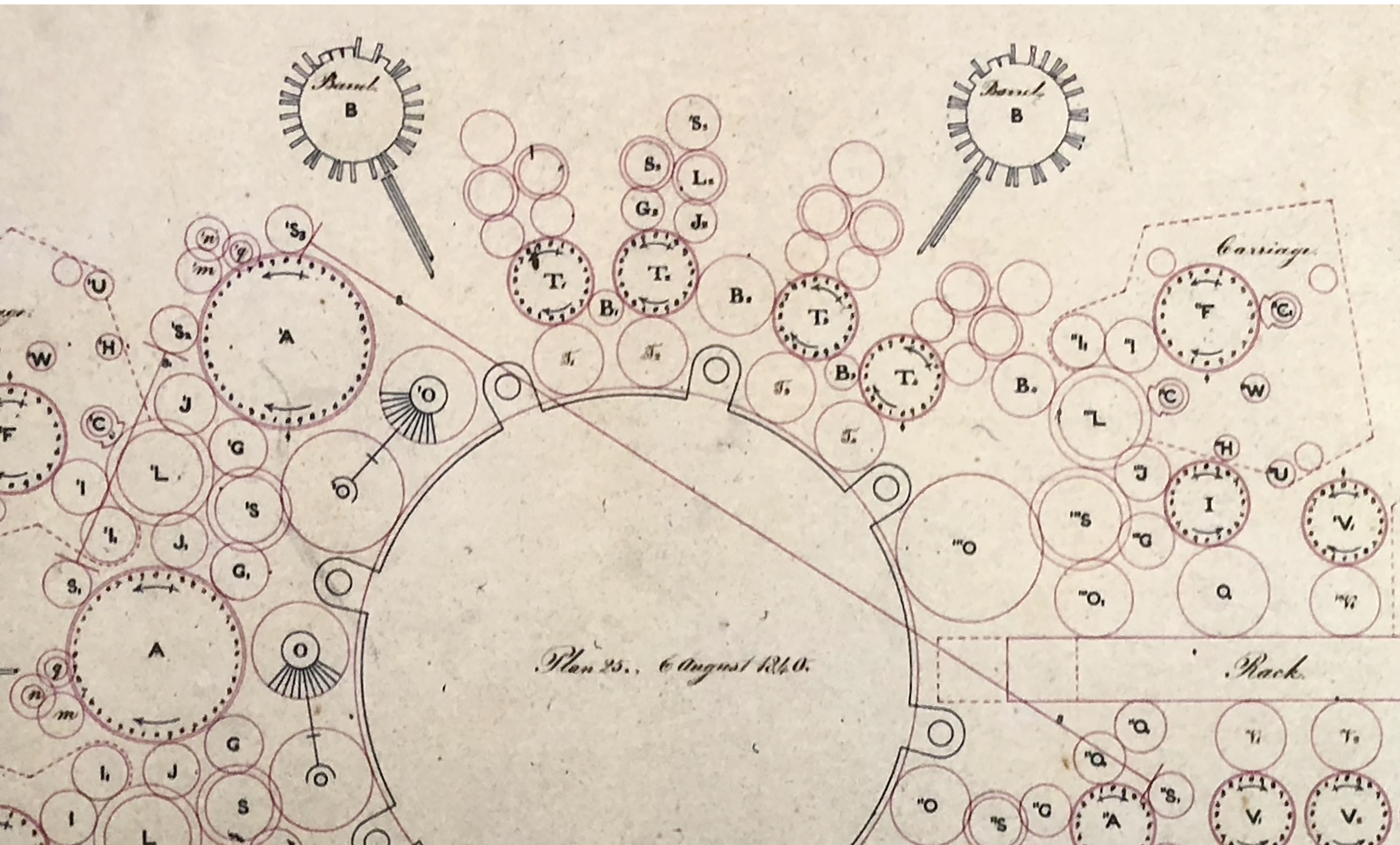
forall *x* @ syr

An Introduction to Formal Logic

By P. D. Magnus, Tim Button, and Michael Rieppel



Cover Image: plan for Babbage's Analytical Engine (1840). Available at Wikimedia Commons.

The LaTeX source for this book is available on GitHub at https://github.com/mrieppel.

The present version was released on October 30, 2019.

Preface

I first learned formal logic from Michael Byrd at UW-Madison, using Lemmon's *Beginning Logic*, and first taught logic in 2008 as a teaching assistant for Branden Fitelson at UC Berkeley, using Graeme Forbes' *Modern Logic*. When I started to develop my own logic course, I continued to use Forbes' book, which I liked for its rigorous and thorough treatment of the three central components of any introductory logic class: symbolization, semantics, and natural deduction.

However, over time I became dissatisfied with *Modern Logic* for two reasons. First, the Lemmon-style notation that it uses for natural deduction is much less accessible to beginners than the Fitch-style notation found in other texts. And second, Oxford University Press started producing fewer copies of the book, making it rather expensive — more expensive, at any rate, than I thought an introductory logic text should be. By the time I began teaching at Syracuse in 2015, I therefore started listing Forbes' book only as a recommended text, and relied heavily on the detailed lecture notes I had put together over the years.

But in the longer term, I faced a choice: either select a new textbook, or transform my own notes into a book. The open-source nature of Tim Button's *forall x: Cambridge* offered me way to do both: I could take his already excellent text (which *inter alia* included a Fitch-style deduction system) and supplement it with material of my own. And so I've come to make my own addition to the "groaning shelves" of logic textbooks, to borrow Forbes' description — though with electronic distribution, the groan has thankfully become more metaphorical. The main additions I've made to *forall x: Cambridge* are the following:

- ▷ I've changed the first chapter to more closely reflect my own introductory lecture. For example, I elucidate the modal notion of validity using possible worlds, and emphasize logic's focus on *formal* validity a bit more explicitly.
- ▷ I've added more on the semantics of truth-functional and especially first-order logic, particularly as concerns the construction of countermodels.
- ▷ I've made some changes to set of natural deduction rules in both parts.

Besides the obvious debt the present text owes to the *forall x* editions that P.D. Magnus and Tim Button have so generously made available, and to Forbes' *Modern Logic*, the present text also draws on ideas taken from Barwise and Etchemendy's *Language, Proof, and Logic*, Belnap's *The Art of Logic*, Goldfarb's *Deductive Logic*, *forall x: Calgary Remix*, and lecture notes by Branden Fitelson and Daniel Warren.

Contents

“When you come to any passage you don’t understand, *read it again*: if you *still* don’t understand it, *read it again*: if you fail, even after *three* readings, very likely your brain is getting a little tired. In that case, put the book away, and take to other occupations, and next day, when you come to it fresh, you will very likely find that it is *quite* easy.

“If possible, find some genial friend, who will read the book along with you, and will talk over the difficulties with you. *Talking* is a wonderful smoother-over of difficulties. When I come upon anything — in Logic or in any other hard subject — that entirely puzzles me, I find it a capital plan to talk it over, *aloud*, even when I am all alone.”

Lewis Carroll, *Symbolic Logic* (1897)

What is Logic? 1

1.1 Arguments

This book provides an introduction to logic. But what is logic? This is a surprisingly difficult question, still debated by philosophers. But generally speaking, logic is about distinguishing *valid* from *invalid* arguments.

In everyday language, the word 'argument' is often used to describe an activity that people engage in. On twitter, or youtube, or the evening news, people often have heated disagreements, and you've probably had arguments like this with your family or friends. Logicians tend to be a pretty sedate crowd, and they mean something very different by 'argument'. In logic, an argument is just a collection of statements. More specifically:

> An ARGUMENT is a collection of one or more *statements*, exactly one of which is the argument's *conclusion* and the rest of which are its *premises*.

Here is an example of an argument:

(1) All rabbits are mammals.
Bugs Bunny is a rabbit.
$\therefore$ Bugs Bunny is a mammal.

This argument consists of three statements. One of them is argument's conclusion, which we indicated by the three dots $\therefore$ read as "therefore." The rest are its premises. This argument has two premises, but arguments can have any number of premises (though, again, only one conclusion).

Notice that our logician's definition of an argument is very permissive. Consider the following:

There is a bassoon-playing dragon in the *Cathedra Romana*.
$\therefore$ Salvador Dali was a poker player.

We have a premise and a conclusion, and so we have an argument. Admittedly, it is a *terrible* argument. But it is still an argument.

Here's another argument, one that's not as obviously terrible:

(2) All rabbits are mammals.
Winnie the Pooh is a mammal.
$\therefore$ Winnie the Pooh is a rabbit.

In this case the premises at least involve the same concepts as the conclusion. But this argument still isn't as good as (1) from earlier: unlike the earlier example, this argument isn't valid — its conclusion doesn't follow from its premises. But what exactly does validity, or "following from," consist in? What's wrong with argument (2) as compared to (1)?

One thing that's bad about the second argument is that its conclusion is false: Pooh isn't a rabbit, he's a bear! But that isn't what distinguishes valid from invalid arguments, because there are valid arguments with false conclusions, and invalid arguments with true conclusions:

(3) All rabbits are birds.
Winnie the Pooh is a rabbit.
∴ Winnie the Pooh is a bird.

(4) All rabbits are mammals
Bugs Bunny is a mammal.
∴ Bugs Bunny is a rabbit.

The first of these is valid, but has a false conclusion. And the second has a true conclusion, as well as true premises, but it still isn't valid because its conclusion doesn't follow from its premises.

Validity isn't about whether the premises or conclusion are in fact true. It rather has to do with the *relationship* between the premises and the conclusion. When we ask about validity we want to know whether, if all the premises were true, the conclusion would also *have to be true*. Put another way:

> An argument is VALID if and only if it is *impossible* for all of its premises to be true but its conclusion to be false.

Let's unpack some of the concepts involved in the two definitions we've encountered a little bit more.

■ Exercises 1.1

As you've seen, we always put the conclusion at the end of an argument and indicate it using the three "therefore dots" ∴. Informally presented arguments don't always have the conclusion at the end, however — it can appear at the beginning, or even in the middle. In each of the following arguments, highlight the phrase which expresses the conclusion:

1. It is sunny. So I should take my sunglasses.
2. It must have been sunny. I did wear my sunglasses, after all.
3. No one but you has had their hands in the cookie-jar. And the scene of the crime is littered with cookie-crumbs. You're the culprit!
4. Miss Scarlett and Professor Plum were in the study at the time of the murder. And Reverend Green had the candlestick in the ballroom, and we know that there is no blood on his hands. Hence Colonel Mustard did it in the kitchen with the lead-piping. Recall, after all, that the gun had not been fired.

1.2 Background Concepts

First, we said that an argument is a collection of STATEMENTS. Statements are sentences that are either true or false. Truth and falsity are called TRUTH-VALUES. The truth-value of a statement is determined by what the world is like: a statement like 'Syracuse is in New York State' describes the world as being a certain way, and is true or false depending on whether the world is that way. As the ancient philosopher (and logician!) Aristotle put it in his book *Metaphysics*:

> "To say of what is that it is not, or of what is not that it is, is false, while to say of what is that it is, and of what is not that it is not, is true." (1011b25)

It's important to notice that not all English sentences count as statements in this sense. For example, none of the following sentences can be assessed as true or false:

- Welcome to the Syracuse Airport!
- Please have your ID ready.
- Are there any liquids in your bag?

A sentence like 'please have your ID ready' isn't meant to describe the world, but to ask someone to do something. Similarly, 'Welcome to Syracuse Airport' is just a greeting, and isn't meant to offer an accurate or inaccurate description of the world. And although the answer to the last question on this list has a truth-value, the question itself doesn't. In general, things like greetings, requests, orders, and questions don't have truth-values, and therefore don't count as statements and cannot be premises or conclusions of arguments. Though it's important to keep this point in mind, moving forward we'll generally use the words "statement" and "sentence" interchangeably.

Next, notice that because a statement's truth value depends on what the world is like, its truth-value could have been different if the world had been different. For example, the sentence'Rieppel is a philosopher' is in fact true, but if I had taken up a different career it would have been false. Conversely, 'Rieppel is a professional juggler' is false, but if I had gone to juggling school instead of continuing with philosophy, it would have been true.

Philosophers often invoke the notion of a POSSIBLE WORLD in this connection. The idea is that besides the actual world, there are various other possible worlds, other ways things could have been — alternative histories, or alternative universes, if you like. 'Rieppel is a professional juggler' is false at the actual world, but it is true at various other possible worlds. Sentences like this, which are true in some possible worlds but not others, are said to be contingent.

Other sentences are not contingent. For example, 'Syracuse either is or is not in New York State' isn't just true at the actual world, it's true at *every* possible world, that is, it's a necessary truth. Mathematical truths are another example: $2+2=4$ is again true in every possible world, and therefore a necessary truth. At the other extreme, a sentence like 'Syracuse both is and is not in New York State' is false in every possible world, or necessarily false.

Returning to arguments, you can think of the notion of validity in terms of possible worlds too. We said that an argument is valid just in case it's *impossible* for all of its premises to be true but it's conclusion false. Put in terms of possible worlds, we can say that an argument is valid if and only if there is *no possible world* where all its premises are true but its conclusion is false. Equivalently put: its conclusion is true at every possible world at which all its premises are true. So again, whether an argument is valid or not doesn't depend on whether its premises and conclusion are in fact true or false. It's about the connection between them — whether it is *possible* for the conclusion to be false while all the premises are true.

There are various logical concepts that we'll encounter in this class that involve the notions of necessity and possibility, besides validity. Here are some of them:

- ▷ A sentence is CONTINGENT if and only if it is possible for it to be true, and also possible for it to be false.
- ▷ A sentence is a NECESSARY TRUTH if and only if it is not possible for it to be false.
- ▷ A sentence is a NECESSARY FALSEHOOD if and only if it is not possible for it to be true.
- ▷ Two sentences are CONTRADICTORY if and only if they necessarily have opposite truth values.
- ▷ Two sentence are EQUIVALENT if and only if they necessarily have the same truth value.
- ▷ A collection of sentences is JOINTLY CONSISTENT if and only if it is possible for all of them to be true together, and JOINTLY INCONSISTENT otherwise.

Notice that these concepts apply to different things. Whereas the first three concern properties of single sentences, the next two concern relations between two sentences, and the last ones properties of collections of sentences. Validity is again slightly different, because it is a property had (or lacked) by only those collections of sentences that also have a designated conclusion.

■ Exercises 1.2

A. For each of the following: is it necessarily true, necessarily false, or contingent?

1. Caesar crossed the Rubicon.
2. Someone once crossed the Rubicon.
3. No one has ever crossed the Rubicon.
4. If Caesar crossed the Rubicon, then someone has.
5. Even though Caesar crossed the Rubicon, no one has ever crossed the Rubicon.
6. If anyone has ever crossed the Rubicon, it was Caesar.

B. Consider the following sentences:

G1. There are at least four giraffes at the wild animal park.
G2. There are exactly seven gorillas at the wild animal park.
G3. There are not more than two martians at the wild animal park.

G4. Every giraffe at the wild animal park is a martian.

Now, for each of the following, determine if the sentences in question are jointly consistent or jointly inconsistent:

1. G2, G3, and G4
2. G1, G3, and G4
3. G1, G2, and G4
4. G1, G2, and G3

C. Could there be:

1. Jointly consistent sentences, one of which is necessarily false?
2. Jointly consistent sentences, one of which is a necessary truth?
3. Jointly inconsistent sentences, one of which is a necessary truth?

In each case: if so, give an example; if not, explain why not.

1.3 Good and Bad Arguments

Being valid is certainly one thing that makes for a good argument, intuitively speaking. But there's more to being a good argument than that.

First off, if an argument has an obviously false premise, then even if it is valid, it remains of limited interest because it doesn't establish its conclusion. By contrast, if an argument is valid and all of its premises are true, then we know that its conclusion must be true too. Arguments of this latter sort are said to be *sound*:

> An argument is SOUND if and only it (i) is valid, and (ii) has premises that are in fact true.

Arguments are generally intended to be not just valid, but sound. So if you're faced with an argument, in a philosophy class or elsewhere, whose conclusion you want to resist, you have two options: you can either try to show that the argument is not valid, or you can try to show that one of its premises is false. What you can't do is accept it as valid, and concede that its premises are true, but still reject the conclusion as false.

Although its very important in practice to determine whether or not the premises of an argument are true, it is (for the most part) not the job of logic to do this. The job of logic is just to determine whether or not an argument is valid. The task of determining whether the argument's premises are in fact true (and the argument sound) is usually best left to experts in the relevant field: biologists, philosophers, historians, physicists, economists, or whomever.

A second way in which validity is not all there is to good argumentation comes out if you consider the following:

In January 2015, it snowed in Syracuse.
In January 2016, it snowed in Syracuse.

In January 2017, it snowed in Syracuse.
In January 2018, it snowed in Syracuse.
In January 2019, it snowed in Syracuse.
So: It snows every January in Syracuse.

This argument generalizes from observations about several past cases to a conclusion about all cases. The argument isn't valid, because even if it snowed in many recent years, that doesn't mean it's impossible for it not to snow in some future year. The argument could be made stronger by adding additional premises, about other snowy Syracuse Januaries in the past. But however many premises of this sort we add, the argument will remain invalid.

That doesn't mean that it's a bad argument. Arguments like this one are called *inductive* arguments, and they are often used legitimately and with great success in science and everyday life. In this book, we shall set aside (entirely) the difficult question of what makes for a good inductive argument. What logic studies is the different notion of *deductive* validity — where truth of the premises has to *guarantee* truth of the conclusion — and this will be the focus of our concern.

■ Exercises 1.3

Here are some exercises to test your understanding of deductive validity and related concepts we've discussed. For these questions, you don't need to worry about the distinction between validity and "validity in virtue of logical form." That is, you can just assume the definition of validity we gave in §1.1 above.

A. Which of the following arguments are valid? Which are invalid?

1. Socrates is a man.
   All men are carrots.
   ∴ Socrates is a carrot.

2. Abe Lincoln was either born in Illinois or he was once president.
   Abe Lincoln was never president.
   ∴ Abe Lincoln was born in Illinois.

3. If I pull the trigger, Abe Lincoln will die.
   I do not pull the trigger.
   ∴ Abe Lincoln will not die.

4. Abe Lincoln was either from France or from Luxemborg.
   Abe Lincoln was not from Luxemborg.
   ∴ Abe Lincoln was from France.

5. If the world were to end today, then I would not need to get up tomorrow morning.
   I will need to get up tomorrow morning.
   ∴ The world will not end today.

6. Joe is now 19 years old.
   Joe is now 87 years old.

∴ Bob is now 20 years old.

B. Could there be:

1. A valid argument that has one false premise and one true premise?
2. A valid argument that has only false premises but a true conclusion?
3. A valid argument with only false premises and a false conclusion?
4. A valid argument with only true premise but a false conclusion?
5. An invalid argument with only true premises and a true conclusion?
6. An invalid argument with only false premises but a true conclusion?
7. A sound argument with a false conclusion?
8. A sound argument with at least one false premise?
9. An invalid argument that can be made valid by the addition of a new premise?
10. A valid argument that can be made invalid by the addition of a new premise?
11. A valid argument, the conclusion of which is necessarily false?
12. An invalid argument, the conclusion of which is necessarily true?
13. A valid argument whose premises are jointly inconsistent?
14. A valid argument with only one premise?

In each case: if so, give an example; if not, explain why not.

1.4 Formal Validity

There's one last complication we have to address before setting out on our investigation of logic. Consider the following arguments:

(5) The sculpture is green all over.
∴ The sculpture is not red all over.

(6) Reihan is a bachelor.
∴ Reihan is not married.

In both cases it is impossible for the premise to be true and the conclusion false: if something's green all over it can't be any other color, and being unmarried is part of what it is to be a bachelor. Both arguments are therefore valid.

But there's an important difference between valid arguments like these and one like the following:

(7) Jenny is either happy or sad.
Jenny is not happy.
∴ Jenny is sad.

This argument is also valid, but there's more. It has a special structure, or logical form, that we might represent as follows:

A or *B*

not-A
$\therefore B$

This is an excellent form for an argument to have, because any argument of this form will be valid, no matter what sentences occur in place of A and B! Or consider our Bugs Bunny argument, which has the structure represented to the right:

(1)	All rabbits are mammals.	All F are G
	Bugs Bunny is a rabbit.	a is F
	$\therefore$ Bugs Bunny is a mammal.	$\therefore a$ is G

Again, any argument of this form will be valid, no matter what predicates we put in for F and G or what name we put in for a.

We can put the general point by saying that arguments like (7) and (1) are valid simply *in virtue of their logical form*. They each exhibit a structure which renders any argument with that structure valid. By contrast, arguments (5) and (6), though valid, are not valid in virtue of their logical form. For example, here's another argument that has the same form as (6) but that isn't valid:

Reihan is a runner.	a is F
$\therefore$ Reihan is not married.	$\therefore a$ is not-G.

What made argument (6) valid wasn't its logical form, but the connection between the meanings of the specific words 'bachelor' and 'married' that occur in its premise and conclusion. Logic is all about patterns, and only cares about *formally* valid arguments like (1) and (7), not arguments like (5) and (6) that are valid for reasons other than their logical form.

1.5 A Look Ahead

Due to logic's concern with form, we will approach the task of distinguishing valid from invalid arguments in an indirect way. We will first introduce a formal language in which we can symbolize English arguments, thereby letting us represent their logical forms. We will then give a precise definition of validity for arguments cast in this formal notation, which will in turn give us our indirect means of distinguishing valid from invalid arguments in English: an English argument is formally valid if it can be symbolized as a valid argument in our formal notation.

In fact, we will study two systems of logic, involving two different formal languages. These systems will differ in what words they treat as *logical constants*, i.e. what words they treat as indicative of logically significant structure. The first system we will study is *Truth-Functional Logic* (or TFL). It will let us represent the structure of arguments like (7) via the symbolization to the right:

(7)	Jenny is either happy or sad.	$(A \lor B)$
	Jenny is not happy.	$\neg A$
	$\therefore$ Jenny is sad.	$\therefore B$

This language will treats words like ‘either . . . or’ and ‘not’ as logical constants, and will use upper-case letters to represent complete sentences.

The second system of logic we will study is *first-order logic* (or FOL). It will let us represent the structure of things like the Bugs Bunny argument via the symbolization to the right:

(1)	All rabbits are mammals.	$\forall x(Fx \to Gx)$
	Bugs Bunny is a rabbit.	Fa
	$\therefore$ Bugs Bunny is a mammal.	$\therefore Ga$

This system extends truth-functional logic by additionally treating words like ‘all’ as logical constants, and using upper- and lower-case letters to represent English predicates and names, respectively.[1] With this short preview out of the way, let’s get started with logic!

[1] At this point you might be wondering why logicians treat some words (like ‘either . . . or’, ‘not’, and ‘all’) as logical constants, but not other words (like ‘happy’ or ‘bachelor’). This is a difficult philosophical question about logic that we won’t get into here. If you’re interested, check out the *Stanford Encyclopedia of Philosophy’s* entries on Logical Constants and Logical Consequence.

I

Truth Functional Logic

Symbolization in TFL 2

2.1 Atomic sentences

In this chapter, we'll look at how to symbolize English arguments in the language of *truth-functional logic* (or TFL), thereby revealing their truth-functional logical structure, or form.[1] Here is an example of a symbolization in TFL:

(1) It is raining outside.
If it is raining outside, then Jenny is miserable.
∴ Jenny is miserable.

A	A
If A, then C	$(A \to C)$
$\therefore C$	$\therefore C$

In symbolizing this argument, I began by replacing *subsentences* of larger sentences with uppercase letters. For example, 'it is raining outside' is a subsentence of the complex sentence 'If it is raining outside, then Jenny is miserable', and we replaced this subsentence with the letter 'A'. Similarly, we used 'C' to symbolize the subsentence 'Jenny is miserable'.

Our artificial language, TFL, uses uppercase letters as its ATOMIC SENTENCES. These will be the basic building blocks, or "atoms," out of which more complex TFL sentences are built. There are only twenty-six letters of the alphabet, but in principle there is no limit to the number of atomic sentences that we might want to consider. By adding numerical subscripts to letters, we obtain as many atomic sentences as we need. So the following all count as atomic sentences of TFL:

$$A, P, P_1, P_2, A_{234}$$

To indicate which atomic sentence of TFL is being used to represent which English sentence, we provide a SYMBOLIZATION KEY, such as the following:

A:	It is raining outside
C:	Jenny is miserable

Doing this does *not* fix this symbolization *once and for all*. We are just saying that, for the time being, we will use the atomic TFL sentence 'A' to symbolize the English sentence 'It

[1]What does "truth-functional" mean? We'll get to that in the next chapter, on the semantics of TFL.

is raining outside', and 'C' to symbolize 'Jenny is miserable'. Later, when we are dealing with different sentences or different arguments, we can provide a new symbolization key that associates these atomic TFL sentences with different English sentences.

It is important to recognize that whatever internal structure an English sentence might have is lost if it is symbolized by an atomic sentence of TFL. From the point of view of TFL, an atomic sentence has no internal (or "subatomic") structure. It can be used to build more complex sentences, but it cannot be taken apart.

For this reason, English sentences that have an internal logical structure to them, like the conditional 'If it's raining outside, then Jenny is miserable', should not be symbolized using atomic sentences of TFL. If we just symbolized this sentence as 'P', for example, our symbolization would obscure the fact that it has the form of an 'if ... then ... ' statement, and that it contains subsentences that also occur on their own as premise and conclusion in our argument (1) above. We would thefore miss out on the logical form in virtue of which argument (1) is valid.

For this reason we have to symbolize logically complex sentences of English via complex (i.e. non-atomic) sentence of TFL, in this case '$(A \rightarrow C)$'. Complex sentences of TFL are built up by combining atomic sentences with *connectives*.

TFL connectives will be used to symbolize English connectives like 'if ... then', 'or' and 'not'. Just as these English connectives can be applied to sentences to form new, bigger sentences, so TFL connectives can be applied to atomic sentences to build larger, complex sentences of TFL. in total, there are five connectives in TFL. This table summarizes them, and gives you a rough indication of their meaning:

symbol	**name**	**rough meaning**
$\neg$	negation	'It is not the case that...'
$\wedge$	conjunction	'Both... and ...'
$\vee$	disjunction	'Either... or ...'
$\rightarrow$	conditional	'If ... then ...'
$\leftrightarrow$	biconditional	'... if and only if ...'

For the remainder of this chapter, we have two main objectives, a practical one and a technical one. The first, practical objective is to learn how to symbolize logically complex English sentences using our TFL connectives. The second, more technical objective will be to give a precise grammar, or syntax, for the language of TFL, which explains exactly how TFL connectives combine with atomic sentences to produce complex TFL sentences. We'll turn to this technical task later, after we've gotten some practice using the language of TFL to symbolize English sentences.

2.2 Negation

Consider how we might symbolize these sentences:

(2) Mary is in Barcelona.
(3) It is not the case that Mary is in Barcelona.
(4) Mary is not in Barcelona.

In order to symbolize (2), we will need an atomic sentence. We might offer this symbolization key:

B: Mary is in Barcelona.

Since (3) is obviously related to (2), we do not want to symbolize it with a completely different sentence, say 'A'. Roughly, sentence (3) means something like 'It is not the case that B'. In order to symbolize this, we need a symbol for negation. We will use '$\neg$'. Now we can symbolize (3) with '$\neg B$'.

Sentence (4) also contains the word 'not'. And it is obviously equivalent to (3). As such, we can also symbolize it with '$\neg B$'.

A sentence can be symbolized as $\neg\mathscr{A}$ if it can be paraphrased in English as 'It is not the case that…'.

Here's another example:

(5) The widget can be replaced.
(6) The widget is irreplaceable.
(7) The widget is not irreplaceable.

Let's use the following symbolization key:

R: The widget is replaceable

Sentence (5) can then be symbolized by 'R'. Next, (6) says the widget is irreplaceable, which means that it is not the case that the widget is replaceable. So even though (6) does not contain the word 'not', we can still symbolize it as '$\neg R$'. Sentence (7) can be paraphrased as 'It is not the case that the widget is irreplaceable.' Which can again be paraphrased as 'It is not the case that it is not the case that the widget is replaceable'. So we can symbolize this with the TFL sentence '$\neg\neg R$'.

At this point you might be wondering: don't double negatives cancel out, so that '$\neg\neg R$' is equivalent to plain 'R'? We'll get into meaning, or semantics, of TFL sentences in the next chapter, but the quick answer is that, yes, these are equivalent, and (7) *could* also be symbolized as 'R'. Still, '$\neg\neg R$' is the *preferred* symbolization, since it represents more of the logical structure implicit in the English sentence (7).

Some care is needed when handling negations. Consider:

(8) Jane is happy.
(9) Jane is unhappy.

If we let 'H' symbolize 'Jane is happy', we can symbolize (8) as 'H'. However, it would be a mistake to symbolize (9) with '$\neg H$'. Sentence (9) does not mean the same thing as 'It is not the case that Jane is happy'. Jane might be neither happy nor unhappy; she might be in a state of blank indifference. In order to symbolize (9), then, we would need a new atomic sentence of TFL.

2.3 Conjunction

Consider the sentence:

(10) Adam is athletic, and Barbara is athletic.

We will need separate atomic sentences to symbolize the two subsentences that occur in (10).

A: Adam is athletic.
B: Barbara is athletic.

We will use '$\wedge$' to symbolize 'and', and thus symbolize (10) as '$(A \wedge B)$'. This connective is called CONJUNCTION. We also say that 'A' and 'B' are the two CONJUNCTS of the conjunction '$(A \wedge B)$'.

There are few things to notice about conjunction. First, in English the word 'and' doesn't always conjoin two sentences:

(11) Barbara is athletic and energetic.
(12) Barbara and Adam are both athletic.

In (11) the word 'and' conjoins two adjectives, rather than two sentences. But it can be paraphrased as 'Barbara is athletic and Barbara is energetic' where 'and' now does conjoin two complete sentences. So if we use 'E' symbolize 'Barbara is energetic', we can symbolize the entire sentence as '$(B \wedge E)$'. In sentence (12), 'and' conjoins two names. Again, though, this can be paraphrased in terms of a conjunction of two sentences, as 'Barbara is athletic and Adam is athletic', and can therefore be symbolized as '$(B \wedge A)$'.[2]

Second, conjunction can be expressed in English using words other than 'and':

(13) Although Barbara is energetic, she is not athletic.
(14) Adam is athletic, but Barbara is not.

In (13), the word 'although' sets up a contrast between the first part of the sentence and the second part. Nevertheless, the sentence tells us both that Barbara is energetic and that she is not athletic. So we can symbolize it as a conjunction:

B: Barbara is athletic.
E: Barbara is energetic.

Symbolization of (13): $(E \wedge \neg B)$

[2]Some care is needed with this. Not *all* sentences where 'and' conjoins two names can be paraphrased in a way where 'and' conjoins two sentences. For example, 'Barbara and Adam carried the piano upstairs' may not mean the same as 'Barbara carried the piano upstairs and Adam carried the piano upstairs', since the latter (but not the former) is compatible with them each carrying it individually rather than together.

Of course we have lost all sorts of nuance in this symbolization. There is a distinct difference in tone between the English sentence (13) and ‘Both Barbara is energetic and it is not the case that Barbara is athletic’. TFL does not (and cannot) preserve these nuances, however, and so we do not attend to them when symbolizing English into TFL. Notice also that in the symbolization key we replaced the pronoun ‘she’ with ‘Barbara’, since it might otherwise be unclear who ‘she’ is meant to refer to. In general: always use names in place of pronouns in your symbolization key!

Sentence (14) raises similar issues. The word ‘but’ sets up a contrast between the two parts of the sentence, but this is not something that TFL can deal with. We can paraphrase the sentence as ‘*Both* Adam is athletic, *and* Barbara is not athletic’. Notice that the second conjunct involves a negation as well. Using the sentence letters already introduced, we can symbolize (14) as ‘$(A \wedge \neg B)$’.

There are other words besides ‘although’ and ‘but’ that can be used to express conjunction. For example, ‘Barbara is energetic, however she is not athletic’ and ‘Barbara is energetic despite not being athletic’ expresses the same conjunctive claim as sentence (13) from above, and get symbolized using the same TFL sentence. In general, the symbolization guideline for conjunction is:

A sentence can be symbolized as $(\mathscr{A} \wedge \mathscr{B})$ if (nuance aside) it can be paraphrased in English as ‘Both…, and…’.

You might be wondering why we always put *brackets* around the conjunctions. The reason can be brought out by thinking about negation interacts with conjunction. Consider:

(15) You will not get both soup and salad.
(16) You will not get soup but you will get salad.

Sentence (15) can be paraphrased as ‘It is not the case that: both you will get soup and you will get salad’. Using the following symbolization key:

S_1: You get soup.
S_2: You get salad.

we would symbolize ‘both you will get soup and you will get salad’ as ‘$(S_1 \wedge S_2)$’. To symbolize the whole of sentence (15), then, we negate this, giving us ‘$\neg(S_1 \wedge S_2)$’. Sentence (16), on the other hand, gets symbolized as a conjunction whose first conjunct is negated ‘$(\neg S_1 \wedge S_2)$’.

These English sentences are very different, and their symbolizations differ accordingly. In one of them, the entire conjunction is negated. In the other, just one conjunct is negated. Brackets help us to keep track of things like the *scope* of the negation: whether it applies to the whole conjunction, or just to the first conjunct.

2.4 Disjunction

Consider these sentences:

(17) Either Denison will play golf with me, or he will watch movies.

(18) Either Denison or Ellery will play golf with me.

We can use the following symbolization key for these sentences (notice that we, as always, replace pronouns with names):

D:	Denison will play golf with me.
E:	Ellery will play golf with me.
M:	Denison will watch movies.

Sentence (17) is and 'either ... or' statement, and gets symbolized as '$(D \vee M)$'. The connective is called DISJUNCTION. We also say that 'D' and 'M' are the DISJUNCTS of the disjunction '$(D \vee M)$'.

Sentence (18) is only slightly more complicated. Here 'or' occurs between two names rather than two complete sentences. However, we can paraphrase sentence (18) as 'Either Denison will play golf with me, or Ellery will play golf with me' where 'or' now connects two complete sentences. So we can symbolize it as '$(D \vee E)$'.

A sentence can be symbolized as $(\mathscr{A} \vee \mathscr{B})$ if it can be paraphrased in English as 'Either..., or....'

Sometimes in English, the word 'or' excludes the possibility that both disjuncts are true. This is called an EXCLUSIVE OR. An *exclusive or* is intended when it says, on a restaurant menu, 'Entrees come with either soup or salad': you may have soup; you may have salad; but, if you want *both* soup *and* salad, then you have to pay extra. At other times, the word 'or' allows for the possibility that both disjuncts might be true. This is probably the case with sentence (18), above: I might play golf with Denison, with Ellery, or with both Denison and Ellery. Sentence (18) merely says that I will play with *at least* one of them. This is an INCLUSIVE OR.

Importantly, the TFL symbol '$\vee$' expresses *inclusive or*. Whenever you see the English words 'either ... or' in this book, you can assume that the inclusive sense is intended, and symbolize such sentences using '$\vee$'. When an exclusive sense is intended, we will always add an explicit 'but not both', as in:

(19) The entree will come with either soup or salad, but not both.

Using 'S_1' for 'the entree will come with soup' and 'S_2' for 'the entree will come with salad', we can symbolize (19) as '$((S_1 \vee S_2) \wedge \neg(S_1 \wedge S_2))$'. So although the TFL symbol '$\vee$' always symbolizes *inclusive or*, we can symbolize an *exclusive or* in TFL. We just have to use a few other TFL symbols in addition to '$\vee$'!

There are some futher interesting interactions between disjunction and negation that we should attend to. Consider the following:

(20) Either Barbara will not get soup, or she will not get salad.
(21) Barbara will get neither soup nor salad.

Sentence (20) can be paraphrased as: '*Either* it is not the case that Barbara will get soup, *or* it is not the case that Barbara will get salad'. Using the following symbolization key:

P: Barbara will get soup.
D: Barbara will get salad.

we can symbolize (20) as $(\neg P \lor \neg D)$. This has the form of a disjunction both disjuncts of which are negated. Sentence (21) has a different structure. It can be paraphrased as, '*It is not the case that*: either Barbara will get soup or Barbara will get salad'. Since this negates the entire disjunction, we symbolize sentence (21) as '$\neg(P \lor D)$'. This differs from our symbolization of (20), as it should given that the two English sentences mean different things.

You may have noticed that (20) means the same thing as 'Barbara will not get both soup and salad.' The latter can be symbolized as a negated conjunction, '$\neg(P \land D)$'. Since '$\neg(P \land D)$' symbolizes an English sentence that's equivalent to (20), and (20) is symbolized as '$(\neg P \lor \neg D)$', we can conclude that the TFL sentences '$\neg(P \land D)$' and '$(\neg P \lor \neg D)$' are themselves equivalent.

Similarly, (21) means the same as 'Barbara will not get soup and Barbara will not get salad', which can be symbolized as a conjunction of two negations '$(\neg P \land \neg D)$'. So again, since '$(\neg P \land \neg D)$' symbolizes an English sentence that's equivalent to (21), and (21) is symbolized as '$\neg(P \lor D)$', we can conclude that these two TFL sentences are themselves equivalent. What we've discovered are:

DEMORGAN'S LAWS:

$\neg(\mathscr{A} \lor \mathscr{B})$ is equivalent to $(\neg\mathscr{A} \land \neg\mathscr{B})$

$\neg(\mathscr{A} \land \mathscr{B})$ is equivalent to $(\neg\mathscr{A} \lor \neg\mathscr{B})$

These laws are named after Augustus DeMorgan who first explicitly formulated them in the nineteenth century. These laws will stay with us throughout our study of logic, and we will see how to prove that these equivalences hold in the next chapter.[3]

2.5 Conditional

Consider these sentences:

(22) If Jean is in Paris, then Jean is in France.
(23) Jean is in France if Jean is in Paris.
(24) Jean is in France only if Jean is in Paris.

Let's use the following symbolization key:

P: Jean is in Paris.
F: Jean is in France

Sentence (22) is roughly of this form: 'if P, then F'. We will use the symbol '$\rightarrow$' to symbolize the English 'if…, then…' structure. So we symbolize sentence 23 by '$(P \rightarrow F)$'.

[3]It is noteworthy that these equivalences only hold in TFL given that '$\lor$' expresses inclusive disjunction. The fact that the corresponding English sentences are also intuitively equivalent again shows that English 'or' often expresses inclusive disjunction.

The connective is called THE CONDITIONAL. Here, '*P*' is called the ANTECEDENT of the conditional '$(P \to F)$', and '*F*' is called the CONSEQUENT.

Sentence (23) looks different from (22) since the word 'if' occurs in the middle of the sentence rather than at the beginning. But clearly (23) it is equivalent to (22), so we can also symbolize it as '$(P \to F)$'. In general, the 'if'-clause of an English conditional always introduces the *antecedent*, whether or not it occurs first in the English sentence, with the rest of the sentence functioning as the *consequent*.

Sentence (24) is also a conditional. Since the word 'if' appears in the second half of the sentence, it might be tempting to symbolize this in the same way as sentence (23), as '$(P \to F)$'. But that would be a mistake. My knowledge of geography tells me that sentence (23) is unproblematically true: there is no way for Jean to be in Paris that doesn't involve Jean being in France. But sentence (24) is not so straightforward: were Jean in Marseilles, Lyons, or Toulouse, Jean would be in France without being in Paris, thereby rendering sentence (24) false. Since geography alone dictates the truth of sentence (23), whereas travel plans (say) are needed to know the truth of sentence (24), they must mean different things, and (24) can't be symbolized as '$(P \to F)$.

The moral is that 'only if' means something very different from plain 'if'. The 'if'-clause of a conditional introduces a SUFFICIENT CONDITION: (22) and (23) say that Jean's being in Paris is *sufficient for* his being in France (which is unproblematically true). 'Only if', by contrast, introduces a NECESSARY CONDITION: sentence (24) claims that Jean's being in Paris *is necessary for* his being in France (likely false, since there are other ways to be in France). In TFL, the antecedent $\mathscr{A}$ of a conditional $(\mathscr{A} \to \mathscr{B})$ always indicates the sufficient condition, whereas the consequent $\mathscr{B}$ indicates the the necessary condition. Since Jean's being in Paris is claimed as a necessary condition in (24), this sentence is symbolized as '$(F \to P)$'. In general, whereas 'if' introduces the antecedent of a conditional, 'only if' introduces the consequent. So our symbolization guidelines for conditionals are:

A sentence can be symbolized as $(\mathscr{A} \to \mathscr{B})$ if it can be paraphrased in English as 'If A, then B', or as 'B if A', or as 'A only if B'.

The fact that (22) is symbolized as '$(P \to F)$' means that this 'if. . . then' statement can also be paraphrased as an 'only if' statement, 'Jean is in Paris only if Jean is in France'. That's intuitively correct: since his being in Paris is sufficient for his being in France, it's also true that his being in France is necessary for his being in Paris. This connection between conditionals and necessary and sufficient conditions shows that the conditional can represent many English constructions that don't involve the word 'if' at all. Consider:

(25) It is necessary condition on Bugs' being a rabbit that he be a mammal.
(26) For Bugs to be a rabbit, it is required that Bugs be a mammal.
(27) For Bugs to be a mammal, it is enough that Bugs be a rabbit.
(28) It is a sufficient for Bugs' being a mammal that he be a rabbit.

If we think about it, all four of these sentences mean the same as 'If Bugs is a rabbit, then Bugs is a mammal'. So using '*R*' for 'Bugs is a rabbit' and '*M*' for 'Bugs is a mammal, all four can all be symbolized as '$(R \to M)$'.

It is important to bear in mind that the connective '$\to$' tells us only that, if the antecedent is true, then the consequent is true. It says nothing about a *causal* connection between two events (for example). In fact, we lose a huge amount when we use '$\rightarrow$' to symbolize English conditionals. We will look more closely at the discrepancies between English 'if. . . then' and the TFL connective '$\rightarrow$' in §3.3.

