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, by itself, either true or false.

Some Sample Propositions

- I am not throwing away my shot.
- I’m just like my country.
- I’m young, scrappy, and hungry.
- I’m not throwing away my shot.
- I’m ‘a get a scholarship to King’s College.
- I prob’ly shouldn’t brag, but dag, I amaze and astonish.
- The problem is I got a lot of brains but no polish.

Things That Aren't Propositions

Commands cannot be true or false.

Things That Aren't Propositions

Questions cannot be true or false.

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 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:

$$\boldsymbol{p \land q}$$

- 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:

$$\boldsymbol{p \lor q}$$

- 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:

$$\neg \qquad \wedge \qquad \vee$$

Summary of Important Points

- The ∨ 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.)

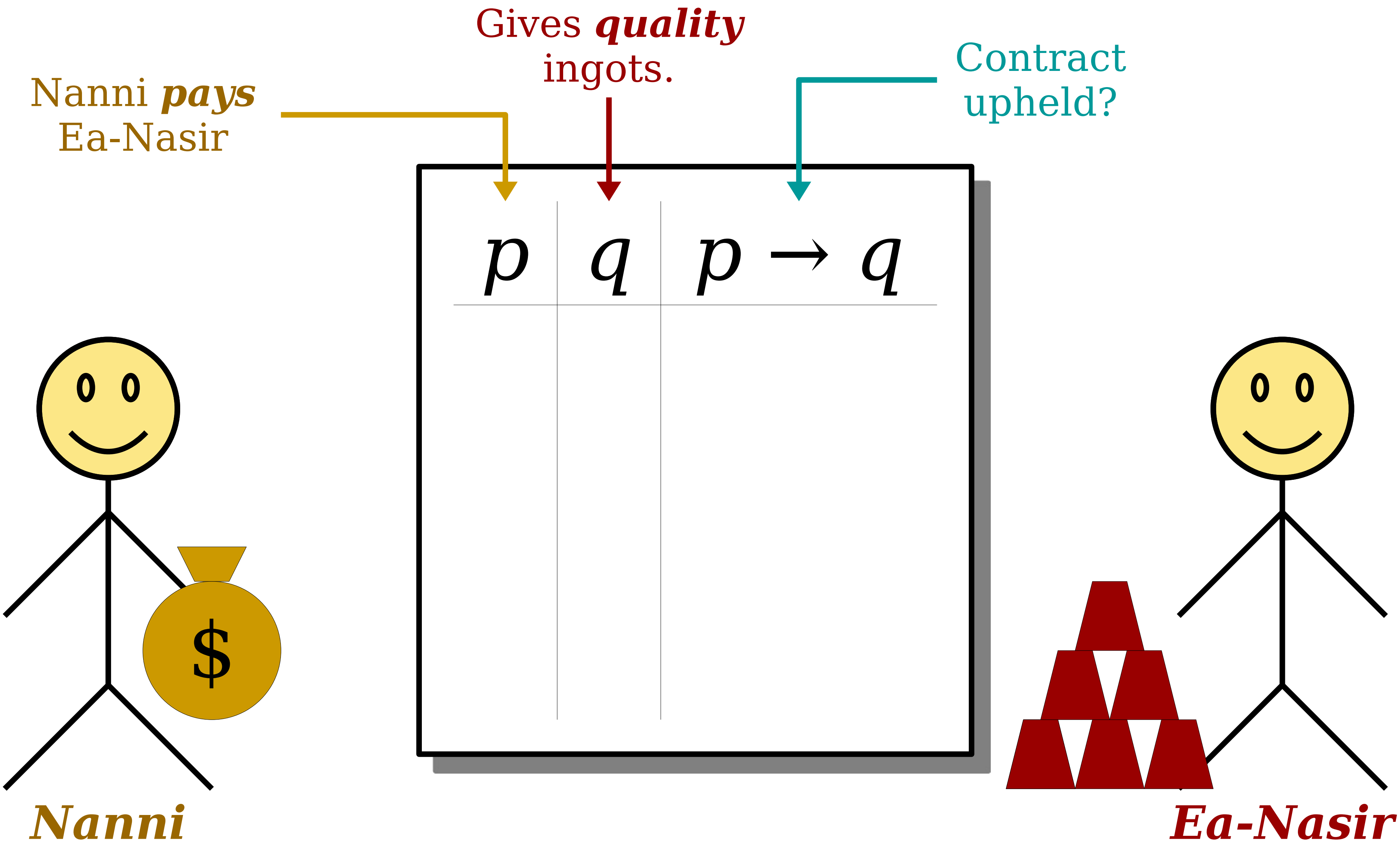
Mathematical Implication

Implication

- We can represent implications using this connective:

