Canonical Forms
– sum-of-products and product-of-sums
– both are useful

Corollaries of construction:
– any function can be formed with just $\neg$, $\lor$, $\land$
– actually, just $\neg$, $\lor$ (De Morgan’s laws)
– actually, just $A$ (HW1 Q4)

NAND and NOR also have this property
Predicate Logic

• Propositional Logic
  – Allows us to analyze complex propositions in terms of their simpler constituent parts (a.k.a. atomic propositions) joined by connectives

• Predicate Logic
  – Lets us analyze them at a deeper level by expressing how those propositions depend on the objects they are talking about

“All positive integers $x, y,$ and $z$ satisfy $x^3 + y^3 \neq z^3$.”
Predicate Logic

Adds two key notions to propositional logic

- Predicates

- Quantifiers
Predicates

Predicate

– A function that returns a truth value, e.g.,

Cat(x) := “x is a cat”
Prime(x) := “x is prime”
HasTaken(x, y) ::= “student x has taken course y”
LessThan(x, y) ::= “x < y”
Sum(x, y, z) ::= “x + y = z”
GreaterThan5(x) ::= “x > 5”
HasNChars(s, n) ::= “string s has length n”

Predicates can have varying numbers of arguments and input types.
Domain of Discourse

For ease of use, we define one “type”/“domain” that we work over. This non-empty set of objects is called the “domain of discourse”.

For each of the following, what might the domain be?

(1) “$x$ is a cat”, “$x$ barks”, “$x$ ruined my couch”

“mammals” or “sentient beings” or “cats and dogs” or …

(2) “$x$ is prime”, “$x = 0$”, “$x < 0$”, “$x$ is a power of two”

“numbers” or “integers” or “integers greater than 5” or …

(3) “student $x$ has taken course $y$” “$x$ is a pre-req for $z$”

“students and courses” or “university entities” or …
We use **quantifiers** to talk about collections of objects.

\[ \forall x \ P(x) \]

*P(x)* is true for every *x* in the domain
read as “for all *x*, *P* of *x*”

\[ \exists x \ P(x) \]

There is an *x* in the domain for which *P(x)* is true
read as “there exists *x*, *P* of *x*”


**Domain of Discourse**  
Positive Integers

| Predicate Definitions |  
|-----------------------|---|
| Even(x) ::= “x is even” | Greater(x, y) ::= “x > y” |
| Odd(x) ::= “x is odd” | Equal(x, y) ::= “x = y” |
| Prime(x) ::= “x is prime” | Sum(x, y, z) ::= “x + y = z” |

Determine the truth values of each of these statements:

- $\exists x \text{ Even}(x)$  
  - T  
  - e.g. 2, 4, 6, ...

- $\forall x \text{ Odd}(x)$  
  - F  
  - e.g. 2, 4, 6, ...

- $\forall x (\text{Even}(x) \lor \text{Odd}(x))$  
  - T  
  - every integer is either even or odd

- $\exists x (\text{Even}(x) \land \text{Odd}(x))$  
  - F  
  - no integer is both even and odd

- $\forall x \text{ Greater}(x+1, x)$  
  - T  
  - adding 1 makes a bigger number

- $\exists x (\text{Even}(x) \land \text{Prime}(x))$  
  - T  
  - Even(2) is true and Prime(2) is true
Statements with Quantifiers (Literal Translations)

<table>
<thead>
<tr>
<th>Domain of Discourse</th>
<th>Predicate Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Integers</td>
<td>Even(x) ::= “x is even” Greater(x, y) ::= “x &gt; y”</td>
</tr>
<tr>
<td></td>
<td>Odd(x) ::= “x is odd”  Equal(x, y) ::= “x = y”</td>
</tr>
<tr>
<td></td>
<td>Prime(x) ::= “x is prime” Sum(x, y, z) ::= “x + y = z”</td>
</tr>
</tbody>
</table>

Translate the following statements to English

$$\forall x \, \exists y \, \text{Greater}(y, x)$$
For every positive integer $$x$$, there is a positive integer $$y$$, such that $$y > x$$.

$$\exists y \, \forall x \, \text{Greater}(y, x)$$
There is a positive integer $$y$$ such that, for every pos. int. $$x$$, we have $$y > x$$.

$$\forall x \, \exists y \, (\text{Greater}(y, x) \land \text{Prime}(y))$$
For every positive integer $$x$$, there is a pos. int. $$y$$ such that $$y > x$$ and $$y$$ is prime.