2.6 Biconditional

Consider the following sentence:

(29) Shergar is a horse if and only if he is a mammal

Let's use the following symbolization key:

H:	Shergar is a horse
M:	Shergar is a mammal

Sentence 29 can be paraphrased as 'Shergar is a horse if he is a mammal, and Shergar is a horse only if Shergar is a mammal'. This is just the conjunction of two conditionals we already know how to symbolize. So we can symbolize it as '$((H \rightarrow M) \wedge (M \rightarrow H))$'. We call this a BICONDITIONAL, because it amounts to stating both directions of the conditional. That is, (29) says (falsely, as it happens) that Shergar's being a mammal is both necessary and sufficient for his being a horse.

We could treat every biconditional this way, as a conjunction of two conditionals. So, just as we do not need a new TFL symbol to deal with *exclusive or*, we do not really need a new TFL symbol to deal with biconditionals. However, since biconditionals occur a lot in logic an philosophy, we'll use the dedicated connective '$\leftrightarrow$' to symbolize them. We can then can symbolize sentence 29 with the TFL sentence '$(H \leftrightarrow M)$'.

Since 'if and only if' gets used so much in logic and philosophy, it is often abbreviated with a single, snappy word, 'iff'. I shall follow this practice. So 'if' with only *one* 'f' is the English conditional. But 'iff' with *two* 'f's is the English biconditional. Armed with this we can say:

A sentence can be symbolized as $(\mathscr{A} \leftrightarrow \mathscr{B})$ if it can be paraphrased in English as 'A iff B', that is to say, as 'A if and only if B'.

A word of caution. Ordinary speakers of English often use 'if . . . , then. . . ' when they really mean to use something more like '. . . if and only if . . . '. Perhaps your parents told you, when you were a child: 'if you don't eat your greens, you won't get any dessert'. Suppose you ate your greens, but that your parents then refused to give you any dessert, on the grounds that they were only committed to the *conditional* (roughly 'if you get dessert, then you will have eaten your greens'), rather than the biconditional (roughly, 'you get dessert iff you eat your greens'). Well, a tantrum would rightly ensue. So, be aware of this when interpreting people; but in your own writing, make sure to use the biconditional iff you mean to.

2.7 ‘Unless’

We have now seen all of the connectives of TFL. We can use them together to symbolize many kinds of sentences. But some cases are harder than others. And a typically nasty case is the English-language connective ‘unless’:

(30) Unless you wear a jacket, you will catch a cold.
(31) You will catch a cold unless you wear a jacket.

These two sentences are clearly equivalent. To symbolize them, we shall use the symbolization key:

J:	You will wear a jacket.
D:	You will catch a cold.

Both sentences mean that if you do not wear a jacket, then you will catch a cold. With this in mind, we might symbolize them as ‘$(\neg J \rightarrow D)$’.

Equally, both sentences mean that if you do not catch a cold, then you must have worn a jacket. With this in mind, we might symbolize them as ‘$(\neg D \rightarrow J)$’.

Equally, both sentences mean that either you will wear a jacket or you will catch a cold. With this in mind, we might symbolize them as ‘$(J \vee D)$’.

All three are correct symbolizations. Indeed, in the next chapter we shall see that all three symbolizations are equivalent in TFL. For simplicity we’ll use the guideline:

> If a sentence can be paraphrased as ‘Unless A, B’ or ‘B unless A’, then it can be symbolized as $(\mathscr{A} \vee \mathscr{B})$.

Again, though, there is a little complication. ‘Unless’ can be symbolized as a conditional. But as noted above, people often use the conditional (on its own) when they mean to use the biconditional. Equally, ‘unless’ can be symbolized as a disjunction; but there are two kinds of disjunction (exclusive and inclusive). So it will not surprise you to discover that ordinary speakers of English often use ‘unless’ to mean something more like the biconditional, or like exclusive disjunction. Suppose I say: ‘I will go running unless it rains’. I probably mean something like ‘I will go running if and only if it does not rain’ (i.e. the biconditional), or ‘either I will go running or it will rain, but not both’ (i.e. exclusive disjunction). However, in this book we’ll always use ‘unless’ in the strict sense in which it can be symbolized using a (inclusive) disjunction or a conditional.

■ Exercises 2.7

A. Using the symbolization key given, symbolize each English sentence in TFL.

M:	Those creatures are men in suits.
C:	Those creatures are chimpanzees.
G:	Those creatures are gorillas.

1. Those creatures are not men in suits.

2. Those creatures are men in suits, or they are not.
3. Those creatures are either gorillas or chimpanzees.
4. Those creatures are neither gorillas nor chimpanzees.
5. If those creatures are chimpanzees, then they are neither gorillas nor men in suits.
6. Unless those creatures are men in suits, they are either chimpanzees or they are gorillas.

B. Using the symbolization key given, symbolize each English sentence in TFL.

A:	Mister Ace was murdered.
B:	The butler did it.
C:	The cook did it.
D:	The Duchess is lying.
E:	Mister Edge was murdered.
F:	The murder weapon was a frying pan.

1. Either Mister Ace or Mister Edge was murdered.
2. If Mister Ace was murdered, then the cook did it.
3. If Mister Edge was murdered, then the cook did not do it.
4. Either the butler did it, or the Duchess is lying.
5. The cook did it only if the Duchess is lying.
6. If the murder weapon was a frying pan, then the culprit must have been the cook.
7. If the murder weapon was not a frying pan, then the culprit was either the cook or the butler.
8. Mister Ace was murdered if and only if Mister Edge was not murdered.
9. The Duchess is lying, unless it was Mister Edge who was murdered.
10. If Mister Ace was murdered, he was done in with a frying pan.
11. Since the cook did it, the butler did not.
12. Of course the Duchess is lying!

C. Using the symbolization key given, symbolize each English sentence in TFL.

E_1:	Ava is an electrician.
E_2:	Harrison is an electrician.
F_1:	Ava is a firefighter.
F_2:	Harrison is a firefighter.
S_1:	Ava is satisfied with her career.
S_2:	Harrison is satisfied with his career.

1. Ava and Harrison are both electricians.
2. If Ava is a firefighter, then she is satisfied with her career.
3. Ava is a firefighter, unless she is an electrician.
4. Harrison is an unsatisfied electrician.
5. Neither Ava nor Harrison is an electrician.
6. Both Ava and Harrison are electricians, but neither of them find it satisfying.
7. Harrison is satisfied only if he is a firefighter.
8. If Ava is not an electrician, then neither is Harrison, but if she is, then he is too.
9. Ava is satisfied with her career if and only if Harrison is not satisfied with his.

10. If Harrison is both an electrician and a firefighter, then he must be satisfied with his work.
11. It cannot be that Harrison is both an electrician and a firefighter.
12. Harrison and Ava are both firefighters if and only if neither of them is an electrician.

D. Give a symbolization key and symbolize the following English sentences in TFL.

1. Alice and Bob are both spies.
2. If either Alice or Bob is a spy, then the code has been broken.
3. If neither Alice nor Bob is a spy, then the code remains unbroken.
4. The German embassy will be in an uproar, unless someone has broken the code.
5. Either the code has been broken or it has not, but the German embassy will be in an uproar regardless.
6. Either Alice or Bob is a spy, but not both.

E. Give a symbolization key and symbolize the following English sentences in TFL.

1. If there is food to be found in the pridelands, then Rafiki will talk about squashed bananas.
2. Rafiki will talk about squashed bananas unless Simba is alive.
3. Rafiki will either talk about squashed bananas or he won't, but there is food to be found in the pridelands regardless.
4. Scar will remain as king if and only if there is food to be found in the pridelands.
5. If Simba is alive, then Scar will not remain as king.

F. For each argument, write a symbolization key and symbolize all of the sentences of the argument in TFL.

1. If Dorothy plays the piano in the morning, then Roger wakes up cranky. Dorothy plays piano in the morning unless she is distracted. So if Roger does not wake up cranky, then Dorothy must be distracted.
2. It will either rain or snow on Tuesday. If it rains, Neville will be sad. If it snows, Neville will be cold. Therefore, Neville will either be sad or cold on Tuesday.
3. If Zoog remembered to do his chores, then things are clean but not neat. If he forgot, then things are neat but not clean. Therefore, things are either neat or clean; but not both.

G. We symbolized *exclusive or*, or *xor*, using '$\vee$', '$\wedge$', and '$\neg$'. How could you symbolize an *xor* using only two connectives? Hint: think about how you might use DeMorgan's Laws to eliminate one of the connectives in your symbolization.

2.8 The Syntax of TFL

In the course of learning to symbolize English sentences in TFL, we've gotten a pretty good intuitive sense of how to build up complex TFL sentences from atomic ones using our five connectives. But as we move ahead, it will be necessary for us to be a bit more precise about the structure of our logical language. Because just as not every string of English words counts as a grammatical English sentence (e.g. 'shelf on book red lies' is just gibberish), so not every string of symbols counts as grammatical, or well-formed sentence of TFL.

We have seen that there are three kinds of symbols in TFL:

Atomic sentences:	$A, B, C, \ldots, Z, \ldots, A_1, B_1, Z_1, A_2, A_{25}, J_{375}, \ldots$
Connectives:	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Brackets:	(,)

This constitutes the LEXICON of TFL. Now define an EXPRESSION OF TFL to be any string of symbols of TFL. That is: write down any sequence of symbols from the lexicon of TFL, in any order, and you have an expression of TFL.

Strings like '$(A \leftarrow B)$' or '$(p \vee C)$' or '$\neg(\phi \wedge A)$' are not expressions of TFL because they contain symbols like '$\leftarrow$' (leftward arrows) 'p' (lowercase letters) and 'ϕ' (Greek letters) that are not even in the Lexicon of TFL. On the other hand, '$(A \wedge B)$' is an expression of TFL, and so are '$(\neg)A \rightarrow)$' and '$\neg)(\vee() \wedge (\neg\neg())((B$'. However, whereas the first of these expressions also counts as a *sentence* of TFL, the rest are just *gibberish*. What we want are some rules to tell us precisely which TFL expressions count as sentences.

Obviously, individual atomic sentences like 'A' and 'G_{13}' should count as sentences. We can form further sentences out of these by using the various connectives. Using negation, we can get '$\neg A$' and '$\neg G_{13}$'. Using conjunction, we can get '$(A \wedge G_{13})$', '$(G_{13} \wedge A)$', '$(A \wedge A)$', and '$(G_{13} \wedge G_{13})$'. We could also apply negation repeatedly to get sentences like '$\neg\neg A$' and '$\neg\neg\neg A$', or apply negation to one of our conjunctions to get sentences like '$\neg(A \wedge G_{13})$' and '$\neg(G_{13} \wedge \neg G_{13})$'. There are infinitely many possible combinations, even starting with just these two sentence letters. And of course there are infinitely many sentence letters. So there is no point in trying to list all of the sentences of TFL one by one.

Instead, we will describe the process by which sentences can be *constructed*. Consider negation: given any sentence $\mathscr{A}$ of TFL, putting a negation in front of it gives us a sentence $\neg\mathscr{A}$. We can say similar things for each of the other connectives. For instance, if $\mathscr{A}$ and $\mathscr{B}$ are sentences of TFL, then $(\mathscr{A} \wedge \mathscr{B})$ is a sentence of TFL. (What's up with the funny cursive letters, which are *not* in the lexicon of TFL? We'll get to that in §2.8 below.)

Providing clauses like this for all of the connectives, we arrive at the following SYNTAX, or formal definition of what counts as a SENTENCE OF TFL:

THE SYNTAX OF TFL:

1. Every atomic sentence is a sentence.
2. If $\mathscr{A}$ is a sentence, then $\neg\mathscr{A}$ is a sentence.
3. If $\mathscr{A}$ and $\mathscr{B}$ are sentences, then $(\mathscr{A} \wedge \mathscr{B})$ is a sentence.
4. If $\mathscr{A}$ and $\mathscr{B}$ are sentences, then $(\mathscr{A} \vee \mathscr{B})$ is a sentence.
5. If $\mathscr{A}$ and $\mathscr{B}$ are sentences, then $(\mathscr{A} \rightarrow \mathscr{B})$ is a sentence.
6. If $\mathscr{A}$ and $\mathscr{B}$ are sentences, then $(\mathscr{A} \leftrightarrow \mathscr{B})$ is a sentence.
7. Nothing else is a sentence.

Definitions like this are called *recursive*. Recursive definitions begin with some list of base elements (in this case, atomic sentences), and then present ways to generate indefinitely many more elements by compounding together previously generated elements. We can then determine if any given TFL expression counts as a sentence by checking whether it can be generated by applying these recursive rules of syntax.

For example, suppose we want to know whether or not '$\neg\neg D$' is a sentence of TFL. Looking at clause 2 of the definition, we know that '$\neg\neg D$' is a sentence *if* '$\neg D$' is a sentence. So now we need to ask whether or not '$\neg D$' is a sentence. Again looking at clause 2 of the definition, '$\neg D$' is a sentence *if* 'D' is. And 'D' is an atomic sentence of TFL, so we know that 'D' is a sentence by the clause 1 of the definition. So by applying the clauses of our definition repeatedly, we see that our original sentence '$\neg\neg D$' can be generated by applying the rules of our syntax to atomic sentences, and thus counts as a TFL sentence.

Next, consider a more complex example: '$\neg(P \wedge \neg(\neg Q \vee R))$'. Looking at clause 2 of the definition, this is a sentence if '$(P \wedge \neg(\neg Q \vee R))$' is. And by clause 3 the latter is a sentence if *both* 'P' *and* '$\neg(\neg Q \vee R)$' are sentences. The former is an atomic sentence, and the latter is a sentence if '$(\neg Q \vee R)$' is a sentence. Looking at clause 4, we see this is a sentence if both '$\neg Q$' and 'R' are sentences. And both are! So we've shown that the expression we started with is indeed a sentence.

Notice that negation differs from our other operators. It attaches to a *single* sentence to form a new sentence, and is therefore called a UNARY OPERATOR. By contrast, the other operators all operate on *pairs* of sentences to form new sentences, and are therefore called BINARY OPERATORS.

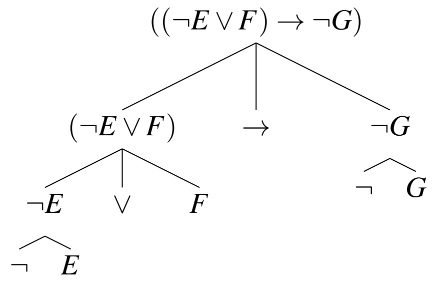
Our syntactic rules tell us that any sentence formed by applying a binary operator to a pair of sentences must be enclosed by parentheses. For example, when putting 'S' and 'R' together using a conjunction, the resulting sentence '$(S \wedge R)$' must have brackets around it. So '$S \wedge R$' is not technically a sentence of TFL, but a mere expression. This is *not* the case for negation, however! Putting a negation in front of a sentence never requires adding parentheses. So '$\neg\neg D$' and '$\neg(S \wedge R)$' are well-formed sentences, but '$(\neg(\neg(D)))$' or '$(\neg(S \wedge R))$' are not sentences. (We'll return to the rationale behind these bracketing rules in §2.8 below.)

Some Syntactic Notions

Ultimately, every TFL sentence is constructed nicely out of atomic sentences on the basis of our syntactic rules. When we are dealing with any complex (i.e. non-atomic) sentence, we can see that there must be some sentential connective that was introduced *most recently* when constructing that sentence. We call that the MAIN OPERATOR of the sentence:

> The MAIN OPERATOR of a complex TFL sentence is the one that was introduced *most recently* in the process of constructing that sentence.

In the case of '$\neg\neg D$', the main operator is the very first '$\neg$' sign. In the case of '$((\neg E \vee F) \to \neg G)$', the main operator is '$\to$' because the last step in constructing this sentence is to connect '$(\neg E \vee F)$' and '$\neg G$' using '$\to$' (and putting brackets around the result). One can visually represent the process in which a sentence is constructed from its parts via a SYNTACTIC TREE for the sentence. For example, the syntactic tree for '$((\neg E \vee F) \to \neg G)$' looks like this:



This shows that '$((\neg E \vee F) \to \neg G)$' was constructed by connecting '$(\neg E \vee F)$' and '$\neg G$' using '$\to$', and '$(\neg E \vee F)$' was in turn constructed by connecting '$\neg E$' and 'F' using '$\vee$', and '$\neg E$' was constructed by putting '$\neg$' in front of 'E', and so on. If we represent the construction process in terms of a tree structure like this, then the main operator of a sentence is whichever operator occurs on a branch of its own at the *first level* below the sentence as a whole (which again, in this case, is '$\to$').

The syntactic structure of sentences in TFL also allows us to give a formal definition of the *scope* of a negation (mentioned in §2.3). The scope of a '$\neg$' in a given sentence is whatever subsentence of that sentence has '$\neg$' as its main logical operator. For example, consider the complex TFL sentence:

$$(\neg(R \wedge B) \leftrightarrow Q)$$

This was constructed by putting a biconditional between '$\neg(R \wedge B)$' and 'Q' . So '$\neg(R \wedge B)$' is a *subsentence* of the sentence as a whole. And the main logical operator for this subsentence is '$\neg$'. So the scope of the negation in '$(\neg(R \wedge B) \leftrightarrow Q)$' is just '$\neg(R \wedge B)$'. More generally:

The SCOPE of any connective in a sentence is the subsentence for which that connective is the main logical operator.

So again in the case of '$(\neg(R \wedge B) \leftrightarrow Q)$', the scope of '$\leftrightarrow$' is the sentence as a whole (since it is the main operator of the whole sentence), and the scope of '$\wedge$' is '$(R \wedge B)$', since $(R \wedge B)$ is the subsentence of which '$\wedge$' is the main operator.

Bracketing

As mentioned in §2.3 and §2.8 above, brackets are an important part of the syntax of TFL. This is because they demarcate the scope of connectives. For example, there is an important difference between '$\neg(P \wedge Q)$' and '$(\neg P \wedge Q)$'. In the case of '$\neg(P \wedge Q)$' the scope of the negation operator is the whole sentence, that is, it is the main operator of the sentence and it serves to negate the entire conjunction it is attached to. In the case of '$(\neg P \wedge Q)$', the scope of the negation is just the subsentence '$\neg P$', and the main operator of the sentence as a whole is '$\wedge$'.

Strictly speaking, therefore, a string like '$\neg P \wedge Q$' is *not* a sentence of TFL, but a mere expression, because it is missing brackets. As things stand, it is not clear where in '$\neg P \wedge Q$' the brackets are supposed to go, that is, whether it is supposed to be a negated conjunction, i.e. '$\neg(P \wedge Q)$', or rather a conjunction with a negated left conjunct, i.e. '$(\neg P \wedge Q)$'. When working with TFL, however, it will make our lives easier if we are sometimes a little less strict. So, here are some convenient conventions.

First, we'll allow ourselves to omit the *outermost* brackets on a sentence. Thus we allow ourselves to write '$Q \wedge R$' instead of '$(Q \wedge R)$'. However, we have to put the brackets back in when we want to embed this sentence into another, larger sentence! So we cannot write '$P \to Q \wedge R$', but must write $P \to (Q \wedge R)$' instead. With this convention in place, something like '$\neg P \wedge Q$' can now be interpreted as missing its outermost parentheses, and thus being a shorthand for '$(\neg P \wedge Q)$' rather than '$\neg(P \wedge Q)$'.

Second, it can be a bit painful to stare at long sentences with many nested pairs of brackets. To make things a bit easier on the eyes, we will allow ourselves to use square brackets, '[' and ']', instead of rounded ones. So there is no logical difference between '$(P \vee Q)$' and '$[P \vee Q]$', for example. Combining this convention with the first one, we can rewrite the unwieldy sentence:

$$(((H \to I) \vee (I \to H)) \wedge (J \vee K))$$

more simply as follows:

$$\big[(H \to I) \vee (I \to H)\big] \wedge (J \vee K)$$

The scope of each connective is now much more visually apparent.

Metalanguage and Metavariables

Our recursive definition of TFL sentences included clauses like the following:

3. If $\mathscr{A}$ and $\mathscr{B}$ are sentences, then $(\mathscr{A} \wedge \mathscr{B})$ is a sentence.

But notice that '$(\mathscr{A} \wedge \mathscr{B})$' is *not* a sentence of TFL. In fact, it isn't even an expression of TFL! After all, it includes cursive letters, and these are not among the symbols that constitute the the lexicon of TFL. Atomic TFL sentences are just ordinary uppercase roman letters (possibly subscripted) like 'A' or 'B' or 'S_{21}', not cursive letters. So what's going on in our recursive clauses?

The answer is that in these clauses, we are using cursive letters as *variables* that "range over" arbitrary expressions of TFL. Consider how, in a math class, you might explain to someone that if m and n are any two positive integers, it holds that $m+n \geq m$. In this case we are using 'm' and 'n' as variables that range over arbitrary positive integers, and saying that $m+n \geq m$ holds no matter what positive integers m and n are. On the other hand, if the variables were allowed to range over *all* integers, positive and negative, the claim $m+n \geq m$ would not longer hold — now m might be 2 and n might be -3, in which case $m+n \not\geq m$.

In the same way, we are here using cursive letters as variables, except that we are using them as variables that range over arbitrary expressions in the language of TFL (rather than over integers, say). The language of TFL has been the object of our study for the past several sections, and we therefore call it the OBJECT LANGUAGE. But we have been conducting our discussion of TFL in English, so English is our METALANGUAGE: it is the language in which we talk about the object language. Of course these notions aren't fixed. If we had been discussing the syntax of Korean, say, rather than TFL, then our metalanguage would still have been English, but our object language would have been Korean.

For this reason, variables like '$\mathscr{A}$' and '$\mathscr{B}$' are called METAVARIABLES: they form part of our metalanguage, English, and they range over arbitrary expressions in our object language, TFL. But again, these metavariables are *only* part of our metalanguage, and not themselves included in the object language TFL.

'$\mathscr{A}$' is a metavariable in our metalanguage that we use to talk about arbitrary expressions of TFL. 'A', by contrast, is a particular atomic sentence of our object language TFL .

So what clause 3 in our definition says is that if $\mathscr{A}$ and $\mathscr{B}$ are any arbitrary sentences of TFL, then the result of writing whatever $\mathscr{A}$ is, followed by '$\wedge$', followed by whatever $\mathscr{B}$ is, and enclosing the result in parentheses, produces a sentence of TFL. For example, if $\mathscr{A}$ is the TFL sentence '$\neg D$' and $\mathscr{B}$ is the TFL sentence '$(S \to T)$', then clause 3 tells us that '$(\neg D \wedge (S \to T))$' is a sentence of TFL. But again, the string of symbols '$(\mathscr{A} \wedge \mathscr{B})$' is not itself a sentence (nor even an expression) of TFL.

■ Exercises 2.8

A. For each of the following: (a) Is it a sentence of TFL, strictly speaking? (b) Is it a sentence of TFL, allowing for our relaxed bracketing conventions?

1. (A)
2. $J_{374} \vee \neg J_{374}$
3. $\neg\neg\neg\neg F$
4. $\neg\neg\neg(\neg F)$
5. $\neg \wedge S$
6. $(G \wedge \neg G)$
7. $(A \to (A \wedge \neg F)) \vee (D \leftrightarrow E)$
8. $(A \wedge (B \wedge ((C \wedge D) \wedge E)))$
9. $[(Z \leftrightarrow S) \to W] \wedge [J \vee X]$
10. $(F \leftrightarrow \neg D \to J) \vee (C \wedge D)$

B. What is the main operator in each of the following? And what are the scopes of the other operators in these sentences?

1. $\neg(A \to \neg(\neg C \leftrightarrow B))$
2. $\neg A \to \neg(\neg C \leftrightarrow B)$
3. $(A \wedge (B \vee ((C \vee D) \wedge E)))$
4. $\big[(H \to I) \vee (I \to H)\big] \wedge (J \vee K)$

C. Are there any sentences of TFL that contain no atomic sentences? Does any TFL sentence contain a greater number of binary connectives than atomic sentences? Explain your answers.

The Semantics of TFL 3

We ended the last chapter by looking at the SYNTAX, or grammar, for the language of TFL. In this chapter, we'll be concerned with the SEMANTICS, or meaning of TFL sentences. More specifically, we're going to look at the meanings of our five TFL connectives, and see how the meaning of a complex TFL sentence is determined by the connectives it contains. Once we've done that, we can use our semantics to give a precise definition of various logical notions, like validity.

3.1 Meanings for TFL Connectives

Negation Let's begin with negation. The meaning of the TFL connective '$\neg$' should roughly resemble that of the English word 'not'. But what does 'not' mean? This might seem like a baffling question. What are meanings, anyhow?

To make the issue more tractable, let's ask a simpler question: if you put 'not' into a sentence, what does that do to the *truth-value* of the sentence? Take a true sentence, like 'Frida Kahlo was a painter'. If you add a 'not' into it, you get the false sentence 'Frida Kahlo was not a painter'. And similarly, if you take a false sentence and negate it, you get a true sentence.

So we can characterize the meaning of the TFL connective '$\neg$' as a mapping between truth values: given something true, it returns something false, and given something false, it returns something true. We'll abbreviate 'True' with 'T' and 'False' with 'F'. We can then represent the meaning of the TFL connective '$\neg$' via the following *characteristic truth table* for negation:

$\mathscr{A}$	$\neg\mathscr{A}$
T	F
F	T

What this says is that for any TFL sentence $\mathscr{A}$: if $\mathscr{A}$ is true, then $\neg\mathscr{A}$ is false, and if $\mathscr{A}$ is false, then $\neg\mathscr{A}$ is true.

Conjunction A similar line of thought goes for conjunction. If you take two true sentences, and put an 'and' between them, the conjunction you've formed is true. On the other hand, if even one of the two conjoined sentences is false, the entire conjunction is false. So the characteristic truth table for conjunction looks like this:

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A} \wedge \mathscr{B})$
T	T	T
T	F	F
F	T	F
F	F	F

Note that conjunction is *symmetrical*. The truth value for $(\mathscr{A} \wedge \mathscr{B})$ is always the same as the truth value for $(\mathscr{B} \wedge \mathscr{A})$.

Disjunction Recall that '$\vee$' represents inclusive or. So, for any sentences $\mathscr{A}$ and $\mathscr{B}$, $(\mathscr{A} \vee \mathscr{B})$ is true iff at least one of $\mathscr{A}$ and $\mathscr{B}$ is true. This gives us the following characteristic truth table for disjunction:

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A} \vee \mathscr{B})$
T	T	T
T	F	T
F	T	T
F	F	F

Like conjunction, disjunction is symmetrical: '$(\mathscr{A} \vee \mathscr{B})$' always has the same truth value as '$(\mathscr{B} \vee \mathscr{A})$'.

As we saw in §2.4, the English construction 'either ... or' is sometimes used to express *exclusive* disjunction, which "excludes" the possibility of both disjuncts' being true. If we liked, we could expand TFL by introducing a new connective $\oplus$ (sometimes also called XOR) with the following characteristic truth table:

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A} \oplus \mathscr{B})$
T	T	F
T	F	T
F	T	T
F	F	F

However, as we discussed, we don't need to (and won't) go this route, since the effect of the exclusive $(\mathscr{A} \oplus \mathscr{B})$ can be achieved using the TFL connectives we already have, via $(\mathscr{A} \vee \mathscr{B}) \wedge \neg(\mathscr{A} \wedge \mathscr{B})$.

Conditional Conditionals are considerably more contentious. In fact, we might as well be up front about it: they are a mess. We'll simply stipulate that, in TFL, $(\mathscr{A} \rightarrow \mathscr{B})$ is false if $\mathscr{A}$ is true and $\mathscr{B}$ is false, and true in *all* other circumstances. This gives us the following characteristic truth table for the conditional:

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A} \rightarrow \mathscr{B})$
T	T	T
T	F	F
F	T	T
F	F	T

Notice that the conditional is *asymmetrical*: $(\mathscr{A} \rightarrow \mathscr{B})$ and $(\mathscr{B} \rightarrow \mathscr{A})$ need not have the same truth value. For example, if $\mathscr{A}$ is true and $\mathscr{B}$ false, then $(\mathscr{A} \rightarrow \mathscr{B})$ is false but $(\mathscr{B} \rightarrow \mathscr{A})$ is true.

You can perhaps already see that this is a controversial way to symbolize English 'if ... then' constructions. The above truth table tells us that any TFL conditional with a false antecedent is true (and similarly, any TFL conditional with a true consequent is true). But it's far from clear that any English conditional whose antecedent turns out to be false is therefore automatically true. This is known as "the paradox of material implication." In the case of conditionals, there in other words isn't just a worry about TFL bypassing certain *subtleties* of meaning, but of missing out on the meaning of the corresponding English expression altogether. We'll look at some of these issues in §3.3 below.

Biconditional Since a biconditional is to be the same as the conjunction of a conditional running in both directions, the characteristic truth table for the biconditional has to be as follows, given our truth table for the conditional:

$\mathscr{A}$	$\mathscr{B}$	$(\mathscr{A} \leftrightarrow \mathscr{B})$
T	T	T
T	F	F
F	T	F
F	F	T

An easy way to remember this is that a biconditional is true when both sides have the *same* truth value, and false when the two sides have different truth values. The biconditional is therefore symmetrical. It's truth table is in effect the opposite of exclusive disjunction. As we'll soon be able to show, this truth table is indeed the same as the one we would get for $(\mathscr{A} \rightarrow \mathscr{B}) \wedge (\mathscr{B} \rightarrow \mathscr{A})$.

3.2 Truth-Functionality

The fact that we can give characteristic truth tables like these for our TFL connectives means that they are *truth-functional*:

> A connective is TRUTH-FUNCTIONAL iff the truth value of a sentence with that connective as its main logical operator is uniquely determined by the truth value(s) of the constituent sentence(s).

Indeed, this is what gives TFL its name: *truth-functional logic*.

Many languages have connectives that are not truth-functional. In English, for example, we can form a new sentence from any simpler sentence by prefixing it with the unary connective 'It is necessarily the case that...'. The truth value of this new sentence is not fixed solely by the truth value of the original sentence. For consider two true sentences:

1. $2+2=4$
2. Shostakovich wrote fifteen string quartets

Whereas it is necessarily the case that $2+2=4$, it is not *necessarily* the case that Shostakovich wrote fifteen string quartets. If he had died earlier or later, he might have written fewer or more quartets than he in fact did. So the English unary connective 'It is necessarily the case that. . . ' is not *truth-functional*. TFL cannot represent non-truth-functional connectives like these; it can only represent truth-functional connectives like e.g. 'It is not the case that . . . ' or '. . . and . . . '.

Since TFL's connectives are all truth-functional, we have to ignore everything except the truth-functional aspects of English when symbolizing English sentences or arguments into TFL. A lot is inevitably lost in the process. There are subtleties to our ordinary claims that far outstrip their mere truth values: sarcasm, poetry, snide implicature, emphasis. These are all important parts of everyday discourse, but none of it is retained in TFL.

For example, as already remarked in in §2.3, TFL cannot capture the subtle differences between the following English sentences:

1. Adam is energetic and Adam is not athletic.
2. Although Adam is energetic, he is not athletic.
3. Despite being energetic, Adam is not athletic.
4. Adam is energetic, albeit not athletic.

They all get symbolized with the same TFL sentence, perhaps '$(E \wedge \neg A)$'. Similarly, in symbolizing 'Adam is energetic' as 'E', we are ignoring all aspects of its meaning except its truth value.

This is why we talk of *symbolizing* English sentences. Some logic textbooks talk about *translating* English sentences into TFL. But a good translation should preserve more than mere truth values and truth-functional aspects of meaning. So we can't really *translate* English into TFL, properly speaking.

3.3 Conditionals in TFL and English

When we introduced the truth table for '$\rightarrow$', we didn't provide any justification for it. In fact, we noticed that it seems problematic as a symbolization of English 'if . . . then'. But there are some things to be said in favor of the truth table we provided.

First, the TFL conditional has some attractive *logical* features given our truth table.[1] The following all seem correct for English 'if . . . then' statements:

- Arguments of the form 'If $\mathscr{A}$ then $\mathscr{B}$; $\mathscr{A}$; therefore $\mathscr{B}$' are valid. (This form of argument is called *modus ponens*.)
- Arguments of the form 'If $\mathscr{A}$ then $\mathscr{B}$; $\mathscr{B}$; therefore $\mathscr{A}$' are not valid. (This is called the fallacy of *affirming the consequent*.)
- Statements of the form 'If $\mathscr{A}$, then $\mathscr{A}$' are necessarily true.

As we will see once we define validity and other logical notions in TFL, the same holds for the corresponding TFL symbolizations: arguments of the form $(\mathscr{A} \rightarrow \mathscr{B}), \mathscr{A} \therefore \mathscr{B}$ are valid in TFL, ones of the form $(\mathscr{A} \rightarrow \mathscr{B}), \mathscr{B} \therefore \mathscr{A}$ are not valid in TFL (at least if $\mathscr{A}$ and $\mathscr{B}$ are atomic sentences), and any TFL sentence of the form $(\mathscr{A} \rightarrow \mathscr{A})$ is a logical necessity (or

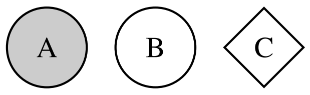
[1] I owe the following observation to Branden Fitelson.

"tautology") in TFL. And importantly, out of the sixteen possible binary truth functions, the one we have assigned to '$\to$' is the *only* one that has all these logical properties! So if we have to pick a truth-functional connective to symbolize English 'if . . . then', then '$\to$' is the best one among the sixteen available.

Second, there's an argument to suggest that the truth table we've given for '$\to$' captures at least certain uses of English 'if . . . then'.[2] Suppose Lara has drawn several shapes on a piece of paper, and colored some of them grey. I have not seen them, but I claim:

If any shape is grey, then it is also circular.

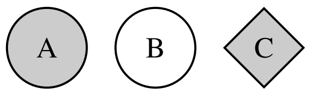
As it happens, Lara has drawn the following:



In this case, my general conditional claim is true. And this in turn means that each of its *instances* must be true:

- If A is grey, then it is circular (true antecedent, true consequent)
- If B is grey, then it is circular (false antecedent, true consequent)
- If C is grey, then it is circular (false antecedent, false consequent)

However, if Lara had drawn the following:



then my claim would have been false, because it would then have had a false instance:

- If C is grey, then it is a circular (true antecedent, false consequent)

Notice that this distribution of truth values exactly matches that in our truth-table for '$\to$'. So this suggests that the truth values of at least some English 'if . . . then' statements match those predicted by our truth table.

At the same time, it's clear that there are other uses of 'if . . . then' in English that aren't adequately symbolized using '$\to$'. Consider the following two sentences:

(1) If Hillary Clinton had won the 2016 US election, then she would have been the first female president of the US.
(2) If Hillary Clinton had won the 2016 US election, then she would have turned into a helium balloon and floated away into the sky.

[2]Versions of this argument are given by Dorothy Edgington (2014), 'Conditionals', in the *Stanford Encyclopedia of Philosophy* (`http://plato.stanford.edu/entries/conditionals/`) and Warren Goldfarb (2003), in his textbook *Deductive Logic*.

Intuitively, sentence 1 is true and sentence 2 is false. But both have false antecedents and false consequents. (Hillary did not win; she did not become the first female president of the US; and she of course did not turn into a helium balloon.) So our truth table would incorrectly count both sentence true.

These are examples of *subjunctive conditionals*, because they are in the subjunctive mood (that is, they involve words like 'had' and 'would'). They ask us to imagine something contrary to fact— a world in which Hillary won the 2016 election — and then ask us to evaluate what *would* have happened in that case. What we've seen is that subjunctive conditionals are not adequately symbolized using '$\rightarrow$'. In fact, we've seen that subjunctive conditionals aren't even *truth functional*! After all, (1) and (2) have antecedents and consequents with the same truth values (all false), but the two conditionals themselves have different truth values. Since subjunctive conditionals aren't truth-functional, there is no hope of symbolizing them in the truth-functional language of TFL.

Still, for the reasons given earlier, '$\rightarrow$' is the best candidate we have for symbolizing at least certain uses of English 'if ... then'. We'll therefore continue to symbolize them this way, while remaining mindful of the simplification involved.

3.4 Complete Truth Tables

We've seen what the characteristic truth tables for the five TFL connectives are. Our next step is to use these truth tables to build up truth tables for complex TFL sentences that contain multiple connectives. To construct a truth table for a complex sentence like '$(H \wedge I) \rightarrow H$' we have to start with truth-values for the atomic sentences, and then calculate the truth value of the complex sentence.

So far, we've used symbolization keys to assign truth values to TFL sentences. For example, we might say that the TFL sentence 'B' is to symbolize 'Big Ben is in London'. Since Big Ben *is* in London, this symbolization would make 'B' true. But we can also assign truth values *directly*. We could simply stipulate that 'B' is to be true, or stipulate that it is to be false. Such stipulations are called *valuations*:

A VALUATION is any assignment of truth values to particular atomic sentences of TFL.

To construct the COMPLETE TRUTH TABLE for a complex TFL sentence, we will have to calculate its truth value on every possible valuation of the atomic sentences it contains.

Let's look at an example. Take the sentence '$(H \wedge I) \rightarrow H$'. There are four possible ways to assign True and False to the atomic sentence 'H' and 'I'—four possible valuations. We can represent these as follows:

H	I	$(H \wedge I) \rightarrow H$
T	T	
T	F	
F	T	
F	F	

To calculate the truth value of the entire sentence '$(H \wedge I) \rightarrow H$', we first copy the truth values for the atomic sentences and write them underneath the letters in the sentence:

H	I	$(H$	$\land$	$I)$	$\to$	H
T	T	T		T		T
T	F	T		F		T
F	T	F		T		F
F	F	F		F		F

Next we have to consider the subsentence '$(H \land I)$'. This is a conjunction, and the characteristic truth table for conjunction tells us that a conjunction is true iff both conjuncts are true. Since 'H' and 'I' are both true on (and only on) the first line of the truth table, the conjunction '$(H \land I)$' is true on the first row of the table and false on the rest:

H	I	$(H$	$\land$	$I)$	$\to$	H
T	T	T	T	T		T
T	F	T	F	F		T
F	T	F	F	T		F
F	F	F	F	F		F

Notice how we've recorded the truth value for the subsentence '$(H \land I)$' on each row underneath its main operator, '$\land$'.

Now, our TFL sentence as a whole is a conditional, $\mathscr{A} \to \mathscr{B}$, with '$(H \land I)$' as $\mathscr{A}$ and with 'H' as $\mathscr{B}$. So to determine the truth-value of the whole conditional, we have to look at the truth values of '$(H \land I)$' and 'H' on each row. On the second row, for example, '$(H \land I)$' is false and 'H' is true. Since a conditional is true when the antecedent is false, we write a 'T' in the second row underneath the conditional symbol. We continue for the other three rows and get this:

H	I	$(H$	$\land$	$I)$	$\to$	H
T	T		T		T	T
T	F		F		T	T
F	T		F		T	F
F	F		F		T	F

The conditional is the main logical operator of this sentence. The column of 'T's underneath '$\to$' therefore tells us the truth value for the sentence as a whole on each of the four possible valuations of its atomic constituents 'H' and 'I'. What the table shows is that '$(H \land I) \to H$' is true regardless of the truth values of 'H' and 'I'. They can be true or false in any combination, and the complex sentence still comes out true. Since we have considered all four possible valuations, this means that '$(H \land I) \to H$' is true on *every* valuation.

In this example, I erased some 'T's and 'F's as we went along to make things more readable. When actually writing truth tables on paper, however, it is impractical to erase whole columns or rewrite the whole table for every step. Although it is more crowded, the complete truth table with no columns erased looks like this:

H	I	$(H$	$\land$	$I)$	$\to$	H
T	T	T	T	T	**T**	T
T	F	T	F	F	**T**	T
F	T	F	F	T	**T**	F
F	F	F	F	F	**T**	F

Most of the columns underneath the sentence are only there for bookkeeping purposes. The column that matters most is the column underneath the *main logical operator* for the sentence, since this tells you the truth value of the entire sentence. I have emphasized this column by putting it in bold. When you work through truth tables yourself, you should similarly emphasize the column under the main operator (perhaps by highlighting it or circling it).

As you can see from this example, a complete truth table has a row for every possible assignment of True and False to the relevant atomic sentences. Each of these rows represents a *valuation*. The number of rows depends on the number of different atomic sentences involved. A sentence that contains only one atomic sentence requires only two rows, as in the characteristic truth table for negation. This is true even if the same letter is repeated many times, as in the sentence '$[(C \leftrightarrow C) \rightarrow C] \wedge \neg(C \rightarrow C)$'. The complete truth table requires only two row because there are only two possibilities: 'C' can be true or it can be false. The truth table for this sentence looks like this:

C	$[(C$	$\leftrightarrow$	$C)$	$\rightarrow$	$C]$	$\wedge$	$\neg$	$(C$	$\rightarrow$	$C)$
T	T	T	T	T	T	**F**	F	T	T	T
F	F	T	F	F	F	**F**	F	F	T	F

Looking at the column underneath the main logical operator, we see that the sentence is false on both rows of the table; i.e., the sentence is false regardless of whether 'C' is true or false. It is false on every valuation.

There will be four rows in the complete truth table for a sentence containing two atomic sentences, like '$(H \wedge I) \rightarrow H$'. And there will be eight rows in the complete truth table for a sentence containing three atomic sentences, e.g.:

M	N	P	M	$\wedge$	$(N$	$\vee$	$P)$
T	T	T	T	**T**	T	T	T
T	T	F	T	**T**	T	T	F
T	F	T	T	**T**	F	T	T
T	F	F	T	**F**	F	F	F
F	T	T	F	**F**	T	T	T
F	T	F	F	**F**	T	T	F
F	F	T	F	**F**	F	T	T
F	F	F	F	**F**	F	F	F

So truth tables grow quickly! Four atomic sentences would require 16 rows, five atomic sentences require 32 rows, six atomic sentences require 64 rows, and so on.

A complete truth table for a sentence with n atomic sentences must have 2^n rows, representing the 2^n possible valuations.

