q \quad \text{is equivalent to} \quad \neg(p \land \neg q) \]
  \[ \text{is equivalent to} \quad \neg p \lor \neg \neg q \]
  \[ \text{is equivalent to} \quad \neg p \lor q \]
• Thus \( p \to q \) is equivalent to \( \neg p \lor q \)
Another Important Equivalence

- Here's a useful equivalence. Start with:
  \[ p \rightarrow q \quad \text{is equivalent to} \quad \neg(p \land \neg q) \]
- By de Morgan's laws:
  \[ p \rightarrow q \quad \text{is equivalent to} \quad \neg p \lor \neg \neg q \]
  \[ p \rightarrow q \quad \text{is equivalent to} \quad \neg p \lor q \]
- 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.
New Stuff!
First-Order Logic
What is First-Order Logic?

- **First-order logic** is a logical system for reasoning about properties of objects.
- Augments the logical connectives from propositional logic with
  - *predicates* that describe properties of objects,
  - *functions* that map objects to one another, and
  - *quantifiers* that allow us to reason about multiple objects.
Some Examples
Likes(You, Eggs) ∧ Likes(You, Tomato) → Likes(You, Shakshuka)
Likes(You, Eggs) \land Likes(You, Tomato) \to Likes(You, Shakshuka)

Learns(You, History) \lor ForeverRepeats(You, History)

In(MyHeart, Havana) \land TookBackTo(Him, Me, EastAtlanta)
Likes(You, Eggs) \land Likes(You, Tomato) \rightarrow Likes(You, Shakshuka)

Learns(You, History) \lor ForeverRepeats(You, History)

In(MyHeart, Havana) \land TookBackTo(Him, Me, EastAtlanta)
Likes(You, Eggs) ∧ Likes(You, Tomato) → Likes(You, Shakshuka)

Learns(You, History) ∨ ForeverRepeats(You, History)

In(MyHeart, Havana) ∧ TookBackTo(Him, Me, EastAtlanta)

These blue terms are called constant symbols. Unlike propositional variables, they refer to objects, not propositions.
Likes(You, Eggs) ∧ Likes(You, Tomato) → Likes(You, Shakshuka)

Learns(You, History) ∨ ForeverRepeats(You, History)

In(MyHeart, Havana) ∧ TookBackTo(Him, Me, EastAtlanta)
\[ \text{Likes}(\text{You}, \text{Eggs}) \land \text{Likes}(\text{You}, \text{Tomato}) \rightarrow \text{Likes}(\text{You}, \text{Shakshuka}) \]

\[ \text{Learns}(\text{You}, \text{History}) \lor \text{ForeverRepeats}(\text{You}, \text{History}) \]

\[ \text{In}(\text{MyHeart}, \text{Havana}) \land \text{TookBackTo}(\text{Him}, \text{Me}, \text{EastAtlanta}) \]

The red things that look like function calls are called \textit{predicates}. Predicates take objects as arguments and evaluate to true or false.
\text{Likes}(\text{You}, \text{Eggs}) \land \text{Likes}(\text{You}, \text{Tomato}) \rightarrow \text{Likes}(\text{You}, \text{Shakshuka})

\text{Learns}(\text{You}, \text{History}) \lor \text{ForeverRepeats}(\text{You}, \text{History})

\text{In}(\text{MyHeart}, \text{Havana}) \land \text{ TookBackTo}(\text{Him, Me, EastAtlanta})
What remains are traditional propositional connectives. Because each predicate evaluates to true or false, we can connect the truth values of predicates using normal propositional connectives.
Reasoning about Objects

• To reason about objects, first-order logic uses *predicates*.

• Examples:

  \[ \textit{Cute}(\textit{Quokka}) \]
  \[ \textit{ArgueIncessantly}(\textit{Democrats, Republicans}) \]

• Applying a predicate to arguments produces a proposition, which is either true or false.

• Typically, when you’re working in FOL, you’ll have a list of predicates, what they stand for, and how many arguments they take. It’ll be given separately than the formulas you write.
First-Order Sentences

- Sentences in first-order logic can be constructed from predicates applied to objects:
  
  \[ \text{Cute}(a) \rightarrow \text{Dikdik}(a) \lor \text{Kitty}(a) \lor \text{Puppy}(a) \]
  
  \[ \text{Succeeds}(\text{You}) \iff \text{Practices}(\text{You}) \]

  \[ x < 8 \rightarrow x < 137 \]

The less-than sign is just another predicate. Binary predicates are sometimes written in infix notation this way.