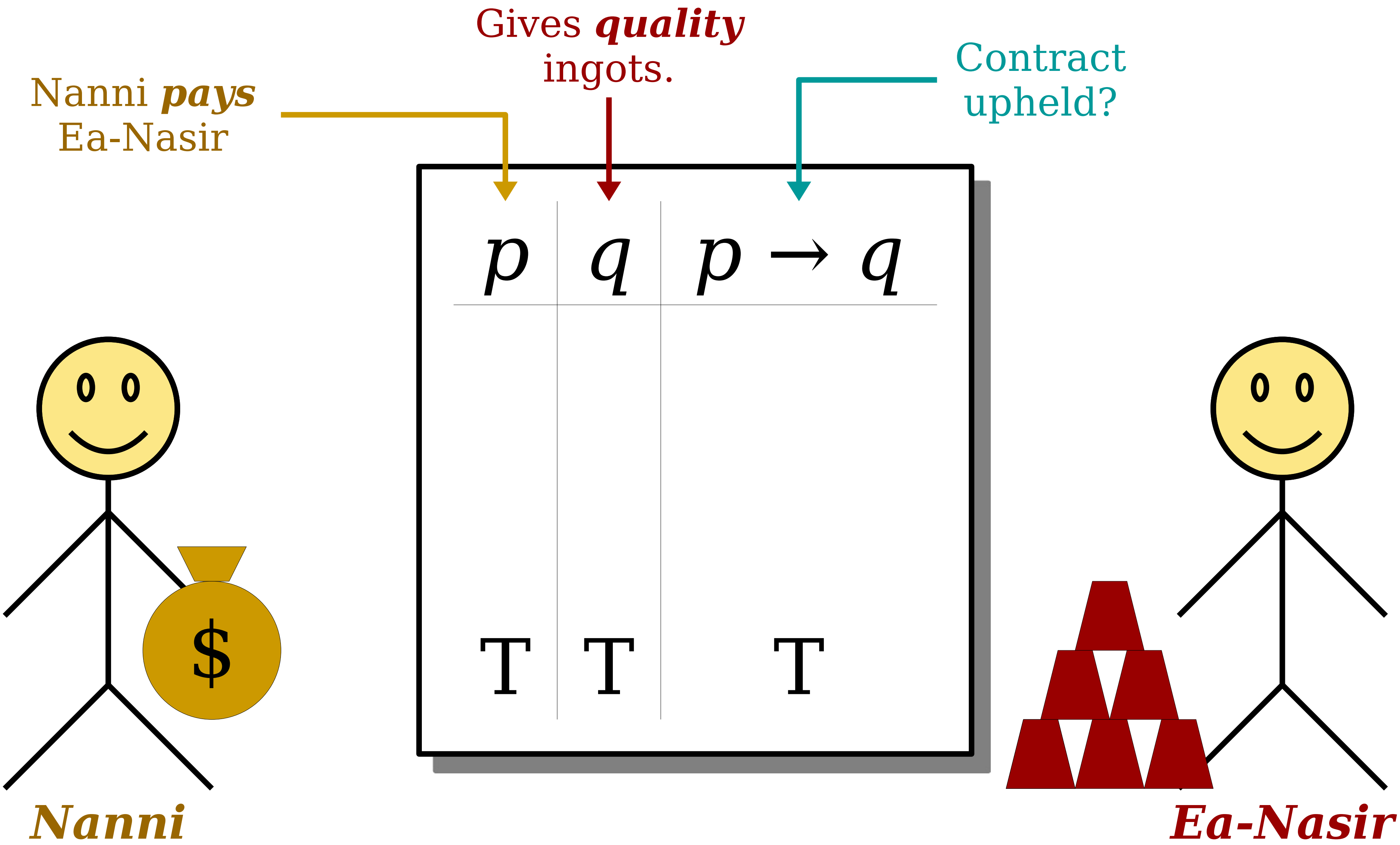
$$\boldsymbol{p \to q}$$

- You'd read this out loud as "p implies q."
  - The fancy name for this is the ***material conditional***.
- ***Question:*** What should the truth table for $p \to q$ look like?
- Pull out a sheet of paper, make a guess, and talk things over with your neighbors!