In order to fill in the columns under the atomic sentences, begin with the right-most atomic sentence ('P' in the table above) and alternate between 'T' and 'F'. In the next column to the left (the one under 'N' in our table), write two 'T's followed by two 'F's, and repeat. For the third atomic sentence ('M' in our table), write four 'T's followed by four 'F's. This yields an eight line truth table like the one above. For a 16 line truth table, the next column of atomic sentences should have eight 'T's followed by eight 'F's. For a 32 line table, the next column would have 16 ‘T’s followed by 16 ‘F’s. And so on. In general, you should construct your truth tables according to the following rules:

1. Write down the complex sentence you are working with, and to its left list the atomic sentences it contains *in alphabetical order*.
2. Determine how many rows your table will require given how many atomic sentences are involved. Again, for n atomic sentences you need 2^n rows.
3. Fill in the truth values for each atomic sentence according to the pattern described above. The column under the right-most atomic sentence will follow the pattern T F T F T F, the next column to the left will have the pattern T T F F T T F F, the column to the left of that the pattern T T T T F F F F, and so on.
4. Then calculate the truth value for the complex sentence as a whole on every row of the truth table (i.e. on every possible valuation). When you’re done, remember to highlight or circle the column under the sentences’s main logical operator.

These rules give us a canonical format for truth tables, which makes it easier to compare (and grade) truth tables written by different people.

■ Exercises 3.4

A. Construct complete truth tables in canonical format (i.e. by following the rules we gave) for each of the following:

1. $A \rightarrow A$
2. $C \rightarrow \neg C$
3. $(A \leftrightarrow B) \leftrightarrow \neg(A \leftrightarrow \neg B)$
4. $(A \rightarrow B) \vee (B \rightarrow A)$
5. $(A \wedge B) \rightarrow (B \vee A)$
6. $\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B)$
7. $\big[(A \wedge B) \wedge \neg(A \wedge B)\big] \wedge C$
8. $[(A \wedge B) \wedge C] \rightarrow B$
9. $\neg\big[(C \vee A) \vee B\big]$

B. Show that ‘$((A \vee B) \wedge \neg(A \wedge B))$’ has a truth table that matches that for exclusive disjunction given in §3.1 above.

If you want additional practice, you can construct truth tables for any of the sentences and arguments in the exercises for the previous chapter.

3.5 Semantic Concepts

Now that we know how to construct complete truth tables for complex sentences, we’ll introduce some semantic concepts and see how to use truth tables to test whether they apply. In §1.2, we looked at the notions of *necessary truth* and *necessary falsity*. Both notions have surrogates in TFL. We’ll start with a surrogate for necessary truth.

A sentence $\mathscr{A}$ is a TF TAUTOLOGY iff it is true on every valuation.

That is, a TFL sentence is a TF tautology if it is true on every row of its complete truth table, since rows represent valuations. If you look back at the truth table for '$(H \wedge I) \rightarrow H$' from §3.4 you'll see that it's a tautology. Other tautologies include sentences of the form $(\mathscr{A} \rightarrow \mathscr{A})$ and ones of the form $(\mathscr{A} \vee \neg\mathscr{A})$; the latter is called the LAW OF EXCLUDED MIDDLE.

Notice that this is only a *surrogate* for necessary truth. There are some necessary truths that we cannot adequately symbolize in TFL. For example, '$2+2=4$' and 'Every city either is or is not in France' are both necessary truths, but if if we symbolize them in TFL, the best we can offer is an atomic sentence, and no atomic sentence is a tautology. Still, if we can adequately symbolize some English sentence using a TFL sentence which is a tautology, then that English sentence expresses a necessary truth.

We have a similar surrogate for necessary falsity:

A sentence $\mathscr{A}$ is a TF CONTRADICTION iff it is false on every valuation.

A sentence is a TF contradiction if it is false on every line of its complete truth table. The truth table for '$[(C \leftrightarrow C) \rightarrow C] \wedge \neg(C \rightarrow C)$' we constructed in §3.4 shows that this sentence is a contradiction. Sentences of the form $(\mathscr{A} \wedge \neg\mathscr{A})$ or $\neg(\mathscr{A} \rightarrow \mathscr{A})$ are other examples of contradictions. Notice that the negation of any tautology is a contradiction. Lastly, we have a surrogate for contingency: a sentence $\mathscr{A}$ is TF CONTINGENT iff it is neither a tautology nor a contradiction, i.e. if it is true on at least one valuation and false on at least one valuation.

These notions all apply to *single* sentences of TFL. Another useful notion—which we've occasionally already made use of—is that of *equivalence*. This is a property that applies to *pairs* of sentences of TFL:

$\mathscr{A}$ and $\mathscr{B}$ are TF EQUIVALENT iff they have the same truth value on every valuation.

Notice that by this definition, any two tautologies, and any two contradictions, are equivalent. Equivalence is more interesting when we find pairs of contingent sentences that are equivalent. For example, we've observed that a biconditional like '$(A \leftrightarrow B)$' is equivalent to a conjunction of two conditionals, '$((A \rightarrow B) \wedge (B \rightarrow A))$'. Similarly, since $(\mathscr{A} \rightarrow \mathscr{B})$ is true if $\mathscr{A}$ is true or if $\mathscr{B}$ is false (and false otherwise), TFL conditionals are equivalent to disjunctions of the form $(\neg\mathscr{A} \vee \mathscr{B})$. DEMORGAN'S LAWS, which we looked at in 2.4, are another example of equivalences.

Again, we can test for TF equivalence using truth tables. Consider the sentences '$\neg(P \vee Q)$' and '$\neg P \wedge \neg Q$'. To find out whether they're TF equivalent, we construct JOINT TRUTH TABLE that includes both sentences at once:

P	Q	$\neg\,(P \vee Q)$	$\neg P \wedge \neg Q$
T	T	**F** T T T	F T **F** F T
T	F	**F** T T F	F T **F** T F
F	T	**F** F T T	T F **F** F T
F	F	**T** F F F	T F **T** T F

Joint truth tables like these are constructed by listing to the left (in alphabetical order) all the atomic sentences that occur in *any* of the sentences being compared. We then include two separate sections in the table, one for each sentence, and calculate the truth value of each sentence in every row, i.e. on every valuation.

Looking at the columns under the main logical operators of the two sentences in the table above (negation for the first sentence, conjunction for the second) we see that both are false on the first three rows, and both are true on the last row. Since they match on every row, they have the same truth value on every valuation, and are therefore TF equivalent. This pair of sentences is an instance of one of DeMorgan's Laws, so the table shows that the law does indeed hold in TFL.

Another notion that applies to pairs of sentences is the following:

> $\mathscr{A}$ and $\mathscr{B}$ are TF CONTRADICTORY iff they have opposite truth values on every valuation.

We can again test for this property by drawing a joint truth table for the two sentences to be compared, and checking to see that the truth values listed underneath their main connectives are different in every row of the table. It's important not to confuse the notion of two sentences being *contradictory* with the notion of a sentence's being a *contradiction*: the former is a property of pairs of sentences, the latter a property of a single sentence. But the two notions are connected (as the names suggest): if $\mathscr{A}$ and $\mathscr{B}$ are contradictory, then their conjunction $(\mathscr{A} \wedge \mathscr{B})$ is a contradiction.

Next, here is a notion that now applies to arbitrarily large *collections* of sentences:

> $\mathscr{A}_1, \ldots, \mathscr{A}_n$ are JOINTLY TF CONSISTENT iff there is at least one valuation which makes them all true.

Collections of sentences are also said to be JOINTLY TF INCONSISTENT iff they are not TF consistent, i.e. iff there is no valuation on which they are all true. Again, it is easy to test for joint TF consistency (and inconsistency) using joint truth tables:

P	Q	P	$\wedge$	Q	P	$\vee$	Q	P	$\to$	Q
T	T	T	**T**	T	T	**T**	T	T	**T**	T
T	F	T	**F**	F	T	**T**	F	T	**F**	F
F	T	F	**F**	T	F	**T**	T	F	**T**	T
F	F	F	**F**	F	F	**F**	F	F	**T**	F

We can see from this that '$P \wedge Q$', '$P \vee Q$', and '$P \to Q$' are consistent, because there is a row in their joint table on which they are all true, namely the first.

The following notion is closely related to that of joint consistency:

> $\mathscr{A}_1, \ldots, \mathscr{A}_n$ TF ENTAIL $\mathscr{C}$ iff no valuation makes all of $\mathscr{A}_1, \ldots, \mathscr{A}_n$ true and $\mathscr{C}$ false.

Again, we can test for TF entailment with a joint truth table. To test whether '$\neg L \to (J \vee L)$' and '$\neg L$' TF entail 'J' we construct the following joint truth table:

J	L	$\neg L \rightarrow (J \vee L)$	$\neg L$	J
T	T	F T **T** T T T	**F** T	**T**
T	F	T F **T** T T F	**T** F	**T**
F	T	F T **T** F T T	**F** T	**F**
F	F	T F **F** F F F	**T** F	**F**

The only row on which both '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' are true is the second row. But that is a row on which 'J' is also true! So there is no valuation that makes both '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' true and '$J$' false, meaning that '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' TF entail '$J$'.

The notion of TF entailment is particularly important because it functions as our surrogate for *validity*. In §1.1 we said that an argument is TF valid iff it's impossible for the premises to be true and the conclusion false. Similarly, we now say that a TFL argument is valid iff there's no valuation on which the premises are true and the conclusion false, that is to say:

An argument $\mathscr{A}_1, \ldots, \mathscr{A}_n \therefore \mathscr{C}$ is TF VALID iff the premises $\mathscr{A}_1, \ldots, \mathscr{A}_n$ TF entail the conclusion $\mathscr{C}$.

The connection to the intuitive notion of validity from §1.1 is that if there's no valuation that makes the premises of an argument true but it's conclusion false, then it's not possible for its premises to be true but its conclusion false. So TFL gives us a way to test for the validity of English arguments! First, we symbolize them in TFL. Then we test for TF validity using truth tables. If the symbolized argument is TF valid, the English argument is also valid.

The Limits of Our Tests

This is an important milestone: a test for the validity of arguments! But we shouldn't get carried away. It is important to understand the *limits* of our achievement. We can illustrate these limits with a few examples. First, consider the argument:

1. Daisy has four legs. ∴ Daisy has more than two legs.

To symbolize this argument in TFL, we would have to use two different atomic sentences — perhaps 'F' and 'T' — for the premise and the conclusion respectively. Obviously 'F' does not TF entail 'T', and the argument $F \therefore T$ is therefore not TF valid. And yet the English argument is valid, i.e. it's impossible for the premise to be true and the conclusion false!

This case perhaps isn't so bad. As we discussed in §1.4, logic only aims to identify arguments that are *formally* valid, that is, valid in virtue of their logical structure. And the above argument isn't formally valid. But now consider this argument:

2. Some parallelograms are squares. All squares are equilateral. ∴ Some parallelograms are equilateral.

To symbolize this in TFL, we'd again have to use different atomic sentences for the premises and conclusion, something like:

$$P, S \therefore E$$

Again this argument is not TF valid. And yet the English argument is valid, and indeed formally valid this time! So here TFL really does fail us. To capture the structure in virtue of which this argument is valid, we need a stronger system of logic, namely FOL, which we'll look at in the second part of this book.

Similar shortcomings beset our other truth table tests. Consider the sentence:

3. Jan is neither bald nor not-bald.

We could symbolize this in TFL as '$\neg(J \vee \neg J)$', or (given DeMorgan's Laws) as '$\neg J \wedge \neg\neg J$'. If we constructed a truth-table for this, we'd see that it's a TF contradiction. But the English sentence (3) does not seem like a contradiction: maybe Jan is on the borderline between being bald and not-bald, and (3) is in fact true! So from the fact that an English sentence receives a contradictory symbolization in TFL, we can't always conclude that the English sentence is necessarily false.

Lastly, consider the following sentence:

4. It's not the case that, if God exists, then God answers malevolent prayers.

Symbolising this in TFL, we would offer something like '$\neg(G \rightarrow M)$'. Now, '$\neg(G \rightarrow M)$' is TF equivalent to '$G \wedge \neg M$' and therefore TF entails 'G' (check this with a truth table). So if we symbolize the English sentence (4) in TFL, it seems to entail that God exists. But that's strange: surely even the atheist can accept (4), and not thereby commit herself to the existence of God!

In different ways, all of these examples highlight some of the limits of working with a language like TFL that can *only* handle truth-functional connectives. These limits give rise to some interesting questions in the field of *philosophical logic*. Our first two arguments raise questions about the distinction between validity and formal validity, and how these are related to our surrogate notion in TFL. The case of Jan's baldness raises the question of what logic we should use when dealing with *vague* language. And the case of the atheist raises the question of how to deal with the *paradoxes of material implication*, which arise from differences between English 'if ... then' constructions and the TFL conditional '$\rightarrow$' (see §3.3). Part of the purpose of this course is to equip you with the tools needed to explore these questions in philosophical logic. But we have to walk before we can run. We have to become proficient in using TFL before we can adequately discuss its limits, and consider alternatives.

3.6 The Double-Turnstile Notation

Because TF entailment is such an important concept, we'll introduce some new notation in connection with it. Rather than saying that the sentences $\mathcal{A}_1, \ldots, \mathcal{A}_n$ TF entail $\mathcal{C}$, we will abbreviate this by writing:

$$\mathcal{A}_1, \ldots, \mathcal{A}_n \vDash \mathcal{C}$$

The symbol '$\vDash$' is known as *the double-turnstile*, since it looks like a turnstile with two horizontal beams.

There is no limit on the number of TFL sentences that can be mentioned before the symbol '$\vDash$'. Indeed, we can even consider the limiting case:

$$\vDash \mathcal{C}$$

This says that there is no valuation which makes all the sentences mentioned on the left side of '$\vDash$' true while making $\mathscr{C}$ false. Since *no* sentences are mentioned on the left side of '$\vDash$' in this case, this just means that there is no valuation which makes $\mathscr{C}$ false. But that just means that $\mathscr{C}$ is a tautology! So writing $\vDash \mathscr{C}$ gives us a short way to say that $\mathscr{C}$ is a tautology.

Sometimes we will want to deny that a TF entailment holds. We will write:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n \nvDash \mathscr{C}$$

to say that $\mathscr{A}_1, \ldots, \mathscr{A}_n$ *do not* TF entail $\mathscr{C}$, i.e. to say that there *does* exist a valuation that makes all of $\mathscr{A}_1, \ldots, \mathscr{A}_n$ true but $\mathscr{C}$ false. Similarly $\nvDash \mathscr{C}$ means that $\mathscr{C}$ is *not* a tautology, i.e. that there exists a valuation that makes $\mathscr{C}$ false.

Lastly, we can abbreviate the claim that $\mathscr{A}$ and $\mathscr{B}$ are equivalent as follows:

$$\mathscr{A} \Dashv\vDash \mathscr{B}$$

This encapsulates the idea that $\mathscr{A}$ and $\mathscr{B}$ are equivalent iff they mutually entail each other. That makes sense: if $\mathscr{A} \vDash \mathscr{B}$ then there is no valuation that makes $\mathscr{A}$ true but $\mathscr{B}$ false, and if $\mathscr{B} \vDash \mathscr{A}$ then there is no valuation that makes $\mathscr{B}$ true but $\mathscr{A}$ false. Putting it together, this means that there's no valuation on which $\mathscr{A}$ and $\mathscr{B}$ have different truth values, meaning that $\mathscr{A}$ and $\mathscr{B}$ are equivalent. Similarly reasoning holds the other direction, from equivalence to mutual entailment

It's important to be clear that '$\vDash$' is not a symbol in the language of TFL. Rather, it is a symbol of our metalanguage (recall the difference between object language and metalanguage from §2.8). The following is a claim in our metalanguage, not in the language TFL:

- $P, P \rightarrow Q \vDash Q$

This is just shorthand for the following metalinguistic claim:

- There is no valuation that makes 'P' and '$P \rightarrow Q$' true but 'Q' false

For this reason it is also important not to confuse the symbols '$\rightarrow$' and '$\vDash$'. The conditional '$\rightarrow$' is a symbol in our object language, TFL. The TFL sentence '$P \rightarrow Q$' just says that it's not the case that 'P' is true and 'Q' is false. By contrast '$P \vDash Q$' is a sentence in the metalanguage. It doesn't just claim that it is not the case that 'P' is true and 'Q' is false, but makes the stronger — and as it happens false — claim that *there exists no valuation at all* that makes 'P' true and 'Q' false.

Despite this important difference, there are some close connections between conditionals in TFL and claims about entailment in the metalanguage. Observe the following:

- $\mathscr{A} \vDash \mathscr{C}$ iff no valuation makes $\mathscr{A}$ true and $\mathscr{C}$ false.
- $\mathscr{A} \rightarrow \mathscr{C}$ is a tautology iff no valuation makes $\mathscr{A} \rightarrow \mathscr{C}$ false. Since a conditional is only false if its antecedent is true and its consequent false, this means that $\mathscr{A} \rightarrow \mathscr{C}$ is a tautology iff no valuation makes $\mathscr{A}$ true and $\mathscr{C}$ false.

Combining these two observations, we see that:

$$\mathscr{A} \vDash \mathscr{C} \text{ iff } \vDash \mathscr{A} \rightarrow \mathscr{C}$$

(recall that $\vDash \mathscr{A} \rightarrow \mathscr{C}$ abbreviates the claim that $\mathscr{A} \rightarrow \mathscr{C}$ is a tautology). This means that if $\mathscr{A} \vDash \mathscr{C}$ holds, then the corresponding conditional $\mathscr{A} \rightarrow \mathscr{C}$ is a tautology, and must therefore be true. However, the mere truth of the conditional $\mathscr{A} \rightarrow \mathscr{C}$ does not suffice for it to be the case that $\mathscr{A} \vDash \mathscr{C}$. For the latter to hold, '$\mathscr{A} \rightarrow \mathscr{C}$' doesn't just have to be *true*, it has to be a *tautology*. More generally, we have:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n, \mathscr{B} \vDash \mathscr{C} \text{ iff } \mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{B} \rightarrow \mathscr{C}$$

■ Exercises 3.6

A. Revisit your answers to §3.4**A**. Determine which sentences were tautologies, which were contradictions, and which were neither tautologies nor contradictions.

B. Construct joint truth tables to determine whether the following hold:

1. $(A \leftrightarrow B) \Dashv\vDash ((A \rightarrow B) \wedge (B \rightarrow A))$
2. $(A \rightarrow B) \Dashv\vDash (B \rightarrow A)$
3. $(A \rightarrow B) \Dashv\vDash \neg A \vee B$
4. $((A \wedge B) \wedge C) \Dashv\vDash (A \wedge (B \wedge C))$ (That is: is $\wedge$ is associative?)
5. $((A \vee B) \vee C) \Dashv\vDash (A \vee (B \vee C))$ (That is: is $\vee$ is associative).
6. $((A \rightarrow B) \rightarrow C) \Dashv\vDash (A \rightarrow (B \rightarrow C))$ (That is: is $\rightarrow$ associative?)
7. $((A \leftrightarrow B) \leftrightarrow C) \Dashv\vDash (A \leftrightarrow (B \leftrightarrow C))$ (That is: is $\leftrightarrow$ is associative?)

C. Use joint truth tables to determine whether these sentences are jointly consistent, or jointly inconsistent:

1. $A \rightarrow A$, $\neg A \rightarrow \neg A$, $A \wedge A$, $A \vee A$
2. $A \vee B, A \rightarrow C, B \rightarrow C$
3. $B \wedge (C \vee A), A \rightarrow B, \neg(B \vee C)$
4. $A \leftrightarrow (B \vee C), C \rightarrow \neg A, A \rightarrow \neg B$

D. Use joint truth tables to determine if the following entailments hold:

1. $A \rightarrow A \vDash A$
2. $A \rightarrow (A \wedge \neg A) \vDash \neg A$
3. $A \vee (B \rightarrow A) \vDash \neg A \rightarrow \neg B$
4. $A \vee B, B \vee C, \neg A \vDash B \wedge C$
5. $(B \wedge A) \rightarrow C, (C \wedge A) \rightarrow B \vDash (C \wedge B) \rightarrow A$

E. Answer each of the questions below and justify your answer.

1. Suppose that $\mathscr{A} \Dashv\vDash \mathscr{B}$. Is it the case that $\vDash \mathscr{A} \leftrightarrow \mathscr{B}$?
2. Suppose that $\vDash \mathscr{A} \leftrightarrow \mathscr{B}$. Is it the case that $\mathscr{A} \Dashv\vDash \mathscr{B}$?
3. Suppose that $(\mathscr{A} \wedge \mathscr{B}) \rightarrow \mathscr{C}$ is neither a tautology nor a contradiction. Is it the case that $\mathscr{A}, \mathscr{B} \vDash \mathscr{C}$?
4. Suppose that $\mathscr{A}$, $\mathscr{B}$ and $\mathscr{C}$ are jointly inconsistent. What can you say about $(\mathscr{A} \wedge \mathscr{B} \wedge \mathscr{C})$?
5. Suppose $\mathscr{A}$ and $\mathscr{B}$ are joint inconsistent. Must $\mathscr{A}$ and $\mathscr{B}$ be contradictory?
6. Suppose that $\mathscr{A}$ is a contradiction. Is it the case that $\mathscr{A}, \mathscr{B} \vDash \mathscr{C}$?

7. Suppose that $\mathscr{C}$ is a contradiction. Is it the case that $\mathscr{A}, \mathscr{B} \vDash \mathscr{C}$?
8. Suppose that $\mathscr{C}$ is a tautology. Is it the case that $\mathscr{A}, \mathscr{B} \vDash \mathscr{C}$?
9. Suppose that $\mathscr{A}$ is a tautology. Is it the case that $\mathscr{A}, \mathscr{B} \vDash \mathscr{C}$?
10. Suppose that $\mathscr{A}$ and $\mathscr{B}$ are equivalent. What can you say about $(\mathscr{A} \vee \mathscr{B})$?
11. Suppose that $\mathscr{A}$ and $\mathscr{B}$ are *not* equivalent. What can you say about $(\mathscr{A} \vee \mathscr{B})$?
12. Suppose $\mathscr{A}$ and $\mathscr{B}$ are equivalent. What can you say about $(\mathscr{A} \to \mathscr{B})$?
13. Suppose $\mathscr{A}$ and $\mathscr{B}$ are contradictory. Must $(\mathscr{A} \to \mathscr{B})$ be a contradiction?
14. Suppose $\mathscr{A}$ is a contradiction. What can you say about $(\mathscr{A} \to \mathscr{B})$?
15. Suppose $\mathscr{B}$ is a contradiction. What can you say about $(\mathscr{A} \to \mathscr{B})$?

F. Consider the following principle:

- Suppose $\mathscr{A}$ and $\mathscr{B}$ are equivalent. Then for given argument that contains $\mathscr{A}$ (either as a premise or as its conclusion), replacing $\mathscr{A}$ with $\mathscr{B}$ will not affect that argument's validity.

Is this principle correct? Explain your answer.

3.7 Truth Table Shortcuts

As you become better at constructing truth tables, you will quickly notice that you can use shortcuts to lighten your work. For example, you know for sure that a disjunction is true whenever one of the disjuncts is true. So once you find one true disjunct, there is no need to work out the truth values of the other disjuncts. Thus you might offer:

P	Q	$(\neg P$	$\vee$	$\neg Q)$	$\vee$	$\neg P$
T	T	F	F	F	**F**	F
T	F	F	T	T	**T**	F
F	T				**T**	T
F	F				**T**	T

We don't need to know what the truth value of '$(\neg P \vee \neg Q)$' is on the third and fourth row, because we already know that '$\neg P$' is true on these rows, meaning that sentence as a whole must be true too. What we ultimately care about is the column under the main connective, so you only need to do as much work as is needed to determine the truth-value under this connective.

Similarly, you know for sure that a conjunction is false whenever one of the conjuncts is false. So if you find one false conjunct, there is no need to work out the truth value of the other conjunct. Thus you might offer:

P	Q	$\neg$	$(P \wedge$	$\neg Q)$	$\wedge$	$\neg P$
T	T				**F**	F
T	F				**F**	F
F	T	T	F		**T**	T
F	F	T	F		**T**	T

There's no need to look at the truth value of '$\neg(P \wedge \neg Q)$ on the first and second row since we already know that the second conjunct '$\neg P$' is false on these rows.

A similar short cut is available for conditionals. You immediately know that a conditional is true if either its consequent is true, or its antecedent is false. Thus you might present:

P	Q	$((P\to Q)\to P)\to P$		
T	T			**T**
T	F			**T**
F	T	T	F	**T**
F	F	T	F	**T**

So '$((P \to Q) \to P) \to P$' is a tautology. In fact, it is an instance of *Peirce's Law*, named after Charles Sanders Peirce.

■ Exercises 3.7

A. Using shortcuts, check whether each sentence is a tautology (true on every row), a contradiction (false on every row), or contingent (true on at least one row, false on at least one).

1. $\neg B \wedge B$
2. $\neg D \vee D$
3. $(A \wedge B) \vee (B \wedge A)$
4. $\neg[A \to (B \to A)]$
5. $A \leftrightarrow [A \to (B \wedge \neg B)]$
6. $\neg(A \wedge B) \leftrightarrow A$
7. $A \to (B \vee C)$
8. $(A \wedge \neg A) \to (B \vee C)$
9. $(B \wedge D) \leftrightarrow [A \leftrightarrow (A \vee C)]$

Joint Truth Table Shortcuts

In §3.5, we saw how to use truth tables to test for TF entailment, or validity. To apply that test we look for "bad" lines in the joint truth table: lines where the premises are all true and the conclusion is false. A line like this is said to be a COUNTEREXAMPLE to the entailment. Now:

- If the conclusion is true on a line, then that line is not a counterexample. (And we don't need to evaluate *anything else* on that line to confirm this.)
- If any premise is false on a line, then that line is not a counterexample. (And we don't need to evaluate *anything else* on that line to confirm this.)

With this in mind, we can speed up our tests for validity considerably. Consider how we might test the following argument for validity:

$$\neg L \to (J \vee L), \neg L \therefore J$$

The *first* thing we should do is evaluate the conclusion. If we find that the conclusion is *true* on some row, then that row is not a counterexample, and we can ignore it. In this case that leaves us with only the third and fourth rows to consider:

J	L	$\neg L \to (J \vee L)$	$\neg L$	J
T	T			T
T	F			T
F	T	?	?	F
F	F	?	?	F

with the question-marks indicating where we need to keep digging. The easiest premise to evaluate is the second, so we do that next. Filling in rows three and four for it gives us:

J	L	$\neg L \to (J \vee L)$	$\neg L$	J
T	T			T
T	F			T
F	T		F	F
F	F	?	T	F

Since '$\neg L$' is false on row three, that row is certainly not a counterexample and we can ignore it. So this leaves us with only row four to consider. Filling it in gives us:

J	L	$\neg L \to (J \vee L)$	$\neg L$	J
T	T			T
T	F			T
F	T		F	F
F	F	T **F** F	T	F

The truth table has no counterexample rows, so the argument is valid. Any valuation which makes the conclusion false also makes at least one premise false.

Here's another example. Suppose we want to test:

$$A \vee B, \neg(B \wedge C) \therefore (A \vee \neg C)$$

Again, we start by evaluating the conclusion. Since this is a disjunction, it is true whenever either disjunct is true, and we can speed things up a bit:

A	B	C	$A \vee B$	$\neg(B \wedge C)$	$(A \vee \neg C)$
T	T	T			**T**
T	T	F			**T**
T	F	T			**T**
T	F	F			**T**
F	T	T	?	?	**F** F
F	T	F			**T** T
F	F	T	?	?	**F** F
F	F	F			**T** T

Since the conclusion is only false on rows five and six, these are the only two rows that we need to consider further. Evaluating the premises we get:

A	B	C	$A \vee B$	$\neg(B \wedge C)$	$(A \vee \neg C)$
T	T	T			**T**
T	T	F			**T**
T	F	T			**T**
T	F	F			**T**
F	T	T	**T**	**F** T	**F** F
F	T	F			**T** T
F	F	T	**F**		**F** F
F	F	F			**T** T

So no row is a counterexample, and the entailment holds!

3.8 Partial Truth Tables

Using the shortcuts like these can save a lot of work. But we can get even more efficient. Recall that truth tables grow exponentially: to test an argument involving n atomic sentences, we have to consider a joint truth table with 2^n rows. So if an argument involves 5 atomic sentences, for example, that would mean setting up a 32 row table!

We can be more efficient by using the method of *constructing partial truth tables*. To show that an entailment fails, it suffices to find a single counterexample, i.e. a single valuation that makes all the premises true and the conclusion false. So rather than to set up a complete joint truth table and determine whether any row meets this condition, it is often quicker to try and actively construct a truth table row that does the trick. There are two possible outcomes:

- ▷ We might succeed in constructing a counterexample row. We can then conclude that the entailment fails and the argument is *invalid*.
- ▷ We might discover that it's *impossible* to construct a counterexample. In this case, we can conclude that the entailment holds and the argument is *valid*.

Example 1 Suppose we have to test whether the following is valid:

$$P \leftrightarrow \neg R, (P \vee Q) \rightarrow \neg S \mathrel{\therefore} P \rightarrow (S \vee Q)$$

Rather than set up a sixteen row joint truth table, we'll see if we can "reverse engineer" an assignment of truth values to the atomic sentences 'P', 'Q', 'R', and 'S' that makes both premises true and the conclusion false. That is, we want to know whether there's a way to fill in truth-values for atomic sentences so as to get a truth-table row that looks as follows:

P Q R S	$P \leftrightarrow \neg R$	$(P \vee Q) \rightarrow \neg S$	$P \rightarrow (S \vee Q)$
? ? ? ?	**T**	**T**	**F**

Let's begin with the conclusion. To make '$P \rightarrow (S \vee Q)$' false, we have to make 'P' true and '$(S \vee Q)$' false, which means making both 'S' and 'Q' false:

P Q R S	$P \leftrightarrow \neg R$	$(P \vee Q) \rightarrow \neg S$	$P \rightarrow (S \vee Q)$
T F ? F	**T**	**T**	T **F** F F F

Next, given that 'P' is true, in order to make '$P \leftrightarrow \neg R$' true we have to make $\neg R$ true as well, meaning 'R' has to be false:

P	Q	R	S	P	$\leftrightarrow$	$\neg$	R	(	P	$\vee$	Q	)	$\rightarrow$	$\neg$	S	P	$\rightarrow$	(	S	$\vee$	Q	)
T	F	F	F	T	**T**	T	F						**T**			T	**F**		F	F	F	

At this point we have a valuation that covers all the atomic sentences. And we know that it makes the conclusion false and the first premise true. To make sure that our valuation really constitutes a counterexample to the entailment, we have be sure that it makes the second premise true as well. And it does:

P	Q	R	S	P	$\leftrightarrow$	$\neg$	R	(	P	$\vee$	Q	)	$\rightarrow$	$\neg$	S	P	$\rightarrow$	(	S	$\vee$	Q	)
T	F	F	F	T	**T**	T	F		T	T	F		**T**	T	F	T	**F**		F	F	F	

Since the valuation we've constructed succeeds as a counterexample, we can conclude that the entailment does not hold and that the argument is not TF valid. Constructing this partial truth table was a lot quicker than calculating a 16 row table!

Example 2 For another example, consider following TFL argument:

$$A \rightarrow (D \wedge C), B \leftrightarrow \neg D \therefore A \rightarrow (\neg B \wedge C)$$

Again, to test if the premises entail the conclusion, we have to determine whether there is a truth-table row that looks like this:

A	B	C	D	A	$\rightarrow$	(	D	$\wedge$	C	)	B	$\leftrightarrow$	$\neg$	D	A	$\rightarrow$	(	$\neg$	B	$\wedge$	C	)
?	?	?	?		**T**							**T**				**F**						

We begin with the conclusion: to make '$A \rightarrow (\neg B \wedge C)$' false, we have to make 'A' true and '$(\neg B \wedge C)$' false. We could make '$(\neg B \wedge C)$' false by either making '$\neg B$' false or making 'C' false. We don't know which yet, so all we have at this stage is:

A	B	C	D	A	$\rightarrow$	(	D	$\wedge$	C	)	B	$\leftrightarrow$	$\neg$	D	A	$\rightarrow$	(	$\neg$	B	$\wedge$	C	)
T	?	?	?		**T**							**T**			T	**F**				F		

However, since we're trying to make the first premise '$A \rightarrow (D \wedge C)$' true, and we've made '$A$' true, we have to make both 'D' and 'C' true:

A	B	C	D	A	$\rightarrow$	(	D	$\wedge$	C	)	B	$\leftrightarrow$	$\neg$	D	A	$\rightarrow$	(	$\neg$	B	$\wedge$	C	)
T	?	T	T	T	**T**		T	T	T			**T**			T	**F**				F		

Next, since 'D' is true, '$\neg D$' must be false, so given that we're trying to make '$B \leftrightarrow \neg D$' true, we have to make B false as well:

A	B	C	D	A	$\rightarrow$	(	D	$\wedge$	C	)	B	$\leftrightarrow$	$\neg$	D	A	$\rightarrow$	(	$\neg$	B	$\wedge$	C	)
T	F	T	T	T	**T**		T	T	T		F	**T**	F	T	T	**F**				F		

We now have truth values assigned to all our atomic sentences. But there's a problem: we've had to make 'B' false and 'C' true, which means that '$(\neg B \wedge C)$ is true. But that now makes our conclusion '$A \rightarrow (\neg B \wedge C)$' true:

A	B	C	D	A	$\rightarrow$	(	D	$\wedge$	C	)	B	$\leftrightarrow$	$\neg$	D	A	$\rightarrow$	(	$\neg$	B	$\wedge$	C	)
T	F	T	T	T	**T**		T	T	T		F	**T**	F	T	T	**F/T!**		T	F	T	T	

wheres we were trying to construct a valuation that makes it false! What we've discovered is that it's impossible to construct such a valuation. So we can conclude that the entailment holds, and that the argument is TF valid.

Notice that the truth table row we've constructed technically does not, itself, show that the entailment holds. All it shows is that the particular valuation on which 'A', 'C', and 'D' are true and 'B' is false does not make the premises true and the conclusion false. To show that the argument is valid, we'd have to show that no other valuation makes the premises true and the conclusion false either. But the *reasoning* we used to arrive at our row does implicitly show that no other valuation will work: only this one has any hope of doing the trick, and it doesn't work.

To show that the entailment holds, we can give a verbal proof in English that recapitulates the reasoning we went through. The proof looks like this:

> *Claim:* $A \rightarrow (D \wedge C), B \leftrightarrow \neg D \vDash A \rightarrow (\neg B \wedge C)$
>
> *Proof:* assume (for reductio) that there exists a valuation, let's call it v, that makes '$A \rightarrow (D \wedge C)$' and '$B \leftrightarrow \neg D$' true but '$A \rightarrow (\neg B \wedge C)$' false. Since '$A \rightarrow (\neg B \wedge C)$' is false, '$A$' must be true. And since '$A \rightarrow (D \wedge C)$' is true and '$A$' is true, we know '$(D \wedge C)$' must be true, meaning that both 'D' and 'C' are true. Further, since 'D' is true, 'B' must be false in order for '$B \leftrightarrow \neg D$' to be true. But now if 'B' is false and 'C' is true, '$\neg B \wedge C$' is true, meaning that the conclusion '$A \rightarrow (\neg B \wedge C)$ is true as well. This contradicts our original assumption that '$A \rightarrow (\neg B \wedge C)$' is false on v. Since our assumption lead to a contradiction, we can conclude that the assumption is false, that is, that there *does not* exist a valuation that makes '$A \rightarrow (D \wedge C)$' and '$B \leftrightarrow \neg D$' true but '$A \rightarrow (\neg B \wedge C)$' false. So the entailment holds. QED[3]

This style of proof is called a proof by *reductio ad absurdum*: we began with an assumption (that there exists a valuation that makes the premises true and the conclusion false), showed that a contradiction (or "absurdity") results from it, and concluded that the assumption is false (that there exists no such valuation).

Instead of giving a reductio proof like this in English, we could instead construct a full 16 row truth table, and demonstrate validity that way. In any case, the moral is that whereas a single truth table row suffices to show that a TFL argument is *invalid*, more work is needed to show that an argument is valid.

[3]Here 'QED' abbreviates the latin phrase "*quod erad demonstrandum*", meaning "which was to be proven." Writing QED at the end of a proof is a signal that the proof is complete, and establishes the claim we set out to prove.

■ Exercises 3.8

A. Use partial truth tables to determine whether each argument is valid or invalid. Remember: if you find it's valid, you need to either give a full truth table (shortcuts are OK) or a *reductio* proof to demonstrate this. A one-row partial table only suffices to demonstrate *invalidity*.

1. $A \lor \big[A \to (A \leftrightarrow A)\big] \therefore A$
2. $A \leftrightarrow \neg(B \leftrightarrow A) \therefore A$
3. $A \to B, B \therefore A$
4. $A \lor B, B \lor C, \neg B \therefore A \land C$
5. $A \leftrightarrow B, B \leftrightarrow C \therefore A \leftrightarrow C$
6. $A \to (C \lor E), B \to D \therefore (A \lor B) \to (C \to (D \lor E))$
7. $D \lor \neg A, \neg(B \lor C) \to \neg D \therefore A \to (B \,\&\, C)$
8. $A \to (B \lor C), \neg(A \& B) \therefore A \to C$
9. $A \to (B \,\&\, E), D \to (A \lor C), \neg E \therefore D \to B$
10. $(D \to H) \to P, D \to \neg(C \lor G), C \lor H \therefore D \to P$
11. $\neg A \lor (B \to C), E \to (B \& A), C \to E \therefore C \leftrightarrow A$
12. $\neg C \to \neg(B \lor D), C \to \neg A, B \lor A \therefore A \leftrightarrow \neg C$

Testing for Other Semantic Notions

We can use partial truth tables to test for other semantic notions, besides entailment.

Tautology To test whether '$(U \land T) \to (S \land W)$' is a tautology, we can set up a partial truth table and see whether it's possible to make the sentence false:

S	T	U	W	$(U$	$\land$	$T)$	$\to$	$(S$	$\land$	$W)$
							F			

Since this is a conditional, and we're trying to make it flase, the antecedent must be true and the consequent must be false:

S	T	U	W	$(U$	$\land$	$T)$	$\to$	$(S$	$\land$	$W)$
					T		**F**		F	

In order for the '$(U \land T)$' to be true, both 'U' and 'T' must be true.

S	T	U	W	$(U$	$\land$	$T)$	$\to$	$(S$	$\land$	$W)$
	T	T		T	T	T	**F**		F	

Now we just need to make '$(S \land W)$' false. To do this, we need to make at least one of 'S' and 'W' false. We can make both 'S' and 'W' false if we want. All that matters is that the whole sentence turns out false on this line. Making an arbitrary decision, we finish the table in this way:

S	T	U	W	$(U$	$\land$	$T)$	$\to$	$(S$	$\land$	$W)$
F	T	T	F	T	T	T	**F**	F	F	F

So we now have a partial truth table which shows that there is a valuation which makes '$(U \wedge T) \rightarrow (S \wedge W)$' false, namely, the valuation which makes 'S' false, 'T' true, 'U' true and 'W' false. So we can conclude that '$(U \wedge T) \rightarrow (S \wedge W)$' is not a tautology.

Our partial truth table suffices to show that this sentence is *not* a tautology. But a partial truth table does not suffice to show that a sentence *is* a tautology, just as it doesn't suffice to show that an argument is valid. To show that a sentence is a tautology, i.e. to show that it's true on *every* valuation, we'd have to either give a full truth table, or a *reductio* argument in English showing that it's *impossible* to construct a valuation that makes the sentence false.

Contradiction. To test whether a sentence is a contradiction, we see whether we can construct a valuation that makes it true:

S	T	U	W	$(U \wedge T) \rightarrow (S \wedge W)$
				T

To make the sentence true, it will suffice to ensure that the antecedent is false. Since the antecedent is a conjunction, we can just make one of them false. Making an arbitrary choice, let's make 'U' false; we can then assign any truth value we like to the other atomic sentences.

S	T	U	W	$(U \wedge T) \rightarrow (S \wedge W)$
F	T	F	F	F F T **T** F F F

Since there is a valuation that makes the sentence true, our partial table shows that it is *not* a contradiction. Again, to show that something *is* a contradiction (false on *every* valuation), we'd have to give a full truth table or a *reductio* argument showing it is impossible to make the sentence true.

Consistency. To test some sentences for consistency, we would test whether we can construct a partial truth table which makes all of the sentence true. If we succeed, that is sufficient to demonstrate consistency. To demonstrate *inconsistency* we'd have to give a full truth table, or a *reductio* argument showing that it is impossible to make all the sentences in question true.

Equivalence To test two sentences for equivalence, we would test whether we can construct a partial truth table on which the two sentences have different truth values. If we succeed, that is sufficient to show that the sentences are *not* equivalent. To demonstrate equivalence, we'd have to give a full truth table, or a reductio argument showing that it's impossible to make the sentences have different truth values (or alternatively, two reductio arguments showing that the sentences mutually entail each other).

This table summarises what is required:

	Yes	**No**
Entailment?	complete table or *reductio*	partial truth table
Tautology?	complete table or *reductio*	partial truth table
Contradiction?	complete table or *reductio*	partial truth table
Consistent?	partial truth table	complete table or *reductio*
Equivalent?	complete table or *reductio*	partial truth table

■ Exercises 3.8

A. Use the partial truth table method to determine whether these pairs of sentences are equivalent (and remember, if you find they *are* equivalent you need to give a full table or a reductio proof):

1. A, $\neg A$
2. A, $A \vee A$
3. $A \rightarrow A$, $A \leftrightarrow A$
4. $A \vee \neg B$, $A \rightarrow B$
5. $A \wedge \neg A$, $\neg B \leftrightarrow B$
6. $\neg(A \wedge B)$, $\neg A \vee \neg B$
7. $\neg(A \rightarrow B)$, $\neg A \rightarrow \neg B$
8. $(A \rightarrow B)$, $(\neg B \rightarrow \neg A)$

B. Use the partial truth table method to determine whether these sentences are jointly consistent or inconsistent (and remember, to show they are *inconsistent* you need to give a full table or a reductio proof):

1. $A \wedge B$, $C \rightarrow \neg B$, C
2. $A \rightarrow B$, $B \rightarrow C$, A, $\neg C$
3. $A \vee B$, $B \vee C$, $C \rightarrow \neg A$
4. A, B, C, $\neg D$, $\neg E$, F

Natural deduction for TFL 4

4.1 The Very Idea of Natural Deduction

We've seen how to use truth tables to determine whether a TFL argument is valid. Truth tables are nice because they give us a completely mechanical test for validity: we just crunch through the table and see whether there is any valuation that makes all the premises true and the conclusion false. But truth tables do not give us much *insight* into why arguments are valid. Consider these two TFL arguments:

$$P \to Q, P \therefore Q$$
$$P \lor Q, \neg P \therefore Q$$

Clearly, these are valid. You could confirm that they are by constructing four-line truth tables. But another way to tell that they are valid is by noticing that they make use of intuitively correct *forms of inference*. The first form of inference goes by the fancy Latin name of *modus ponens*, and the second is called *disjunctive syllogism* (its Latin name is *modus tollendo ponens*).