Numbers are not "built in" to first-order logic. They're constant symbols just like "You" and "a" above.
Equality

• First-order logic is equipped with a special predicate $\equiv$ that says whether two objects are equal to one another.

• Equality is a part of first-order logic, just as $\rightarrow$ and $\neg$ are.

• Examples:

  \[
  \text{TomMarvoloRiddle} = \text{LordVoldemort} \\
  \text{MorningStar} = \text{EveningStar}
  \]

• Equality can only be applied to objects; to state that two propositions are equal, use $\leftrightarrow$. 
Let's see some more examples.
\[ \text{FavoriteMovieOf(You)} \neq \text{FavoriteMovieOf(Date)} \land \text{StarOf(FavoriteMovieOf(You))} = \text{StarOf(FavoriteMovieOf(Date))} \]
\[ \text{FavoriteMovieOf(You)} \neq \text{FavoriteMovieOf(Date)} \land \text{StarOf(FavoriteMovieOf(You))} = \text{StarOf(FavoriteMovieOf(Date))} \]
\[ \text{FavoriteMovieOf}(\text{You}) \neq \text{FavoriteMovieOf}(\text{Date}) \land \\
\text{StarOf} (\text{FavoriteMovieOf}(\text{You})) = \text{StarOf} (\text{FavoriteMovieOf}(\text{Date})) \]
$\text{FavoriteMovieOf(You)} \neq \text{FavoriteMovieOf(Date)} \land \\
\text{StarOf(FavoriteMovieOf(You))} = \text{StarOf(FavoriteMovieOf(Date))}$
\[ \text{FavoriteMovieOf(You)} \neq \text{FavoriteMovieOf(Date)} \land \text{StarOf(FavoriteMovieOf(You))} = \text{StarOf(FavoriteMovieOf(Date))} \]

These purple terms are **functions**. Functions take objects as input and produce objects as output.
\[ \text{FavoriteMovieOf}(\text{You}) \neq \text{FavoriteMovieOf}(\text{Date}) \land \text{StarOf}(\text{FavoriteMovieOf}(\text{You})) = \text{StarOf}(\text{FavoriteMovieOf}(\text{Date})) \]
\[
FavoriteMovieOf(You) \neq FavoriteMovieOf(Date) \land 
\text{StarOf}(FavoriteMovieOf(You)) = \text{StarOf}(FavoriteMovieOf(Date))
\]
\[
\text{FavoriteMovieOf}(\text{You}) \neq \text{FavoriteMovieOf}(\text{Date}) \land \text{StarOf}(\text{FavoriteMovieOf}(\text{You})) = \text{StarOf}(\text{FavoriteMovieOf}(\text{Date}))
\]
Functions

• First-order logic allows **functions** that return objects associated with other objects.

• Examples:

  \[\text{ColorOf(Money)}\]
  \[\text{MedianOf}(x, y, z)\]
  \[x + y\]

• As with predicates, functions can take in any number of arguments, but always return a single value.

• Functions evaluate to **objects**, not **propositions**.
Objects and Predicates

- When working in first-order logic, be careful to keep objects (actual things) and propositions (true or false) separate.
- You cannot apply connectives to objects:
  \[ \text{Venus} \rightarrow \text{TheSun} \]
- You cannot apply functions to propositions:
  \[ \text{StarOf}(\text{IsRed}(\text{Sun}) \land \text{IsGreen}(\text{Mars})) \]
- Ever get confused? Just ask!
The Type-Checking Table

<table>
<thead>
<tr>
<th>Connectives ($\leftrightarrow, \land, \text{etc.}$) ...</th>
<th>... operate on ...</th>
<th>... and produce</th>
</tr>
</thead>
<tbody>
<tr>
<td>propositions</td>
<td>propositions</td>
<td>a proposition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicates ($=, \text{etc.}$) ...</th>
<th>... operate on ...</th>
<th>... and produce</th>
</tr>
</thead>
<tbody>
<tr>
<td>objects</td>
<td>objects</td>
<td>a proposition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Functions ...</th>
<th>... operate on ...</th>
<th>... and produce</th>
</tr>
</thead>
<tbody>
<tr>
<td>objects</td>
<td>an object</td>
<td></td>
</tr>
</tbody>
</table>
One last (and major) change
Some muggle is intelligent.
Some muggle is intelligent.