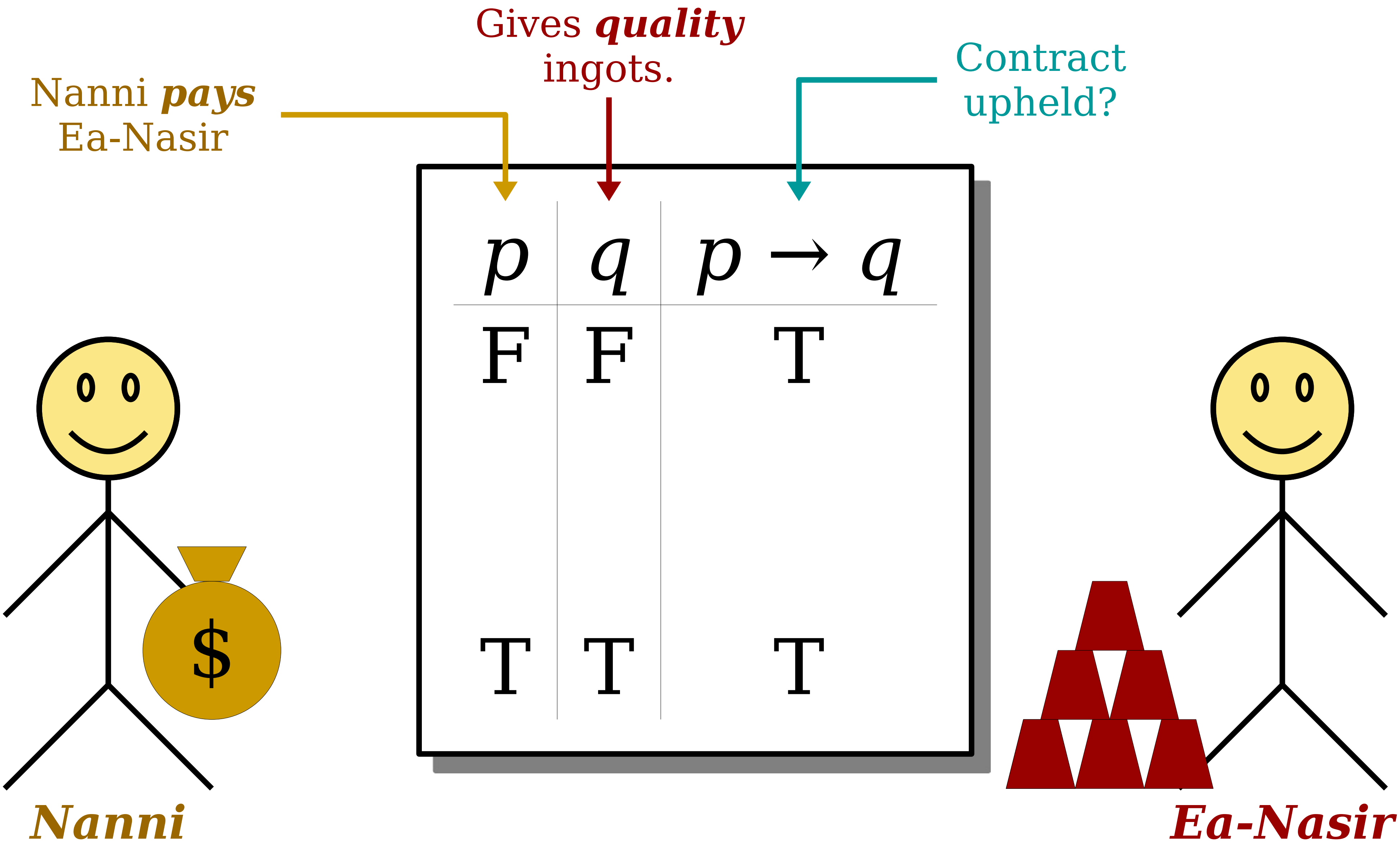
Ancient Contract:

If Nanni pays money to Ea-Nasir, then
Ea-Nasir will give Nanni quality copper ingots.