This suggests a rather different approach to logic. Rather than constructing a truth table, we can assess an argument's validity by seeing whether it is possible to *deduce* its conclusion from its premises via a series of simple, intuitively correct steps of inference. Think of it as the Sherlock Holmes approach to logic, in contrast to the more computational approach embodied by truth tables.

One aim of a *natural deduction system* is to formalize this process of deduction. We'll begin with a few very basic rules of inference, which can then be combined into more complicated chains of reasoning. Indeed, with just a small starter pack of rules, we will be able to capture *all* valid arguments. Whereas truth tables are completely mechanical, natural deduction requires insight and ingenuity. This makes it harder, but also more interesting.

The move to natural deduction can be motivated by more than the search for insight, however. It might also be motivated by *necessity*. In TFL, truth tables give us a completely mechanical test for validity. Of course they can get unmanageably big as the number of atomic sentences increase, but you could in principle program a computer to crunch through them for you. When we get to FOL, in the second part of this book, things will look very different. There is nothing like the truth table test for validity available in FOL. The fact that there is no completely mechanical test for validity in FOL is a deep mathematical result, independently proved by Alan Turing and Alonzo Church in 1936. So in FOL, using methods that require ingenuity and insight become indispensable, and we will have to rely on natural deduction to prove arguments valid.

The modern development of natural deduction dates from simultaneous papers from 1934 by Gerhard Gentzen and Stanisław Jaśkowski. Later, in 1952, Frederic Fitch introduced the graphical "Fitch notation" for natural deduction proofs that we will use here.

4.2 Setting up Natural Deduction Proofs

The NATURAL DEDUCTION system we will develop will includes a pair of rules for every connective. INTRODUCTION RULES allow us to prove a sentence that has that connective as the main logical operator, and ELIMINATION RULES allow us to prove something *from* a sentence that has that connective as the main logical operator.

Our natural deduction proofs will be *formal proofs*. They will consist of a sequence of lines, with the premises listed at the top and the conclusion at the bottom. All the lines in between have to be justified as following from earlier lines via some rule of inference. As an illustration, consider the following instance of DeMorgan's Law:

$$\neg(A \lor B) \therefore \neg A \land \neg B$$

We would start this proof by writing the premise:

$$\begin{array}{r|l} 1 & \neg(A \lor B) \\ \cline{2-2} \end{array}$$

Note that we have numbered the premise, since we will want to refer back to it. Indeed, every line on a proof is numbered so that we can refer back to it. Note also that we have drawn a vertical line to the left and a horizontal line underneath the premise. Everything written above the horizontal line is an *assumption* — so the premise is introduced into the proof as an assumption. Everything written below the horizontal line will either be something which follows from this assumption, or it will be some new assumption.

We are hoping to conclude that '$\neg A \land \neg B$'. So we are hoping ultimately to end our proof with a line that looks like this:

$$\begin{array}{r|l} n & \neg A \land \neg B \end{array}$$

for some number n. It doesn't matter what line number we end on, but we would obviously prefer a short proof to a long one.

Or to take another example, suppose we wanted to prove the following valid:

$$A \lor B, \neg(A \land C), \neg(B \land \neg D) \therefore \neg C \lor D$$

The argument has three premises, so we start by writing them all down, numbered, and drawing a vertical line the the left and a horizontal line underneath:

$$\begin{array}{r|l} 1 & A \lor B \\ 2 & \neg(A \land C) \\ 3 & \neg(B \land \neg D) \\ \cline{2-2} \end{array}$$

We are hoping to conclude with a line that looks like this:

$$n \quad | \; \neg C \vee D$$

What we have to learn are rules of inference, and how to chain them together to move in a step-by-step fashion from the premises to the conclusion.

Before we look at the rules at our disposal, however, we'll introduce some new terminology and notation having to do with proofs. We will use the following expression:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$$

to mean that $\mathscr{C}$ is *provable* from $\mathscr{A}_1, \ldots, \mathscr{A}_n$, that is, that there exists a proof which ends with $\mathscr{C}$ and whose premises include at most $\mathscr{A}_1, \ldots, \mathscr{A}_n$. We'll call a provability claim of this form a SEQUENT. By providing a natural deduction proof, we can demonstrate that a sequent holds, i.e. demonstrate that there is indeed a proof of $\mathscr{C}$ from $\mathscr{A}_1, \ldots, \mathscr{A}_n$. When we want to say that it is *not* the case that there exists a proof which ends with $\mathscr{C}$ from $\mathscr{A}_1, \ldots, \mathscr{A}_n$, we write:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n \nvdash \mathscr{C}$$

Natural deduction does not give us a way to verify claims like these about the non-existence of a proof — more complicated reasoning would be required to show this kind of thing.

The symbol '$\vdash$' is called the *single turnstile*. This is *not* the same as the double turnstile symbol '$\vDash$' that we used to symbolize entailment in chapter 3. The single turnstile '$\vdash$' says something about the existence of a certain kind of *proof* (one that begins with certain premises and ends in a certain conclusion). The double turnstile '$\vDash$' says something about the non-existence of a certain kind of *valuation* (one that makes the premises true and the conclusion false). Valuations are very different things from proofs, so it's important not to confuse '$\vdash$' (a *proof theoretic* notion) with '$\vDash$' (a *semantic* notion).

That said, the system of natural deduction that we will develop is specifically designed to deliver a proof whenever a semantic entailment holds. That is, it is designed to ensure:

COMPLETENESS: If $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{C}$ then $\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$

meaning that if $\mathscr{C}$ is semantically entailed by $\mathscr{A}_1, \ldots, \mathscr{A}_n$, then there is also a proof of $\mathscr{C}$ from $\mathscr{A}_1, \ldots, \mathscr{A}_n$. Our natural deduction system is also designed to guarantee the other direction:

SOUNDNESS: If $\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$ then $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{C}$

' meaning that whenever $\mathscr{C}$ is provable from $\mathscr{A}_1, \ldots, \mathscr{A}_n$, then $\mathscr{C}$ is also semantically entailed by $\mathscr{A}_1, \ldots, \mathscr{A}_n$.[1] All good proof systems should be both sound and complete, to ensure that the proof-theoretic notion of provability matches up perfectly with the semantic notion of entailment. In a more advanced logic class, you learn how to provide "meta-logical" proofs showing that a proof system is both sound and complete, but for now you'll just have to take my word for it that our natural deduction system will have both of these features.

We also have a proof-theoretic analogue of the semantic notion of TF equivalence:

Two sentences $\mathscr{A}$ and $\mathscr{B}$ are PROVABLY EQUIVALENT iff each is provable from the other; i.e., both $\mathscr{A} \vdash \mathscr{B}$ and $\mathscr{B} \vdash \mathscr{A}$, also written $\mathscr{A} \dashv\vdash \mathscr{B}$.

[1]Note that this is a different notion of soundness from the one we discussed in §1.3

Given that our natural deduction system is both sound and complete, we will again have it that any two TF equivalent sentences are also provably equivalent, and vice versa. Let's now look at the rules that constitute our system of natural deduction.

4.3 Conjunction Rules

Suppose I want to show that Jacob is both hypocritical and reactionary. One obvious way to do this would be as follows: first I show that Jacob is hypocritical; then I show that Jacob is reactionary; then I put these two demonstrations together to obtain the conjunction.

Our natural deduction system will capture this thought via the rule of ∧-Introduction, or ∧*I* for short. Perhaps I am working through a proof, and have obtained '*H*' on line 8 and '*R*' on line 15. Then on any subsequent line I can obtain '$H \wedge R$' thus:

$$
\begin{array}{r|ll}
8 & H & \\
 & \vdots & \\
15 & R & \\
 & \vdots & \\
 & H \wedge R & \wedge\text{I } 8,\ 15
\end{array}
$$

Every line of our proof must either be an assumption, or must be justified by some rule like this. We cite '∧I 8, 15' here to indicate that '$H \wedge R$' is obtained by the rule of conjunction introduction applied to lines 8 and 15. We could equally well have conjoined the conjuncts in the opposite order to infer '$R \wedge H$' rather than '$H \wedge R$', though then we should also adjust our rule citation to read '∧I 15, 8', with the line numbers of the two conjuncts listed in the opposite order.

More generally, our conjunction introduction rule is:

$$
\begin{array}{r|ll}
m & \mathscr{A} & \\
n & \mathscr{B} & \\
 & \mathscr{A} \wedge \mathscr{B} & \wedge\text{I } m,\ n
\end{array}
$$

Here lines *m* and *n* can occur in either order, i.e. $\mathscr{A}$ could occur first in the proof followed by $\mathscr{B}$ later on, or $\mathscr{B}$ could occur first followed by $\mathscr{B}$ later on.

The rule is called "conjunction *introduction*" because it introduces the symbol '∧' into our proof where it may have been absent. Correspondingly, we have a rule that *eliminates* that symbol. Suppose you have shown that Jacob is both hypocritical and reactionary. Then you're entitled to infer that Jacob is hypocritical, and you're also entitled to infer that Jacob is reactionary. This gives us our conjunction elimination rule(s):

$$\begin{array}{r|ll} m & \mathscr{A} \wedge \mathscr{B} & \\ & \mathscr{A} & \wedge\text{E } m \end{array}$$

and equally:

$$\begin{array}{r|ll} m & \mathscr{A} \wedge \mathscr{B} & \\ & \mathscr{B} & \wedge\text{E } m \end{array}$$

The point is simply that, when you have a conjunction on some line of a proof, you can obtain either of its two conjuncts by ∧E.

One point is worth emphasizing: you can only apply this rule (as well as the other rules we'll introduce) to the main logical operator of a sentence. So the following would not be a legitimate use of $\wedge E$:

$$\begin{array}{r|ll} 1 & P \wedge (Q \wedge R) & \\ \cline{2-2} 2 & R & \wedge\text{E } 1 \end{array}$$

I can only apply $\wedge E$ to the main operator of '$P \wedge (Q \wedge R)$', giving me either 'P' or '$Q \wedge R$'. I could then apply $\wedge E$ to the latter to get 'R' itself, but I can't get R directly from '$P \wedge (Q \wedge R)$'.

Here's an example that illustrates this. The following argument is valid (showing that ∧ is associative):

$$A \wedge (B \wedge C) \therefore (A \wedge B) \wedge C$$

To provide a proof for this argument, we start by writing the premise as an assumption:

$$\begin{array}{r|l} 1 & A \wedge (B \wedge C) \\ \cline{2-2} \end{array}$$

From the premise, we can get 'A' and '$(B \wedge C)$' by applying ∧E twice. And we can then apply ∧E twice more to get a proof that looks like this:

$$\begin{array}{r|ll} 1 & A \wedge (B \wedge C) & \\ \cline{2-2} 2 & A & \wedge\text{E } 1 \\ 3 & B \wedge C & \wedge\text{E } 1 \\ 4 & B & \wedge\text{E } 3 \\ 5 & C & \wedge\text{E } 3 \end{array}$$

Again: we *cannot* get 'B' or 'C' by applying $\wedge E$ to line 1. We first have to get '$B \wedge C$' from 1, and then get 'B' and 'C' out of *this* by applying $\wedge E$ again. To get to our desired conclusion, we now just put the various atomic sentences back together using $\wedge I$:

1	$A \wedge (B \wedge C)$	
2	A	$\wedge$E 1
3	$B \wedge C$	$\wedge$E 1
4	B	$\wedge$E 3
5	C	$\wedge$E 3
6	$A \wedge B$	$\wedge$I 2, 4
7	$(A \wedge B) \wedge C$	$\wedge$I 6, 5

Notice that whereas $\wedge E$ gets applied to a single line, $\wedge I$ gets applied to two lines. However, $\wedge I$ doesn't necessarily have to be applied to two *different* lines. If we wanted, for example, we could formally prove '$A \wedge A$' from 'A' as follows:

1	A	
2	$A \wedge A$	$\wedge$I 1, 1

And we could now apply $\wedge E$ to line 2 to prove (rather uninterestingly) that 'A' follows from 'A':

1	A	
2	$A \wedge A$	$\wedge$I 1, 1
3	A	$\wedge$E 2

4.4 Conditional Rules

Consider the following argument:

If Jane is smart then she is fast. Jane is smart. So Jane is fast.

This argument is certainly valid. And it suggests a conditional elimination rule ($\rightarrow$E):

m	$\mathscr{A} \rightarrow \mathscr{B}$	
n	$\mathscr{A}$	
	$\mathscr{B}$	$\rightarrow$E m, n

This rule implements the *modus ponens* form of inference mentioned earlier: given a conditional, and given its antecedent, we can infer its consequent. Again, this is an elimination rule, because it allows us to obtain a sentence that may not contain '$\rightarrow$', having started with a sentence that did contain '$\rightarrow$'. Note that the conditional and its antecedent can appear in either order in our proof. However, in the citation for →E, we should cite the conditional first, followed by the antecedent.

The rule for conditional introduction is also quite easy to motivate. The following argument should be valid:

> Jacob is hypocritical. Therefore, if Jacob is reactionary, then Jacob is both hypocritical *and* reactionary.

If someone doubted that this was valid, we might try to convince them otherwise by explaining ourselves as follows:

> We are given that Jacob is hypocritical. Now, assume additionally, for the sake of argument, that Jacob is also reactionary. Then by conjunction introduction—which we just discussed—Jacob is both hypocritical and reactionary. Of course, this only follows given our additional assumption that Jacob is reactionary. But this just means that, *if* Jacob is reactionary, then he is both hypocritical and reactionary.

Transferred into natural deduction format, here is the pattern of reasoning that we just used. We started with one premise, 'Jacob is hypocritical', thus:

$$\begin{array}{r|l} 1 & H \\ \cline{2-2} \end{array}$$

The next thing we did is to make an *additional, temporary assumption* ('Jacob is reactionary'), for the sake of argument. To indicate that we are no longer dealing *merely* with our original assumption ('H', the premise), but with an additional assumption, we continue our proof as follows:

$$\begin{array}{r|ll} 1 & H & \\ \cline{2-2} 2 & & \begin{array}{|l} R \\ \hline \end{array} \end{array}$$

Introducing 'R' as an additional assumption opens up a *subproof*. Inside this subproof we can now reason under the assumption, or hypothesis, that 'R' holds. We indicate this by drawing a line under 'R' (to indicate that it is an assumption) and by indenting it with a further vertical line (to indicate that we have entered a new subproof headed by this assumption).

With this extra assumption in place, we are in a position to use ∧I:

$$\begin{array}{r|ll} 1 & H & \\ \cline{2-2} 2 & & \begin{array}{|l} R \\ \hline \end{array} \\ 3 & & \begin{array}{|ll} H \wedge R & \wedge\text{I } 1, 2 \end{array} \end{array}$$

So we have now shown that, under the assumption that 'R' holds, we can obtain '$H \wedge R$'. We can therefore conclude that, *if* 'R' obtains, so does '$H \wedge R$'. Or, to put it more briefly, we can conclude '$R \to (H \wedge R)$':

```
1  | H
   |----
2  |  | R
   |  |----
3  |  | H ∧ R            ∧I 1, 2
4  | R → (H ∧ R)        →I 2–3
```

Observe that we have popped back out of the subproof opened by our additional assumption. This indicates that we have now *discharged* the additional assumption, 'R', since the conditional itself follows just from our original premise, 'H'.

The general pattern at work here is the following: to prove a conditional $\mathscr{A} \rightarrow \mathscr{B}$, we *assume* the antecedent $\mathscr{A}$ temporarily or "for the sake of argument", and then try to prove the consequent $\mathscr{B}$ from that additional assumption. If we succeed, we can discharge the assumption and conclude that the conditional $\mathscr{A} \rightarrow \mathscr{B}$ holds:

```
m  |  | 𝒜
   |  |----
   |  | ⋮
n  |  | ℬ
   | 𝒜 → ℬ         →I m–n
```

Here's another illustration of →I in action. Suppose we want to prove:

$$P \rightarrow Q, Q \rightarrow R \therefore P \rightarrow R$$

We start by listing both of our premises as assumptions. Then, since we are aiming to prove a conditional, namely, '$P \rightarrow R$', we assume the antecedent to that conditional as an additional assumption, which opens a new subproof:

```
1  | P → Q
2  | Q → R
   |--------
3  |  | P
   |  |----
   |  | ⋮
```

Our goal is not to prove R from this additional assumption, inside of our subproof. Given 'P' , we can use →E on the first premise. This will yield 'Q'. And we can then use →E on the second premise to get 'R'. So, by assuming 'P' we were able to prove 'R'! We can now apply the →I rule—discharging 'P'—and finish the proof:

```
1 │ P → Q
2 │ Q → R
  ├──────
3 │  │ P
  │  ├───
4 │  │ Q        →E 1, 3
5 │  │ R        →E 2, 4
6 │ P → R       →I 3–5
```

Notice that when applying →I to obtain '$P \to R$', we have to cite the *entire* subproof that begins with 'P' and ends with 'R'. So we use a dash, rather than just a comma, between the two line numbers (writing '3–5' rather than '3, 5').

4.5 Additional assumptions and subproofs

The rule →I invoked the idea of opening subproofs via additional assumptions. This needs to be handled with some care. Consider this proof:

```
1 │ A
  ├───
2 │  │ B
  │  ├───
3 │  │ B ∧ B      ∧I 2, 2
4 │  │ B          ∧E 3
5 │ B → B         →I 2–4
```

This is a perfectly legitimate, if somewhat unusual, proof. What it shows is that the argument $A \therefore B \to B$ is valid. This is as it should be: '$B \to B$' is a tautology, and any argument with a tautology as its conclusion is valid. But suppose we now tried to continue the proof as follows:

```
1 │ A
  ├───
2 │  │ B
  │  ├───
3 │  │ B ∧ B      ∧I 2, 2
4 │  │ B          ∧E 3
5 │ B → B         →I 2–4
6 │ B             No!! →E 5, 4
```

If we were allowed to do this, it would be a disaster: our proof would now purport to show that the argument $A \therefore B$ is valid. We could in this way prove any conclusion we liked from any premise whatsoever. That would obviously destroy the soundness of our natural deduction system.

What has gone wrong here is that on line 6 we've illegitimately tried to apply →E to line 4, which occurs inside a subproof we've already "popped out of." A subproof can be thought of as essentially posing this question: *what could we show, if we also make this additional assumption?* While we are working within the subproof, we can refer to the additional assumption that we made to open the subproof, and to anything that we obtained from our premises. After all, those premises still hold. But at some point we have to return from the subproof to the main proof. At this point, we say that the subproof is CLOSED. Once a subproof is closed, the additional assumption that opened it has been DISCHARGED, and it becomes illegitimate to draw upon anything that depends upon that discharged assumption. Thus we stipulate:

> To cite any individual line when applying a rule, that line must (1) occur before the application of the rule, but (2) not occur within a closed subproof.

The application of →E in the faulty proof above involves citing a line (namely line 4) that occurs within a subproof that has (by line 6) been closed. This is illegitimate.

Once we have started thinking about what we can show by making additional assumptions, nothing stops us from posing the question of what we could show if we were to make *even more* assumptions. We can in other words introduce a subproof within a subproof. Here is an example of such nested subproofs:

```
1 | A
  |---
2 |   | B
  |   |---
3 |   |   | C
  |   |   |---
4 |   |   | A ∧ B              ∧I 1, 2
5 |   | C → (A ∧ B)            →I 3–4
6 | B → (C → (A ∧ B))          →I 2–5
```

This proof gets set up as follows: we begin with the premise 'A' and the goal of proving '$B \to (C \to (A \wedge B))$'. Since the conclusion is a conditional, we assume its antecedent 'B' and set ourselves the new goal of proving its consequent '$(C \to (A \wedge B))$' using this additional assumption. But our new goal '$(C \to (A \wedge B))$' is *itself* a conditional, so we repeat the same process: assume its antecedent 'C', and try to prove its consequent '$A \wedge B$' in the subproof we've opened. Proving '$A \wedge B$' is easy: we can just apply ∧I to our original premise from line 1 and our first additional assumption from line 2. Referring back to lines 1 and 2 in step 4 of the proof in this manner is legitimate, since neither line occurs in a subproof that has been closed by the time of step 4.

But it would now *not* be legitimate to continue the proof as follows:

1	A	
2	$\quad B$	
3	$\quad\quad C$	
4	$\quad\quad A \wedge B$	$\wedge$I 1, 2
5	$\quad C \rightarrow (A \wedge B)$	$\rightarrow$I 3–4
6	$B \rightarrow (C \rightarrow (A \wedge B))$	$\rightarrow$I 2–5
7	$C \rightarrow (A \wedge B)$	**No!!** $\rightarrow$I 3–4

This would be awful. This proof would purport to show that '$C \rightarrow (A \wedge B)$' can be deduced from the premise 'A'. But the argument '$A \therefore C \rightarrow (A \wedge B)$' is certainly not valid. Again, if we were allowed to do this kind of thing, our proof system would no longer be sound.

The problem is that the subproof that began with the assumption 'C' occurs within the scope of (i.e. within the subproof opened by) assumption 'B' on line 2. By line 6, we have *discharged* assumption 'B'. So it is cheating to try to help ourselves (on line 7) to a subproof that occurs within the scope of an assumption that has already been discharged. Here the problem isn't that we cited an individual *line* that occurs inside a closed subproof, but that we cited an entire *subproof* that occurs within a closed subproof. So we expand our stipulation to cover rules that cite entire subproofs:

> To cite a subproof when applying a rule, the subproof must (1) come before the application of the rule, but (2) not occur within some other closed subproof.

Our proof above violates this stipulation, since the subproof of lines 3–4 occurs within the subproof spanning lines 2–5, which has already been closed by the point we get to line 7.

One last thing to remember about subproofs: in principle, you can always make any additional assumption you like, at any point in a proof. However, once you make an assumption, you also have to have a strategy for discharging the assumption and closing the subproof that it opens, so as to ultimately return back to your main proof. For this reason it is very important to *only* make assumptions when you have a discharge strategy like this in mind. At this point, we have only one rule that tells us to make an additional assumption, and that is the rule of $\rightarrow$I: if your goal is to prove a conditional $\mathscr{A} \rightarrow \mathscr{B}$, you should assume its antecedent $\mathscr{A}$ and try to prove its consequent $\mathscr{B}$ inside the subproof opened by that assumption. But at this point, this is the only time at which you should be making assumptions — when aiming to prove conditionals.

4.6 Proving Theorems and Reiterating

We said that the sequent $\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$ means that there exists a proof ending in $\mathscr{C}$ whose premises — or as we can now say more generally, whose undischarged assumptions — include at most $\mathscr{A}_1, \ldots, \mathscr{A}_n$. Similarly, we will write:

$$\vdash \mathscr{A}$$

to say that there is a proof of $\mathscr{A}$ with no undischarged assumptions whatsoever. Sentences which are provable with no undischarged assumptions are called THEOREMS. Notice the similarity with our notation in semantics, where we used $\vDash \mathscr{A}$ to say that $\mathscr{A}$ is a tautology. Given that our system of natural deduction is both is sound and complete, every tautology should be a theorem of our proof system, and every theorem should be a tautology.

For example, since $(A \land B) \to A$ is a tautology, we should be able to prove it with no undischarged assumptions. Here's how that looks like:

```
1 │  │ A∧B
  │  ├─────
2 │  │ A                 ∧E 1
3 │ (A∧B) → A)           →I 1–2
```

Notice that, unlike in any of the other proofs we've looked at, the leftmost vertical line, which appears next to our conclusion, has no premise or assumption listed at its top. This graphically indicates that '$(A \land B) \to A$' is a theorem in our proof system, i.e. something that is provable without any premises or undischarged assumptions.

As another example, take the tautology '$A \to (B \to A)$'. This is provable as a theorem as follows:

```
1 │  │ A
  │  ├───
2 │  │  │ B
  │  │  ├───
3 │  │  │ A∧B            ∧I 1, 2
4 │  │  │ A              ∧E 3
5 │  │ B → A             →I 2–4
6 │ A → (B → A)          →I 1–5
```

This proof is a bit odd: since we're trying to prove '$(B \to A)$' on line 5, the subproof that begins with 'B' on line 2 has to end with 'A'. We already have 'A' as an assumption on line 1, but the only way to get it to appear at the end of our second subproof is to first conjoin it with 'B' to get '$(A \land B)$' on line 3, and then to use $\land$E to get it back on its own on line 4.

In order to avoid having to use the trick of using $\land$I and $\land$E in this way to repeat earlier lines at later stages in a proof, we'll allow ourselves to use the following shortcut rule:

> REITERATION RULE: at any point in a proof, we may write down any sentence that (1) occurs before that point in the proof, and (2) does not occur within a closed subproof.

This lets us shorten the above proof to:

```
1  │ │ A
   │ ├──
2  │ │ │ B
   │ │ ├──
3  │ │ │ A              Reit 1
4  │ │ B → A            →I 2–3
5  │ A → (B → A)        →I 1–4
```

Reiteration also gives us a quick way to prove that '$B \rightarrow B$' is a theorem:

```
1  │ │ B
   │ ├──
2  │ │ B          Reit 1
3  │ B → B        →I 1–2
```

In fact, just as $\wedge$I can be applied to a single line to go from 'A' to '$A \wedge A$', so $\rightarrow$I can in principle be applied to a subproof that consists of just one line. So an even shorter proof of the theorem '$B \rightarrow B$' can be given like this:

```
1  │ │ B
   │ └──
2  │ B → B        →I 1–1
```

■ Exercises 4.6

A. Prove the following sequents (these require only conjunction rules and $\rightarrow$E):

1. $A \wedge (B \wedge C) \vdash C \wedge B$
2. $P \wedge Q, (Q \wedge P) \rightarrow R \vdash R$
3. $A \wedge B, B \rightarrow (A \rightarrow C) \vdash C \wedge B$

B. Prove the following sequents (these now also require $\rightarrow$I, and note that the last one asks you to prove an equivalence, which requires two deductions):

1. $A \rightarrow B, B \rightarrow C \vdash A \rightarrow (B \wedge C)$
2. $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$
3. $A \rightarrow B \vdash (A \wedge C) \rightarrow (B \wedge C)$
4. $A \rightarrow (B \rightarrow C) \dashv\vdash (A \wedge B) \rightarrow C$

C. Prove the following theorems:

1. $\vdash (P \wedge Q) \rightarrow (Q \wedge P)$
2. $\vdash (P \rightarrow Q) \rightarrow ((Q \rightarrow R) \rightarrow (P \rightarrow R))$
3. $\vdash A \rightarrow (B \rightarrow B)$

4.7 Biconditional Rules

The rules for the biconditional will be like double-barrelled versions of the rules for the conditional. In order to prove '$F \leftrightarrow G$', for instance, you must be able to prove 'G' on the assumption 'F' *and* prove 'F' on the assumption 'G'. The biconditional introduction rule ↔I therefore requires two subproofs. Schematically, the rule works like this:

```
i  │  │ 𝒜
   │  ├────
   │  │ ⋮
j  │  │ ℬ
k  │  │ ℬ
   │  ├────
   │  │ ⋮
l  │  │ 𝒜
   │ 𝒜 ↔ ℬ        ↔I i–j, k–l
```

There can be as many lines as you like between i and j, and as many lines as you like between k and l. Moreover, the subproofs can come in any order, and the second subproof does not need to come immediately after the first.

The biconditional elimination rule ↔E is a bit like →E in both directions. If you have the left-hand subsentence of the biconditional, you can obtain the right-hand subsentence, and if you have the right-hand subsentence, you can obtain the left-hand subsentence:

```
m  │ 𝒜 ↔ ℬ
n  │ 𝒜
   │ ℬ              ↔E m, n
```

and:

```
m  │ 𝒜 ↔ ℬ
n  │ ℬ
   │ 𝒜              ↔E m, n
```

As usual, lines m and n can occur in either order, but in the citation for ↔E, we always cite the line number of the biconditional first.

4.8 Disjunction Rules

Suppose Alice is a logician. Then certainly Alice is either a logician or a chemist. After all, to say that Alice is either a logician or a chemist is to say something weaker than to say that Alice is a logician. In fact, we can weaken the claim however we like. Suppose Jacob a logician. It follows that Alice is *either* a logician *or* a kumquat. Equally, it follows that *either* Alice is a logician *or* oranges are the only fruits on earth. Equally, it follows that *either* Alice is a logician *or* the earth is flat. Many of these things are strange inferences to draw. But there is nothing *logically* wrong with them (even if they violate implicit conversational norms).

Our disjunction introduction rules implement this process of weakening a claim:

$$\begin{array}{r|ll} m & \mathscr{A} & \\ & \mathscr{A} \vee \mathscr{B} & \vee\text{I } m \end{array}$$

and

$$\begin{array}{r|ll} m & \mathscr{A} & \\ & \mathscr{B} \vee \mathscr{A} & \vee\text{I } m \end{array}$$

Notice that $\mathscr{B}$ can be *any* sentence whatsoever. So the following is a perfectly kosher proof:

$$\begin{array}{r|ll} 1 & M & \\ \cline{2-2} 2 & M \vee ([(A \leftrightarrow B) \to (C \wedge D)] \leftrightarrow [E \wedge F]) & \vee\text{I } 1 \end{array}$$

Using a complete truth table to show this would have taken 128 lines.

The disjunction elimination rule is slightly trickier. Suppose that Alice is either a logician or a chemist. What can you conclude? Not that Alice is a logician; she might be a chemist instead. And equally, not that Alice is a chemist; for she might be a logician instead. Disjunctions, just by themselves, are hard to work with.

But suppose that we could somehow show both of the following: first, that Alice's being a logician entails that she has a PhD; second, that Alice's being a chemist entails that she has a PhD. Then if we know that Alice is either a logician or a chemist, we know that, whichever she happens to be, she has a PhD. Our disjunction elimination rule $\vee$E formalizes this insight:

$$
\begin{array}{r|lll}
m & \mathscr{A} \lor \mathscr{B} & & \\
i & & \begin{array}{|l} \mathscr{A} \\ \hline \vdots \end{array} & \\
j & & \begin{array}{|l} \mathscr{C} \end{array} & \\
k & & \begin{array}{|l} \mathscr{B} \\ \hline \vdots \end{array} & \\
l & & \begin{array}{|l} \mathscr{C} \end{array} & \\
 & \mathscr{C} & & \lor\text{E } m,\ i\text{–}j,\ k\text{–}l
\end{array}
$$

This is a bit more complicated than our previous rules, but the idea is fairly simple. Suppose we have some disjunction, $\mathscr{A} \lor \mathscr{B}$. If we can now give two subproofs, one showing that $\mathscr{C}$ follows from the assumption that $\mathscr{A}$, and the other that $\mathscr{C}$ follows from the assumption that $\mathscr{B}$, then we can infer $\mathscr{C}$ itself. You can think of this as formally implementing the kind of *reasoning by cases* that we discussed in §3.8. The disjunction $\mathscr{A} \lor \mathscr{B}$ tells us that one of two cases obtains, either $\mathscr{A}$ holds or $\mathscr{B}$ does. If it can be shown that $\mathscr{C}$ must hold in either case, then we can conclude that $\mathscr{C}$ holds on the basis of the original disjunction.

Notice that the citation for $\lor$E is fairly complex. We have to cite *three* things: the line number of the original disjunction, and the two subproofs. As usual, there can be as many lines as you like between i and j, and as many lines as you like between k and l. Moreover, the subproofs and the disjunction can come in any order, and do not have to be adjacent.

Some examples will help illustrate the rule. Consider this argument:

$$(P \land Q) \lor (P \land R) \therefore P$$

The premise tells us that either '$(P \land Q)$' holds or '$(P \land R)$' holds. But in either case, 'P' must holds, so 'P' follows from our premise. Here's how this looks as a proof:

$$
\begin{array}{r|lll}
1 & (P \land Q) \lor (P \land R) & & \\
\hline
2 & & \begin{array}{|l} P \land Q \\ \hline P \end{array} & \\
3 & & & \land\text{E } 2 \\
4 & & \begin{array}{|l} P \land R \\ \hline P \end{array} & \\
5 & & & \land\text{E } 4 \\
6 & P & & \lor\text{E } 1,\ 2\text{–}3,\ 4\text{–}5
\end{array}
$$

Here is a slightly harder example:

$$A \land (B \lor C) \therefore (A \land B) \lor (A \land C)$$

And here is a proof corresponding to this argument:

1	$A \wedge (B \vee C)$		
2	A		$\wedge$E 1
3	$B \vee C$		$\wedge$E 1
4		B	
5		$A \wedge B$	$\wedge$I 2, 4
6		$(A \wedge B) \vee (A \wedge C)$	$\vee$I 5
7		C	
8		$A \wedge C$	$\wedge$I 2, 7
9		$(A \wedge B) \vee (A \wedge C)$	$\vee$I 8
10	$(A \wedge B) \vee (A \wedge C)$		$\vee$E 3, 4–6, 7–9

Don't be alarmed if you think that you wouldn't have been able to come up with this proof yourself. The ability to construct proofs comes with practice. But hopefully you can, by looking at the proof, see that it conforms with the rules that we have laid down.

In general, the strategy working with $\vee$E is that whenever you have a disjunction $\mathcal{A} \vee \mathcal{B}$ as a premise, or as something that is implied by a premise or by an assumption, then you should split your reasoning into two cases, and prove your goal from each disjunct separately. In our case here, the premise on line 1 implies the disjunction '$B \vee C$'. So applying the $\vee$E strategy, that means first assuming B, and proving our conclusion $(A \wedge B) \vee (A \wedge C)$ from that, and then assuming C, and proving that the conclusion also follows from that. Given that either 'B' or 'C' holds (as per line 3), and having shown that the conclusion follows in either case, we can then conclude that the conclusion holds on the basis of $\vee$E.

■ Exercises 4.8

A. Prove the following sequents:

1. $A \leftrightarrow B \vdash B \leftrightarrow A$
2. $A \leftrightarrow B, B \leftrightarrow C \vdash A \leftrightarrow C$
3. $K \wedge L \vdash K \leftrightarrow L$
4. $(A \wedge B) \leftrightarrow (A \wedge C) \vdash A \rightarrow (B \leftrightarrow C)$
5. $A \rightarrow B \vdash A \rightarrow (C \vee B)$
6. $A \vee B \vdash B \vee A$
7. $A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C)$
8. $S \leftrightarrow T \vdash S \leftrightarrow (T \vee S)$
9. $(C \wedge D) \vee E \vdash E \vee D$
10. $A \rightarrow C \vdash (A \vee C) \rightarrow C$
11. $A \vee B \vdash (A \rightarrow B) \rightarrow B$
12. $(Z \wedge K) \vee (K \wedge M), K \rightarrow D \vdash D$
13. $(Z \wedge K) \leftrightarrow (Y \wedge M), D \wedge (D \rightarrow M) \vdash Y \rightarrow Z$
14. $\vdash P \leftrightarrow P$

4.9 Negation Rules

We have only one connective left: negation. In the context of natural deduction, negation is unusual because the rules governing it involve another notion, that of *contradiction*.

Consider that an effective way to argue against someone is to show that the assumptions they are making lead to a contradiction. At that point, you have your opponent in a bind — they have to give up at least one of their assumptions. This argumentative strategy involves the style of *reductio ad absurdum* reasoning that we encountered in §3.8. We are going to make use of this idea in our proof system by adding a new symbol, '$\bot$'. This should be read as something like 'contradiction!' or 'reductio!' or 'but that's absurd!'

We can introduce this symbol into a proof whenever we explicitly contradict ourselves, i.e. whenever we find both a sentence and its negation appearing in the proof. This gives us our elimination rule for negation:

$$\begin{array}{r|ll} m & \mathscr{A} & \\ n & \neg\mathscr{A} & \\ & \bot & \neg\text{E } m, n \end{array}$$

It does not matter what order the sentence and its negation appear in, and they do not need to appear on adjacent lines. However, we should cite the sentence first, followed by its negation. The rule is called ¬E, since the negation sign is eliminated. (We could equally have called this rule '$\bot$I', since it introduces '$\bot$'.)

Next, we need a rule for negation introduction. This rule will be a formal implementation of the *reductio ad absurdum* proof strategy: if assuming something leads you to a contradiction, then the assumption must be wrong. The rule looks like this:

$$\begin{array}{r|l|ll} i & & \underline{\mathscr{A}} & \\ & & \vdots & \\ j & & \bot & \\ & \neg\mathscr{A} & & \neg\text{I } i\text{–}j \end{array}$$

There can be as many lines between i and j as you like. As with other rules that require subproofs, you need to cite the entire subproof when applying ¬I. Notice that since the subproof has to end with the contradiction symbol $\bot$, we will usually have to use ¬E in the course of using ¬I, since ¬E is the rule that lets us introduce $\bot$ into a proof.

Here's an example of how this works. Suppose we want to show:

$$\neg A \vdash \neg(A \wedge B)$$

To prove this, we assume '$A \wedge B$', derive a contradiction from this assumption together with our premise, and then conclude '$\neg(A \wedge B)$' by ¬I, as follows:

$$\begin{array}{r|lll}
1 & \neg A & & \\
\cline{2-2}
2 & & \multicolumn{1}{|l}{A \wedge B} & \\
\cline{3-3}
3 & & \multicolumn{1}{|l}{A} & \wedge\text{E } 2 \\
4 & & \multicolumn{1}{|l}{\bot} & \neg\text{E } 1, 3 \\
5 & \neg(A \wedge B) & & \neg\text{I } 2\text{–}4
\end{array}$$

The strategy of reasoning by *reductio ad absurdum* can take another form too, however. The ¬I rule says that in order to show that some sentence $\mathscr{A}$ is false (i.e. that $\neg\mathscr{A}$ holds), we can assume $\mathscr{A}$ and reduce that to a contradiction. But instead of using *reductio* to show that something is false, we can also use it to show that something is true: to prove that $\mathscr{A}$ is true, suppose that it is false (i.e. that $\neg\mathscr{A}$ holds), and reduce that to a contradiction.

Interestingly, the rules we have assembled won't yet let us replicate this second style of reductio reasoning. So we'll add it to our system of deduction as as one last primitive rule, which we'll call *indirect proof*:

$$\begin{array}{r|lll}
i & & \multicolumn{1}{|l}{\neg\mathscr{A}} & \\
\cline{3-3}
 & & \multicolumn{1}{|l}{\vdots} & \\
j & & \multicolumn{1}{|l}{\bot} & \\
 & \mathscr{A} & & \text{IP } i\text{–}j
\end{array}$$

This is called indirect proof because it lets us prove $\mathscr{A}$ "indirectly", by assuming its negation and deriving a contradiction from that. This is a very powerful rule, because any proof whatsoever can in principle be done using IP as the overall strategy! Just assume the negation of whatever conclusion you're trying to prove, derive a contradiction from that. But be careful: proofs done in this indirect way tend to be longer and more complicated. So IP should only be used as a *last resort*, when you're sure that there's no other way to complete the proof!

Here is an example of something that can *only* be proven using IP:

$$P \wedge \neg P \vdash Q$$

The corresponding entailment $P \wedge \neg P \vDash Q$ holds: there is no valuation that makes the premise true and the conclusion false, simply because no valuation makes the premise true. So if our system of deduction is to be complete, we had better be able to provide a natural deduction proof of this as well. And we can, as follows:

1	$P \wedge \neg P$	
2	$\quad \neg Q$	
3	$\quad P$	1 $\wedge$E
4	$\quad \neg P$	1 $\wedge$E
5	$\quad \bot$	3,4 $\neg$E
6	Q	2-5 IP

This proof illustrates the EXPLOSION PRINCIPLE: from a contradiction, like $P \wedge \neg P$, anything whatsoever can be proven. We here proved Q, but exactly the same sequence of steps would have equally well let us prove A, or B, or $A \rightarrow B$, or anything else. The principle can be stated in general terms as $\bot \vdash \mathscr{A}$.

Another thing that can only be proven using IP is the LAW OF EXCLUDED MIDDLE, which says that $\vdash \mathscr{A} \vee \neg\mathscr{A}$, i.e. that any statement of the form $\mathscr{A} \vee \neg\mathscr{A}$ is a theorem. Proponents of *intuitionistic logic* reject the Law of Excluded Middle, because they reject the assumption we've been making throughout this book, that for any statement, it or its negation must be true. So they must reject our rule IP, since it lets us prove Excluded Middle.

And other logicians reject the Explosion Principle. For example, proponents of *relevance logic* hold that there must always be some "relevant connection" between the premises and conclusion of a valid argument — something that wouldn't hold if Explosion arguments are valid. And proponents of *paraconsistent logic* hold the view that some contradictions are true, and that accepting a contradiction should therefore not allow you to infer anything whatsoever. So they reject the Explosion Principle, and therefore have to use a different set of rules that doesn't prove it.

Intuitionistic logic, relevance logic, and paraconsistent logic are all varieties of NON-CLASSICAL LOGIC. However, in CLASSICAL LOGIC, which is what we are here studying, both the Law of Excluded Middle and the Explosion Principle hold. Since we can't prove these without IP, we have to add this rule into our natural deduction system so as to render it complete with respect to classical logic. This rule, together with the various introduction and elimination rules we've covered, form the basic rules of our system of natural deduction.

4.10 Proof strategies

There is no simple recipe for proofs, and there is no substitute for practice. Here, though, are some strategies to keep in mind.

Work backwards from your goal. The ultimate goal is to obtain the conclusion. Look at the conclusion and ask what the introduction rule is for its main logical operator. This gives you an overall strategy, and tells you what should happen *just before* the last line of the proof. Then you can treat this line as your new goal. Now ask what *its* main operator is, thereby identifying a strategy to prove it, and so on.