$\exists m. (Muggle(m) \land Intelligent(m))$
Some muggle is intelligent.

\[ \exists m. (\text{Muggle}(m) \land \text{Intelligent}(m)) \]

\[ \exists \] is the **existential quantifier** and says "for some choice of m, the following is true."
The Existential Quantifier

• A statement of the form

   \[ \exists x \text{. some-formula} \]

   is true if there exists a choice of \( x \) where \textit{some-formula} is true when that \( x \) is plugged into it.

• Examples:

   \[ \exists x \text{. } (Even(x) \land Prime(x)) \]
   \[ \exists x \text{. } (TallerThan(x, me) \land LighterThan(x, me)) \]
   \[ (\exists w \text{. Will(w)}) \rightarrow (\exists x \text{. Way(x)}) \]
The Existential Quantifier

∃x. Smiling(x)
The Existential Quantifier

∃x. Smiling(x)
The Existential Quantifier

\( \exists x \).

Smiling(\( x \))
The Existential Quantifier

$\exists x. Smiling(x)$
The Existential Quantifier

$\exists x. Smiling(x)$
The Existential Quantifier

∃x. Smiling(x)
The Existential Quantifier

\[ \exists x. \text{Smiling}(x) \]
The Existential Quantifier

$$\exists x. \text{Smiling}(x)$$

Since \(\text{Smiling}(x)\) is true for some choice of \(x\), this statement evaluates to true.
The Existential Quantifier

\[ \exists x. \text{Smiling}(x) \]

Since \( \text{Smiling}(x) \) is true for some choice of \( x \), this statement evaluates to true.
The Existential Quantifier

\[ \exists x. \text{Smiling}(x) \]
The Existential Quantifier

$\exists x. Smiling(x)$
The Existential Quantifier

$\exists x. \text{Smiling}(x)$
The Existential Quantifier

$\exists x. \text{Smiling}(x)$
The Existential Quantifier

$\exists x.\ Smiling(x)$
The Existential Quantifier

$\exists x. \text{Smiling}(x)$
The Existential Quantifier

$\exists x. \text{Smiling}(x)$
The Existential Quantifier

\[ \exists x. \text{Smiling}(x) \]

Since \( \text{Smiling}(x) \) is not true for any choice of \( x \), this statement evaluates to false.
The Existential Quantifier

\[ \exists x. Smiling(x) \]

Since \( Smiling(x) \) is not true for any choice of \( x \), this statement evaluates to false.
The Existential Quantifier

\((\exists x. \text{Smiling}(x)) \rightarrow (\exists y. \text{WearingHat}(y))\)
(∃x. Smiling(x)) → (∃y. WearingHat(y))
The Existential Quantifier

$\exists x. \text{Smiling}(x) \rightarrow \exists y. \text{WearingHat}(y)$

Is this part of the statement true or false?
The Existential Quantifier

Is this part of the statement true or false?

$$(\exists x. \text{Smiling}(x)) \rightarrow (\exists y. \text{WearingHat}(y))$$
The Existential Quantifier

(∃x. Smiling(x)) → (∃y. WearingHat(y))

Is this part of the statement true or false?
The Existential Quantifier

(∃x. Smiling(x)) → (∃y. WearingHat(y))

Is this part of the statement true or false?
The Existential Quantifier

(∃x. Smiling(x)) → (∃y. WearingHat(y))

Is this overall statement true or false?
The Existential Quantifier

The Existential Quantifier is used to denote that there exists at least one element in a set that satisfies a certain property. It is denoted by the symbol "\( \exists \)".