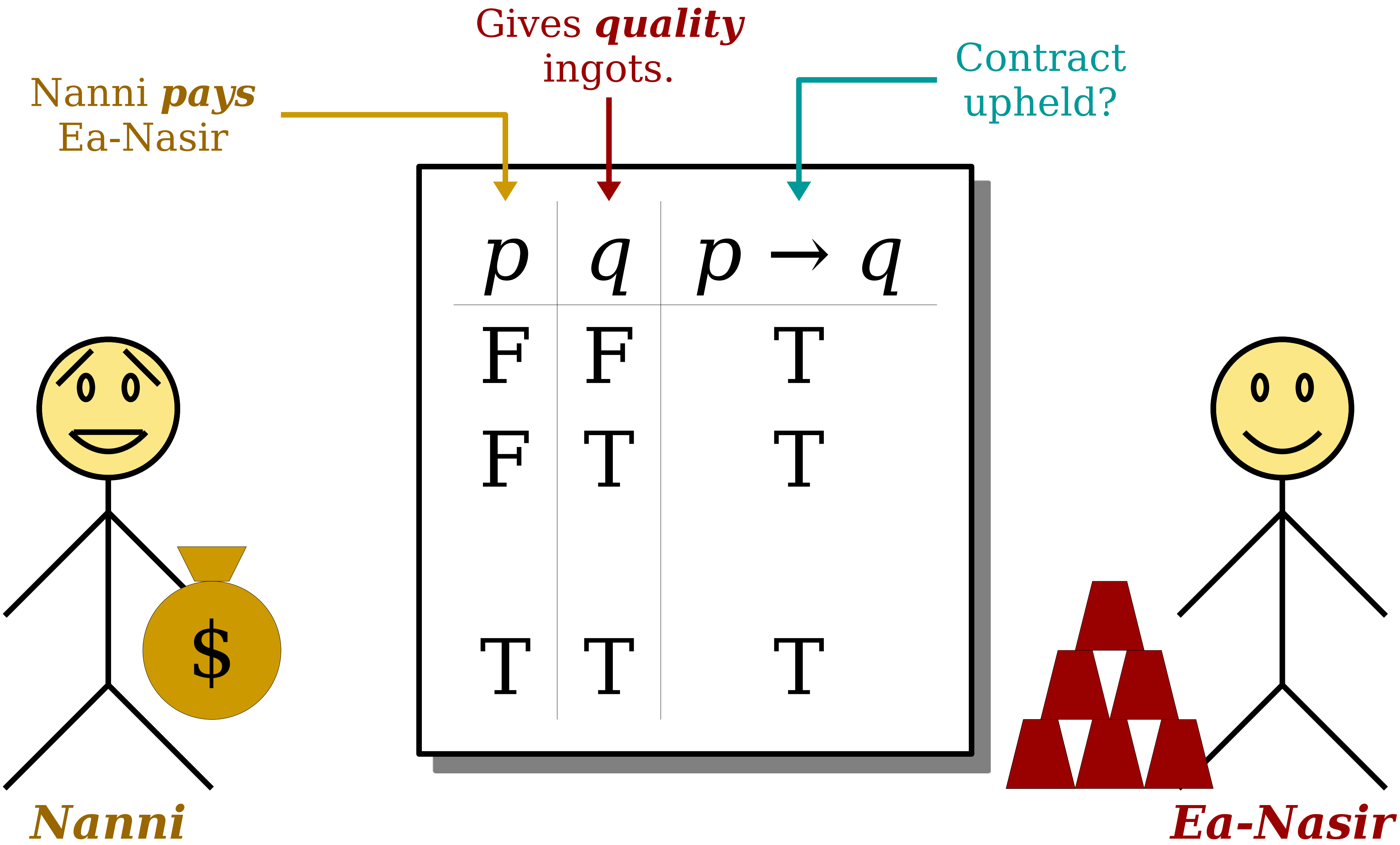
Ancient Contract:

If Nanni pays money to Ea-Nasir, then
Ea-Nasir will give Nanni quality copper ingots.