For example: If your conclusion is a conditional $\mathscr{A} \rightarrow \mathscr{B}$, plan to use the $\rightarrow$I as your strategy. This requires opening a subproof in which you assume $\mathscr{A}$. The subproof ought to end with $\mathscr{B}$. So now your goal is to prove $\mathscr{B}$ inside the subproof. Or if your conclusion is a negated sentence $\neg\mathscr{A}$, plan to use $\neg$I. That means opening a subproof where you assume $\mathscr{A}$ and which ends with $\bot$. So now your new goal is to prove $\bot$ inside the subproof.

Work forwards from what you have. When you are starting a proof, look at the premises. Think about the elimination rules for the main operators of these sentences, and note whether there are any obvious consequences you can draw. For example, if you have a conjunction, you know you can get either conjunct any time you want using $\wedge$E.

Importantly, if you see a disjunction $\mathscr{A} \vee \mathscr{B}$ among your premises, or as something that you can easily derive from your premises (perhaps in addition with assumptions you've already made), it's almost always a good strategy to set up an $\vee$E proof. That means opening up two subproofs, one that begins with an assumption of $\mathscr{A}$ and another that begins with $\mathscr{B}$. Inside each of them, you now have to prove whatever your current goal is.

Try an indirect proof. If you can't find any way to prove your goal $\mathscr{A}$ directly, try an indirect proof: assume $\neg\mathscr{A}$ and try to derive a contradiction. If you succeed, you can infer your goal $\mathscr{A}$ by IP. This strategy should only be used as a last resort, however! Indirect proofs are often longer and more complicated than direct proofs.

Persist. Try different things. If one approach fails, try something else. I'll never ask you to prove something that cannot be proven.

Let's look at one more example. Take the following English argument:

> If Guatemala is in Canada, then it is in North America. So if Guatemala is not in North America, it also isn't in Canada.

This is intuitively valid, so we should be able to give a proof of it. We can symbolize the argument as: $C \rightarrow A \therefore \neg A \rightarrow \neg C$. Our goal here is to prove a conditional, '$\neg A \rightarrow \neg C$'. So we use $\rightarrow$I as our strategy, meaning we assume '$\neg A$' and set ourselves the new goal of proving '$\neg C$' from that assumption. And now, since '$\neg C$' has a negation as its main operator, we use $\neg$I as our strategy, meaning we assume 'C' and prove a contradiction from that.

Notice how I reasoned "backwards" from the conclusion in order to discover this strategy. Written out, the proof looks like this:

```
1 │ C → A
  ├──────
2 │ │ ¬A
  │ ├────
3 │ │ │ C
  │ │ ├───
4 │ │ │ A          →E 1, 3
5 │ │ │ ⊥          ¬E 2, 4
6 │ │ ¬C           ¬I 3–5
7 │ ¬A → ¬C        →I 2–6
```

■ Exercises 4.10

A. The following three proofs are missing their rule citations. Add them, to turn them into bona fide proofs. Additionally, write down the sequent (i.e. single-turnstile $\vdash$ statement) that each proof demonstrates.

$$
\begin{array}{r|l}
1 & P \land S \\
2 & S \to R \\
\hline
3 & P \\
4 & S \\
5 & R \\
6 & R \lor E
\end{array}
$$

$$
\begin{array}{r|l}
1 & A \to D \\
\hline
2 & \quad \vert\; \underline{A \land B} \\
3 & \quad \vert\; A \\
4 & \quad \vert\; D \\
5 & \quad \vert\; D \lor E \\
6 & (A \land B) \to (D \lor E)
\end{array}
$$

$$
\begin{array}{r|l}
1 & \neg L \to (J \lor L) \\
2 & \neg L \\
\hline
3 & J \lor L \\
4 & \quad \vert\; \underline{J} \\
5 & \quad \vert\; J \land J \\
6 & \quad \vert\; J \\
7 & \quad \vert\; \underline{L} \\
8 & \quad \vert\; \bot \\
9 & \quad \vert\; J \\
10 & J
\end{array}
$$

B. Prove each of the following sequents:

1. $J \to \neg J \vdash \neg J$
2. $Q \to (Q \land \neg Q) \vdash \neg Q$
3. $P \lor Q, \neg P \vdash Q$
4. $\neg F \to G, F \to H \vdash G \lor H$
5. $D \vdash \neg\neg D$
6. $P \land (Q \lor R), P \to \neg R \vdash Q \lor E$
7. $\neg C \lor (A \to B) \vdash (C \land A) \to B$
8. $C \to (E \land G), \neg C \to G \vdash G$
9. $M \land (\neg N \to \neg M) \vdash (N \land M) \lor \neg M$
10. $(W \lor X) \lor (Y \lor Z), X \to Y, \neg Z \vdash W \lor Y$

C. Show that the following are provably equivalent:

1. $\neg(P \land Q) \dashv\vdash \neg P \lor \neg Q$
2. $\neg(P \lor Q) \dashv\vdash \neg P \land \neg Q$
3. $P \to Q \dashv\vdash \neg Q \to \neg P$
4. $P \to Q \dashv\vdash \neg P \lor Q$
5. $\neg(P \to Q) \dashv\vdash P \land \neg Q$
6. $\neg P \leftrightarrow Q \dashv\vdash \neg(P \leftrightarrow Q)$

D. Prove the following theorems:

1. $\vdash \neg A \to (A \to B)$
2. $\vdash J \leftrightarrow [J \vee (L \wedge \neg L)]$
3. $\vdash (A \to B) \vee (B \to A)$
4. $\vdash A \vee \neg A$
5. $\vdash ((P \to Q) \to P) \to P$

E. We noted that the Law of Excluded Middle and the Explosion Principle can only be proven with IP. Another thing that can, somewhat surprisingly, only be proven via IP is *double negation elimination*: $\neg\neg\mathscr{A} \vdash \mathscr{A}$. Indeed, we could have added this as the final rule in our proof system, instead of IP, and thereby also obtained a complete proof system:

m	$\neg\neg\mathscr{A}$	
	$\mathscr{A}$	m DNE

Do two things. First, give a proof of:

$$\neg\neg A \vdash A$$

using IP, showing that it can be used in place of DNE. Second, show that DNE can similarly be used in place of IP. In particular, give a proof of the following using DNE, but without using IP:[2]

$$\neg A \to \bot \vdash A$$

4.11 Derived Rules

We have introduced the introduction and elimination rules for each of our five connectives. Together with IP, this gives us a complete proof system: every valid argument can be proven using just the rules we have. In this section, we're going to introduce some additional rules to shorten our proofs and make our proof system easier to work with. It's important to note at the outset that these additional rules are not necessary. They represent a *conservative* extension of our proof system: anything proven using these new rules can also be proven using just our basic set of rules.

To illustrate the motivation for additional rules, consider the following argument:

Alice is either a logician or a chemist. She is not a chemist. So she is a logician.

This involves a very natural form of inference called *Disjunctive Syllogism*. We could symbolize the argument as $L \vee C, \neg C \therefore L$, and we then give a natural deduction proof to show that it is valid.

But now consider this: by giving a proof of 'L' from '$L \vee C$' and '$\neg C$', we have implicitly shown that given *any* sentences of the form $\mathscr{A} \vee \mathscr{B}$ and $\neg\mathscr{B}$, it is possible to prove $\mathscr{A}$. If we

[2]Perhaps think about how you would prove this using IP first. Then think about how to prove it using DN, but without using IP. Hint: you'll have to use $\neg$I along the way.

substitute the metavariables $\mathscr{A}$ and $\mathscr{B}$ for the sentences 'L' and 'C' in our proof, we get a PROOF TEMPLATE for the disjunctive syllogism form of inference:

$$\begin{array}{l|lll}
m & (\mathscr{A} \vee \mathscr{B}) & & \\
n & \neg\mathscr{A} & & \\
k_0 & \vert\ \underline{\mathscr{A}} & & \\
k_1 & \vert\ \vert\ \underline{\neg\mathscr{B}} & & \\
k_2 & \vert\ \vert\ \bot & & \neg\text{E } n, k_0 \\
k_3 & \vert\ \mathscr{B} & & \text{IP } k_0\text{–}k_3 \\
k_4 & \vert\ \underline{\mathscr{B}} & & \\
k_5 & \vert\ \mathscr{B} \wedge \mathscr{B} & & \wedge\text{I } k_4, k_4 \\
k_6 & \vert\ \mathscr{B} & & \wedge\text{E } k_5 \\
k_7 & \mathscr{B} & & \vee\text{E } m, k - k_3, k_4 - k_6
\end{array}$$

Now, if at any time, in the context of any proof whatsoever, we face the task of proving some sentence $\mathscr{A}$ from two sentences of the form $\mathscr{A} \vee \mathscr{B}$ and $\neg\mathscr{B}$, we can simply "slot in" an instance of the above proof template. In other words, once we've proven one instance of disjunctive syllogism, we can use that as a template to prove disjunctive syllogism again in the context of any other proof.

Given this, we might as well just introduce a derived rule into our proof system that lets us skip the actual proof and make the disjunctive syllogism inference *directly*:

$$\begin{array}{l|lll}
m & \mathscr{A} \vee \mathscr{B} & & \\
n & \neg\mathscr{A} & & \\
 & \mathscr{B} & & \text{DS } m, n
\end{array}$$

DS is a DERIVED RULE in the sense that it can be shown to hold using only the *primitive* rules of our system. You can think of derived rules like promissory notes: "I am here justifying my inference by writing 'DS', but I promise that, if you demanded it, I could slot in a series of steps using only the primitive rules of our natural deduction system." Derived rules shorten our proofs, but add no power into our proof system — any proof that appeals to a derived rule could be expanded into one that only appeals to primitive rules, by slotting in an instance of the proof template corresponding to the derived rule.

In fact, we implicitly already added a derived rule to our system in §4.6 when we introduced Reiteration. Reiteration is just a shortcut to let us skip the $\wedge$I+$\wedge$E trick that I used in steps k_5 and k_6 in the above proof template.[3] There are many further useful derived rules we can add to our proof system. For example, consider the following argument:

[3]Indeed, in this particular case we could have avoided using the $\wedge$I+$\wedge$E trick, and shortened our proof template, by treating line $k+4$ as a whole subproof that begins with $\mathscr{B}$ and ends with $\mathscr{B}$ (see the end of §4.6).

> If Alice is a chemist, then she has a PhD. Alice does not have a PhD. So she is not a chemist.

This inference pattern is called *modus tollens*, and we can introduce a derived rule for it:

$$\begin{array}{r|ll}
m & \mathscr{A} \to \mathscr{B} & \\
n & \neg\mathscr{B} & \\
 & \neg\mathscr{A} & \text{MT } m, n
\end{array}$$

Again, this adds no power to our system because it is simply a shortcut for a series of steps involving only primitive rules, as illustrated by the following proof template:

$$\begin{array}{r|l|ll}
m & \multicolumn{2}{l}{\mathscr{A} \to \mathscr{B}} & \\
n & \multicolumn{2}{l}{\neg\mathscr{B}} & \\
k_0 & & \mathscr{A} & \\ \cline{3-3}
k_1 & & \mathscr{B} & \to\text{E } m, k_0 \\
k_2 & & \bot & \neg\text{E } k_1, n \\
k_3 & \multicolumn{2}{l}{\neg\mathscr{A}} & \neg\text{I } k_0\text{–}k_2
\end{array}$$

In §4.10 we gave a seven step proof showing that $C \to A \therefore \neg A \to \neg C$ is valid. Using our derived rule MT, we can now shorten this to just four steps:

$$\begin{array}{r|l|ll}
1 & \multicolumn{2}{l}{C \to A} & \\ \hline
2 & & \neg A & \\ \cline{3-3}
3 & & \neg C & \text{MT } 1, 2 \\
4 & \multicolumn{2}{l}{\neg A \to \neg C} & \to\text{I } 2\text{–}3
\end{array}$$

Here is a complete list of the derived rules that you'll be able to use in natural deduction proofs from this point forward:

Sequent:	**Derived Rule:**
$\mathscr{A} \vee \mathscr{B}, \neg\mathscr{B} \vdash \mathscr{A}$	DS
$\mathscr{A} \vee \mathscr{B}, \neg\mathscr{A} \vdash \mathscr{B}$	DS
$\mathscr{A} \rightarrow \mathscr{B}, \neg\mathscr{B} \vdash \neg\mathscr{A}$	MT
$\mathscr{A} \vdash \mathscr{B} \rightarrow \mathscr{A}$	PMI
$\neg\mathscr{A} \vdash \mathscr{A} \rightarrow \mathscr{B}$	PMI
$\mathscr{A} \rightarrow \mathscr{B} \dashv\vdash \neg\mathscr{A} \vee \mathscr{B}$	Imp
$\neg(\mathscr{A} \rightarrow \mathscr{B}) \dashv\vdash \mathscr{A} \wedge \neg\mathscr{B}$	NegImp
$\neg(\mathscr{A} \wedge \mathscr{B}) \dashv\vdash \neg\mathscr{A} \vee \neg\mathscr{B}$	DeM
$\neg(\mathscr{A} \vee \mathscr{B}) \dashv\vdash \neg\mathscr{A} \wedge \neg\mathscr{B}$	DeM
$\mathscr{A} \dashv\vdash \neg\neg\mathscr{A}$	DN
$\mathscr{A} @ \mathscr{B} \vdash \mathscr{B} @ \mathscr{A}$	Com
$\bot \vdash \mathscr{A}$	EX
$\vdash \mathscr{A} \vee \neg\mathscr{A}$	LEM

(where @ can be any of the three commutative connectives $\vee, \wedge, \leftrightarrow$). The way understand this list of derived rules is as follows:

▷ For any sequent $\mathscr{A}_1 \ldots \mathscr{A}_n \vdash \mathscr{C}$, if it has the form of one of the sequents on the above list, and if sentences $\mathscr{A}_1 \ldots \mathscr{A}_n$ occur on lines $j_1 \ldots j_n$ in your proof (none of them inside a closed subproof), then you may directly infer $\mathscr{C}$ and justify it by citing the name of the relevant derived rule followed by the numbers of lines $j_1 \ldots j_n$.

For example, if you have the sentence 'P' on a line numbered m in your proof, then you are now allowed to directly infer '$Q \rightarrow P$' with the justification 'PMI m' (for "Paradox of Material Implication"), or to infer '$\neg\neg P$' with the justification 'DN m' (for "Double Negation"). Or if you have a sentence '$\neg K \vee \neg L$' on a line m in you proof, you may directly infer '$\neg(K \wedge L)$' with the justification 'DeM m' (for "DeMorgan's Law"), or you may infer '$K \rightarrow \neg L$' with the justification 'Imp m'. The second-to-last derived rule, EX, is the Explosion Principle: it lets you infer any sentence whatsoever from a contradiction. And the last derived rule, LEM, is the Law of Excluded Middle: it lets you write down any sentence of the form $\mathscr{A} \vee \neg\mathscr{A}$ at any point in your proof.

For another example of these rules in action, consider the following theorem:

$$\vdash (P \rightarrow Q) \vee (Q \rightarrow P)$$

This was one of the exercises in §4.10. Proving this using only basic rules is quite difficult, as you will have noticed if you tried that exercise. With derived rules, we can give a much quicker and more intuitive proof of this theorem, by starting out with an instance of the Law of Excluded Middle and then pursuing an $\vee$E strategy:

1	$P \vee \neg P$	LEM
2	$\quad\mid\ \underline{P}$	
3	$\quad\mid\ Q \rightarrow P$	PMI 2
4	$\quad\mid\ (P \rightarrow Q) \vee (Q \rightarrow P)$	$\vee$I 3
5	$\quad\mid\ \underline{\neg P}$	
6	$\quad\mid\ P \rightarrow Q$	PMI 5
7	$\quad\mid\ (P \rightarrow Q) \vee (Q \rightarrow P)$	$\vee$I 6
8	$(P \rightarrow Q) \vee (Q \rightarrow P)$	$\vee$E 1,2-4,5-7

As an exercise, you might try to re-write this proof by "slotting in" a subproof involving only primitive rules wherever the above proof appeals to a derived rule. This involves showing how LEM, and the two versions of PMI, can be proven using only primitive rules of our system.

■ Exercises 4.11

A. The following proofs are missing their citations (rule and line numbers). Add them wherever they are required:

1	$\mid\ W \rightarrow \neg B$
2	$\mid\ A \wedge W$
3	$\mid\ \underline{B \vee (J \wedge K)}$
4	$\mid\ W$
5	$\mid\ \neg B$
6	$\mid\ J \wedge K$
7	$\mid\ K$

1	$\mid\ L \leftrightarrow \neg O$
2	$\mid\ \underline{L \vee \neg O}$
3	$\mid\quad\mid\ \underline{\neg L}$
4	$\mid\quad\mid\ \neg O$
5	$\mid\quad\mid\ L$
6	$\mid\quad\mid\ \bot$
7	$\mid\ L$

1	$\mid\ Z \rightarrow (C \wedge \neg N)$
2	$\mid\ \underline{\neg Z \rightarrow (N \wedge \neg C)}$
3	$\mid\quad\mid\ \underline{\neg(N \vee C)}$
4	$\mid\quad\mid\ \neg N \wedge \neg C$
5	$\mid\quad\mid\ \neg N$
6	$\mid\quad\mid\ \neg N \vee \neg\neg C$
7	$\mid\quad\mid\ \neg(N \wedge \neg C)$
8	$\mid\quad\mid\ \neg\neg Z$
9	$\mid\quad\mid\ Z$
10	$\mid\quad\mid\ \neg C$
11	$\mid\quad\mid\ \neg C \vee \neg\neg N$
12	$\mid\quad\mid\ \neg(C \wedge \neg N)$
13	$\mid\quad\mid\ \neg Z$
14	$\mid\quad\mid\ \bot$
15	$\mid\ N \vee C$

B. Prove the following sequents using derived rules:

1. $(A \vee B) \rightarrow C, \neg C \vdash \neg A$
2. $E \vee F, F \vee G, \neg F \vdash E \wedge G$
3. $\neg(A \rightarrow (B \vee \neg C)) \vdash (B \vee C) \rightarrow A$
4. $M \vee (N \rightarrow M) \therefore \neg M \rightarrow \neg N$
5. $A \rightarrow (B \vee C) \vdash (A \rightarrow B) \vee (A \rightarrow C)$
6. $(M \vee N) \wedge (O \vee P), N \rightarrow P, \neg P \therefore M \wedge O$
7. $(B \wedge C) \vee \neg(A \rightarrow \neg D), \neg C \vdash A \wedge D$
8. $(X \wedge Y) \vee (X \wedge Z), \neg(X \wedge D), D \vee M \therefore M$

If you want more practice, you can also re-do any of the earlier proofs in this chapter using derived rules.

C. Provide proof templates (like those I provided for DS and MT) that justify the addition of the De Morgan rules, and the Imp and NegImp rules, as derived rules. If you don't want to bother with metavariables, you can just prove instances of the sequents corresponding to DeM, Imp, and NegImp. But in any case, be sure to only use primitive rules in your proofs.

II

First-order logic

Symbolization in FOL 5

Consider the following argument, which is obviously valid in English:

Willard is a logician.
All logicians wear funny hats.
∴ Willard wears a funny hat.

To symbolize it in TFL, we could offer the following symbolization key:

L:	Willard is a logician.
A:	All logicians wear funny hats.
F:	Willard wears a funny hat.

The argument then gets symbolized as:

$$L, A \therefore F$$

But the truth-table test will indicate that this is *invalid*. What has gone wrong?

The problem is not that we have made a mistake while symbolizing the argument. This is the best symbolization we can give *in TFL*. The problem lies with TFL itself! This argument is not valid in virtue of its *truth-functional* structure, but rather in virtue of its *subsentential* structure. For example, 'All logicians wear funny hats' establishes a certain relationship between being a logician and hat-wearing. But in TFL, the best we can do is symbolize it as an atomic sentences. Because TFL doesn't let us represent any subsentential structure, we lose the connection between Willard's being a logician and Willard's wearing a hat in the TFL symbolization..

To symbolize arguments like the preceding one, we will have to develop a new logical language which will allow us to *split the atom*. That is to say: it will let us represent logically significant structure *inside* atomic sentences. This will be the language of *first-order logic*, or *FOL*.

The details of FOL will be explained throughout this chapter, but here is the basic idea for splitting the atom. A sentence like 'Willard is a logician' is internally composed of a name, 'Willard', and a predicate, '____is a logician'. In FOL, we'll use lowercase letters to symbolize names, and uppercase letters to symbolize predicates. So we might use '*a*' to symbolize the name 'Willard', and '*L*' to symbolize the predicate '____is a logician'. The whole sentence can then be symbolized as '*La*', thereby representing the fact that this atomic sentence is internally composed of a name and a predicate.

A sentence like 'All logicians wear funny hats' involves two predicates: '____is a logician' and '____wears funny hats'. It also involves the word 'all', which relates the two predicates. This is called a *quantifier*, because it tells us something about “quantities” — in this case that *all* individuals who are logicians wear funny hats, rather than just some of them. FOL will have two quantifiers, ‘$\forall$’ and ‘$\exists$’. ‘$\exists$’ will roughly convey ‘There is at least one thing such that . . . ’ and ‘$\forall$’ will convey ‘Every thing is such that . . . ’.

In sum, FOL has three new ingredients: names, predicates, and quantifiers. And we can use these ingredients to represent the internal structure of atomic sentences. That is the general idea. But FOL is significantly more complex than TFL, so we’ll build up slowly.

5.1 Names and Predicates

In English, a *singular term* is a word or phrase that refers to a *specific* person, place, or thing. The word ‘dog’ is not a singular term, because there are a great many dogs. The phrase ‘Bertie’ is a singular term, because it refers to a specific terrier. Likewise, the phrase ‘Philip’s dog Bertie’ is a singular term, because it refers to a specific little terrier.

Proper names are a particularly important kind of singular term. These are expressions that pick out individuals without describing them. The name ‘Emerson’ is a proper name, and the name alone does not tell you anything about Emerson. Of course, some names are traditionally given to boys and other are traditionally given to girls. If ‘Hilary’ is used as a singular term, you might guess that it refers to a woman. You might, though, be guessing wrongly. Indeed, the name does not necessarily mean that the person referred to is even a person: Hilary might be a giraffe, for all you could tell just from the name.

In FOL, our NAMES are lower-case letters ‘a’ through to ‘r’. We can add subscripts if we want to use some letter more than once. So the following are all names in FOL:

$$a, b, c, \ldots, r, a_1, f_{32}, j_{390}, m_{12}$$

These should be thought of along the lines of proper names in English. But with one difference. ‘Syracuse’ is a proper name, but it is the name of both a city in New York State and of a city in Italy. And there are over thirty towns in the US that have the name ‘Springfield’. We live with this kind of ambiguity in English, allowing context to determine that ‘Syracuse’ is being used to refer to a city in the US rather than to one in Italy. In FOL, we do not tolerate any such ambiguity. Each name must pick out *exactly* one thing. (However, two different names may pick out the same thing.)

As with TFL, we’ll provide symbolization keys. These indicate, temporarily, what object a name will pick out. So we might offer:

e:	Elsa
g:	Gregor
m:	Marybeth

The second ingredient in FOL are predicates. The simplest predicates express properties of individuals. They are things you can say about an object. Here are some examples of English predicates:

____ is a dog
____ is a member of Monty Python
A piano fell on ____

In general, you can think about predicates as things which combine with singular terms to make sentences. Conversely, you can start with sentences and make predicates out of them by removing terms. Consider the sentence, 'Vinnie borrowed the family car from Nunzio.' By removing a singular term, we can obtain any of three different predicates:

> ____ borrowed the family car from Nunzio
> Vinnie borrowed ____ from Nunzio
> Vinnie borrowed the family car from ____

FOL predicates are capital letters A through Z, with or without subscripts. We might write a symbolization key for predicates like this:

> A: ____ is angry
> H: ____ is happy

If we combine our two symbolization keys, we can start to symbolize some English sentences that use these names and predicates in combination. For example, consider the English sentences:

(1) Elsa is angry.
(2) Gregor and Marybeth are angry.
(3) If Elsa is angry, then so are Gregor and Marybeth.

Sentence (1) is just symbolized as 'Ae'. Sentence (2) is a conjunction of two simpler sentences. The simple sentences can be symbolized just by 'Ag' and 'Am'. Then we help ourselves to our resources from TFL, and symbolize the entire sentence by '$Ag \wedge Am$'. This illustrates an important point: FOL has all of the truth-functional connectives of TFL! Lastly, sentence (3) is a conditional, whose antecedent is sentence (1) and whose consequent is sentence (2). So we can symbolize it as '$Ae \rightarrow (Ag \wedge Am)$'.

We can also use TFL connectives to symbolize sentences that involve COMPOUND PREDICATES, that is, predicates formed out of simpler ones. Consider the following sentence:

(4) Herbie is a white car

It involves the compound predicate '____is a white car'. But we can paraphrase the sentence as a conjunction involving simpler predicates: 'Herbie is white and Herbie is a car'. Using the following symbolization key:

> W: ____ is white
> C: ____ is a car
> h: Herbie

we can symbolize (4) as '$Wh \wedge Ch$'.

In this case, the compound predicate was formed out of an adjective and a noun. But there are other ways to do this too:

(5) Herbie is a car from Germany.
(6) Herbie is a car that is fast.

Sentence (5) involves a compound predicate formed from a noun and the prepositional phrase ‘from Germany’, and (6) involves a compound predicate formed from a noun and the relative clause ‘that is fast’. These can also be symbolized as conjunctions of simple predications. Using ‘G’ for ‘____is from Germany’ and ‘F’ for ‘____is fast’, (5) can be symbolized as ‘$Ch \wedge Gh$’ and (6) as ‘$Ch \wedge Fh$’.

One does, occasionally have to be careful when symbolizing compound predicates. Consider the following:

(7) Damon Stoudamire is a short basketball player.
(8) Damon Stoudamire is a man.
(9) Damon Stoudamire is a short man.

We might symbolize (7) as ‘$Sd \wedge Bd$’, sentence (8) as ‘Md’, and sentence (9) as ‘$Sd \wedge Md$’. But that would be a mistake. For this now suggests that sentences (7) and (8) together entail sentence (9). But they do not. At 5’10”, Damon Stoudamire is one of the shortest professional basketball players of all time, but he is nevertheless an averagely-tall man. The point is that sentence (7) says that Damon is short *for a basketball player*, even though he is of average height *for a man*. So here you’d need to symbolize ‘____ is a short basketball player’ and ‘____ is a short man’ using different predicate letters.

5.2 Quantifiers

Next up are quantifiers. Consider these sentences:

(10) Everyone is happy.
(11) Someone is angry.

Sentence (10) superficially looks like it has the same kind of structure as something like ‘Elsa is happy’. So you might be tempted to symbolize (10) as ‘He’, with the explanation that ‘e’ is to symbolize ‘everyone’. But that would be a serious mistake.

‘Everyone’ is not a proper name — it doesn’t pick out any particular individual — and so it should not be symbolized using a name like ‘e’ in FOL. The word ‘everyone’ is rather a quantifier. Logically, quantifiers behave very differently from names. For example, whereas ‘Either Elsa is happy or Elsa is not happy’ is a necessary truth, ‘Either everyone is happy or everyone is not happy’ is not a necessary truth. In fact, it’s presumably false: some people are happy and others are not.

To express claims about every individual in a set, we’ll use the FOL symbol ‘$\forall$’. This is called the UNIVERSAL QUANTIFIER. A quantifier in FOL always has to be followed by a variable. FOL variables are italic lowercase letters ‘s’ through ‘z’, with or without subscripts. So we might symbolize sentence (10) as ‘$\forall x Hx$’. The variable ‘x’ is serving as a kind of placeholder. The expression ‘$\forall x$’ intuitively means that you can pick anyone and put them in as ‘x’. The subsequent ‘Hx’ indicates, of that thing you picked out, that it is happy. So ‘$\forall x Hx$’ can be read as saying “every individual x is such that x is happy.”

I should say that there is no special reason to use ‘x’ rather than some other variable. The sentences ‘$\forall x Hx$’, ‘$\forall y Hy$’, ‘$\forall z Hz$’, and ‘$\forall x_5 Hx_5$’ use different variables, but they are all logically equivalent, and any of them could be used to symbolize (10).

To symbolize sentence (11), we introduce another new symbol: the EXISTENTIAL QUANTIFIER, '$\exists$'. Like the universal quantifier, the existential quantifier requires a variable. Sentence 11 can be symbolized by '$\exists xAx$'. You can read '$\exists xAx$' as saying "there is some individual x such that x is angry'. Once again, the variable is just a kind of placeholder; we could just as easily have symbolized sentence (11) with '$\exists zAz$', '$\exists w_{256}Aw_{256}$', or whatever.

Some more examples will help. Consider these sentences:

(12) No one is angry.
(13) There is someone who is not happy.
(14) Not everyone is happy.

Sentence (12) can be paraphrased as, 'It is not the case that someone is angry'. We can therefore symbolize it using negation together with an existential quantifier: '$\neg\exists xAx$'. On the other hand, (12) could also be paraphrased as 'Everyone is not angry'. With this in mind, it can be symbolized using negation and a universal quantifier: '$\forall x\neg Ax$' (you can read this as "every individual x is such that x is not angry"). Both of these are acceptable symbolizations. Indeed, as we will see, it holds in general that $\forall\mathscr{v}\neg\mathscr{A}$ is logically equivalent to $\neg\exists\mathscr{v}\mathscr{A}$. (Notice that I have here returned to the practice of using '$\mathscr{A}$' as a metavariable, now over FOL sentences; and $\mathscr{v}$ as a metavariable over FOL variables.) Symbolizing a sentence one way, rather than the other, might seem more 'natural' in some contexts, but it is not much more than a matter of taste.

Sentence (13) is most naturally paraphrased as, 'There is some x, such that x is not happy'. This then becomes '$\exists x\neg Hx$'. Of course, if there is someone who's not happy, then it's not the case that everyone is happy, and vice versa. So we could equally well have symbolized (13) as '$\neg\forall xHx$'. And that is also a perfectly adequate symbolization of sentence (14). This illustrates that in general $\neg\forall\mathscr{v}\mathscr{A}$ is equivalent to $\exists\mathscr{v}\neg\mathscr{A}$. So we have:

QUANTIFIER EQUIVALENCE LAWS:

$\forall\mathscr{v}\neg\mathscr{A}$ is equivalent to $\neg\exists\mathscr{v}\mathscr{A}$

$\neg\forall\mathscr{v}\mathscr{A}$ is equivalent to $\exists\mathscr{v}\neg\mathscr{A}$

Given the symbolization key we have been using, '$\forall xHx$' symbolizes 'Everyone is happy'. Who is included in this *everyone*? When we use sentences like this in English, we usually do not mean everyone now alive on the Earth, let alone everyone who was ever alive or who will ever live. We usually mean something more modest: everyone now in the building, everyone enrolled in the ballet class, or whatever.

In order to eliminate this ambiguity, we will need to specify a DOMAIN. The domain is the set of things that we are talking about. So if we want to talk about people in Chicago, we define the domain to be people in Chicago. We write this at the beginning of the symbolization key, like this:

Domain: people in Chicago

The quantifiers *range over* the domain. So '$\forall x$' should really be read as "every object x *in the domain* is such that ..." and '$\exists x$' should be read as "there is some object x in the domain such that ...'. If the domain is people in Chicago, then '$\forall x$' can be read as "every object x which is a person in Chicago is such that ...", and similarly for the existential quantifier

In FOL, the domain must always include at least one thing. Moreover, since in English we can legitimately infer 'something is angry' from 'Gregor is angry', we will want to be able to infer '$\exists x Ax$' from 'Ag' in FOL. So we will require that each name must pick out exactly one thing in the domain. If we want to name people in places beside Chicago, then we need to specify a wider domain that includes those people.

A domain must have *at least* one member. Every name must pick out *exactly* one member of the domain. But a member of the domain may be picked out by one name, many names, or none at all.

We're going to be a bit more restrictive about domain selection later. But before we get to that we'll look at some more complex symbolizations.

5.3 Common Quantifier Phrases

Consider these sentences:

(15) Some dogs are poodles.
(16) Every dog is a canine.

Let's use the following symbolization key:

Domain:	animals
D:	____ is a dog
P:	____ is a poodle
C:	____ is a canine

Sentence (15) gets symbolized using an existential quantifier as '$\exists x(Dx \wedge Px)$'. You can read this as "there is some object x (in the domain of animals) such that x is a dog and x is a poodle", which does capture the intent of (15).

Sentence (16) gets symbolized using a universal quantifier. You might be tempted to to symbolize it as '$\forall x(Dx \wedge Cx)$', using a universal quantifier together with a conjunction, just as we used an existential quantifier together with a conjunction for (15). But that would be a mistake: '$\forall x(Dx \wedge Cx)$' means "every object x (in the domain) is such that x is a dog and x is a canine", or more simply, "every object (in the domain) is both a dog and a canine." That's not at all what (16) says!

To see the route towards the correct symbolization, notice that (16) can be paraphrased as "for any object x in the domain, *if* x is a dog, then x is a canine." So (16) gets symbolized using a universal quantifier together with a conditional, as '$\forall x(Dx \rightarrow Cx)$'. As general guidelines, we have the following:

A sentence can be symbolized as $\exists x(\mathcal{F}x \wedge \mathcal{G}x)$ if it can be paraphrased in English as 'some F is G'.

A sentence can be symbolized as $\forall x(\mathcal{F}x \rightarrow \mathcal{G}x)$ if it can be paraphrased in English as 'every F is G'.

Notice that just as it was essential to use a conditional (rather than a conjunction) with the universal quantifier, so it's essential to use a conjunction (rather than a conditional) with the existential. Suppose we had instead symbolized (15) as '$\exists x(Dx \to Px)$'. That would mean that there is some object in the domain of which '$(Dx \to Px)$' is true. Recall that, in TFL, $\mathscr{A} \to \mathscr{B}$ is tautologically equivalent to $\neg\mathscr{A} \lor \mathscr{B}$. This equivalence will also hold in FOL. So '$\exists x(Dx \to Px)$' is true if there is some object in the domain, such that '$(\neg Dx \lor Px)$' is true of that object, that is, if some animal is *either* not a dog *or* is a poodle. And of course there are lots of animals that are not dogs! So it is *far too easy* for '$\exists x(Tx \to Dx)$' to be true, and it doesn't at all capture the meaning of (15).

The same patterns apply to quantified sentences that involve compound predicates:

(17) Some pet dogs are poodles.
(18) Every wild dog is a canine.

Let's use 'T' for '____is a pet' and 'W' for '____is wild'. Recall from §5.1 that in general, compound predicates get symbolized as conjunctions of the component predicates. So we can symbolize (17) as $\exists x((Tx \land Dx) \land Px)$, with the conjunction '$(Tx \land Dx)$' symbolizing the compound predicate 'is a pet dog'. Similarly, (18) gets symbolized as '$\forall x((Wx \land Dx) \to Cx)$', with '$(Wx \land Dx)$' symbolizing the compound predicate 'is a wild dog'. This can be read as: every object x (in the domain) is such that *if* x is wild and x is a dog, *then* x is a canine.

When symbolizing more complex sentences like these, it is helpful to distinguish the RESTRICTOR PREDICATE from the MAIN PREDICATE:

$$\text{Some } \underbrace{\text{pet dogs}}_{\text{Restrictor}} \underbrace{\text{are poodles}}_{\text{Main}}$$

$$\text{Every } \underbrace{\text{wild dog from Africa}}_{\text{Restrictor}} \underbrace{\text{is a canine}}_{\text{Main}}$$

Intuitively, the restrictor predicate tells you what class of things the sentence says something about — e.g. pet dogs, or wild dogs from Africa — and the main predicate then says what is true of some or all of them. As we've seen, in the case of an existential quantifier, the restrictor predicate and main predicate get connected by a conjunction. And in the case of a universal quantifier, the restrictor predicate and main predicate get connected by a conditional. So if we use 'A' for '____is from Africa', the second of the above sentences gets symbolized as '$\forall x(((Wx \land Dx) \land Ax) \to Cx)$': for every object x in the domain, *if* x is wild and x is a dog and x is from Africa, *then* x is a canine.

Next consider the following examples involving "negative" quantifiers:

(19) Not all dogs are poodles.
(20) No dog is a poodle.

Sentence (19) can be paraphrased as 'It is not the case that every dog is a poodle'. So it just gets symbolized as the negation of a universal sentence: '$\neg\forall x(Dx \to Px)$'. Sentence (20) means something very different: it doesn't just say that it's not the case that *every* dog is a poodle (which is true), but that it is not the case that there exists even one dog that is a poodle (which is false). Accordingly, it gets symbolized as the negation of an existential sentence: $\neg\exists x(Dx \wedge Px)$.

You may have noticed that there are other ways to paraphrase these two sentences. Sentence (19) could also be paraphrased as saying ‘Some dog (i.e. at least one) is not a poodle’. So instead of symbolizing it as a negated universal ‘$\neg\forall x(Dx \rightarrow Px)$’, it could also be symbolized as an existential: $\exists x(Dx \wedge \neg Px)$’.

Similarly, sentence (20) could also be paraphrased as ‘every dog is a non-poodle’. So instead of symbolizing it as a negated existential ‘$\neg\exists x(Dx \wedge Px)$’, it could also be symbolized as a universal: $\forall x(Dx \rightarrow \neg Px)$. In both of these cases, the two possible symbolizations are equivalent to each other. We’ll learn how to use natural deduction to prove this later, but you can already see the reason for the equivalence given the Quantifier Equivalence Laws introduced in §5.2 together with some of the equivalences we know from TFL.

First, the original symbolization ‘$\neg\forall x(Dx \rightarrow Px)$’ of (19) is, by the Quantifier Equivalence Laws, equivalent to ‘$\exists x\neg(Dx \rightarrow Px)$’. Then, applying NegImp inside the scope of the existential, the latter is equivalent to ‘$\exists x(Dx \wedge \neg Px)$’, which was our second possible symbolization. Similarly, by the Quantifier Equivalence Laws, the original symbolization ‘$\neg\exists x(Dx \wedge Px)$’ of (20) is equivalent to ‘$\forall x\neg(Dx \wedge Px)$’. Applying DeMorgan’s Law inside the scope of the universal then gives us ‘$\forall x(\neg Dx \vee \neg Px)$’, and the latter is, by applying Imp inside the scope of the universal, then equivalent to ‘$\forall x(Dx \rightarrow \neg Px)$’, which was our second possible symbolization.

5.4 Picking a domain

So far we’ve been pretty loose about picking domains. But in practice, picking a domain can be a delicate matter, and can affect what the appropriate symbolization of a sentence is. Suppose we want to symbolize the English sentence:

(21) Every dog is a poodle.

Let’s again use ‘D’ for ‘____is a dog’ and ‘P’ for ‘____is a poodle’. Now suppose I pick animals living on my street as the domain. Then ‘$\forall x(Dx \rightarrow Px)$’ would only convey the claim that every dog *living on my street* is a poodle. But ordinarily, someone who asserts (21) would be taken to have made a stronger claim, probably about dogs in general. So by picking a domain that includes only animals living on my street, we have failed to capture the intended meaning of the English sentence: it might be true that all dogs on my street are poodles, but on its intended meaning (21) says something false. To rectify that problem, we have to pick a bigger domain, like all animals. Relative to this larger domain, ‘$\forall x(Dx \rightarrow Px)$’ now captures the intended meaning of (21). The moral is that by picking the wrong domain, you may fail to capture important aspects of the meaning of the English sentence you set out to symbolize.

At this point you might wonder what happens if we pick a domain that just consists of dogs. In that case, the symbolization of (21) can be radically simplified, to just ‘$\forall xPx$’. After all, ‘$\forall xPx$’ says that every objects x in the domain — which in this case consists of all dogs — is a poodle, which is indeed the claim that (21) is making.

In fact, you could in principle always avoid complex symbolizations by just restricting your domain appropriately. A sentence that standardly gets symbolized via the pattern

$\forall x(\mathcal{F}x \rightarrow \mathcal{G}x)$ could just be symbolized $\forall x\mathcal{G}x$ by making the domain consist of whatever things the restrictor predicate $\mathcal{F}x$ is true of. And similarly, a sentence that standardly gets symbolized via the pattern $\exists x(\mathcal{F}x \wedge \mathcal{G}x)$ could just be symbolized as $\exists x\mathcal{G}x$ by picking a domain that consists of whatever things $\mathcal{F}x$ is true of.

However, this gain in convenience comes at a cost. If we pick a domain of dogs in our symbolization key for (21), then we can no longer use the same symbolization key to symbolize sentences that talk about things other than dogs. That could be a problem. For example, 'Every dog is a poodle' might appear as the first premise in an argument, whose second premise is 'No pelican is a poodle', and whose conclusion is 'No pelican is a dog'. Since the second premise and the conclusion talk about pelicans, we can't symbolize them using a domain of just dogs. We'll instead have to go back to using a larger domain, like all animals, and revert to the more complex symbolization of 'Every dog is a poodle'.

In fact, we're going to be fairly restrictive about what domains to use. In order to standardize symbolizations, you should always pick one of two domains in your symbolization key: either a domain that consists of things in general (dogs, people, other animals, plants, stars, numbers etc.), or a domain that consists of all people. You can use the following guidelines to determine which domain to pick:

- ▷ When symbolizing a sentence or argument all of whose quantifiers are of the sort 'everyone', 'someone', 'no one', 'anyone' that concern only people, and which contains no names of anything other than people, you may use a domain consisting of just people.
- ▷ When symbolizing any other kind of sentence or argument, you should use the domain of things in general.

So to symbolize something like 'every dog is a poodle' or 'some wild roses are pink', you should use a domain of things in general. On the other hand, for something like 'everyone is happy' or 'someone is angry', you can just use a domain of people.

It's important to notice, though, that you don't *have* to use a domain of people in the latter case. Using a domain of people, 'everyone is happy' gets symbolized as '$\forall xHx$'. But you could also use a domain of things in general, and then make the restriction to people explicit by using an additional predicate 'P' for '____is a person', symbolizing the sentence as '$\forall x(Px \rightarrow Hx)$'. In fact, using an extra predicate like this is what you would have to do if 'everyone is happy' occurred as part of a larger sentence or argument that also involved reference to things other than people.