$$\forall x \, (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \lor \text{Odd}(x)))$$
For each positive integer $$x$$, if $$x$$ is prime, then $$x = 2$$ or $$x$$ is odd.

$$\exists x \, \exists y \, (\text{Sum}(x, 2, y) \land \text{Prime}(x) \land \text{Prime}(y))$$
There exist positive integers $$x$$ and $$y$$ such that $$x + 2 = y$$ and $$x$$ and $$y$$ are prime.
Translate the following statements to English

\( \forall x \ \exists y \ \text{Greater}(y, x) \)

For every positive integer \( x \), there is a positive integer \( y \), such that \( y > x \).

\( \exists y \ \forall x \ \text{Greater}(y, x) \)

There is a positive integer \( y \) such that, for every pos. int. \( x \), we have \( y > x \).

\( \forall x \ \exists y \ \left( \text{Greater}(y, x) \land \text{Prime}(y) \right) \)

For every positive integer \( x \), there is a pos. int. \( y \) such that \( y > x \) and \( y \) is prime.
Translate the following statements to English:

$$\forall x \ \exists y \ \text{Greater}(y, x)$$

For every positive integer, there is some larger positive integer.

$$\exists y \ \forall x \ \text{Greater}(y, x)$$

There is a positive integer that is larger than every other positive integer.

$$\forall x \ \exists y \ (\text{Greater}(y, x) \land \text{Prime}(y))$$

For every positive integer, there is a prime that is larger.

Sound more natural without introducing variable names.
English to Predicate Logic

**Domain of Discourse**

Mammals

<table>
<thead>
<tr>
<th>Predicate Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat(x) ::= “x is a cat”</td>
</tr>
<tr>
<td>Red(x) ::= “x is red”</td>
</tr>
<tr>
<td>LikesTofu(x) ::= “x likes tofu”</td>
</tr>
</tbody>
</table>

“All red cats like tofu”

\[ \forall x \ ((\text{Red}(x) \wedge \text{Cat}(x)) \rightarrow \text{LikesTofu}(x)) \]

“Some red cats don’t like tofu”

\[ \exists y \ ((\text{Red}(y) \wedge \text{Cat}(y)) \wedge \neg \text{LikesTofu}(y)) \]
English to Predicate Logic

Domain of Discourse
Mammals

Predicate Definitions
Cat(x) ::= “x is a cat”
Red(x) ::= “x is red”
LikesTofu(x) ::= “x likes tofu”

“**All Red cats like tofu**”

When putting two predicates together like this, we use an “and”.

“**Some red cats don’t like tofu**”

When restricting to a smaller domain in a “for all” we use implication.

“When restricting to a smaller domain in an “exists” we use and.”

“Some” means “there exists”.
Statements with Quantifiers (Literal Translations)

<table>
<thead>
<tr>
<th>Domain of Discourse</th>
<th>Predicate Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Integers</td>
<td>Even(x) ::= “x is even” Greater(x, y) ::= “x &gt; y”</td>
</tr>
<tr>
<td></td>
<td>Odd(x) ::= “x is odd”  Equal(x, y) ::= “x = y”</td>
</tr>
<tr>
<td></td>
<td>Prime(x) ::= “x is prime” Sum(x, y, z) ::= “x + y = z”</td>
</tr>
</tbody>
</table>

Translate the following statements to English

\[ \forall x \ (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \lor \text{Odd}(x))) \]
For each positive integer \( x \), if \( x \) is prime, then \( x = 2 \) or \( x \) is odd.

\[ \exists x \ \exists y \ (\text{Sum}(x, 2, y) \land \text{Prime}(x) \land \text{Prime}(y)) \]
There exist positive integers \( x \) and \( y \) such that \( x + 2 = y \) and \( x \) and \( y \) are prime.
Statements with Quantifiers (Literal Translations)

<table>
<thead>
<tr>
<th>Domain of Discourse</th>
<th>Positive Integers</th>
</tr>
</thead>
</table>

Predicate Definitions

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even(x)</td>
<td>“x is even”</td>
</tr>
<tr>
<td>Odd(x)</td>
<td>“x is odd”</td>
</tr>
<tr>
<td>Prime(x)</td>
<td>“x is prime”</td>
</tr>
<tr>
<td>Greater(x, y)</td>
<td>“x &gt; y”</td>
</tr>
<tr>
<td>Equal(x, y)</td>
<td>“x = y”</td>
</tr>
<tr>
<td>Sum(x, y, z)</td>
<td>“x + y = z”</td>
</tr>
</tbody>
</table>

Translate the following statements to English

\( \forall x \ (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \lor \text{Odd}(x))) \)

Every prime number is either 2 or odd.