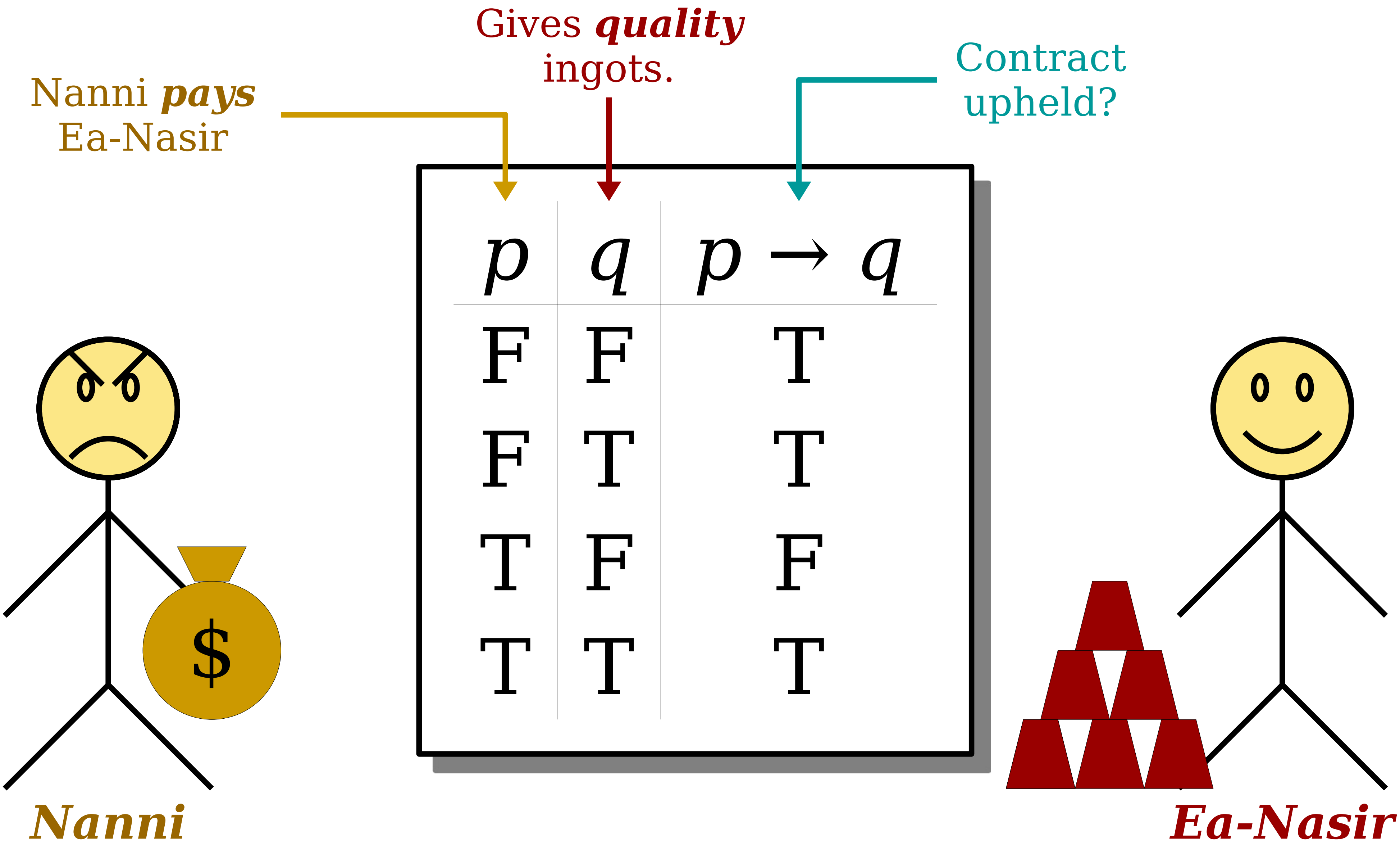
Ancient Contract:

If Nanni pays money to Ea-Nasir, then
Ea-Nasir will give Nanni quality copper ingots.