In the context of the provided statement:

\[ (\exists x. \text{Smiling}(x)) \rightarrow (\exists y. \text{WearingHat}(y)) \]

Is this overall statement true or false?
Fun with Edge Cases

∃x. Smiling(x)
Fun with Edge Cases

Existentially-quantified statements are false in an empty world, since nothing exists, period!

\[ \exists x. \text{Smiling}(x) \]
Some Technical Details
Variables and Quantifiers

- Each quantifier has two parts:
  - the variable that is introduced, and
  - the statement that's being quantified.
- The variable introduced is scoped just to the statement being quantified.

\[(\exists x. Loves(You, x)) \land (\exists y. Loves(y, You))\]
Variables and Quantifiers

- Each quantifier has two parts:
  - the variable that is introduced, and
  - the statement that's being quantified.
- The variable introduced is scoped just to the statement being quantified.

\[(\exists x. \text{Loves}(\text{You}, x)) \land (\exists y. \text{Loves}(y, \text{You}))\]
Variables and Quantifiers

- Each quantifier has two parts:
  - the variable that is introduced, and
  - the statement that's being quantified.
- The variable introduced is scoped just to the statement being quantified.

\[(\exists x. \text{Loves}(\text{You}, x)) \land (\exists y. \text{Loves}(y, \text{You}))\]
Variables and Quantifiers

• Each quantifier has two parts:
  • the variable that is introduced, and
  • the statement that's being quantified.

• The variable introduced is scoped just to the statement being quantified.

\((\exists x. \text{Loves}(\text{You}, x)) \land (\exists x. \text{Loves}(x, \text{You}))\)
Variables and Quantifiers

• Each quantifier has two parts:
  • the variable that is introduced, and
  • the statement that's being quantified.

• The variable introduced is scoped just to the statement being quantified.

\[(\exists x. \text{Loves}(You, x)) \land (\exists x. \text{Loves}(x, You))\]

The variable \(x\) just lives here.

A different variable, also named \(x\), just lives here.
Operator Precedence (Again)

• When writing out a formula in first-order logic, quantifiers have precedence just below ¬.

• The statement

\[ \exists x. P(x) \land R(x) \land Q(x) \]

is parsed like this:

\[ \triangleright (\exists x. P(x)) \land (R(x) \land Q(x)) \quad \triangleleft \]

• This is syntactically invalid because the variable \( x \) is out of scope in the back half of the formula.

• To ensure that \( x \) is properly quantified, explicitly put parentheses around the region you want to quantify:

\[ \exists x. (P(x) \land R(x) \land Q(x)) \]
“For any natural number \( n \), \( n \) is even if and only if \( n^2 \) is even”
“For any natural number $n$, $n$ is even if and only if $n^2$ is even”

\[ \forall n. \ (n \in \mathbb{N} \rightarrow (\text{Even}(n) \leftrightarrow \text{Even}(n^2))) \]
“For any natural number $n$, $n$ is even if and only if $n^2$ is even”

$\forall n. \ (n \in \mathbb{N} \rightarrow (\text{Even}(n) \leftrightarrow \text{Even}(n^2)))$

$\forall$ is the universal quantifier and says “for any choice of $n$, the following is true.”
The Universal Quantifier

- A statement of the form
  \[ \forall x. \text{some-formula} \]
  is true if, for every choice of \( x \), the statement \( \text{some-formula} \) is true when \( x \) is plugged into it.

- Examples:
  \[ \forall p. (\text{Puppy}(p) \rightarrow \text{Cute}(p)) \]
  \[ \forall a. (\text{EatsPlants}(a) \lor \text{EatsAnimals}(a)) \]
  \[ \text{Tallest}(\text{SultanKösen}) \rightarrow \]
  \[ \forall x. (\text{SultanKösen} \neq x \rightarrow \text{ShorterThan}(x, \text{SultanKösen})) \]
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

\( \forall x. \text{Smiling}(x) \)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

\[ \forall x. \text{Smiling}(x) \]
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)

Since Smiling(x) is true for every choice of x, this statement evaluates to true.
The Universal Quantifier

∀x. Smiling(x)

Since Smiling(x) is true for every choice of x, this statement evaluates to true.
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)
The Universal Quantifier

∀x. Smiling(x)

Since \( Smiling(x) \) is false for this choice \( x \), this statement evaluates to false.
The Universal Quantifier

\[ \forall x. \text{Smiling}(x) \]