\( \exists x \ \exists y \ (\text{Sum}(x, 2, y) \land \text{Prime}(x) \land \text{Prime}(y)) \)

There exist prime numbers that differ by two.

Spot the domain restriction patterns
English to Predicate Logic

Domain of Discourse
Mammals

Predicate Definitions
Cat(x) ::= “x is a cat”
Red(x) ::= “x is red”
LikesTofu(x) ::= “x likes tofu”

“All Red cats like tofu”

“Red cats like tofu”

When there’s no leading quantification, it means “for all”.

“Some red cats don’t like tofu”

“A red cat doesn’t like tofu”

“A” means “there exists”.

“Mammals”
Translations often (not always) sound more natural if we

1. Notice “domain restriction” patterns

\[ \forall x \ ( \text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \lor \text{Odd}(x))) \]

Every prime number is either 2 or odd.

2. Avoid introducing unnecessary variable names

\[ \forall x \ \exists y \ \text{Greater}(y, x) \]

For every positive integer, there is some larger positive integer.

3. Can sometimes drop “all” or “there is”

\[ \neg \exists x \ (\text{Even}(x) \land \text{Prime}(x) \land \text{Greater}(x, 2)) \]

No even prime is greater than 2.
(*) $\forall x \text{PurpleFruit}(x)$ ("All fruits are purple")

What is the negation of (*)?

(a) “There exists a purple fruit”
(b) “There exists a non-purple fruit”
(c) “All fruits are not purple”

Try your intuition! Which one seems right?
Negations of Quantifiers

Predicate Definitions

PurpleFruit(x) ::= “x is a purple fruit”

(*) \( \forall x \text{ PurpleFruit}(x) \) (“All fruits are purple”)

What is the negation of (*)?
(a) “there exists a purple fruit”
(b) “there exists a non-purple fruit”
(c) “all fruits are not purple”

Domain of Discourse

\{plum, apple\}

(*) \text{PurpleFruit}(plum) \land \text{PurpleFruit}(apple)

(a) \text{PurpleFruit}(plum) \lor \text{PurpleFruit}(apple)
(b) \neg \text{PurpleFruit}(plum) \lor \neg \text{PurpleFruit}(apple)
(c) \neg \text{PurpleFruit}(plum) \land \neg \text{PurpleFruit}(apple)
De Morgan’s Laws for Quantifiers

\[ \neg \forall x \ P(x) \equiv \exists x \ \neg P(x) \]
\[ \neg \exists x \ P(x) \equiv \forall x \ \neg P(x) \]
De Morgan’s Laws for Quantifiers

\[ \neg \forall x \ P(x) \equiv \exists x \ \neg P(x) \]
\[ \neg \exists x \ P(x) \equiv \forall x \ \neg P(x) \]

“There is no integer larger than every other integer”

\[ \neg \exists x \ \forall y \ (x \geq y) \]
\[ \equiv \ \forall x \ \neg \ \forall y \ (x \geq y) \]
\[ \equiv \ \forall x \ \exists y \ (y > x) \]

“For every integer, there is a larger integer”
De Morgan’s Laws for Quantifiers

\[ \neg \forall x \ P(x) \equiv \exists x \ \neg P(x) \]
\[ \neg \exists x \ P(x) \equiv \forall x \ \neg P(x) \]

These are **equivalent** but not **equal**

They have different English translations, e.g.:

- There is no unicorn \( \neg \exists x \ Unicorn(x) \)
- Every animal is not a unicorn \( \forall x \ \neg Unicorn(x) \)
De Morgan’s Laws for Quantifiers

\[
\neg \forall x \ P(x) \equiv \exists x \ \neg P(x) \\
\neg \exists x \ P(x) \equiv \forall x \ \neg P(x)
\]