Ancient Contract:

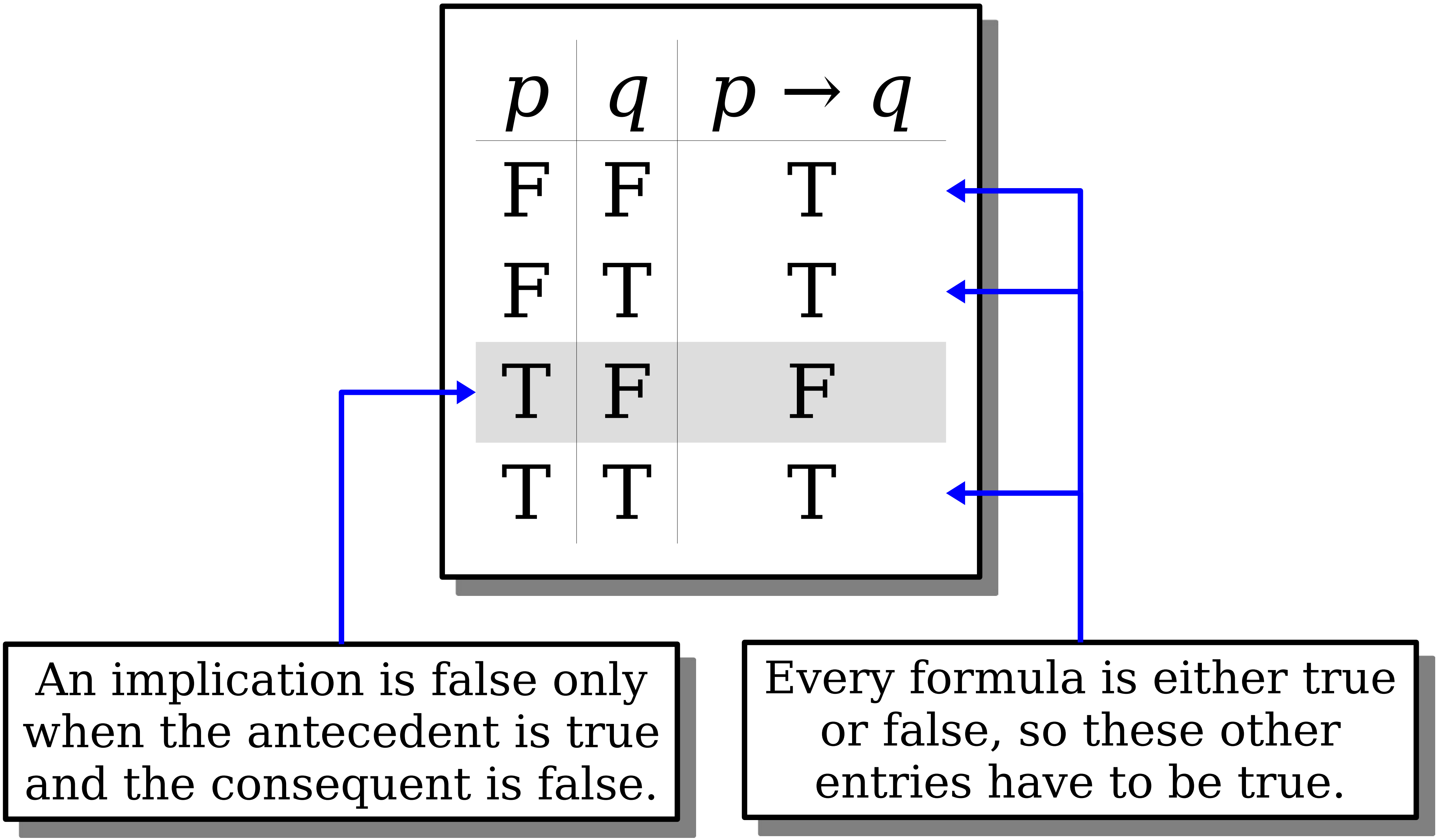
If Nanni pays money to Ea-Nasir, then
Ea-Nasir will give Nanni quality copper ingots.



Ancient Contract:

If Nanni pays money to Ea-Nasir, then
Ea-Nasir will give Nanni quality copper ingots.

p	q	$p \to q$
F	F	T
F	T	T
T	F	F
T	T	T


p | q | p → q
F F T
F T T
T F F
T T T
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.

p	q	$p \to q$
F	F	T
F	T	T
T	F	F
T	T	T

Important observation:
The statement $p \to q$ is true whenever $p \wedge \neg q$ is false.

p	q	$p \to q$
F	F	T
F	T	T
T	F	F
T	T	T

An implication with a false antecedent is called ***vacuously true***.

An implication with a true consequent is called ***trivially true***.

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

Please commit this table to memory. We’re going to need it, extensively, over the next couple of weeks.