Since \( \text{Smiling}(x) \) is false for this choice of \( x \), this statement evaluates to false.
The Universal Quantifier

\[(\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))\]
The Universal Quantifier

\((\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))\)
The Universal Quantifier

$(\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))$
The Universal Quantifier

$$(\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))$$

Is this part of the statement true or false?
The Universal Quantifier

\[(\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))\]

Is this part of the statement true or false?
The Universal Quantifier

\[(\forall x. \text{Smiling}(x)) \rightarrow (\forall y. \text{WearingHat}(y))\]

Is this part of the statement true or false?
The Universal Quantifier

(∀x. Smiling(x)) → (∀y. WearingHat(y))

Is this overall statement true or false in this scenario?
The Universal Quantifier

(∀x. Smiling(x)) → (∀y. WearingHat(y))

Is this overall statement true or false in this scenario?
∀x. Smiling(x)
Fun with Edge Cases

Universally-quantified statements are said to be *vacuously true* in empty worlds.

$$\forall x. \text{Smiling}(x)$$
Translating into First-Order Logic
Translating Into Logic

- First-order logic is an excellent tool for manipulating definitions and theorems to learn more about them.
- Need to take a negation? Translate your statement into FOL, negate it, then translate it back.
- Want to prove something by contrapositive? Translate your implication into FOL, take the contrapositive, then translate it back.
Translating Into Logic

• When translating from English into first-order logic, we recommend that you 
  think of first-order logic as a mathematical programming language.

• Your goal is to learn how to combine basic concepts (quantifiers, connectives, etc.) together in ways that say what you mean.
Using the predicates

- \textit{Smiling}(x), which states that \(x\) is smiling, and
- \textit{WearingHat}(x), which states that \(x\) is wearing a hat,

write a sentence in first-order logic that says

\textbf{some smiling person wears a hat.}
"Some smiling person wears a hat."

\[ \exists x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \]

\[ \exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]
"Some smiling person wears a hat."

\[ \exists x. \ (Smiling(x) \land WearingHat(x)) \]

\[ \exists x. \ (Smiling(x) \rightarrow WearingHat(x)) \]
∃x. (Smiling(x) ∧ WearingHat(x))

∃x. (Smiling(x) → WearingHat(x))
∃x. (Smiling(x) ∧ WearingHat(x))

∃x. (Smiling(x) → WearingHat(x))
"Some smiling person wears a hat."  True

\[ \exists x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \]
\[ \exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]
"Some smiling person wears a hat." True

$\exists x. \ (\text{Smiling}(x) \land \text{WearingHat}(x))$ True

$\exists x. \ (\text{Smiling}(x) \rightarrow \text{WearingHat}(x))$
∃x. (Smiling(x) ∧ WearingHat(x))

∃x. (Smiling(x) → WearingHat(x))
Some smiling person wears a hat. 

\[
\exists x. (\text{Smiling}(x) \land \text{WearingHat}(x)) 
\]

\[
\exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) 
\]
∃x. (Smiling(x) ∧ WearingHat(x))

∃x. (Smiling(x) → WearingHat(x))
∃x. (Smiling(x) ∧ WearingHat(x))

∃x. (Smiling(x) → WearingHat(x))

“Some smiling person wears a hat.”
"Some smiling person wears a hat."

\[ \exists x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \]

\[ \exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]
"Some smiling person wears a hat."

\[ \exists x. \ (Smiling(x) \land WearingHat(x)) \]

\[ \exists x. \ (Smiling(x) \rightarrow WearingHat(x)) \]
"Some smiling person wears a hat."  False

\[ \exists x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \]

\[ \exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]
∃x. (Smiling(x) ∧ WearingHat(x))
∃x. (Smiling(x) → WearingHat(x))

"Some smiling person wears a hat." False
"Some smiling person wears a hat."  \[\text{False}\]

\[\exists x. (\text{Smiling}(x) \land \text{WearingHat}(x))\]  \[\text{False}\]

\[\exists x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x))\]
"Some smiling person wears a hat."  False

\[ \exists x. (Smiling(x) \land WearingHat(x)) \]  False

\[ \exists x. (Smiling(x) \rightarrow WearingHat(x)) \]  True
∃x. (Smiling(x) ∧ WearingHat(x))  \[\text{False}\]