Similarly, if you wanted to symbolize an assertion of 'everyone is happy' where it is clear that the speaker really only meant to refer to, say, people in Chicago, then you can make that restriction explicit using an additional predicate 'C' for '____is in Chicago'. Using a domain of people, our sentence would then get symbolized as '$\forall x(Cx \rightarrow Hx)$'. Or using a domain of things in general, the sentence would get symbolized as '$\forall x((Px \wedge Cx) \rightarrow Hx)$' — i.e. every object x in the domain of all things is such that if x is a person and x is in Chicago, then x is happy.[1]

[1]That said, when you see quantifiers like 'everyone' or 'someone' in the practice exercises, you can always interpret these as making claims about people in general, not just members of some restricted set of people.

5.5 Quantifier Scope

Consider the following two sentences:

(22) Someone is a logician and someone is an architect.
(23) Someone is a logician and an architect.

These sentences clearly mean different things, and must therefore receive different symbolizations. Sentence (22) says that there is someone who is a logician, and also that there is someone (maybe a different person) who is an architect. So it is a conjunction of two existential sentences. Using the following symbolization key:

domain:	people
L:	____ is a logician
A:	____ is an architect

sentence (22) can be symbolized as '$\exists xLx \wedge \exists xAx$'.

Notice that we could equally well have symbolized it as '$\exists xLx \wedge \exists yAy$' or '$\exists zLz \wedge \exists v_3Av_3$'. As we observed in §5.2, sentences like '$\exists xAx$' and '$\exists yAy$' that differ only in which variable gets used are equivalent to each other, so '$\exists xLx \wedge \exists xAx$' is therefore equivalent to'$\exists xLx \wedge \exists yAy$'. In particular, using a single variable as in '$\exists xLx \wedge \exists xAx$' does *not* mean that the person who is a logician is the same as the one who is an architect, and using different variables as in '$\exists xLx \wedge \exists yAy$' does not mean it's a different person. Both of these FOL sentences just say that there is someone who is a logician, and that there is someone (maybe the same, maybe different) who is an architect.

Sentence (23), on the other hand, says that there exists some *one* individual who is *both* a logician and an architect. It gets symbolized as $\exists x(Lx \wedge Ax)$. You can read this as: there is some object in the domain of people such that both 'Lx' and 'Ax' are true of that object. The difference between this FOL sentence and the earlier '$\exists xLx \wedge \exists xAx$' has to do with the SCOPE of the quantifiers. The way scope works with quantifiers is very similar to it works with negation, which we discussed in §2.8.

In the sentence '$\neg P \wedge \neg Q$', the first $\neg$ has scope over just the first conjunct, and the second $\neg$ has scope over just the second conjunct. It is the conjunction $\wedge$ that functions as the main logical operator. Similarly, in '$\exists xLx \wedge \exists xAx$', the first quantifier has scope over just the first conjunct, and the second quantifier over just the second conjunct, with the conjunction $\wedge$ functioning as the main logical operator.

On the other hand, in '$\exists x(Lx \wedge Vx)$' the quantifier is the main logical operator, and has scope over the entire sentence, with the conjunction $\wedge$ occurring inside the scope of the quantifier. This is similar to how in '$\neg(P \wedge Q)$', the negation is the main logical operator, with the conjunction occurring inside its scope. We'll give a more precise definition of the notion of scope in relation to quantifiers in §5.9, but the analogy with negation should give you a working handle for now.

Closely connected to the notion of scope are the notions of FREE and BOUND variables. A quantifier is said to *bind* the variables that occur in its scope. So in '$\exists x(Lx \wedge Vx)$', the quantifier '$\exists x$' binds the variable in 'Lx' and also the one in 'Vx', since they both occur inside its scope. On the other hand, in '$\exists xLx \wedge \exists xAx$', the first quantifier binds the variable in 'Lx', since it occurs in its scope, but does not bind the variable in 'Ax' — that variable is bound by the second quantifier, in whose scope it occurs.

Or compare the following two FOL sentence:

(24) $\forall x(Dx \rightarrow Px)$
(25) $\forall x Dx \rightarrow Px$

In (24), the quantifier '$\forall x$ is the main operator, and it has scope over the entire sentence; it therefore binds both occurrences of the variable 'x'. On the other hand, in (25), the main operator is the conditional, with the quantifier only taking scope over the antecedent. So here, the quantifier binds the variable in 'Dx', but not the one in 'Px'. Variables like this, that are not bound by any quantifier, are said to be *free* variables.

When symbolizing English sentences, you should never have any free variables in your symbolization. Whereas (24) symbolizes the English sentence 'Every dog is a poodle', (25) doesn't really make any sense, and doesn't symbolize any English sentence. Expressions like 25 that contain free variables are called OPEN FORMULAS. English sentences should never be symbolized using open formulas; they are always symbolized by CLOSED FORMULAS, i.e. ones where all variables are bound, like in (25). The moral is that parentheses are very important! They indicate the scope of quantifiers (just like they indicate the scope of negation), and therefore indicate what variables a quantifier binds.

Here is another example to illustrate the importance of scope:

(26) If everyone is a bassist, then Kurt Cobain is a bassist
(27) Everyone is such that, if they are a bassist, then Kurt Cobain is a bassist.

We'll use a domain of people, 'B' for '____is a bassist', and 'c' for the name 'Kurt Cobain'. Sentence (26) is a conditional, whose antecedent is 'everyone is a bassist'. So we will symbolize it with '$\forall x Bx \rightarrow Bc$'. This sentence is *necessarily* true: if absolutely *everyone* is a bassist, then Kurt Cobain has to be a bassist too. (Of course the antecedent is in fact false; but the conditional as a whole is a necessary truth.)

Sentence (27), by contrast, is best paraphrased as 'every person x is such that, if x is a bassist, then Kurt Cobain is a bassist'. This is symbolized by '$\forall x(Bx \rightarrow Bc)$'. Not only is this not a necessary truth, it is false. The claim is that no matter what person x you pick, the conditional '$(Bx \rightarrow Bc)$' will be true. That's not the case: we could pick Kim Deal, who is in fact a bassist — but the conditional 'If Kim Deal is a Bassist, then Kurt Cobain is a bassist' is false (unlike Deal, Cobain was not a bassist).

The difference between '$\forall x Bx \rightarrow Bc$' and '$\forall x(Bx \rightarrow Bc)$' again concerns the scope of the quantifiers. In '$\forall x Bx \rightarrow Bc$', the conditional $\rightarrow$ is the main logical operator, and the quantifier '$\forall x$' only has scope over the antecedent. In '$\forall x(Bx \rightarrow Bc)$' on the other hand, the scope of '$\forall x$' extends over the entire sentence, and it functions as the main operator. Notice also that even though the quantifier in '$\forall x Bx \rightarrow Bc$' only has scope over the antecedent, this sentence has no free variables (as we would expect, since it symbolizes an English sentence). That's because although 'c' occurs outside the scope of the quantifier, it is not a variable, but a name. And only variables can be bound or free.

5.6 The Utility of Paraphrase

As the previous examples have shown, it is important to get the structure of the sentences you want to symbolize right. Sometimes you will be able to move from English directly to a sentence of FOL. Other times, it helps to paraphrase the sentence one or more times. Each successive paraphrase should move from the original sentence closer to something that you can finally symbolize directly in FOL.

For the next several examples, we will use this symbolization key:

domain:	people
B:	____ is a bassist
R:	____ is a rock star
k:	Kim Deal

Now consider these sentences:

(28) If Kim Deal is a bassist, then she is a rock star.
(29) If someone is a bassist, then she is a rock star.

These sentences look similar, and even have the same words in the consequent ('… she is a rock star'), but they mean very different things and will requires different symbolizations. To arrive at the correct symbolization, it helps to paraphrase the original sentences, removing pronouns.

Sentence (28) can be paraphrased as, 'If Kim Deal is a bassist, then *Kim Deal* is a rock-star'. This can obviously be symbolized as the conditional '$Bk \rightarrow Rk$'. Sentence (29) must be paraphrased differently: 'If some person is a bassist, then *that person* is a rock star'. This sentence is not about any particular person, so we need a variable rather than a name. As a halfway house, we can paraphrase this as, 'For any person x, if x is a bassist, then x is a rockstar'. Now this can be symbolized as '$\forall x(Bx \rightarrow Rx)$'.

This is the same we would have symbolized 'Everyone who is a bassist is a rock star'. And on reflection, (29) is indeed true iff everyone who is a bassist is a rock star. This example illustrates a surprising but important fact about English: some English sentences that contain an existential quantifier like 'someone' have to be paraphrased using a universal quantifier in FOL.

Next, consider these sentences:

(30) If anyone is a bassist, then Kim Deal is a rock star.
(31) If anyone is a bassist, then she is a rock star.

The same words appear as the antecedent in sentences (30) and (31) ('If anyone is a bassist…'). But again, they mean very different things, and paraphrase will come to our aid.

Sentence (30) can be paraphrased, 'If there exists at least one bassist, then Kim Deal is a rock star'. It is now clear that this is a conditional whose antecedent is an existentially quantified sentence. We can symbolize the entire sentence with a conditional as the main logical operator: '$\exists xBx \rightarrow Rk$'.

Sentence (31) can be paraphrased, ‘For all people x, if x is a bassist, then x is a rock star’. Or, in more natural English, it can be paraphrased by ‘All bassists are rock stars’. It is best symbolized as a universally quantified sentence, ‘$\forall x(Bx \rightarrow Rx)$’, just like sentence (29) from earlier. What these examples illustrate is that the English quantifier ‘anyone’ sometimes gets symbolized as an existential quantifier in FOL, and at other times as a universal quantifier. To determine which, try paraphrasing the sentence using words *besides* ‘any’ or ‘anyone’.

■ Exercises 5.6

A. Below are the syllogistic figures identified by Aristotle and his successors, along with their medieval names. These formed the foundation of formal logic for over two millennia, until the end of the 19th century. Formalize each figure in FOL.

- **Barbara.** All G are F. All H are G. So: All H are F
- **Celarent.** No G are F. All H are G. So: No H are F
- **Ferio.** No G are F. Some H is G. So: Some H is not F
- **Darii.** All G are F. Some H is G. So: Some H is F.
- **Camestres.** All F are G. No H are G. So: No H are F.
- **Cesare.** No F are G. All H are G. So: No H are F.
- **Baroko.** All F are G. Some H is not G. So: Some H is not F.
- **Festino.** No F are G. Some H are G. So: Some H is not F.
- **Datisi.** All G are F. Some G is H. So: Some H is F.
- **Disamis.** Some G is F. All G are H. So: Some H is F.
- **Ferison.** No G are F. Some G is H. So: Some H is not F.
- **Bokardo.** Some G is not F. All G are H. So: Some H is not F.
- **Camenes.** All F are G. No G are H So: No H is F.
- **Dimaris.** Some F is G. All G are H. So: Some H is F.
- **Fresison.** No F are G. Some G is H. So: Some H is not F.

B. Using the following symbolization key:

domain:	people
K:	____ knows the combination to the safe
S:	____ is a spy
V:	____ is a vegetarian
h:	Hofthor
i:	Ingmar

symbolize the following sentences in FOL:

1. Neither Hofthor nor Ingmar is a vegetarian.
2. No spy knows the combination to the safe.
3. No one knows the combination to the safe unless Ingmar does.
4. Hofthor is a spy, but no vegetarian is a spy.

C. Using this symbolization key:

domain:	all animals
A:	____ is an alligator.
M:	____ is a monkey.
R:	____ is a reptile.
Z:	____ lives at the zoo.
a:	Amos
b:	Bouncer
c:	Cleo

symbolize each of the following sentences in FOL:

1. Amos, Bouncer, and Cleo all live at the zoo.
2. Bouncer is a reptile, but not an alligator.
3. Some reptile lives at the zoo.
4. Every alligator is a reptile.
5. Any animal that lives at the zoo is either a monkey or an alligator.
6. There are reptiles which are not alligators.
7. If any animal is an reptile, then Amos is.
8. If any animal is an alligator, then it is a reptile.

D. In §5.3 we noted that English sentences of the form 'No $\mathcal{F}$ is $\mathcal{G}$' can be symbolized either as $\neg\exists x(\mathcal{F}x \wedge \mathcal{G}x)$ or as $\forall x(\mathcal{F}x \rightarrow \neg\mathcal{G}x)$, and ones of the form 'Not all $\mathcal{F}$ are $\mathcal{G}$' can be symbolized as either $\neg\forall x(\mathcal{F}x \rightarrow \mathcal{G}x)$ or as $\exists x(\mathcal{F}x \wedge \neg\mathcal{G}x)$. Following these templates, give two different symbolizations for each of the following:

1. No spy is famous.
2. Not all spies are famous.
3. Not every famous villain is a spy.
4. Not all famous spies are villains.
5. No spy is both famous and a villain.
6. No villain is both famous and a spy.
7. Not every spy is both famous and a villain.
8. Some spies are not villains.

E. For each argument, write a symbolization key and symbolize the argument in FOL.

1. Willard is a logician. All logicians wear funny hats. So Willard wears a funny hat
2. Nothing on my desk escapes my attention. There is a computer on my desk. As such, there is a computer that does not escape my attention.
3. All my dreams are black and white. Old TV shows are in black and white. Therefore, some of my dreams are old TV shows.
4. Neither Holmes nor Watson has been to Australia. A person could see a kangaroo only if they had been to Australia or to a zoo. Although Watson has not seen a kangaroo, Holmes has. Therefore, Holmes has been to a zoo.
5. No one expects the Spanish Inquisition. No one knows the troubles I've seen. Therefore, anyone who expects the Spanish Inquisition knows the troubles I've seen.
6. All babies are illogical. Nobody who is illogical can manage a crocodile. Berthold is a baby. Therefore, Berthold is unable to manage a crocodile.

5.7 Quantifiers and Many-Place Predicates

So far, we have only considered sentences with one-place predicates and one quantifier. The full power of FOL really comes out when we start to use many-place predicates and multiple quantifiers, however. Whereas the logic of singly-quantified sentences has been well known for over two millennia since Aristotle, it took until the work of Gottlob Frege in the late 19th for a logic capable of handling sentences with multiple quantifiers to be developed. The system of FOL we are here studying is a fragment of the logic Frege developed in his book *Begriffsschrift* (1879).

ONE-PLACE PREDICATES concern *properties* that objects might have. They have one argument place, or gap, in them. To make a sentence, we simply slot a name into that gap. Other predicates concern *relations* between things. Here are some examples of relational predicates in English:

____ loves ____
____ is to the left of ____
____ is in debt to ____

These are TWO-PLACE PREDICATES: they need to be filled in with two terms in order to make a sentence. Conversely, if we start with an English sentence containing many singular terms, we can remove two singular terms, to obtain different two-place predicates. Consider the sentence 'Vinnie borrowed the family car from Nunzio'. By deleting two singular terms, we can obtain any of three different two-place predicates

Vinnie borrowed ____ from ____
____ borrowed the family car from ____
____ borrowed ____ from Nunzio

And by removing all three singular terms, we obtain a THREE-PLACE PREDICATE:

____ borrowed ____ from ____

Indeed, there is in principle no upper limit on the number of argument places that our predicates may contain.

It's important to realize that the multiple argument places in a predicate can be filled either with the same term, or with different terms, and in various different orders. For example, if we begin with the two-place predicate '____ loves ____', we can fill the gaps with the names 'Karl' and 'Imre' in various different ways, to obtain different English sentences:

(32) Karl loves Imre.
(33) Imre loves Karl.
(34) Karl loves Karl.

In FOL, many-place predicates are symbolized via uppercase letters, just like one-place predicates. To symbolize the above sentences, we can use the following symbolization key:

domain:	people
i:	Imre

k: Karl
L: ____ loves ____

As in the case of one-place predicates, FOL names appear *after* the predicate letter. So sentence (32) will be symbolized as '*Lki*', sentence (33) as '*Lik*', and sentence (34) as '*Lkk*'. You can think of the FOL predicate letter '*L*' as having two invisible argument places after it, into which we can slot the names '*i*' and '*k*'. The convention is that the first gap after the predicate letter represents the first gap in the corresponding English predicate, and the second gap represents the second gap in the English predicate. So since sentence (34) results from putting 'Karl' into the first gap in '____ loves ____' and 'Imre' into the second, its symbolization in FOL has '*k*' in the first gap after '*L*' and '*i*' in the second.

Another way to put it is that the first gap in the English predicate '____ loves ____' is for the *agent* of the relation — the lover, the person doing the loving — and the second gap is for the *patient* of the relation — the beloved, the person who is loved. So given this symbolization key, the first name in the FOL sentence '*Lki*' represents the agent, the lover, and the second name represents the patient, the beloved.

Here are some more sentences that we can symbolize using this key:

(35) Imre loves himself.
(36) Karl loves Imre, but not vice versa.
(37) Karl is loved by Imre.

Sentence (35) can be paraphrased as 'Imre loves Imre', and is symbolized by '*Lii*'. Sentence (36) is a conjunction. We can paraphrase it as 'Karl loves Imre, and Imre does not love Karl', and so symbolize it as '$Lki \wedge \neg Lik$'. Sentence (37) is in the passive voice, but it can be paraphrased in the active voice as 'Imre loves Karl', and so symbolized as '*Lik*'. Of course, there are differences of *tone* between the active and passive voice; but we have preserved the truth conditions.

The difference between active and passive voice illustrates something important. Suppose we had instead used the following symbolization key:

domain: people
i: Imre
k: Karl
L: ____ is loved by ____

Here '*L*' is now used to symbolize an English predicate in the passive voice, meaning that the first gap now represents the patient, the person who is loved, and the second gap the agent, the person doing the loving. Using this symbolization key, the FOL sentence '*Lki*' now means that Karl is loved by Imre, that is to say, that Imre loves Karl. So Sentence (32) — which says that Karl loves Imre, i.e. that Imre is loved by Karl — can no longer be symbolized as '*Lki*', but must be symbolized as '*Lik*'.

The overall moral is simple: *differences in the order of names matter.* When dealing with predicates with more than one argument place, it's important to pay careful attention the order in which the terms appear.

Multiple Generality

Now that we have many-place predicates to work with, we can also symbolize sentences that involve *multiple generality*, i.e. ones that contain more than one quantifier. Consider the sentence 'everyone loves someone', which involves two quantifiers. On its most straightforward reading, this means something like:

(38) For every person, there is someone whom that person loves

But sentences that involve multiple quantifiers are often *ambiguous*. Thus 'everyone loves someone' could also be understood to mean:

(39) There is some particular person whom everyone loves

Using 'L' for '____ loves ____', sentence (38) can be symbolized as '$\forall x \exists y Lxy$'. This would be true of a love-triangle. For example, suppose that our domain contains only Karl, Imre, and Juan. And suppose that Karl loves Imre, that Imre loves Juan, and that Juan loves Karl (and no one loves anyone else). Sentence (38) will then be true: for every individual x in the domain, there is some individual y such that x loves y.

Sentence (39) can be symbolized as '$\exists y \forall x Lxy$'. The difference concerns the *scope* of the quantifiers: in this symbolization, the existential quantifier has the universal quantifier inside its scope, whereas in the symbolization for (38), the universal quantifier had the existential in its scope. This difference in scope results in a difference in meaning.

'$\exists y \forall x Lxy$' is *not* true in the love triangle situation just described. To make it true, we need a situation where there is some one lucky individual y who is loved by everyone. For example, suppose that Karl loves Imre, Juan loves Imre, and that Imre also loves himself. In this situation, it is now true that there exists some individual y such that no matter what x we pick, x loves y — the lucky y is Imre. Notice that it is necessary for Imre to loves himself, otherwise it wouldn't be true that Imre is loved by *everyone* in the domain.

The point is that the scope, or order, of quantifiers matters a great deal. Accidentally switching them around gives rise to the so-called *quantifier shift fallacy*. For example, the following argument is not valid:

	Everything is caused by something.	($\forall\exists$)
$\therefore$	There is some one thing that caused everything.	($\exists\forall$)

Using 'C' for '____ caused ____' (and a domain of things), the premise can be symbolized as '$\forall x \exists y Cyx$': for every x there exists some y such that y caused x. The conclusion, on the other hand, can be symbolized as '$\exists y \forall x Cyx$': there exists some y such that for every x, y caused x. The latter is not implied by '$\forall x \exists y Cyx$'. We'll leave it as an exercise for you to describe a situation that would make the premise true, but the conclusion false.

Such fallacies, though, arise only when we swap around universal with existential quantifiers. With strings of the same quantifier, the order doesn't much matter. For example, consider '$\exists x \exists y Lxy$' and '$\exists y \exists x Lxy$'. These would naturally symbolize the English sentences 'there is someone who loves someone' and 'there is someone who is loved by someone', respectively. But, though these differ in nuance, they are true in exactly the same situations. Also, to return to a point from §5.5, notice that '$\exists x \exists y Lxy$' does not require that x and y be different individuals. This sentence, as well as '$\exists y \exists x Lxy$', would be true in a situation where

Imre loves himself (and no one loves anyone else). After all, that would be a situation where someone loves someone, and also one where someone is loved by someone.

Similar comments apply to pairs like '$\forall x \forall y Lxy$' and '$\forall y \forall x Lxy$': if everyone loves everyone (as per the first sentence) then it follows that everyone is loved by everyone (as per the second), and vice versa. So there can be no situation that makes one true but not the other. And notice that for either to be true, everyone has to, among other things, love themselves. So both imply $\forall x Lxx$.

Lastly, multiply quantified sentences can of course also involve "negative quantifiers." For example:

(40) No one loves everyone.
(41) There's someone who loves no one.

Sentence (40) is the denial of 'there is someone who loves everyone'; since the latter gets symbolized as '$\exists x \forall y Lxy$', sentence (40) gets symbolized as its negation, '$\neg \exists x \forall y Lxy$'. Sentence (41) says that there is some person x such that for any y we pick, x does not love y. It therefore gets symbolized as '$\exists x \forall y \neg Lxy$'.

Another way to think about (41) is as saying: there is some x such that there does not exist any y whom x loves. So we can also symbolize (41) as $\exists x \neg \exists y Lxy$'. This illustrates that the Quantifier Equivalence Laws from §5.2, which govern the movement of negation across quantifiers, continue to hold in multiply quantified sentences. Similarly, if we start with '$\neg \exists x \forall y Lxy$', which was our symbolization of (40), and move the negation across both quantifiers, we end up with '$\forall x \exists y \neg Lxy$'. This says that for every x there is at least one y whom x does not love, which is indeed another way to capture the truth conditions of (40).

Intermediate Steps to Symbolization

As we are starting to see, symbolization in FOL can become tricky. When symbolizing a complex sentence, it is best to proceed by way of several intermediate steps. Let's look at some examples. Consider the following sentences:

(42) Geraldo owns a dog.
(43) Someone owns a dog.
(44) All of Geraldo's friends are dog owners.
(45) Every dog owner is a friend of a dog owner.
(46) Every dog owner's friend owns some friend's dog.

We'll use the following symbolization key:

domain:	things
D:	____ is a dog
F:	____ is a friend of ____
O:	____ owns ____
g:	Geraldo

Sentence (42) can be paraphrased as, 'There is a dog that Geraldo owns'. This can be symbolized by '$\exists x(Dx \wedge Ogx)$'.

Sentence (43) can be paraphrased as, 'There is some y such that y owns a dog'. We can begin by just focusing on the initial quantifier, which gives us '$\exists y(y$ owns a dog)'. Now the fragment 'y owns a dog' is exactly like sentence (42), except it contains a variable instead of a name. Using our symbolization of (42) as a guide, we can symbolize sentence (43) by:

$$\exists y \exists x(Dx \land Oyx)$$

In working out how to symbolize this sentence, we first wrote down '$\exists y(y$ owns a dog)'. To be very clear: this is *neither* an FOL sentence *nor* an English sentence. It uses bits of FOL ('$\exists$', 'y') and bits of English ('owns a dog'). It is really is just an *intermediate step* on the way to symbolizing the English sentence, a bit of "scratch work" we do on the side as we work through the problem.

Sentence (44) can be paraphrased as, 'Everyone who is a friend of Geraldo is a dog owner'. Since being a dog owner is the same as owning a dog, we can in turn paraphrase this as 'Everyone who is a friend of Geraldo owns a dog'. So we can write:

$$\forall x\big[Fxg \to x \text{ owns a dog}\big]$$

as our first intermediate step. Now the consequent of the conditional, 'x is a dog owner', is structurally just like sentence (42). Using our symbolization of (42) as a guide, we get:

$$\forall x\big[Fxg \to \exists y(Dy \land Oxy)\big]$$

Notice that it was essential that we used a variable other than 'x' for the existential quantifier in the consequent. If we had instead written:

$$\forall x\big[Fxg \to \exists x(Dx \land Oxx)\big]$$

we would have had a *clash of variables*. The first variable 'x' after the predicate 'O' represents the agent, the person doing the owning, who is a friend of Geraldo. Accordingly, this variable should get bound by the initial quantifier '$\forall x$' that also binds the 'x' in the antecedent 'Fxg'. But if we now use '$\exists x$' in the consequent, then it should bind every 'x' in its scope, including the first one in 'Oxx'. To avoid this clash of variables, we have to use a different variable for the quantifier in the consequent, as in '$\forall x\big[Fxg \to \exists y(Dy \land Oxy)\big]$'. The broad moral is that a single variable cannot serve two masters simultaneously.

Moving to sentence (45), it can be paraphrased as 'For any x, if x is a dog owner, then x is a friend of some dog owner'. As our first intermediate step, we might have:

$$\forall x\big[x \text{ is a dog owner} \to \exists y(y \text{ is a dog owner} \land Fxy)\big]$$

Again, being a dog owner is the same as owning some dog, and we know how to symbolize that. To avoid a variable clash, we'll have to use an existential quantifier that won't threaten to bind either the 'x' or the 'y' we already have in our intermediate step. So let's use '$\exists z$' in both cases, giving us:

$$\forall x\big[\exists z(Dz \land Oxz) \to \exists y\big(\exists z(Dz \land Oyz) \land Fxy\big)\big]$$

Here '$\exists z(Dz \land Oxz)$' just says that x owns a dog, and '$\exists z(Dz \land Oyz)$' that y owns a dog.

We here decided to use the same variable, ‘z’, in both the antecedent *and* the consequent of the conditional. This is ok, because there is no scope overlap between the two. We might graphically represent the scope of the various quantifiers thus:

$$\overbrace{\forall x\big[\overbrace{\exists z(Dz \wedge Oxz)}^{\text{scope of 1st ‘}\exists z\text{’}} \to \overbrace{\exists y(\overbrace{\exists z(Dz \wedge Oyz)}^{\text{scope of 2nd ‘}\exists z\text{’}} \wedge Fyx)\big]}^{\text{scope of ‘}\exists y\text{’}}}^{\text{scope of ‘}\forall x\text{’}}$$

Since the scopes of the two ‘$\exists z$’ quantifiers don’t overlap, there is no clash of variables. That said, if you want to be absolutely safe, you can always just pick a different variable for each quantifier in your symbolization. So the following, where we use ‘$\exists v$’ in the consequent, would be perfectly good symbolization of (45) as well:

$$\forall x\big[\exists z(Dz \wedge Oxz) \to \exists y\big(\exists v(Dv \wedge Oyv) \wedge Fxy\big)\big]$$

Sentence (46) is the trickiest yet. First we paraphrase it as ‘For any x, if x is a friend of a dog owner, then x owns a dog which is also owned by a friend of x’. As a first intermediate step we get:

$$\forall x\big[x \text{ is a friend of a dog owner} \to x \text{ owns a dog which is owned by a friend of } x\big]$$

Breaking this down a bit more we get:

$$\forall x\big[\exists y(Fxy \wedge y \text{ is a dog owner}) \to \exists y(Dy \wedge Oxy \wedge y \text{ is owned by a friend of } x)\big]$$

I here use ‘$\exists y$’ in both the antecedent and consequent, which is again fine because their scopes do not overlap. To complete the symbolization, we now just have to symbolize ‘y is a dog owner’ and ‘y is owned by a friend of x’, giving us:

$$\forall x\big[\exists y(Fxy \wedge \exists z(Dz \wedge Oyz)) \to \exists y(Dy \wedge Oxy \wedge \exists z(Fzx \wedge Ozy))\big]$$

And we are done!

Again, we could have used different variables for each of the quantifiers, for example:

$$\forall x\big[\exists y(Fxy \wedge \exists z(Dz \wedge Oyz)) \to \exists v(Dv \wedge Oxv \wedge \exists u(Fux \wedge Ouv))\big]$$

Students who are first learning FOL symbolization sometimes want to put all the quantifiers out front, at the beginning of the sentence, perhaps like so:

$$\forall x \exists y \exists z \exists v \exists u\big[(Fxy \wedge (Dz \wedge Oyz)) \to (Dv \wedge Oxv \wedge (Fux \wedge Ouv))\big]$$

This is not a correct symbolization of (46), and indeed, virtually impossible to make sense of. Avoid doing this. Instead, use the strategy of breaking things down via intermediate steps, as we’ve done in these examples. By following this strategy, you will arrive at a correct symbolization, with all the quantifiers in the right places.

■ Exercises 5.7

A. Using this symbolization key:

domain:	things
A:	____ is an alligator
M:	____ is a monkey
R:	____ is a reptile
Z:	____ lives at the zoo
L:	____ loves ____
a:	Amos
b:	Bouncer
c:	Cleo

symbolize each of the following sentences in FOL:

1. If Cleo loves Bouncer, then Bouncer is a monkey.
2. If both Bouncer and Cleo are alligators, then Amos loves them both.
3. Cleo loves a reptile.
4. Bouncer loves all the monkeys that live at the zoo.
5. All the monkeys that Amos loves love him back.
6. Every monkey that Cleo loves is also loved by Amos.
7. There is a monkey that loves Bouncer, but sadly Bouncer does not reciprocate this love.

B. Using this symbolization key:

domain:	things
D:	____ is a dog
S:	____ likes samurai movies
L:	____ is larger than ____
b:	Bertie
e:	Emerson
f:	Fergis

symbolize the following sentences in FOL (provide your own symbolization key for each):

1. Bertie is a dog who likes samurai movies.
2. Bertie, Emerson, and Fergis are all dogs.
3. Emerson is larger than Bertie, and Fergis is larger than Emerson.
4. All dogs like samurai movies.
5. Only dogs like samurai movies.
6. There is a dog that is larger than Emerson.
7. If there is a dog larger than Fergis, then there is a dog larger than Emerson.
8. No animal that likes samurai movies is larger than Emerson.
9. No dog is larger than Fergis.
10. Any animal that dislikes samurai movies is larger than Bertie.
11. There is an animal that is between Bertie and Emerson in size.
12. There is no dog that is between Bertie and Emerson in size.

13. No dog is larger than itself.
14. Every dog is larger than some dog.
15. There is an animal that is smaller than every dog.
16. If there is an animal that is larger than any dog, then that animal does not like samurai movies.

C. Using this symbolization key:

domain:	things
R:	____ has run out.
T:	____ is on the table.
F:	____ is food.
P:	____ is a person.
L:	____ likes ____
e:	Eli
f:	Francesca
g:	the guacamole

symbolize the following English sentences in FOL (be careful here — since the domain consists of things in general, and not just people, quantifiers like 'everyone' have to be symbolizes as 'every person'):

1. All the food is on the table.
2. If the guacamole has not run out, then it is on the table.
3. Everyone likes the guacamole.
4. If anyone likes the guacamole, then Eli does.
5. Francesca only likes the dishes that have run out.
6. Francesca likes no one, and no one likes Francesca.
7. Eli likes anyone who likes the guacamole.
8. Eli likes anyone who likes the people that he likes.
9. If there is a person on the table already, then all of the food must have run out.

D. Using this symbolization key:

domain:	people
D:	____ dances ballet.
F:	____ is female.
M:	____ is male.
C:	____ is a child of ____
S:	____ is a sibling of ____
e:	Elmer
j:	Jane
p:	Patrick

symbolize the following sentences in FOL:

1. All of Patrick's children are ballet dancers.
2. Jane is Patrick's daughter.
3. Patrick has a daughter.

4. Jane is an only child.
5. All of Patrick’s sons dance ballet.
6. Patrick has no sons.
7. Jane is Elmer’s niece.
8. Patrick is Elmer’s brother.
9. Patrick’s brothers have no children.
10. Jane is an aunt.
11. Everyone who dances ballet has a brother who also dances ballet.
12. Every woman who dances ballet is the child of someone who dances ballet.

5.8 Adding Identity

Consider this sentence:

(47) Pavel owes money to everyone

Let’s use a domain of people, ‘p’ for ‘Pavel’, an ‘O’ for ‘____ owes money to ____’. We can then symbolize sentence (47) by ‘$\forall x Opx$’. But this has a (perhaps) odd consequence. It requires that Pavel owes money to *every* member of the domain. Since Pavel himself must be a member of the domain, this entails that Pavel owes money to himself. And maybe we did not want to say that. Maybe what we meant to say was:

(48) Pavel owes money to everyone *else*
(49) Pavel owes money to everyone *other than* Pavel
(50) Pavel owes money to everyone *except* himself

But we do not have any way of dealing with the italicized words yet. The solution is to add a new symbol to FOL.

The symbol ‘$=$’ will be a two-place predicate, denoting the relation of *identity*. Since identity is such a basic logical concept — similar to how e.g. conjunction, negation, or existential quantification are basic logical concepts — ‘$=$’ functions as a LOGICAL CONSTANT in FOL. This means that the symbol ‘$=$’ *has* to be interpreted as ‘____ is identical to ____’; you can’t assign it a different meaning in a symbolization key.

To highlight the fact that identity is special in being the only logical constant among two-place predicates, we adopt a different notational convention for it, and write it *between* two terms rather than in front of them. This notation will be familiar to you from math, where you write things like $\frac{1}{2} = \frac{4}{8}$. Note that in saying that some objects x and y are identical, we don’t merely mean that they are very similar, or indistinguishable in the way that e.g. two cans of Coca Cola are. We mean that they are *one and the same* object.

To put this to use, suppose we want to symbolize this sentence:

(51) Pavel is Mister Checkov.

Using ‘c’ for the name ‘Mister Checkov’, sentence (51) can be symbolized as ‘$p = c$’. This tells us that Pavel and Mister Checkov are one and the same person, and that the names ‘p’ and ‘c’ refer to the same individual.

We can also now deal with sentences (48)–(50). All of these sentences can be paraphrased as ‘Everyone who is not Pavel is owed money by Pavel’. Paraphrasing some more, we get: ‘For all x, if x is not Pavel, then x is owed money by Pavel’. Now that we are armed with our new identity symbol, we can symbolize this as ‘$\forall x(\neg x = p \to Opx)$’.

This last sentence contains the formula ‘$\neg x = p$’. This might look a bit strange, but it just means that we are negating the entire formula, ‘$x = p$’. From math, you’re probably familiar with the notation ‘$\neq$’ for negated identity, so we’ll also adopt this notational convention here, though only as a convenient shorthand:

An FOL sentence of the form $\neg t_1 = t_2$ can be abbreviated as $t_1 \neq t_2$

Using this notational shorthand, we can rewrite our symbolization as ‘$\forall x(x \neq p \to Opx)$’.

In addition to sentences that use the words ‘else’, ‘other than’, and ‘except’, identity will be helpful when symbolizing some sentences that contain the words ‘besides’ and ‘only.’ Consider these examples:

(52) No one besides Pavel owes money to Hikaru.
(53) Only Pavel owes Hikaru money.

Letting ‘*h*’ name Hikaru, sentence (52) can be paraphrased as, ‘No one who is not Pavel owes money to Hikaru’. This can be symbolized by ‘$\neg\exists x(x \neq p \wedge Oxh)$’. Sentence (52) can be paraphrased as ‘for all x, if x owes money to Hikaru, then x is Pavel’. This can be symbolized as ‘$\forall x(Oxh \to x = p)$’. In fact, these two symbolizations are equivalent to each other; and (52) and (53) do seem to express the same claim.

But there is one subtlety here. Our symbolizations imply that anyone who is not Pavel does not owe money to Hikaru. But (52) and (53) also seem to imply that Pavel does owe money to Hikaru. To capture this, we can add ‘Oph’ as a conjunct to either symbolization, giving us e.g. ‘$Oph \wedge \forall x(Oxh \to x = p)$’ as a final symbolization. This, in turn, can be shortened to $\forall x(Oxh \leftrightarrow x = p)$.

Identity can also be used to symbolize claims about *how many* things there are of a particular kind. We’ll go look at three kinds of claims of this sort.

There are at least…

Consider the following ‘at least’ claims:

(54) There is at least one apple
(55) There are at least two apples
(56) There are at least three apples

We’ll use ‘*A*’ for ‘____is an apple’, and a domain of things. Sentence (54) does not require identity. It can be adequately symbolized by ‘$\exists xAx$’: there is some apple; perhaps many, but at least one.

It might be tempting to also translate sentence (55) without identity. But consider the sentence ‘$\exists x\exists y(Ax \wedge Ay)$’. This says that there is some apple *x* in the domain and also some apple *y* in the domain. Since nothing precludes these from being one and the same apple, this would be true even if there were only one apple. (Recall here the point from §5.5 that a difference in variables need not indicate a difference in the objects the variables pick out.) To make sure that we are dealing with *different* apples, we need to use identity, and symbolize (55) as '$\exists x \exists y(Ax \land Ay \land x \neq y)$'.

Sentence (56) requires talking about three different apples. Now we need three existential quantifiers, and we need to make sure that each will pick out something different: '$\exists x \exists y \exists z(Ax \land Ay \land Az \land x \neq y \land y \neq z \land x \neq z)$'. As you can see, by following this pattern, we can symbolize claims of the sort 'there are at least *n* apples' for any (finite) number *n*.

There are at most…

Now consider these sentences:

(57) There is at most one apple
(58) There are at most two apples

Sentence (57) can be paraphrased as, 'It is not the case that there are at least *two* apples'. This is just the negation of sentence (55):

$$\neg \exists x \exists y(Ax \land Ay \land \neg x = y)$$

But sentence (57) can also be approached in another way. It means that if you pick out an object and it's an apple, and then you pick out an object and it's also an apple, you must have picked out the same object both times. With this in mind, it can be symbolized by:

$$\forall x \forall y\big[(Ax \land Ay) \to x = y\big]$$

The two sentences will turn out to be logically equivalent.

Similarly, sentence (58) can be approached in two equivalent ways. It can be paraphrased as 'It is not the case that there are at least *three* apples', which is just the negation of (56) above. Alternatively, we can understand it as saying that if you pick out an apple, and an apple, and an apple, then you will have picked out the same apple at least once. Thus:

$$\forall x \forall y \forall z\big[(Ax \land Ay \land Az) \to (x = y \lor x = z \lor y = z)\big]$$

Again, by following this pattern we can symbolize claims of the sort 'there are at most *n* apples' for any *n*.

There are exactly…

Lastly, there are statements that specify a precise numerical quantity:

(59) There is exactly one apple.
(60) There are exactly two apples.
(61) There are exactly three apples.

Sentence (59) can be paraphrased as 'There is *at least* one apple and there is *at most* one apple'. This is just the conjunction of (54) and (57) from above:

$$\exists x Ax \land \forall x \forall y\big[(Ax \land Ay) \to x = y\big]$$

But it is perhaps more straightforward to paraphrase sentence (59) as, 'There is a thing x which is an apple, and everything which is an apple is just x itself'. Thought of in this way, we'd symbolize it as:

$$\exists x\big[Ax \land \forall y(Ay \rightarrow x = y)\big]$$

Similarly, sentence (60) may be paraphrased as, 'There are *at least* two apples, and there are *at most* two apples', and thus symbolized as the conjunction of (55) and (58). More efficiently, though, we can paraphrase it as 'There are at least two different apples, and every apple is one of those two apples'. Then we offer:

$$\exists x \exists y\big[Ax \land Ay \land x \neq y \land \forall z(Az \rightarrow (x = z \lor y = z))\big]$$

Continuing with this patter, we could symbolize the claim that there are exactly three apples as follows:

$$\exists x \exists y \exists z\big[Ax \land Ay \land Az \land x \neq y \land x \neq z \land y \neq z \land \forall v(Av \rightarrow (x = v \lor y = v \lor z = v))\big]$$

and so on, for any number n of apples.

Finally, consider these sentence:

(62) There are exactly two things
(63) There are exactly two objects

It might be tempting to add a predicate to our symbolization key, to symbolize the English predicate '____ is a thing' or '____ is an object'. But this is unnecessary. Words like 'thing' and 'object' apply trivially to everything. So we can symbolize either sentence as:

$$\exists x \exists y\big[x \neq y \land \forall z(x = z \lor y = z)\big]$$

Logical Truths Involving Identity

We introduced the symbol '$=$' as an additional logical constant, i.e. as a logical symbol whose meaning remains fixed, just like '$\exists$' or '$\neg$'. Part of the motivation for treating identity as a logical constant is that it seems like a very primitive logical concept, much like negation or existence.

Another, related motivation is that there seem to be certain basic logical truths involving identity, just like e.g. the LAW OF EXCLUDED MIDDLE from TFL is a basic logical truth involving negation. One such primitive truth is that everything is identical to itself, which we can express as:

$$\forall x\, x = x$$

This is sometimes called the LAW OF IDENTITY, and it will be a theorem in the system of natural deduction for FOL that we will develop later.

Another logical truth involving identity sometimes goes by the name of LEIBNIZ'S LAW.[2] It says that if x and y are one and the same thing, then x and y must share all their properties.

[2]This law is named after Gottfried Willhelm Leibniz (1646–1716). Leibniz actually endorsed a stronger claim, which says not only that x and y must share all their properties if they are identical, but also (and more controversially!) that if x and y share all their properties, then they are identical. This second claim is called the "Identity of Indiscernibles." Can you think of a potential example of objects x and y that share all their properties but are still distinct?

So if we use 'D' for '____is a dog', the following would be an instance of Leibniz's Law:

$$\forall x\forall y\big[x = y \rightarrow (Dx \leftrightarrow Dy)\big]$$

This says that if x and y are identical, then x is a dog iff y is. Similarly, if we use 'O' for '____ owns ____', then another instance of Leibniz's Law says that if x and y are identical, then x owns a dog iff y does:

$$\forall x\forall y\big[x = y \rightarrow (\exists z(Dz \wedge Oxz) \leftrightarrow \exists z(Dx \wedge Oyz))\big])$$

But Leibniz's Law itself cannot be captured in FOL. Since it makes a claim about *all properties*, we'd need to have quantifiers that bind variables in *predicate* position to really express Leibniz's Law in full generality, writing something like:

$$\forall x\forall y\big[x = y \rightarrow \forall P(Px \leftrightarrow Py)\big]$$

A logic that contains quantifiers like '$\forall P$' is said to be a SECOND-ORDER LOGIC. As its name indicates, FOL is a *first-order* logic. It can express *instances* of Leibniz's Law, concerning particular properties like owning a dog, but it cannot express it in full generality.