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 $p \to q$ and $q \to p$.
- 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:

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

Biconditionals

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

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

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:

$$(\neg p \to \bot) \to p$$

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:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

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

Operator Precedence

- How do we parse this statement?

$$\neg x \rightarrow y \vee z \rightarrow x \vee y \wedge z$$

- Operator precedence for propositional logic:

$$\neg$$

$$\wedge$$

$$\vee$$

$$\rightarrow$$

$$\leftrightarrow$$

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

Operator Precedence

- How do we parse this statement?

$$(\neg x) \rightarrow y \vee z \rightarrow x \vee y \wedge z$$

- Operator precedence for propositional logic:

$$\neg$$

$$\wedge$$

$$\vee$$

$$\rightarrow$$

$$\leftrightarrow$$

- 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:

$$\neg$$
$$\land$$
$$\lor$$
$$\to$$
$$\leftrightarrow$$

- 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:

$$\neg$$

$$\boldsymbol{\land}$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

- 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:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

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

Operator Precedence

- How do we parse this statement?

$$(\neg x) \rightarrow (y \vee z) \rightarrow (x \vee (y \wedge z))$$

- Operator precedence for propositional logic:

$$\neg$$
$$\wedge$$
$$\vee$$
$$\rightarrow$$
$$\leftrightarrow$$

- 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:

$$\neg$$
$$\land$$
$$\lor$$
$$\to$$
$$\leftrightarrow$$

- 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:

$$\neg$$
$$\land$$
$$\lor$$
$$\to$$
$$\leftrightarrow$$

- 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:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

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

Operator Precedence

- The main points to remember:
  - ¬ binds to whatever immediately follows it.
  - ∧ and ∨ bind more tightly than →.
- We will commonly write expressions like $p \land q \to r$ without adding parentheses.
- For more complex expressions, we'll try to add parentheses.
- Confused? ***Please ask!***

The Big Table

Connective	Read Aloud As	C++ Version	Fancy Name
$\neg$	"not"	`!`	Negation
$\wedge$	"and"	`&&`	Conjunction
$\vee$	"or"	`||`	Disjunction
$\to$	"implies"	*see PS2!*	Implication
$\leftrightarrow$	"if and only if"	*see PS2!*	Biconditional
$\top$	"true"	`true`	Truth
$\bot$	"false"	`false`	Falsity

Time-Out for Announcements!

High-Res Course Feedback

- This quarter, we're working with the High-Res Course Feedback (HRCF) team in the CS department to get your input about how we're doing throughout the quarter.
- You'll get two emails over the quarter from ***hrcf@cs.stanford.edu*** asking you to leave feedback.
- If you feel comfortable doing so, please let us know how we're doing! We'd love to know how to improve as we transition back to in-person instruction.

A Note on Bicycling

- Aren’t bikes wonderful? They’re a great way to get around campus.
- However:

☞ ***Please wear a helmet!*** ☜

- Face masks prevent needless suffering due to COVID. Helmets prevent needless suffering due to bike accidents.

Your Questions

“Tips for the computer forum career fair next week?”

“How should I brush up / refresh my technical skills for interviews in the next few weeks? I took CS 106B with you last winter however it's been a while and I don't remember everything super clear.”

If you haven't heard of the Computer Forum, it's a partnership between CS, EE, and industry. They put on some large career fairs each year, one of which is (unfortunately) early in Fall quarter.

If you haven't read "Cracking the Coding Interview," I'd recommend doing so. It's got a mix of great practice problems and general advice. Work some of those problems; they're good practice!

Get on the recruiting list (***recruiting@lists.stanford.edu***) and look for tech talks, interview prep, resume critiques, and the like.

And ask me to elaborate more about this as we get closer to the day!

“What can I do with a CS degree if I don't want to work for a corporation?”

Quite a lot, actually! Here's a sampler of what some of my former students are up to:

- Clerking for a federal judge while working as a civil rights attorney.
- Running a co-op tech collective that builds software for underresourced communities.
- Working for the US Government building out software for HealthCare.gov and other big projects.
- Teaching computer science at the K-8 level.

Also look at careers in government, public policy, and the like. There is a desperate need for CS talents there!

“Favorite book you read in the last 6 months? In the last 6 years?”