∃x. (Smiling(x) → WearingHat(x))  \[\text{True}\]
"Some smiling person wears a hat."  False

\[ \exists x. (Smiling(x) \land WearingHat(x)) \]
False

\[ \exists x. (Smiling(x) \rightarrow WearingHat(x)) \]
True
“Some $P$ is a $Q$” translates as

$\exists x. (P(x) \land Q(x))$
Useful Intuition:

Existentially-quantified statements are false unless there's a positive example.

\[ \exists x. \ (P(x) \land Q(x)) \]

If \( x \) is an example, it must have property \( P \) on top of property \( Q \).
Using the predicates

- $\text{Smiling}(x)$, which states that $x$ is smiling, and
- $\text{WearingHat}(x)$, which states that $x$ is wearing a hat,

write a sentence in first-order logic that says

$\text{every smiling person wears a hat.}$
“Every smiling person wears a hat.”

\[
\begin{align*}
\forall x. (Smiling(x) \land WearingHat(x)) \\
\forall x. (Smiling(x) \to WearingHat(x))
\end{align*}
\]
"Every smiling person wears a hat."

\[ \forall x. \ (Smiling(x) \land WearingHat(x)) \]

\[ \forall x. \ (Smiling(x) \rightarrow WearingHat(x)) \]
∀x. (Smiling(x) ∧ WearingHat(x))
∀x. (Smiling(x) ∧ WearingHat(x))
∀x. (Smiling(x) → WearingHat(x))
“Every smiling person wears a hat.”  True
\[
\forall x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \quad \text{True}
\]
\[
\forall x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \quad \text{True}
\]
"Every smiling person wears a hat."  True

\[ \forall x. (\text{Smiling}(x) \land \text{WearingHat}(x)) \]  True

\[ \forall x. (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]  True
∀x. (Smiling(x) ∧ WearingHat(x))
∀x. (Smiling(x) → WearingHat(x))
“Every smiling person wears a hat.”

\[ \forall x. \ (Smiling(x) \land WearingHat(x)) \]

\[ \forall x. \ (Smiling(x) \rightarrow WearingHat(x)) \]
"Every smiling person wears a hat." True

\[
\forall x. (Smiling(x) \land WearingHat(x))
\]

\[
\forall x. (Smiling(x) \rightarrow WearingHat(x))
\]
\[ \forall x. \, (\text{Smiling}(x) \land \text{WearingHat}(x)) \] and
\[ \forall x. \, (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \]

"Every smiling person wears a hat."  
\[ \text{True} \]
\[ \text{False} \]
∀x. (Smiling(x) ∧ WearingHat(x))

∀x. (Smiling(x) → WearingHat(x))
"Every smiling person wears a hat."

\[
\forall x. \ (\text{Smiling}(x) \land \text{WearingHat}(x)) \quad \text{False}
\]

\[
\forall x. \ (\text{Smiling}(x) \rightarrow \text{WearingHat}(x)) \quad \text{True}
\]
∀x. (Smiling(x) ∧ WearingHat(x))

∀x. (Smiling(x) → WearingHat(x))
“All $P$'s are $Q$'s” translates as

$\forall x. (P(x) \rightarrow Q(x))$
Useful Intuition:

Universally-quantified statements are true unless there's a counterexample.

\[ \forall x. (P(x) \rightarrow Q(x)) \]

If \( x \) is a counterexample, it must have property \( P \) but not have property \( Q \).
Good Pairings

- The ∀ quantifier *usually* is paired with →.
  \[ ∀x. \ (P(x) → Q(x)) \]

- The ∃ quantifier *usually* is paired with ∧.
  \[ ∃x. \ (P(x) \land Q(x)) \]

- In the case of ∀, the → connective prevents the statement from being *false* when speaking about some object you don't care about.

- In the case of ∃, the ∧ connective prevents the statement from being *true* when speaking about some object you don't care about.
Next Time

• *First-Order Translations*
  • How do we translate from English into first-order logic?

• *Quantifier Orderings*
  • How do you select the order of quantifiers in first-order logic formulas?

• *Negating Formulas*
  • How do you mechanically determine the negation of a first-order formula?

• *Expressing Uniqueness*
  • How do we say there’s just one object of a certain type?