One last logical truth involving identity is the following:

$$\exists x\, x = x$$

This says that there exists at least one thing. This is obviously a controversial case: you might think the claim there exists something (rather than nothing) shouldn't just be a truth of logic. But it is a logical truth in FOL, since FOL requires that quantifier domains have at least one member (see the stipulation in §5.2 above), and it will be a theorem in our system of natural deduction. Systems of logic that avoid having this as a logical truth, and allow for empty domains as well as non-referring names, are called FREE LOGICS.

■ Exercises 5.8

A. Explain why:

- '$\exists x\forall y(Ay \leftrightarrow x = y)$' is a good symbolization of 'there is exactly one apple'.
- '$\exists x\exists y\big[\neg x = y \wedge \forall z(Az \leftrightarrow (x = z \vee y = z))\big]$' is a good symbolization of 'there are exactly two apples'.

B. Using the following symbolization key:

domain:	people
K:	____ knows the combination to the safe
S:	____ is a spy
T:	____ trusts ____
h:	Hofthor
i:	Ingmar

symbolize the following sentences in FOL:

1. Hofthor trusts a spy.

2. Everyone who trusts Ingmar also trusts someone else (someone other than Ingmar).
3. Everyone who trusts some spy also trusts anyone who isn't a spy.
4. Everyone who trusts Ingmar trusts someone who trusts a spy.
5. Only Ingmar knows the combination to the safe.
6. Ingmar trusts Hofthor, but no one else.
7. Hofthor trusts everyone except Ingmar.

C. Using the following symbolization key:

domain:	things
A:	____ is a card
B:	____ is black
C:	____ is a club
D:	____ is a deuce
J:	____ is a jack
O:	____ is one-eyed
W:	____ is wild

symbolize each sentence in FOL:

1. All clubs are black cards.
2. There are no wild cards.
3. There are at least two clubs.
4. There is more than one one-eyed jack.
5. There are at most two one-eyed jacks.
6. There are two black jacks.
7. There are four deuces.
8. One-eyed jacks and deuces are wild.
9. If one-eyed jacks are wild, then there are exactly two wild cards.

D. Using the following symbolization key:

domain:	things
B:	____ is in Farmer Brown's field.
H:	____ is a horse.
W:	____ has wings.

symbolize the following sentences in FOL:

1. There are at least three horses.
2. There are at least three things.
3. There is more than one horse in Farmer Brown's field.
4. There are exactly two horses in Farmer Brown's field.
5. There is a single winged horse in Farmer Brown's field, and all other things in the field are wingless.

5.9 The Syntax of FOL

We've been learning to symbolize English sentences in the language of FOL, but it's time to be more precise about the grammar, or syntax of FOL. As in the case of TFL (see §2.8), we will in this section precisely define the notion of a SENTENCE OF FOL.

There are six kinds of symbols that constitute the LEXICON of FOL:

Predicates	$A, B, C, \ldots, Z$
with subscripts, as needed	$A_1, B_1, Z_1, A_2, A_{25}, J_{375}, \ldots$
Constants	$a, b, c, \ldots, r$
with subscripts, as needed	$a_1, b_{224}, h_7, m_{32}, \ldots$
Variables	s, t, u, v, w, x, y, z
with subscripts, as needed	$x_1, y_1, z_1, x_2, \ldots$
Connectives	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Brackets	(,)
Quantifiers	$\forall, \exists$

We define an EXPRESSION OF FOL as any string of symbols of FOL. Take any of the symbols in the lexicon of FOL and write them down, in any order, and you have an expression. But not all expressions are well-formed *sentences* of FOL, so we'll need rules to tell us which expressions count as sentences.

In the case of TFL, in §2.8, we went straight from the statement of the lexicon to the definition of a sentence of TFL. In FOL, we will have to go via a more indirect route. We will first define the notion of being a *formula* of FOL, and then single out the sentences from this larger class of formulas.

Formulas of FOL

To begin, we define the notion of a *term*:

A TERM is any name or any variable.

Using this, we can next define the notion of an *atomic formula*:

1. If $\mathscr{R}$ is an n-place predicate and $\mathscr{t}_1, \ldots, \mathscr{t}_n$ are terms, then $\mathscr{R}\mathscr{t}_1 \ldots \mathscr{t}_n$ is an ATOMIC FORMULA.
2. If $\mathscr{t}_1$ and $\mathscr{t}_2$ are terms, then $\mathscr{t}_1 = \mathscr{t}_2$ is an atomic formula.
3. Nothing else is an atomic formula.

Notice that we are again using cursive *metavariables* in this definition (see the discussion in §2.8). So, '$\mathscr{R}$' is not itself a predicate of FOL. Rather, it is a symbol of our metalanguage which we are using to talk about arbitrary predicates of FOL. Similarly, '$\mathscr{t}_1$' is not a term of FOL, but a symbol of the metalanguage that we are using to talk about arbitrary terms (i.e. variables or constants) of FOL.

If we let 'F' be a one-place predicate,'G' a three-place predicate, and 'S' a six-place predicate, then the following all count as atomic formulas of FOL by our definition:

$$x = a$$
$$Fx$$
$$Fa$$
$$Gxay_2$$
$$Gaaa$$
$$Sx_1x_2abyx_6$$

Given the notion of an *atomic formula*, we can now recursively define the broader class of FOL *formulas* as follows:

1. Every atomic formula is a formula.
2. If $\mathscr{A}$ is a formula, then $\neg\mathscr{A}$ is a formula.
3. If $\mathscr{A}$ and $\mathscr{B}$ are formulas, then $(\mathscr{A} \wedge \mathscr{B})$ is a formula.
4. If $\mathscr{A}$ and $\mathscr{B}$ are formulas, then $(\mathscr{A} \vee \mathscr{B})$ is a formula.
5. If $\mathscr{A}$ and $\mathscr{B}$ are formulas, then $(\mathscr{A} \rightarrow \mathscr{B})$ is a formula.
6. If $\mathscr{A}$ and $\mathscr{B}$ are formulas, then $(\mathscr{A} \leftrightarrow \mathscr{B})$ is a formula.
7. If $\mathscr{A}$ is a formula, $\mathscr{v}$ is a variable, $\mathscr{A}$ contains at least one occurrence of $\mathscr{v}$, and $\mathscr{A}$ contains neither $\forall\mathscr{v}$ nor $\exists\mathscr{v}$, then $\forall\mathscr{v}\mathscr{A}$ is a formula.
8. If $\mathscr{A}$ is a formula, $\mathscr{v}$ is a variable, $\mathscr{A}$ contains at least one occurrence of $\mathscr{v}$, and $\mathscr{A}$ contains neither $\forall\mathscr{v}$ nor $\exists\mathscr{v}$, then $\exists\mathscr{v}\mathscr{A}$ is a formula.
9. Nothing else is a formula.

The first few clauses are similar to those from TFL. What's new are clauses (7) and (8), which tell us how to construct quantified formulas. Letting 'F' be a one-place predicate and 'R' a two-place predicate, the following all count as formulas of FOL:

$$Fc$$
$$Fx$$
$$Rxz$$
$$(Fx \rightarrow Rxz)$$
$$\forall x(Fx \rightarrow Rxz)$$
$$(Fy \leftrightarrow \forall x(Fx \rightarrow Rxz))$$
$$\exists y(Fy \leftrightarrow \forall x(Fx \rightarrow Rxz))$$
$$\forall z\exists y(Fy \leftrightarrow \forall x(Fx \rightarrow Rxz))$$

By contrast, the following are *not* formulas:

$$\forall yRxx$$
$$\forall x\exists xRxx$$

Looking at the first of these, we can say that 'Rxx' is a formula by clause (1), but '$\forall yRxx$' is not a formula, because it results from attaching the quantifier '$\forall y$' to a formula that does not contain at least one occurrence of the variable 'y', thus contravening clause (7). Similarly, in the case of the second example, 'Rxx' is a formula, and by clause (8) '$\exists xRxx$' is a formula. But '$\forall x\exists xRxx$' is not a formula because we've attached the quantifier '$\forall x$' to a formula '$\exists xRxx$' that *already* contains a quantifier '$\exists x$' involving the same variable, which contravenes clause (7).

These constraints have the effect of preventing the kind of *variable clashes* we discussed in §5.7. And in fact, we can now give a formal definition of the notion of *scope* in terms of the notion of *main logical operator*:

> The MAIN LOGICAL OPERATOR in a formula is the operator that was introduced most recently when constructing that formula according to the syntactic rules of FOL.

> The SCOPE of a logical operator in a formula is the subformula for which that operator is the main logical operator.

Since quantifiers are a kind of logical operator, this definition covers the scope of quantifiers alongside the scope of truth functional connectives. The scope of the various quantifiers in one of our earlier examples can be illustrated as follows:

$$\forall z\overbrace{\exists y\overbrace{(Fy \leftrightarrow \overbrace{\forall x(Fx \to Rxz)}^{\text{scope of }\forall x}\ )}^{\text{scope of }\exists y}}^{\text{scope of }\forall z}$$

The problem with a formula like '$\forall x\exists xRxx$' is that we have a quantifier '$\forall x$' inside whose scope there is another quantifier '$\exists x$' involving the same variable, leading to a variable clash.

Sentences of FOL

With these pieces in place, we're now ready to define the notion of a *sentence* of FOL. To see why we need to distinguish sentences from mere formulas of FOL, recall that logic is concerned with *statements*: sentences that can be either true or false. And many formulas are not true or false. For example, consider the formulas Fc and Fx, and suppose we have the following symbolization key:

domain:	people
F:	____ is a philosopher
c:	Confucius

The formula 'Fc' can be assigned a truth value: we just ask ourselves whether the person 'c' refers to is a philosopher. Since Confucius is a philosopher, 'Fc' is true. By contrast, 'Fx' has no truth value. After all, 'x' is just a variable, and doesn't name any specific object in the domain.

Of course, if we put an existential quantifier out front to obtain '$\exists x Fx$', we now have something that's capable of being true or false, since this now says that at least one person is a philosopher. The point is that we need to *bind* the variable in 'Fx' with a quantifier to obtain something true or false.

Since we want all sentences of FOL to be either true or false, we need to exclude formulas like 'Fx' from the class of sentences. We can do this by giving a precise definition of the notions of *bound* and *free* variables that we already informally discussed in §5.5:

An occurrence of a variable $\mathscr{v}$ in a formula is BOUND iff it falls within the scope of either $\forall\mathscr{v}$ or $\exists\mathscr{v}$. A variable which is not bound is FREE.

An OPEN FORMULA is one that contains at least one free variable.

For example, consider the formula:

$$(\forall x(Ex \lor Dy) \rightarrow \exists z(Ex \rightarrow Lzx))$$

The scope of the universal quantifier '$\forall x$' is '$\forall x(Ex \lor Dy)$', so the first 'x' in the formula is bound. However, the 'y' is free. The scope of the existential quantifier '$\exists z$' is '$\exists z(Ex \rightarrow Lzx)$', so '$z$' is bound. But the '$x$'s in this subformula are free in the formula as a whole, since they occur outside the scope of the universal quantifier '$\forall x$'. Since the formula as a whole contains free variables, it is an *open formula*.

Finally we can say the following.

A SENTENCE of FOL is any formula of FOL that contains no free variables. Sentences are also called CLOSED FORMULAS.

By requiring that every variable in a sentence be bound, we ensure that all sentences of FOL are capable of being true or false.

■ Exercises 5.9

A. Identify which variables are bound and which are free.

1. $\exists x Lxy \land \forall y Lyx$
2. $\exists x(Lxy \land \forall y Lyx)$
3. $\forall x Ax \land Bx$
4. $\forall x(Ax \land Bx) \land \forall y(Cx \land Dy)$
5. $\forall x \exists y[Rxy \rightarrow (Jz \land Kx)] \lor Ryx$
6. $\forall x_1(Mx_2 \leftrightarrow Lx_2x_1) \land \exists x_2 Lx_3x_2$

B. For each of the following formulas, rewrite the formula so that all variables are bound, or leave it alone if all variables are already bound. You may *only* change these formulas by adding and removing parentheses (i.e. you can't add more quantifiers).

1. $\exists x Cx \rightarrow \forall y Rxy$
2. $\forall x(\exists y Rxy \rightarrow Rxx)$
3. $\forall x(\exists y Rxy \rightarrow Ryx)$

4. $\forall x((\exists y Rxy \land Fy) \to Rxx)$
5. $\exists x \forall y(\forall z Lzx \to \forall u(Jy \land Lzu))$
6. $\exists x(\forall y \forall z Lzy \to \forall u(Ju \land Lxu))$

Natural deduction for FOL 6

The language of FOL contains all TFL connectives. So proofs in FOL will take over all the natural deduction rules governing from TFL that we studied in chapter 4, as well as the all derived TFL rules introduced in §4.11. We will also continue to use the same proof theoretic notions, in particular, the symbol '$\vdash$'. So:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$$

will continue to mean that $\mathscr{C}$ is *provable* from $\mathscr{A}_1, \ldots, \mathscr{A}_n$, i.e. that there exists a natural deduction proof which ends with $\mathscr{C}$ and whose premises, or more generally, whose undischarged assumptions, include at most $\mathscr{A}_1, \ldots, \mathscr{A}_n$.

What we will need to add to our natural deduction system are rules to govern the new logical symbols that are specific to FOL: the quantifiers '$\forall$' and '$\exists$', and the identity predicate '$=$'. As in the case of TFL, there will be an *introduction* and an *elimination* rule associated with each of these logical symbols.

Again, we will of course want to make sure that the rules we add end up producing a proof system that is both sound and complete:

SOUNDNESS: If $\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$ then $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{C}$

COMPLETENESS: If $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{C}$ then $\mathscr{A}_1, \ldots, \mathscr{A}_n \vdash \mathscr{C}$

Soundness ensures that if $\mathscr{C}$ is provable from $\mathscr{A}_1, \ldots, \mathscr{A}_n$, then $\mathscr{A}_1, \ldots, \mathscr{A}_n$ logically entail the conclusion $\mathscr{C}$. And completeness ensures us that if $\mathscr{A}_1, \ldots, \mathscr{A}_n$ logically entail the conclusion $\mathscr{C}$, then $\mathscr{C}$ is also provable from premises $\mathscr{A}_1, \ldots, \mathscr{A}_n$. We will formally define the concept of logical entailment in Chapter 7.

As in the case of TFL, we won't demonstrate that our proof system is in fact sound and complete — this would require a *meta-logical* proof, a proof *about* our proof system. You'll just have to take my word for it that soundness and completeness hold for our system. The important point for our purposes is that, due to the soundness of our proof system, we can use proofs to show that arguments are valid: if we can deduce a conclusion $\mathscr{C}$ from some premises $\mathscr{A}_1, \ldots, \mathscr{A}_n$, then we can be sure that the corresponding argument is valid.

Similarly, we will continue to write:

$$\vdash \mathscr{A}$$

to mean that $\mathscr{A}$ is a THEOREM of our proof system, i.e. a sentence that is provable using no premises, or undischarged assumptions. Again, given soundess, theorems of our proof system are guaranteed to be *logical truths*, so we can use natural deduction proofs to show that things are logical truths.

6.1 Universal elimination

The elimination rule for the universal quantifier has a very simple idea behind it: from the claim that everything is F, you can infer that any particular thing is F. You name it; it's F. So the following should be fine:

$$\begin{array}{r|ll} 1 & \forall x Rxd & \\ \cline{2-2} 2 & Rad & \forall\text{E } 1 \end{array}$$

We obtained line 2 by dropping the universal quantifier '$\forall x$' and replacing every occurrence of the variable 'x' that this quantifier bound with 'a'. Equally, the following is fine:

$$\begin{array}{r|ll} 1 & \forall x Rxd & \\ \cline{2-2} 2 & Rdd & \forall\text{E } 1 \end{array}$$

Here we obtained line 2 by dropping the universal quantifier and replacing every occurrence of 'x' with 'd'. We could have done the same with any other name we wanted: 'b', 'c', 'e', you name it.

Both 'Rad' and 'Rdd' are said to be *instances* of the quantified sentence '$\forall xRxd$', as are 'Rbd', 'Rcd', 'Red' and so on. In general, this notion is defined as follows:

Given a universally quantified FOL sentence $\forall \mathscr{v}\mathscr{A}(\ldots\mathscr{v}\ldots)$ its INSTANCES are all those FOL sentences $\mathscr{A}(\ldots\mathscr{c}\ldots)$ that are obtained by dropping the quantifier $\forall\mathscr{v}$ and replacing *every* occurrence of the variable $\mathscr{v}$ bound by that quantifier with some name $\mathscr{c}$.

The universal elimination rule $\forall$E then just says that from a universal sentence you may infer any of its instances:

$$\begin{array}{r|ll} m & \forall \mathscr{v}\mathscr{A}(\ldots\mathscr{v}\ldots) & \\ & \mathscr{A}(\ldots\mathscr{c}\ldots) & \forall\text{E } m \end{array}$$

Here, as well as in the definition of the notion of an instance, we are using the notation $\mathscr{A}(\ldots\mathscr{v}\ldots)$ to represent any FOL formula $\mathscr{A}$ that contains at least one occurrence of the variable $\mathscr{v}$, and $\mathscr{A}(\ldots\mathscr{c}\ldots)$ to represent the result of replacing every occurrence of the variable $\mathscr{v}$ in that formula with some name $\mathscr{c}$.

Notice that $\mathscr{A}(\ldots\mathscr{v}\ldots)$ may of course itself contain connectives and even other quantifiers. So if we begin with the complex universally quantified sentence:

$$\forall x(Rax \land \exists yRxy)$$

the following would be instance of it, and could be inferred via $\forall$E:

$$Raa \land \exists yRay$$
$$Rab \land \exists yRby$$

The first results from replacing the variable 'x' with the name 'a', and the second from replacing 'x' with 'b'. By contrast, the following are *not* instances:

$$Rab \land \exists yRxy$$
$$Rac \land \exists yRdy$$

The first isn't because we forgot to replace the second occurrence of 'x' with the name 'b' (thereby leaving this 'x' as a free variable), and the second isn't because we replaced different occurrences of 'x' with different names. Again, the important point is that when using ∀E, the sentence you infer must be an *instance* of the universally quantified sentence you applied the rule to.

It's also important to emphasize that (as with every elimination rule) you can only apply the ∀E rule when the universal quantifier is the *main logical operator*. So the following is *not* a legitimate use of *∀E*:

$$\begin{array}{r|ll}
1 & \forall xBx \to Bc & \\
\hline
2 & Bb \to Bc & \text{illegitimate use of } \forall\text{E } 1
\end{array}$$

This is illegitimate because '$\forall x$' is not the main logical operator on line 1. (If you need a reminder as to why this sort of inference is bad, look back at §5.5.) If line 1 had instead been '$\forall x(Bx \to Bc)$', then the quantifier would have been the main operator, and it would have been legitimate to infer '$Bb \to Bc$'.

Another way to put it is that this use of ∀E is illegitimate because '$Bb \to Bc$' is not an *instance* of '$\forall xBx \to Bc$'. This is because the notion of an instance only applies to sentences that have a quantifier as their main operator. It just doesn't make sense to talk about the "instances" of '$\forall xBx \to Bc$', because this sentence has a conditional as its main operator, and conditionals do not have instances. By contrast, '$\forall x(Bx \to Bc)$' does have a quantifier as its main operator, and '$Bb \to Bc$' is one of its instances, so we can infer it via ∀E.

Using this rule, we can, for example, show that the following argument that we looked at way back in §1.1 is valid:

All rabbits are birds.
Winnie the Pooh is a rabbit.
∴ Winnie the Pooh is a bird.

First, we provide the following FOL symbolization (using the obvious symbolization key):

$\forall x(Rx \to Bx)$
Rp
$\therefore Bp$

And then we can give a simple natural deduction to show that the conclusion follows:

$$
\begin{array}{r|ll}
1 & \forall(xRx \to Bx) & \\
2 & Rb & \\
\hline
3 & Rp \to Bp & \forall\text{E } 1 \\
4 & Bp & \to\text{E } 3, 2
\end{array}
$$

6.2 Existential introduction

The idea behind the existential introduction rules is again very simple: from the claim that some particular thing is F, you can infer that there exists at least one thing that is F. So we ought to allow:

$$
\begin{array}{r|ll}
1 & Raa & \\
\hline
2 & \exists xRax & \exists\text{I } 1
\end{array}
$$

Here, we have replaced just one occurrence of the name '*a*' with a variable '*x*', and then existentially quantified over it. Equally, we would have allowed:

$$
\begin{array}{r|ll}
1 & Raa & \\
\hline
2 & \exists xRxx & \exists\text{I } 1
\end{array}
$$

Here we have replaced both occurrences of the name '*a*' with the variable '*x*', and then existentially generalized. Both kinds of inferences are fine: if Narcissus loves himself, '*Lnn*', then we can infer that there exists someone who loves themselves, '$\exists xLxx$', but we can also infer that there exists someone who Narcissus loves, '$\exists xLnx$', and that there is someone who loves Narcisuss, '$\exists xLxn$'.

Another way to put this is to note that '*Lnn*' is an *instance* of each of the quantified sentences '$\exists xLxx$', '$\exists xLnx$', '$\exists xLxn$'. The notion of an instances as applied to existential sentences is the same as it is for universal ones:

Given any existentially quantified FOL sentence $\exists \mathscr{v}\mathscr{A}(\ldots \mathscr{v} \ldots)$ its INSTANCES are all those FOL sentences $\mathscr{A}(\ldots \mathscr{c} \ldots)$ that are obtained by dropping the quantifier $\exists \mathscr{v}$ and replacing *every* occurrence of the variable $\mathscr{v}$ bound by that quantifier with some name $\mathscr{c}$.

The existential introduction rule $\exists$I then just says that from a sentence containing one or more occurrences of a name $\mathscr{c}$, we may infer *any* existential sentence of which that original sentence is an instance:

$$
\begin{array}{r|ll}
m & \mathscr{A}(\ldots \mathscr{c} \ldots) & \\
 & \exists \mathscr{x}\mathscr{A}(\ldots \mathscr{x} \ldots) & \exists\text{I } m
\end{array}
$$

To take a slightly more complex example, from:

$$Fa \wedge \forall yRay$$

we could infer any of the following using ∃I:

$$\exists x(Fx \wedge \forall yRay)$$
$$\exists x(Fa \wedge \forall yRxy)$$
$$\exists x(Fx \wedge \forall yRxy)$$

because '$Fa \wedge \forall yRay$' is an instance of any of these. Again, we can existentially generalize on just one occurrence of 'a', or on both.

On the other hand, the last of these, '$\exists x(Fx \wedge \forall yRxy)$', could *not* be inferred from:

$$Fa \wedge \forall yRby$$

because this is not one of its instances. What's gone wrong here is that we tried to generalize on two different names, 'a' and 'b', at once, which isn't allowed.

What we could have done instead is to generalize on each of the two names separately, via successive uses of ∃I:

1	$Fa \wedge \forall yRby$	
2	$\exists x(Fx \wedge \forall yRby)$	∃I 1
3	$\exists z\exists x(Fx \wedge \forall yRzy)$	∃I 2

This is fine because line 1 is an instance of line 2, and line 2 is in turn an instance of line 3.

It's important to notice that in moving from line 2 to line 3 here, it was essential that we introduced an existential quantifier involving a *new* variable 'z', which did not yet appear in line 2. The following would *not* have been legitimate:

1	$Fa \wedge \forall yRby$	
2	$\exists x(Fx \wedge \forall yRby)$	∃I 1
3	$\exists y\exists x(Fx \wedge \forall yRyy)$	illegitimate use of ∃I 2

The expression on line 3 involves a *variable clash* (see §5.7) between the newly introduced quantifier '$\exists y$' and the quantifier '$\forall y$' that occurs in its scope. It therefore does not even count as a *formula*, let alone a *sentence* of FOL by the definition we gave in §5.9. For the same reason, we could not have inferred '$\exists x\exists x(Fx \wedge \forall yRxy)$' — this again involves a variable clash, and therefore isn't an FOL formula.

Here's a simple proof that combines our two new quantifier rules:

1	$\forall x Fx$	
2	$\forall y(Fy \to Gy)$	
3	Fa	$\forall$E 1
4	$Fa \to Ga$	$\forall$E 2
5	Ga	$\to$E 4, 3
6	$Ga \land Fa$	$\land$I 5, 3
7	$\exists z(Gz \land Fz)$	$\exists$I 6

Notice that from line 1 I could have inferred some other instance by $\forall$E instead, like ‘Fb, and similarly, from line 2 I could have inferred any other instance, like ‘$Fc \to Gc$’. But if I had used different names in the two instances like this, I could then not have applied $\to$E to them. So although $\forall E$ lets you infer any instance, in the context of a proof you’ll usually have to infer some specific instance.

■ Exercises 6.2

A. Given natural deduction proofs for the following (for these you’ll only have to use $\forall$E and $\exists$I, in addition to TFL rules of course):

1. $\forall x Fx, \forall y(Fy \to Gy) \vdash \exists x(Gx \land \exists y Fy)$
2. $\forall x(Fx \to \forall y Gy), Fc \vdash \exists x(Fx \land Gx)$
3. $\forall x \forall y Rxy \vdash \exists x Rxx$
4. $\forall x(Fx \to Gx) \vdash \forall x Fx \to \exists y Gy$
5. $\exists x Fx \to \forall y Gy \vdash \forall z Fz \to \exists z Gz$
6. $\forall x(Fx \to Gx), \neg Gc \vdash \exists x \exists y(\neg Fx \land Gy)$
7. $\vdash \exists x(Fx \lor \neg Fx)$

6.3 Universal introduction

Suppose you had shown of each particular thing that it is F (and that there are no other things to consider). Then you would be justified in claiming that *everything* is F. This could be used to motivate the following proof rule: if you had established each and every single substitution instance of ‘$\forall x Fx$’, then you can infer ‘$\forall x Fx$’.

Unfortunately, that rule would be utterly unusable. To establish every single substitution instance of ‘$\forall x Fx$’ would require proving ‘Fa’, ‘Fb’, …, ‘Fj_2’, …, ‘Fr_{791}’, … and so on. Since there are infinitely many names in FOL, this process would never end! So we need to be more cunning in coming up with our rule for introducing universal quantifiers.

We can motivate our rule by considering the following:

$$\forall x(Fx \land Gx) \therefore \forall x Fx$$

This argument is obviously valid: if everything is both F *and* G, then everything is F. But how could we prove this? Suppose we begin a proof like this:

1	$\forall x(Fx \land Gx)$	
2	$Fa \land Ga$	$\forall$E 1
3	Fa	$\land$E 2

We have proven 'Fa'. But of course, nothing stops us from using $\forall$E in combination with $\land$E in to likewise prove 'Fb', 'Fc', ..., 'Fj_2', ..., '$Fr_{791}, \ldots$, and so on until we run out of space, time, or patience. So it's clear that from our premise, we could in principle prove prove $F\mathscr{c}$ for *any* name $\mathscr{c}$. Which is just to say that we could in principle prove *every* instance of our goal '$\forall xFx$'. So we should be entitled to infer '$\forall xFx$' by $\forall$I. It's just that we can't *actually* prove every instance, since our proof would never end.

This leads to the following idea: we should be allowed to infer the universal sentence '$\forall xFx$' by the rule of $\forall$I if we are able prove an *arbitrary* instance $F\mathscr{c}$, one that involves some arbitrary name $\mathscr{c}$. For if the name $\mathscr{c}$ is truly arbitrary, then it doesn't matter that we specifically proved this particular instance $F\mathscr{c}$ — we could have picked any other name instead, and thereby proven any other instance of the universal sentence we're aiming for.

Our universal introduction rule $\forall$I implements this idea via a "flagged subproof":

m	$\quad\mathscr{c}$	Flag
	$\quad\vdots$	
n	$\quad\mathscr{A}(\ldots\mathscr{c}\ldots)$	
	$\forall\mathscr{v}\mathscr{A}(\ldots\mathscr{v}\ldots)$	$\forall$I m–n

The Flag-ed name $\mathscr{c}$ may not occur outside the subproof (including in the inferred sentence $\forall\mathscr{v}\mathscr{A}(\ldots\mathscr{v}\ldots)$ itself!)

The Flag step on line m is just a way of officially signaling that the name $\mathscr{c}$ is being introduced as an arbitrary name in the proof, one that we'll use to prove an arbitrary instance of the universal sentence $\forall\mathscr{v}\mathscr{A}(\ldots\mathscr{v}\ldots)$ that we are aiming for. Our proof that $\forall x(Fx \land Gx) \vdash \forall xFx$ can now be presented as follows:

1	$\forall x(Fx \land Gx)$	
2	$\quad a$	Flag
3	$\quad Fa \land Ga$	$\forall$E 1
4	$\quad Fa$	$\land$E 3
5	$\forall xFx$	$\forall$I 2–4

Again, the idea is that although we only proved the one instance, 'Fa', we are allowed to infer the universal sentence '$\forall xFx$' because that instance was arbitrary — we could have just as easily proven any other instance we pleased.

The Flag-ing constraint listed at the bottom of the rule — that the Flag-ed name may not occur outside the subproof — is crucial, because it is what insures that the name we've picked is truly arbitrary.[1] To see the constraint in action, consider this terrible argument:

> Everyone loves Kylie Minogue. Therefore everyone loves themselves.

This argument is obviously not valid. We might symbolize it as:

$$\forall xLxk \therefore \forall xLxx$$

Now, suppose we tried to offer the following "proof" to vindicate this argument:

```
1 | ∀xLxk
  |------
2 | | k          Flag
  | |----
3 | | Lkk        ∀E 1
4 | ∀xLxx        illegitimate use of ∀I
```

It would be bad if this proof were legitimate, since the conclusion doesn't follow. What makes it illegitimate is that the Flag-ed name '*k*' occurs outside the subproof, namely in our premise on line 1. Since '*k*' occurs in the premise, it doesn't have the status of an arbitrary name, and the sentence '*Lkk*' we proved on line 3 doesn't qualify as an arbitrary instance of our goal '$\forall xLxx$': we could *not* have proven *other* instance of '$\forall xLxx$', like e.g. '*Laa*' or '*Ljj*', from our premise.

Notice that the flagged constant also cannot occur in the universal sentence that's being inferred via ∀I either. Consider the following, equally terrible argument:

> Everyone loves themselves. Therefore everyone loves Kylie Minogue.

which we could symbolize as: $\forall xLxx \therefore \forall xLxk$. Now suppose we tried to prove it as follows:

```
1 | ∀xLxx
  |------
2 | | k          Flag
  | |----
3 | | Lkk        ∀E 1
4 | ∀xLxk        illegitimate use of ∀I
```

Again, this prove had better not be legitimate, since the conclusion does not follow. And it isn't legitimate: the Flag-ed name '*k*' occurs outside of the subproof, in the inferred universal sentence '$\forall xLxk$'. Again, although '*Lkk*' is an *instance* of $\forall xLxk$, it doesn't qualify as an *arbitrary* instance because we could not have proven any other instance of '$\forall xLxk$' — like e.g. '*Lak*' or '*Ljk*' — from our premise.

[1]This constraint is actually a little more restrictive than necessary. It would e.g. be alright if the Flag-ed constant occurred earlier in the proof, as long as it doesn't occur in an earlier *undischarged assumption*. But the constraint as we've here formulated it has the advantage of being concise and easy to remember.

For an example of a correct use of ∀I, consider how we might prove that $\forall z(Gz \rightarrow Gz)$ is a theorem. To prove this, we have to open up a flagged subproof inside of which we prove some arbitrary instance of this sentence, such as '$Gd \rightarrow Gd$', as follows:

1	$\mid \quad \mid\ \underline{d}$	Flag
2	$\mid \quad \mid \quad \mid\ \underline{Gd}$	
3	$\mid \quad \mid \quad \mid\ Gd$	Reit 2
4	$\mid \quad \mid\ Gd \rightarrow Gd$	→I 2–3
5	$\mid\ \forall z(Gz \rightarrow Gz)$	∀I 4

The constraints on the legitimate application of ∀I are met, since the name 'd' does not occur outside the subproof. Here '$Gd \rightarrow Gd$' qualifies as an arbitrary instance of '$\forall z(Gz \rightarrow Gz)$': we could just as well have flagged some other name, say 'a', and instead proved the instance '$Ga \rightarrow Ga$' using that name. There was nothing special about 'd'.

6.4 Existential elimination

Suppose we know that *something* is F. The problem is that simply knowing this does not tell us which thing is F. So from '$\exists xFx$' we cannot immediately infer 'Fa', 'Fe_{23}', or any other instance of the sentence. What can we do? How can we deduce anything from existential premises?

Well, suppose we know that something is F, and that everything which is F is G. In English, we might pursue the following line of reasoning:

> Since something is F, there is some particular thing which is an F. We do not know anything about it, other than that it's an F, but for convenience, let's call it 'Obbie'. So: Obbie is F. Since everything which is F is G, it follows that Obbie is G. But since Obbie is G, it follows that something is G. And nothing depended on which object, exactly, Obbie was. Therefore, something is G.

We can capture this reasoning pattern in a proof as follows:

1	$\mid\ \exists xFx$	
2	$\mid\ \underline{\forall x(Fx \rightarrow Gx)}$	
3	$\mid \quad \mid\ \underline{Fo}$	Flag o
4	$\mid \quad \mid\ Fo \rightarrow Go$	∀E 2
5	$\mid \quad \mid\ Go$	→E 4, 3
6	$\mid \quad \mid\ \exists xGx$	∃I 5
7	$\mid\ \exists xGx$	∃E 1, 3–6

Breaking this down: we started by writing down our premises. At line 3, we then made an additional assumption: ‘*Fo*’. The idea here is that premise 1 tell us that *something* is an F. So on line 3 we introduce some arbitrary name ‘*o*’ for that thing, Flag it as arbitrary to the right, and write down the corresponding instance of the existential premise 1. The name we picked is arbitrary, since we’ve assumed nothing about the object named by ‘*o*’ other than that the predicate ‘*F*’ is true of it. On the basis of the assumption *Fo*, we can then establish ‘$\exists xGx$’. Since nothing depended on which specific object ‘*o*’ names, our reasoning patter is perfectly general. We can therefore discharge the assumption ‘*Fo*’, and simply infer ‘$\exists xGx$’ on its own.

Putting this together, we obtain the existential elimination rule ($\exists$E):

$$\begin{array}{r|lll}
m & \exists \mathscr{x}\mathscr{A}(\ldots \mathscr{x} \ldots) & & \\
i & & \vert\ \underline{\mathscr{A}(\ldots \mathscr{c} \ldots)} & \text{Flag } \mathscr{c} \\
j & & \vert\ \mathscr{B} & \\
 & \mathscr{B} & & \exists\text{E } m,\ i\text{–}j
\end{array}$$

The Flag-ed name $\mathscr{c}$ may not occur outside the subproof (including in the original existential $\exists \mathscr{x}\mathscr{A}(\ldots \mathscr{x} \ldots)$ and the inferred sentence $\mathscr{B}$!)

So in general, to prove some sentence $\mathscr{B}$ from an existential sentence $\exists \mathscr{x}\mathscr{A}(\ldots \mathscr{x} \ldots)$, what we do is Flag some arbitrary name $\mathscr{c}$, assume an instance of the existential sentence using this name $\mathscr{c}$, and then prove our goal $\mathscr{B}$ from that instance. Finally, we discharge our assumption and infer $\mathscr{B}$ on its own via $\exists$E.

As with universal introduction, the Flag-ing constraint on the name $\mathscr{c}$ that’s listed at the bottom is very important. To see why, consider the following terrible argument:

> Borges is a librarian. Someone is not a librarian. So Borges is both a librarian and not a librarian.

We might symbolize this obviously invalid argument as follows:

$$Lb, \exists x \neg Lx \therefore Lb \wedge \neg Lb$$

This is clearly a terrible argument: it presumes that the “someone” who is not a librarian according to the second premise is the individual Borges mentioned in the first premise (which can’t be, since Borges is a librarian and the “someone” from premise 2 isn’t). Now, suppose we tried to offer the following proof to vindicate this argument:

$$\begin{array}{r|lll}
1 & Lb & & \\
2 & \underline{\exists x \neg Lx} & & \\
3 & & \vert\ \underline{\neg Lb} & \text{Flag } b \\
4 & & \vert\ Lb \wedge \neg Lb & \wedge\text{E } 1, 3 \\
5 & Lb \wedge \neg Lb & & \text{illegitimate attempt to use } \exists\text{E } 2, 3\text{–}4
\end{array}$$

It would a bad thing if we could prove the conclusion like this, since it doesn't follow from the premises! And the Flag-ing constraint is what prevents us from doing so: the use of ∃E on the last line is not legitimate, because the Flag-ed name on line 3, namely '*b*', continues to appear on line 5, outside the subproof.

We could avoid part of the problem by existentially generalizing line 4 in the subproof to obtain $\exists x(Lx \wedge \neg Lx)$, before discharging our assumption:

1	Lb	
2	$\exists x \neg Lx$	
3	$\quad \neg Lb$	Flag *b*
4	$\quad Lb \wedge \neg Lb$	∧E 1, 3
5	$\quad \exists x(Lx \wedge \neg Lx)$	∃I 4
6	$\exists x(Lx \wedge \neg Lx)$	illegitimate attempt to use ∃E 2, 3–5

But this is no better. If it were legitimate, this proof would vindicate an argument of the following sort:

> Borges is a librarian. Someone is not a librarian. Therefore someone both is and is not a librarian.

This is clearly a bad argument: we can't assume that the "someone" who is not a librarian according to the second premise is, specifically, the individual Borges mentioned in the first premise. And again, the Flag-ing constraint rules out our supposed "proof": the use of ∃E on the the last line is not legitimate because the Flag-ed name '*b*' occurs outside the subproof, namely in the premise on line 1.

The overarching problem with both proofs is that because the name '*b*' already occurs in one of our premises, it does not have the status of an arbitrary name in our proof, and therefore can't be used as an arbitrary name for whatever object premise 2 tell us is not a librarian. The moral is: *if you want to squeeze information out of an existential quantifier, choose a new name for your substitution instance.* That way, you will meet the constraints on the rule for ∃E.

Let's work through a more complicated proof that requires both $\exists E$ and $\forall I$ at the same time. We'll show that the following is valid:

$$\forall x \exists y Lxy, \forall x \forall y(Lxy \rightarrow Lyx) \therefore \forall x \exists y Lyx$$

If we read '*L*' as 'loves', this argument says that if everyone loves someone, and loves is always reciprocated — in the sense that if *x* loves *y*, then *y* loves *x* back — it follows that everyone is loved by someone. We can show that this is valid with the following proof:

1	$\forall x \exists y Lxy$	
2	$\forall x \forall y (Lxy \to Lyx)$	
3	$\quad a$	Flag
4	$\quad \exists y Lay$	$\forall$E 1
5	$\quad\quad Lab$	Flag b
6	$\quad\quad \forall y (Lay \to Lya)$	$\forall$E 2
7	$\quad\quad Lab \to Lba$	$\forall$E 6
8	$\quad\quad Lba$	$\to$E 7, 5
9	$\quad\quad \exists y Lya$	$\exists$I 8
10	$\quad \exists y Lya$	$\exists$E 4, 5–9
11	$\forall x \exists y Lyx$	$\forall$I 3–10

This is a relatively complex proof, so let's think through it systematically. As usual, we work backward from the conclusion we're aiming for: we are trying to prove '$\forall x \exists y Lyx$', i.e. that everyone is loved by someone. Since this is a universal sentence, we use $\forall$I as our overall strategy: we pick an arbitrary name, say 'a', and open up a subproof where we Flag 'a', and make it our goal to prove the 'a'-instance of our conclusion, $\exists y Lya$, which says that someone loves a . If we are able to complete the subproof, we're allowed to infer since a is loved by someone, and a was arbitrary, everyone is loved by someone.

So what do our premises imply about our object a? Well, premise 1 says that everyone loves someone, so we can infer by $\forall$E that a in particular loves someone, as we did on line 4. Since a loves *someone*, we can give that someone a name, say 'b', in order to reason about them. So we can say a loves b. The way this works in our proof is that given the existential sentence '$\exists y Lay$' on line 4, we assume Lab as an arbitrary instance of it on line 5.

Next, premise 2 tells us that love is reciprocal. So given that a loves b, we can conclude that b loves a. In our proof, we did this by obtaining line 7 via two steps of $\forall$E on premise 2, and then doing $\to$E on that. Alright: so given that b loves a, we an conclude that a is loved by *someone*, as on line 9. And at this point, having gotten rid of the name 'b', we can pop out of our subproof by $\exists$E. And finally, since a was arbitrary, we can pop out of our Flag-ed subproof and conclude by $\forall$I that *everyone* is loved by someone, as on line 11.