In the last six months – it’s probably the John McPhee book “Oranges” about, well, oranges: their history, their cultivation, where they’re grown, why they’re grown there, etc. It’s a wide-ranging, wonderfully escapist book.

In the last six years – that is a tough one! Here’s a sampler of ones that really stood out: “Command and Control” by Eric Schlosser (on nuclear weapons safety and institutional failure), “Cadillac Desert” by Marc Reisner (about water policy in the western US), “Radetzky March” by Joseph Roth (about the twilight of the Austro-Hungarian empire), “Catch-22” by Joseph Heller (I reread this one many years after reading it in high school, and it’s brilliant), and “Exhalations” by Ted Chiang (amazingly clever speculative fiction short stories).

Back to CS103!

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: $p \to q$
  - 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: 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."

Some Sample Propositions

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."

$$\neg a \to \neg b$$

“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: 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."

$$a \land \neg c \to \neg b$$

“***p*, but *q***”

translates to

$\boldsymbol{p \wedge q}$

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 $\boldsymbol{p \land q}$ 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

$$\neg(p \wedge q)$$

  is equivalent to

$$\neg p \vee \neg q$$

- We also saw that

$$\neg(p \vee q)$$

  is equivalent to

$$\neg p \wedge \neg q$$

- These two equivalences are called ***De Morgan's Laws***.

de Morgan's Laws in Code

- ***Pro tip:*** Don't write this:

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

- Write this instead:

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

- (This even short-circuits correctly!)

An Important Equivalence

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

$$\boldsymbol{p \to q} \quad \textbf{\textit{is equivalent to}} \quad \boldsymbol{\neg(p \land \neg q)}$$

- Later on, this equivalence will be incredibly useful:

$$\boldsymbol{\neg(p \to q)} \quad \textbf{\textit{is equivalent to}} \quad \boldsymbol{p \land \neg q}$$

Another Important Equivalence

- Here's a useful equivalence. Start with

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

- By de Morgan's laws:

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

$$\textit{is equivalent to} \quad \neg p \lor \neg\neg q$$

$$\textit{is equivalent to} \quad \neg p \lor q$$

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

Another Important Equivalence

- Here's a useful equivalence. Start with

$$p \to q \quad \textit{is equivalent to} \quad \neg(p \land \neg q)$$

- By de Morgan's laws:

$$p \to q \quad \textit{is equivalent}$$

$$\textit{is equivalent}$$

$$\textit{is equivalent}$$

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.

- Thus $p \to q$ is equivalent to $\neg p \lor q$

Why All This Matters

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x + y = 16 \rightarrow x \geq 8 \lor y \geq 8$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x + y = 16 \rightarrow x \geq 8 \lor y \geq 8$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8 \lor y \geq 8) \rightarrow \neg(x + y = 16)$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8 \lor y \geq 8) \to \neg(x + y = 16)$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8 \lor y \geq 8) \rightarrow \neg(x + y = 16)$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8 \lor y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8 \lor y \geq 8) \to x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8 \lor y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8) \land \neg(y \geq 8) \to x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8) \land \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$\neg(x \geq 8) \land \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \land \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x < 8 \land \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x < 8 \land \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \land y < 8 \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x < 8 \land y < 8 \rightarrow x + y \neq 16$$

Why All This Matters

- Suppose we want to prove the following statement:

“If $x + y = 16$, then $x \geq 8$ or $y \geq 8$”

$$x < 8 \land y < 8 \rightarrow x + y \neq 16$$

“If $x < 8$ and $y < 8$, then $x + y \neq 16$”

Theorem: If $x + y = 16$, then $x \geq 8$ or $y \geq 8$.

Proof: We will prove the contrapositive of this statement: if $x < 8$ and $y < 8$, then $x + y \neq 16$.

Let x and y be arbitrary numbers such that $x < 8$ and $y < 8$. We need to show that $x + y \neq 16$. Note that

$$\begin{aligned} x + y &< 8 + y \\ &< 8 + 8 \\ &= 16. \end{aligned}$$

This means that $x + y < 16$, so $x + y \neq 16$, which is what we needed to show. ■

Why This Matters

- Propositional logic is a tool for reasoning about how various statements affect one another.
- To better understand how to prove a result, it often helps to translate what you're trying to prove into propositional logic first.
- That said, propositional logic isn't expressive enough to capture all statements. For that, we need something more powerful.

Next Time

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