“No even prime is greater than 2”

\[
\neg \exists x \ (\text{Even}(x) \land \text{Prime}(x) \land \text{Greater}(x, 2)) \\
\equiv \ \forall x \ \neg (\text{Even}(x) \land \text{Prime}(x) \land \text{Greater}(x, 2)) \\
\equiv \ \forall x \ \neg (\neg (\text{Even}(x) \land \text{Prime}(x)) \lor \neg \text{Greater}(x, 2)) \\
\equiv \ \forall x \ ((\text{Even}(x) \land \text{Prime}(x)) \rightarrow \neg \text{Greater}(x, 2)) \\
\equiv \ \forall x \ ((\text{Even}(x) \land \text{Prime}(x)) \rightarrow \text{LessEq}(x, 2))
\]

“Every even prime is less than or equal to 2.”
We just saw that

$$\neg \exists x \ (P(x) \land R(x)) \equiv \forall x \ (P(x) \rightarrow \neg R(x))$$

Can similarly show that

$$\neg \forall x \ (P(x) \rightarrow R(x)) \equiv \exists x \ (P(x) \land \neg R(x))$$

De Morgan’s Laws respect domain restrictions!
(It leaves them in place and only negates the other parts.)
Scope of Quantifiers

\[ \exists x \ (P(x) \land Q(x)) \quad \text{vs.} \quad \exists x \ P(x) \land \exists x \ Q(x) \]
Scope of Quantifiers

$\exists x \ (P(x) \wedge Q(x))$ \hspace{1cm} vs. \hspace{1cm} $\exists x \ P(x) \wedge \exists x \ Q(x)$

This one asserts $P$ and $Q$ of the same $x$.

This one asserts $P$ and $Q$ of potentially different $x$’s.

Variables with the same name do not necessarily refer to the same object.
Nested Quantifiers

• Bound variable names don’t matter
  \[ \forall x \exists y \ P(x, y) \equiv \forall a \exists b \ P(a, b) \]

• Positions of quantifiers can sometimes change
  \[ \forall x (Q(x) \land \exists y \ P(x, y)) \equiv \forall x \exists y (Q(x) \land P(x, y)) \]

• But: order is important...
Quantifier Order Can Matter

Domain of Discourse
{1, 2, 3, 4}

Predicate Definitions
GreaterEq(x, y) ::= “x ≥ y”

“There is a number greater than or equal to all numbers.”

∃x ∀y GreaterEq(x, y))
Quantifier Order Can Matter

Domain of Discourse
{1, 2, 3, 4}

Predicate Definitions
GreaterEq(x, y) ::= “x ≥ y”

There is a number greater than or equal to all numbers.
\[ \exists x \forall y \text{ GreaterEq}(x, y) \]

Every number has a number greater than or equal to it.
\[ \forall y \exists x \text{ GreaterEq}(x, y) \]
Quantifier Order Can Matter

<table>
<thead>
<tr>
<th>Domain of Discourse</th>
<th>1, 2, 3, 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicate Definitions</td>
<td>GreaterEq(x, y) ::= “x ≥ y”</td>
</tr>
</tbody>
</table>

“There is a number greater than or equal to all numbers.”

\[ \exists x \ \forall y \ \text{GreaterEq}(x, y) \]

“Every number has a number greater than or equal to it.”

\[ \forall y \ \exists x \ \text{GreaterEq}(x, y) \]

The purple statement requires an entire row to be true.
The red statement requires one entry in each column to be true.

**Important**: both include the case \( x = y \)

Different names does not imply different objects!
# Quantification with Two Variables

<table>
<thead>
<tr>
<th>expression</th>
<th>when true</th>
<th>when false</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \forall x \ \forall y \ P(x, y) )</td>
<td>Every pair is true.</td>
<td>At least one pair is false.</td>
</tr>
<tr>
<td>( \exists x \ \exists y \ P(x, y) )</td>
<td>At least one pair is true.</td>
<td>All pairs are false.</td>
</tr>
<tr>
<td>( \forall x \ \exists y \ P(x, y) )</td>
<td>We can find a specific ( y ) for each ( x ). ((x_1, y_1), (x_2, y_2), (x_3, y_3))</td>
<td>Some ( x ) doesn’t have a corresponding ( y ).</td>
</tr>
<tr>
<td>( \exists y \ \forall x \ P(x, y) )</td>
<td>We can find ONE ( y ) that works no matter what ( x ) is. ((x_1, y), (x_2, y), (x_3, y))</td>
<td>For any candidate ( y ), there is an ( x ) that it doesn’t work for.</td>
</tr>
</tbody>
</table>