■ Exercises 6.4

A. Explain why these two 'proofs' are incorrect.

1	$\forall x Rxx$	
2	Raa	$\forall$E 1
3	$\forall y Ray$	$\forall$I 2
4	$\forall x \forall y Rxy$	$\forall$I 3

$$\begin{array}{r|l l}
1 & \underline{\forall x \exists y Rxy} & \\
2 & \exists y Ray & \forall\text{E } 1 \\
3 & \quad \vert\ \underline{Raa} & \\
4 & \quad \vert\ \exists x Rxx & \exists\text{I } 3 \\
5 & \exists x Rxx & \exists\text{E } 2, 3\text{–}4
\end{array}$$

B. The following three proofs are missing their citations (rule and line numbers). Add them, to turn them into full proofs.

$$\begin{array}{r|l}
1 & \forall x \exists y (Rxy \lor Ryx) \\
2 & \underline{\forall x \neg Rmx} \\
3 & \exists y (Rmy \lor Rym) \\
4 & \quad \vert\ \underline{Rma \lor Ram} \\
5 & \quad \vert\ \neg Rma \\
6 & \quad \vert\ Ram \\
7 & \quad \vert\ \exists x Rxm \\
8 & \exists x Rxm
\end{array}$$

$$\begin{array}{r|l}
1 & \forall x (\exists y Lxy \to \forall z Lzx) \\
2 & \underline{Lab} \\
3 & \exists y Lay \to \forall z Lza \\
4 & \exists y Lay \\
5 & \forall z Lza \\
6 & Lca \\
7 & \exists y Lcy \to \forall z Lzc \\
8 & \exists y Lcy \\
9 & \forall z Lzc \\
10 & Lcc \\
11 & \forall x Lxx
\end{array}$$

$$\begin{array}{r|l}
1 & \forall x (Jx \to Kx) \\
2 & \exists x \forall y Lxy \\
3 & \underline{\forall x Jx} \\
4 & \quad \vert\ \underline{\forall y Lay} \\
5 & \quad \vert\ Laa \\
6 & \quad \vert\ Ja \\
7 & \quad \vert\ Ja \to Ka \\
8 & \quad \vert\ Ka \\
9 & \quad \vert\ Ka \land Laa \\
10 & \quad \vert\ \exists x (Kx \land Lxx) \\
11 & \exists x (Kx \land Lxx)
\end{array}$$

C. In §A problem part A, we considered fifteen syllogistic figures of Aristotelian logic. Provide proofs for each of the argument forms. NB: You will find it *much* easier if you symbolize (for example) 'No F is G' as '$\forall x(Fx \to \neg Gx)$'.

D. Aristotle and his successors identified other syllogistic forms which depended upon 'existential import'. Symbolize each of these argument forms in FOL and offer proofs.

- **Barbari.** Something is H. All G are F. All H are G. So: Some H is F
- **Celaront.** Something is H. No G are F. All H are G. So: Some H is not F
- **Cesaro.** Something is H. No F are G. All H are G. So: Some H is not F.
- **Camestros.** Something is H. All F are G. No H are G. So: Some H is not F.
- **Felapton.** Something is G. No G are F. All G are H. So: Some H is not F.
- **Darapti.** Something is G. All G are F. All G are H. So: Some H is F.
- **Calemos.** Something is H. All F are G. No G are H. So: Some H is not F.
- **Fesapo.** Something is G. No F is G. All G are H. So: Some H is not F.
- **Bamalip.** Something is F. All F are G. All G are H. So: Some H are F.

E. Provide a proof of each claim.

1. $\vdash \forall xFx \vee \neg\forall xFx$
2. $\vdash \forall z(Pz \vee \neg Pz)$
3. $\forall x(Ax \to Bx), \exists xAx \vdash \exists xBx$
4. $\forall x(Mx \leftrightarrow Nx), Ma \wedge \exists xRxa \vdash \exists xNx$
5. $\forall x\forall yGxy \vdash \exists xGxx$
6. $\vdash \forall xRxx \to \exists x\exists yRxy$
7. $\vdash \forall y\exists x(Qy \to Qx)$
8. $Na \to \forall x(Mx \leftrightarrow Ma), Ma, \neg Mb \vdash \neg Na$
9. $\forall x\forall y(Gxy \to Gyx) \vdash \forall x\forall y(Gxy \leftrightarrow Gyx)$
10. $\forall x(\neg Mx \vee Ljx), \forall x(Bx \to Ljx), \forall x(Mx \vee Bx) \vdash \forall xLjx$

F. Write a symbolization key for the following argument, symbolize it, and prove it:

> There is someone who likes everyone who likes everyone that she likes. Therefore, there is someone who likes herself.

G. For each of the following pairs of sentences: If they are provably equivalent, give proofs to show this. If they are not, construct an interpretation to show that they are not logically equivalent.

1. $\forall xPx \to Qc, \forall x(Px \to Qc)$
2. $\forall x\forall y\forall zBxyz, \forall xBxxx$
3. $\forall x\forall yDxy, \forall y\forall xDxy$
4. $\exists x\forall yDxy, \forall y\exists xDxy$
5. $\forall x(Rca \leftrightarrow Rxa), Rca \leftrightarrow \forall xRxa$

H. For each of the following arguments: If it is valid in FOL, give a proof. If it is invalid, construct an interpretation to show that it is invalid.

1. $\exists y\forall xRxy \therefore \forall x\exists yRxy$
2. $\exists x(Px \wedge \neg Qx) \therefore \forall x(Px \to \neg Qx)$
3. $\forall x(Sx \to Ta), Sd \therefore Ta$
4. $\forall x(Ax \to Bx), \forall x(Bx \to Cx) \therefore \forall x(Ax \to Cx)$

5. $\exists x(Dx \vee Ex), \forall x(Dx \rightarrow Fx) \therefore \exists x(Dx \wedge Fx)$
6. $\forall x \forall y(Rxy \vee Ryx) \therefore Rjj$
7. $\exists x \exists y(Rxy \vee Ryx) \therefore Rjj$
8. $\forall x Px \rightarrow \forall x Qx, \exists x \neg Px \therefore \exists x \neg Qx$

6.5 Rules for identity

In §5.8, I mentioned that in saying of some objects *a* and *b* that they are *identical*, we don't merely mean that they are very similar to each other, or indistinguishable in the way that e.g. two cans of soda or two pennies might be. Rather, they have to be one and the same object. It follows that no matter how much you tell me about what *a* and *b* are like, qualitatively, this won't suffice to conclude that $a = b$. Indeed, no sentences which do not *already* involve an identity claim could justify an inference to '$a = b$'

However, we can be sure that every object is identical to *itself*. No premises are required to conclude that much. So this will be the identity introduction rule:

$$\begin{array}{l|ll} & \mathcal{c} = \mathcal{c} & =\text{I} \end{array}$$

Notice that this rule does not require referring to any prior lines of the proof. For any name $\mathcal{c}$, you can just write $\mathcal{c} = \mathcal{c}$ at any point, with only the =I rule as justification.

Our elimination rule is more fun. If you have established '$a = b$', then anything that is true of the object named by '*a*' must also be true of the object named by '*b*', since they are one and the same. This means that given any sentence with '*a*' in it, you can replace some or all of the occurrences of '*a*' with '*b*'. For example, from '*Raa*' and '$a = b$', you are justified in inferring '*Rab*', '*Rba*' or '*Rbb*'. More generally:

$$\begin{array}{l|ll} m & \mathcal{a} = \mathcal{b} & \\ n & \mathscr{A}(\ldots \mathcal{a} \ldots \mathcal{a} \ldots) & \\ & \mathscr{A}(\ldots \mathcal{b} \ldots \mathcal{a} \ldots) & =\text{E } m, n \end{array}$$

The notation here should be understood as follows: $\mathscr{A}(\ldots \mathcal{a} \ldots \mathcal{a} \ldots)$ is a sentence containing the name $\mathcal{a}$, and $\mathscr{A}(\ldots \mathcal{b} \ldots \mathcal{a} \ldots)$ is a sentence obtained by replacing one or more occurrences of the name $\mathcal{a}$ with the name $\mathcal{b}$. Lines *m* and *n* can occur in either order, and do not need to be adjacent, but we always cite the statement of identity first.

Symmetrically, we allow:

$$\begin{array}{l|ll} m & \mathcal{a} = \mathcal{b} & \\ n & \mathscr{A}(\ldots \mathcal{b} \ldots \mathcal{b} \ldots) & \\ & \mathscr{A}(\ldots \mathcal{a} \ldots \mathcal{b} \ldots) & =\text{E } m, n \end{array}$$

That is, if we have established $\mathscr{a} = \mathscr{b}$, and we have a sentence that contains the name $\mathscr{b}$, then we are allowed to infer any sentence that results from the first by replacing one or more occurrence of $\mathscr{b}$ with $\mathscr{a}$.

This rule is closely related to LEIBNIZ'S LAW, which we briefly discussed in §5.8. Leibniz's Law says that if x and y are identical, then x has any given property iff y does too. The following, for example, is an instance of Leibniz's Law:

$$\forall x \forall y \big[x = y \to (Dx \leftrightarrow Dy)\big]$$

For example, if 'D' represents the property of being a dog, Leibniz's Law tell us that if $x = y$, then x is a dog iff y is also a dog. We can prove this using =E as follows:

1	$\mid\ \mid\ \underline{a}$	Flag
2	$\mid\ \mid\ \mid\ \underline{b}$	Flag
3	$\mid\ \mid\ \mid\ \mid\ \underline{a = b}$	
4	$\mid\ \mid\ \mid\ \mid\ \mid\ \underline{Da}$	
5	$\mid\ \mid\ \mid\ \mid\ \mid\ Db$	=E 3, 4
6	$\mid\ \mid\ \mid\ \mid\ \mid\ \underline{Db}$	
7	$\mid\ \mid\ \mid\ \mid\ \mid\ Da$	=E 3, 6
8	$\mid\ \mid\ \mid\ \mid\ Da \leftrightarrow Db$	$\leftrightarrow$I 4–5, 6–7
9	$\mid\ \mid\ \mid\ a = b \to (Da \leftrightarrow Db)$	$\to$I 3–8
10	$\mid\ \mid\ \forall y(a = y \to (Da \leftrightarrow Dy))$	$\forall$I 2–9
11	$\mid\ \forall x \forall y(x = y \to (Dx \leftrightarrow Dy))$	$\forall$I 1–10

The =I rule is also related to a logical law discussed in §5.8, the LAW OF IDENTITY, which says that everything is identical to itself, i.e. that $\forall x\, x = x$. We can prove this as a theorem using our =I rule:

1	$\mid\ \mid\ \underline{a}$	Flag
2	$\mid\ \mid\ a = a$	=I
3	$\mid\ \forall x\, x = x$	$\forall$I 1–2

This shows that identity is *reflexive*, i.e. it is a relation that everything bears to itself. The relation of being at-least-as-tall-as would be another example of a reflexive relation, since everyone is at least as tall as themselves. Using our rules, we can prove other properties of the identity relation as well. For example, we can prove that it is *symmetric*:

1	a	Flag
2	b	Flag
3	$a = b$	
4	$a = a$	=I
5	$b = a$	=E 3, 4
6	$a = b \rightarrow b = a$	→I 3–5
7	$\forall y(a = y \rightarrow y = a)$	∀I 2–6
8	$\forall x \forall y(x = y \rightarrow y = x)$	∀I 1–7

Here we obtain line 5 by replacing one instance of '*a*' in line 4 with an instance of '*b*', which is justified given '$a = b$'.

■ Exercises 6.5

A. Here are various important properties that a binary relation *R* could have:

R is **reflexive** iff $\forall x Rxx$

R is **serial** iff $\forall x \exists y Rxy$

R is **symmetric** iff $\forall x \forall y(Rxy \rightarrow Ryx)$

R is **transitive** iff $\forall x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$

R is **euclidean** iff $\forall x \forall y \forall z((Rxy \wedge Rxz) \rightarrow Ryz)$

Above we showed that identity is both reflexive and symmetric. Show that identity also has the other three properties: it is transitive, euclidean, and serial. That is, prove each of the following theorems:

1. $\vdash \forall x \forall y \forall z((x = y \wedge y = z) \rightarrow x = z)$
2. $\vdash \forall x \forall y \forall z((x = y \wedge x = z) \rightarrow y = z)$
3. $\vdash \forall x \exists y\, x = y$

B. Provide a proof of each claim. (Remember that $t_1 \neq t_2$ is shorthand for the negated identity sentence $\neg t_1 = t_2$).

1. $Pa \vee Qb, Qb \rightarrow b = c, \neg Pa \vdash Qc$
2. $m = n \vee n = o, An \vdash Am \vee Ao$
3. $\forall x\, x = m, Rma \vdash \exists x Rxx$
4. $\forall x \forall y(Rxy \rightarrow x = y) \vdash Rab \rightarrow Rba$
5. $\neg \exists x\, x \neq m \vdash \forall x \forall y(Px \rightarrow Py)$
6. $\exists x Jx, \exists x \neg Jx \vdash \exists x \exists y\, x \neq y$
7. $\forall x(x = n \leftrightarrow Mx), \forall x(Ox \vee \neg Mx) \vdash On$

8. $\exists x Dx, \forall x(x = p \leftrightarrow Dx) \vdash Dp$
9. $\exists x\big[(Kx \land \forall y(Ky \to x = y)) \land Bx\big], Kd \vdash Bd$
10. $\vdash Pa \to \forall x(Px \lor x \neq a)$

C. Show that the following are provably equivalent:

- Fa
- $\exists x(x = a \land Fx)$

D. The following are all acceptable ways to symbolize the English sentence 'there is exactly one F':

- $\exists x Fx \land \forall x \forall y\big[(Fx \land Fy) \to x = y\big]$
- $\exists x\big[Fx \land \forall y(Fy \to x = y)\big]$
- $\exists x \forall y(Fy \leftrightarrow x = y)$

Show that they are all provably equivalent. (*Hint*: to show that three claims are provably equivalent, it suffices to show that the first proves the second, the second proves the third and the third proves the first; think about why.)

E. Symbolize the following argument

> There is exactly one F. There is exactly one G. Nothing is both F and G. So: there are exactly two things.

And offer a proof of it.

6.6 Derived Rules for FOL

In §5.2 we noted the QUANTIFIER EQUIVALENCE LAWS, which govern the interaction of quantifiers and negation. For example, '$\forall x \neg Ax$' implies '$\neg \exists x Ax$', as the following proof demonstrates:

1	$\forall x \neg Ax$	
2	$\quad \exists x Ax$	
3	$\quad\quad Ac$	Flag c
4	$\quad\quad \neg Ac$	$\forall$E 1
5	$\quad\quad \bot$	$\neg$E 3, 4
6	$\quad \bot$	$\exists$E 2, 3–5
7	$\neg \exists x Ax$	$\neg$I 2–6

The implication also holds in the other direction, though we'll leave that proof as an exercise. Given this, we can introduce some additional derived rules into our proof system that will let us move negations across quantifiers in this manner. We'll label these rules CQ, for "Conversion of Quantifiers":

$$\begin{array}{r|ll}
m & \forall \mathscr{v} \neg \mathscr{A} & \\
 & \neg \exists \mathscr{v} \mathscr{A} & \text{CQ } m
\end{array}$$

$$\begin{array}{r|ll}
m & \neg \exists \mathscr{v} \mathscr{A} & \\
 & \forall \mathscr{v} \neg \mathscr{A} & \text{CQ } m
\end{array}$$

$$\begin{array}{r|ll}
m & \exists \mathscr{v} \neg \mathscr{A} & \\
 & \neg \forall \mathscr{v} \mathscr{A} & \text{CQ } m
\end{array}$$

$$\begin{array}{r|ll}
m & \neg \forall \mathscr{v} \mathscr{A} & \\
 & \exists \mathscr{v} \neg \mathscr{A} & \text{CQ } m
\end{array}$$

Using these rules, we can give a pair of quick proofs showing that '$\exists x Fx$' is provably equivalent to '$\neg \forall x \neg Fx$':

$$\begin{array}{r|l|ll}
1 & \exists x Fx & & \\ \cline{2-2}
2 & & \forall x \neg Fx & \\ \cline{3-3}
3 & & \neg \exists x Fx & \text{CQ } 2 \\
4 & & \bot & \neg\text{E } 1, 3 \\
5 & \neg \forall x \neg Fx & & \neg\text{I } 2\text{–}4
\end{array}$$

and:

$$\begin{array}{r|ll}
1 & \neg \forall x \neg Fx & \\ \cline{2-2}
2 & \forall x \neg\neg Fx & \text{CQ } 1 \\
3 & \neg\neg Fa & \forall\text{E } 2 \\
4 & Fa & \text{DN } 3 \\
5 & \exists x Fx & \exists\text{I } 4
\end{array}$$

The fact, in general, $\exists \mathscr{v} \mathscr{A}$ is equivalent to $\neg \forall \mathscr{v} \neg \mathscr{A}$ means that we don't really need to have '$\exists$' as a primitive symbol in FOL. We could instead treat $\exists \mathscr{v} \mathscr{A}$ as just a shorthand notation for $\neg \forall \mathscr{v} \neg \mathscr{A}$, much as we've been treating $t_1 \neq t_2$ as a shorthand for $\neg t_1 = t_2$.

■ Exercises 6.6

A. Show that the following are jointly inconsistent, i.e. that they together imply a contradiction:

1. $Sa \to Tm, Tm \to Sa, Tm \wedge \neg Sa$
2. $\neg\exists xRxa, \forall x\forall yRyx$
3. $\neg\exists x\exists yLxy, Laa$
4. $\forall x(Px \to Qx), \forall z(Pz \to Rz), \forall yPy, \neg Qa \wedge \neg Rb$

B. Show that each pair of sentences is provably equivalent:

1. $\forall xFx, \neg\exists x\neg Fx$
2. $\forall x(Ax \to \neg Bx), \neg\exists x(Ax \wedge Bx)$
3. $\neg\forall x(Ax \to Bx), \exists x(Ax \wedge \neg Bx)$
4. $\forall x(\neg Ax \to Bd), \forall xAx \vee Bd$

C. The following pairs of sentences are all equivalent, showing that we can move quantifiers "across" logical operators. Give proofs to that they are equivalent:

1. $\forall x(Fx \wedge Ga), \forall xFx \wedge Ga$
2. $\exists x(Fx \vee Ga), \exists xFx \vee Ga$
3. $\forall x(Ga \to Fx), Ga \to \forall xFx$
4. $\forall x(Fx \to Ga), \exists xFx \to Ga$
5. $\exists x(Ga \to Fx), Ga \to \exists xFx$
6. $\exists x(Fx \to Ga), \forall xFx \to Ga$

When all the quantifiers occur at the beginning of a sentence, that sentence is said to be in *prenex normal form*. These equivalences are sometimes called *prenexing rules*, since they give us a means for putting any sentence into prenex normal form. For example, '$\exists xFx \wedge \forall yGy$' can be put into prenex normal form as '$\exists x\forall y(Fx \wedge Gy)$, or also as '$\forall y\exists x(Fx \wedge Gy)$'.

Semantics of FOL 7

Recall that in TFL, we had the notion of a *valuation*: an assignment of truth values to atomic sentences. We then gave a semantics that allowed us to compute the truth value of an arbitrarily complex TFL sentence on any given valuation. Finally, we used this notion of truth-on-a-valuation to define various logical concepts, like equivalence and entailment.

In this chapter we will do something very similar for FOL, except that the notion of a valuation now gets replaced by the more complex notion of an *interpretation*. We'll first have to see what an FOL interpretation is, and then learn how to compute the truth values of arbitrarily complex FOL sentences in a given interpretation. With that in hand, we can then again define logical concepts — like logical entailment, or validity — for FOL.

Ultimately, we'll be able to use this machinery to show that a given FOL argument is not valid, by giving an interpretation that makes its premises true and its conclusion false. For reasons that will emerge in this chapter, we won't use interpretations to show that arguments are valid. We will use the natural deduction proofs we learned in the last chapter to do this.

7.1 Predicates and their Extensions

TFL was a truth-functional language. Its connectives are all truth-functional, and consequently, TFL only cares about what truth-values sentences have. We can assign a truth value *directly*, via a valuation that just stipulates that the sentence 'P', for example, is to have the value *true*. Alternatively, we can do it *indirectly*, by offering a symbolization key, e.g.:

P: Big Ben is in London

This stipulates that the TFL sentence 'P' is to have the same truth value as the English sentence 'Big Ben is in London' (which, as it happens, is true). But no further aspect of the meaning of the English sentence carries over to TFL.

FOL is similarly impoverished as regards meaning. It goes beyond mere truth values, because it allows us to split atomic sentences into their parts, consisting of terms and predicates. A term is a word that refers to a particular object, and a predicate is a word that is *true of* objects. But FOL doesn't care about any other aspect of predicate's meaning besides what objects it's true of, or any other aspect of a term's meaning besides what it refers to. For example, when we provide a symbolization key for some FOL predicate, such as:

S: ____ is a US state.

this isn't intended to suggest that our FOL predicate '*S*' carries the same *meaning* as the English predicate. It simply tells us that the FOL predicate is to be *true of* exactly those things that the English predicate '____ is a US state' is true of.

Alternatively, we can stipulate what objects a predicate is true of *directly*, by just listing those objects. So we might stipulate that '*S*' is to be true of: Alabama, Alaska, Arizona, ...and so on, listing all 50 states. This is a perfectly legitimate interpretation of an FOL predicate, because, again, all we care about is what objects it's true of, and our stipulation settles this. The things a predicate is true of comprise the EXTENSION of the predicate. FOL is said to be an EXTENSIONAL LANGUAGE because it doesn't care about any aspect of a predicate's meaning besides its extension.

Our stipulations about predicate extensions can be as arbitrary as we like. For example, we could stipulate that '*H*' should have an extension consisting of the following objects:

H: Barack Obama, the number π, the play *Hamlet*

It doesn't matter that these objects have nothing in common, they still form a perfectly good predicate extension. Suppose we add the following names to our symbolization key:

b: Usain Bolt
o: Barack Obama
p: the number π

Together, these stipulations settle the truth value of any atomic FOL sentence formed from the predicate '*H*' and the names '*d*', '*o*', and '*p*': the sentences '*Ho*' and '*Hp*' will both be true on this interpretation, because Obama and π are in the predicate's extension, and '*Hb*' will be false, because Usain Bolt is not one of the objects '*H*' is true of.

Many-Place Predicates Things get slightly more complicated when we move from one-place predicates to two-place predicates. Consider a symbolization key like:

L: ____loves ____

This key should be read as saying something like:

'*L*' is true of *x* and *y* (in that order) iff x loves y.

The qualifier "in that order" is very important here. Since *x* might love *y* without *y* also loving *x*, a two-place predicate like this can apply to a pair of objects in one order but not another.

How should a direct stipulation of the extension of a two-place predicate like this look? This is a bit tricky. If we just *list* objects that '*L*' applies to, we won't know which of the objects are the lovers and which are the objects they love. A simple list would in other words give us no way to indicate the order in which the predicate holds of objects.

To deal wit this, we instead let two-place predicates be true of *pairs* of objects. We could for example stipulate that '*R*' is to be true of, and only of, the following pairs of objects, indicated with angle brackets:

$R:$ ⟨Lenin, Marx⟩
⟨Heidegger, Sartre⟩
⟨Sartre, Heidegger⟩
⟨Marx, Marx⟩

The angle-brackets tell us in what order the predicate R applies to the objects: the first object between the brackets always corresponds to the first argument slot in the predicate, and the second object corresponds to the second argument slot. Suppose, for example, that we add the following stipulations for names:

l:	Lenin
m:	Marx
h:	Heidegger
s:	Sartre

Then 'Rlm' will be true, since the pair ⟨Lenin, Marx⟩ is the extension of 'R'. But 'Rml' will be false, since ⟨Marx, Lenin⟩ is not in the extension of 'R'. However, both 'Rhs' and 'Rsh' will be true, since both ⟨Heidegger, Sartre⟩ and ⟨Sartre, Heidegger⟩ are on our list of pairs, and Rmm will also be true, since the pair ⟨Marx, Marx⟩ is in R's extension.

If we were dealing with a three-place predicate, its extension would consist not of *pairs* of objects, but of ordered *triples* of objects, like ⟨Heiddegger, Marx, Sartre⟩. And a four-place predicate would have ordered *quadruples* of objects in its extension, a five-place predicate would have ordered *quintuples*, and so on. In general, we call ordered things like these TUPLES. So the extension of a many-place predicate can be specified by giving a list of tuples: either of pairs, or of triples, or of quadruples etc. depending on whether we're dealing with a two-, three-, or four-place predicate.

The Identity Predicate Lastly, we have identity, which receives special treatment in FOL. As we've seen, we write it a bit differently from other two-place predicates: '$x = y$' instead of 'Ixy' (for example). More important, though, is that its interpretation is fixed, once and for all. The way to determine the truth value of an identity statement is very simple: $\mathscr{a} = \mathscr{b}$ is true if $\mathscr{a}$ and $\mathscr{b}$ refer to the same object, and false otherwise.

So given our stipulations above, '$m = m$' and '$w = w$' are, for example, both true, whereas '$w = m$' or '$l = s$' are false. It's important to notice, though, that identity statements involving two different names can be still be true if the two names refer to the same object. For example, if we had the following symbolization key:

l:	Lady Gaga
c:	Kurt Cobain
g:	Lady Gaga

then '$l = g$' and '$g = l$' would both be true despite involving different names. Having two names for the same object isn't that unusual, and in fact happens all the time when people have nicknames, pseudonyms, or stage names: 'Stefani Germanotta' and 'Lady Gaga' refer to the same person, and so do 'Jay-Z' and 'Shawn Carter', for example.

7.2 FOL Interpretations

We defined a *valuation* of a sentence $\mathscr{A}$ (or collection of sentences) of TFL to be an assignment of truth values to all the atomic sentences contained in $\mathscr{A}$ (or the collection). In FOL, the role of a valuation will be played by an INTERPRETATION. FOL interpretations are more complex than TFL valuations, because they have *three* components:

An FOL INTERPRETATION of a sentence $\mathscr{A}$ (or of a collection of sentences $\mathtt{C}$) consist of:

1. A specification of a DOMAIN containing at least one object
2. For each name in $\mathscr{A}$ (or in $\mathtt{C}$), an assignment of exactly one object in the domain. This object is the name's REFERENT.
3. For each predicate in $\mathscr{A}$ (or in $\mathtt{C}$), a specification of what objects in the domain (if any), and in what order, that predicate is true of. This constitutes the predicate's EXTENSION.[a]

[a]So notice that a predicate can have an empty extension, in which case it it isn't true of any objects in the domain. By contrast, we don't allow "empty names" that lack a referent.

Symbolization keys like those we used in chapter 5 consequently give us one convenient way to present an interpretation. For example, the following counts as an FOL interpretation of '$(Lw \wedge Tw)$':

Domain:	people
w:	Wittgenstein
L:	____ is a logician
A:	____ is a school teacher

This has all three components: (i) a specification of a domain, (ii) a specification of a referent for every name in '$(Lw \wedge Tw)$', and (iii) a specification of an extension for every predicate in '$(Lw \wedge Tw)$'. The interpretation then determines a truth value for the sentence. In this case, since Wittgenstein was both a logician and a school teacher, both 'L' and 'T' are true of the referent of the name 'w' on this interpretation, and so '$(Lw \wedge Tw)$' is true.

Alternatively, we can specify interpretations by just directly listing the objects that predicates are true of, as discussed in the previous section. In fact, as we move on, it will often be convenient to consider fairly abstract interpretations where the domain consists of natural numbers, i.e. positive integers, rather than people, or plants, or other objects. One possible interpretation of '$\exists x(Fx \wedge Gx)$', for example, would be the following:[1]

Domain:	1, 2, 3
F:	
G:	1, 3

Here the domain contains the numbers 1, 2, and 3, the predicate 'F' is true of none of those objects, and 'G' is true of 1 and 3. As you can probably guess, '$\exists x(Fx \wedge Gx)$' comes out

[1]Although this kind of domain officially consists of just numbers, we could in principle regard it as consisting of any objects we like — we're just calling these objects 'object 1', 'object 2', 'object 3' etc. for simplicity.

false on this interpretation, since there's no object in the domain of which both 'F' and 'G' are true. We'll look at how to determine truth values more closely in the next two sections.

It will often be useful to represent directly-specified interpretations *diagramatically*. Interpretations like the one above that involve only one-place predicates can be represented using a MATRIX DIAGRAM:

	F	G
1	$-$	$+$
2	$-$	$-$
3	$-$	$+$

Here we list the objects in the domain on the left, and then put $+$'s and $-$'s under each predicate to indicate which objects that predicate is true of. If we were also considering some FOL names, we could include those in our diagram by listing each one to the left of whichever object it refers to.

As we discussed in the last section, to directly specify the extension of a *many-place* predicate, we have to give a list of *tuples* of objects, rather than individual objects, to indicate in what order the predicate holds of the objects. So for example, if we take the sentence '$\forall xRxx$' which contains the two-place predicate 'R', one possible interpretation would be the following:

Domain: 1, 2, 3, 4
R: $\langle 1, 4\rangle$, $\langle 4,4\rangle$, $\langle 1, 2\rangle$, $\langle 2, 1\rangle$ $\langle 1, 3\rangle$,

To represent interpretations with two-place predicates diagramatically, we can use ARROW DIAGRAMS like this:

$$\begin{tikzcd}
1 \arrow[r, leftrightarrow] \arrow[d] \arrow[dr] & 2 \\
4 \arrow[loop right] & 3
\end{tikzcd}$$

The arrows represent the order in which 'R' holds of the objects. To indicate that 1 bears the relation represented by 'R' to 4, we draw an arrow from 1 to 4, and to indicate that 4 bears the relation to itself, we draw an arrow that loops from 4 back to 4, and to indicate that 1 bears the relation to 2, and 2 also bears it to 1, we draw a double-arrow between them, and so on. So there is one arrow for each pair in the extension of the predicate.

If we wanted, we could make such arrow diagrams more complex. For example, we could label objects in our diagram with FOL names to indicate which object each name refers to. Equally, to represent the extension of a one-place predicate in an arrow diagram, we could draw a circle around some objects and label the circle with the predicate. And to represent the extensions of multiple two-place predicates in a single arrow diagram, we might use arrows with dashed as opposed to solid lines, or arrows of different colors.

7.3 Truth in FOL

We have introduced interpretations. Our next task is to give a precise account of what it is for an arbitrarily complex FOL sentence to be true or false in a given interpretation. There are three kinds of sentence in FOL: atomic sentences, sentences whose main logical operator is a sentential connective, and lastly, sentences whose main logical operator is a quantifier. We'll go through each kind in turn.

Our explanation will be completely general, However, to make things comprehensible, it will be useful to have a particular interpretation to hand in order to give examples. Let's use the following as our *go-to interpretation*:

Domain:	all positive integers
a:	2
b:	3
E:	____is even
R:	____ is less than ____

I've here specified the extensions of '*E*' and '*R*' indirectly, using English predicates, because I can't list all the numbers, or pairs of numbers, these are true of. But the extension of '*E*' contains even numbers $2, 4, 6, 8\ldots$, and the extension of '*R*' contains the pair $\langle 2, 3\rangle$, as well as $\langle 2, 4\rangle$ and $\langle 2,5\rangle$ and $\langle 3, 5\rangle$, and indeed every pair $\langle$x,y$\rangle$ where *x* is less than *y*.

Atomic Sentences Determining the truth value of an atomic sentence on a given interpretation is fairly straightforward. The atomic sentence '*Ea*', for example, is true just in case '*E*' is true of the referent of '*a*'. Given our go-to interpretation, this sentence is true since '*a*' refers to 2 and 2 is indeed an even number. By contrast, '*Eb*' would be false on our interpretation, because 3, which is the referent of '*b*', is not in the extension of '*E*', since it is not an even number.

Similarly, '*Rab*' is true on our interpretation just in case the referent of '*a*' is less than the referent of '*b*'. Since 2 is indeed less than 3, '*Rab*' is true. By contrast, '*Rba*' is not true, because 3 is not less than 2, or to put it another way, the pair $\langle 3,2\rangle$ is not in the extension of '*R*' on our interpretation. Similarly, neither '*Raa*' nor '*Rbb*' are true on our go-to interpretation, since neither 2 nor 3 is less than itself (i.e. the pairs $\langle 2,2\rangle$ and $\langle 3,3\rangle$ are not in the extension of '*R*'). In general we can say:

> When $\mathscr{R}$ is an *n*-place predicate and $\mathscr{a}_1, \ldots, \mathscr{a}_n$ are names, the sentence $\mathscr{R}\mathscr{a}_1 \ldots \mathscr{a}_n$ is true in a given interpretation **iff** $\mathscr{R}$ is true of the objects referred to by $\mathscr{a}_1, \ldots, \mathscr{a}_n$ (in that order) in that interpretation

There is one additional kind of atomic sentence to consider: those formed by connecting two names with an identity sign. As already mentioned, determining the truth values of identity statements is straightforward:

> For any names $\mathscr{a}$ and $\mathscr{b}$, the sentence $\mathscr{a} = \mathscr{b}$ is true in a given interpretation **iff** $\mathscr{a}$ and $\mathscr{b}$ refer to very same object in that interpretation

So, in our go-to interpretation, '$a = b$' is false, since '*a*' refers to the number 2 and '*b*' to the number 3, and 2 is distinct from 3. On the other hand, both '$a = a$' and '$b = b$' are true.

Adding Truth Functional Connectives Just like in TFL, sentences of FOL can be built up from simpler ones using the truth-functional operators. The semantic rules governing these operators are exactly the same as they were when we considered TFL. Here they are:

$\mathscr{A} \wedge \mathscr{B}$ is true in a given interpretation **iff** both $\mathscr{A}$ and $\mathscr{B}$ are true in that interpretation

$\mathscr{A} \vee \mathscr{B}$ is true in a given interpretation **iff** either $\mathscr{A}$ or $\mathscr{B}$ is true in that interpretation

$\neg\mathscr{A}$ is true in a given interpretation **iff** $\mathscr{A}$ false in that interpretation

$\mathscr{A} \rightarrow \mathscr{B}$ is true in a given interpretation **iff** either $\mathscr{A}$ is false or $\mathscr{B}$ is true in that interpretation

$\mathscr{A} \leftrightarrow \mathscr{B}$ is true in a given interpretation **iff** $\mathscr{A}$ has the same truth value as $\mathscr{B}$ in that interpretation

This is equivalent to the information conveyed by the characteristic truth tables for the connectives; it's just presented in words here rather than truth tables. Some examples will help to illustrate the idea (make sure you understand them!). On our go-to interpretation:

- '$Rab \wedge Ea$' is true because both conjuncts are true
- '$Rab \wedge Eb$' is false because, although 'Rab' is true, 'Eb' is false
- '$a = b \vee Ea$' is true because although '$a = b$' is false, 'Ea' is true
- '$a \neq b \rightarrow Rba$' is false because the antecedent is true and the consequent is false

7.4 Truth for Quantified Sentences

The big innovation in FOL is the use of *quantifiers*. And specifying the truth conditions for quantified sentences turns out to be a little tricky. If we only look at simple sentences, things are pretty straightforward. We can say that '$\exists xEx$' is true iff 'E' is true of at least one object in the domain, and '$\forall xEx$' is true iff 'E' is true of every object in the domain. So on our go-to interpretation, '$\exists xEx$' comes out true, and '$\forall xEx$' comes out false.

But what about a more complex sentence like '$\forall x(Ex \rightarrow Rxb)$'? This has a universal quantifier as its main operator, so we might again try to say that it is true on an interpretation iff '$(Ex \rightarrow Rxb)$' is true of every object in the domain. The trouble is that '$(Ex \rightarrow Rxb)$' is not a predicate, and our interpretation does not directly specify what objects this complex formula is true of. So while our simple-minded approach worked for '$\forall xEx$', it breaks down when we consider more complex sentences. What we need is some *uniform* way of specifying the truth conditions of *any* universally (or existentially) quantified sentence, irrespective of how complex it is.

The way we will do this is by temporarily treating variables like 'x' as if they referred to objects in the domain. For example, we will say that the universal sentence '$\forall x(Ex \rightarrow Rxb)$' is true iff the formula '$(Ex \rightarrow Rxb)$' that the quantifier attaches to is true *no matter what* *object in the domain the variable 'x' is treated as referring to.* Of course, a variable like 'x' doesn't actually refer to any particular thing, since it's not a name, and '$(Ex \rightarrow Rxb)$' doesn't actually have a truth value. But if we temporarily treated 'x' as referring to 1, for example, then the conditional '$(Ex \rightarrow Rxb)$' would be true, since its antecedent 'Ex' would be false (given that 1 is not in the extension of 'E' in our go-to interpretation).

We will use the notion of a VARIABLE ASSIGNMENT to implement this idea of temporarily treating variables as if they refer to objects. We will, for example, write $[x : 1]$ for an assignment that treats 'x' as referring to 1, and $[x : 3]$ for an assignment that treats 'x' as referring to 3. So although the variable 'x' is not a name for any object in the domain, relative to an assignment like $[x : 3]$, the variable works just like a name that refers to the number 3. Such assignments can also cover multiple variables at once. For example, $[x : 1, y : 5, z : 3]$ would be an assignment relative to which 'x' refers 1, 'y' refers to 5, and 'z' refers to 3.

Returning to '$\forall x(Ex \rightarrow Rxb)$', our idea, then, is that to determine whether this is true, we go through each object in our domain, and ask whether '$(Ex \rightarrow Rxb)$' would come out true if 'x' were treated as referring to that object. All of the following would in other words have to hold:[2]

'$(Ex \rightarrow Rxb)$' is true on $[x : 1]$

'$(Ex \rightarrow Rxb)$' is true on $[x : 2]$

'$(Ex \rightarrow Rxb)$' is true on $[x : 3]$

'$(Ex \rightarrow Rxb)$' is true on $[x : 4]$

$\vdots$

and so on for every positive integer we could assign to 'x'. Now '$(Ex \rightarrow Rxb)$' is true on $[x : 1]$ as well as $[x : 3]$, since in both cases the antecedent 'Ex' comes out false (because neither 1 nor 3 are even). And '$(Ex \rightarrow Rxb)$' is also true on $[x : 2]$, since both 'Ex' and 'Rxb' are true on $[x : 2]$ (because 2 is even, and also less than 3, which is what 'b' refers to on our go-to interpretation). An object that makes a formula like '$(Ex \rightarrow Rxb)$' true is said to SATISFY that formula. So the numbers 1, 2, and 3 all satisfy '$(Ex \rightarrow Rxb)$'.

But the number 4 does *not* satisfy it: '$(Ex \rightarrow Rxb)$' is *false* on the assignment $[x : 4]$. That's because 'Ex' is true on $[x : 4]$ (since 4 is even) but 'Rxb' is false on $[x : 4]$ (since 4 is *not* less than 3). So since '$(Ex \rightarrow Rxb)$' is false for some object in the domain (namely 4), we can conclude that the universal sentence '$\forall x(Ex \rightarrow Rxb)$' we started with is itself false. And that's of course the result we want: relative to our go-to interpretation, '$\forall x(Ex \rightarrow Rxb)$' says that every even number is less than three, which is clearly false, since there are lots of even numbers, including 4, that are not less than 3.

In general, then, we explain the truth conditions for quantified sentences as follows:

[2]Having brought assignments onto the scene, we should now technically go back and revise our explanation of the truth conditions for atomic sentences too. That's because we now need to determine the truth value of things like '$(Ex \rightarrow Rxb)$' relative to an assignment of an object to 'x'. Since the latter contains a free variable, it is not a *sentence*, but merely a *formula* of FOL. Our earlier explanations only concerned sentences, however. So we have to go back and extend them to cover formulas as well. See the Appendix for details.

$\forall \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true on [] in a given interpretation **iff** for *every object o* in the domain, $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true on [$\mathcal{v}$:o] in that interpretation.

$\exists \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true on [] in a given interpretation **iff** for *at least one object o* in the domain, $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true on [$\mathcal{v}$:o] in that interpretation.

Here [] is just an arbitrary assignment, and [$\mathcal{v}$:o] is an assignment that treats the variable $\mathcal{v}$ as referring to o, and treats all other variables just like [] does (whatever that might be). The idea, in other words, is that we go through each object *o* in the domain, and check whether $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true when $\mathcal{v}$ is treated as referring to *o*; if so, the universal sentence $\forall \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ is true, if not, it is false. Notice that it's much easier for an existential sentence to be true: for an existential $\exists \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$, we only have to find *one* object in the domain that satisfies $\mathcal{A}(\ldots\mathcal{v}\ldots)$.

This in turn means that that it's very easy for a universal sentence $\forall \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ to be *false*: we just have to find a single object relative to which $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false. For an existential sentence $\exists \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ to be false, on the other hand, we have to make sure that $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false for *every* object in the domain. Let's state these "falsity conditions" for quantified sentences explicitly too:

$\forall \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false on [] in a given interpretation **iff** for *at least one object o* in the domain, $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false on [$\mathcal{v}$:o] in that interpretation.

$\exists \mathcal{v}\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false on [] in a given interpretation **iff** for *every object o* in the domain, $\mathcal{A}(\ldots\mathcal{v}\ldots)$ is false on [$\mathcal{v}$:o] in that interpretation.

It will help to go through a few examples to get a better feel for how to determine the truth values of quantified sentences. Before we do that, however, we will introduce a useful notational shorthand that makes keeping track of variable assignments a bit easier. Instead of always explicitly mentioning the variable assignment we're considering, we will instead use superscripts on the variables themselves to indicate what objects we are temporarily treating them as referring to. So instead of saying something like:

$$\text{`}(Ex \to Rxb)\text{' is true on } [x:2]$$

as we did above, we can just add a superscript to the variable 'x' itself and say that:

$$(Ex^2 \to Rx^2b) \text{ is true}$$

Similarly, instead of saying, for example, that 'Rxy' is true on the assignment $[x:1, y:3]$, we can just say that Rx^1y^3 is true.

It's important to remember, thought, that this really is just a notational shorthand. Things like '$(Ex^2 \to Rx^2b)$' and 'Rx^1y^3' are not sentences of FOL, since the language of FOL doesn't include superscripts on variables. Talk of Rx^1y^3 is just a shorthand for talking about the FOL sentence 'Rxy' relative to an assignment of 1 to 'x' and 3 to 'y'. With this notation in hand, let's now look at some examples.

7.5 Truth in an Interpretation: Examples

Since the number of objects we have to consider increases the larger the domain is, we'll here use interpretations with relatively small domains. Let's start with the following interpretation with just three objects in its domain:

Interpretation A

Domain:	1, 2, 3
F:	1
G:	2, 3
H:	

	F	G	H
1	$+$	$-$	$-$
2	$-$	$+$	$-$
3	$-$	$+$	$-$

Example 1 $\exists x(Fx \wedge Gx)$

For this to be true, it suffices if a single object in the domain satisfies '$Fx \wedge Gx$'. Unfortunately, no matter which object we treat 'x' as referring to, it comes out false. So our original existential sentence is false. Our explanation in other words goes like this:

▷ $\exists x(Fx \wedge Gx)$ is false because '$(Fx \wedge Gx)$' is false on every assignment to 'x':
  ▷ $(Fx^1 \wedge Gx^1)$ is false (since Gx^1 is false), and
  ▷ $(Fx^2 \wedge Gx^2)$ is false (since Fx^2 is false), and
  ▷ $(Fx^3 \wedge Gx^3)$ is false (since Fx^3 is false)

Again, when in the course of this explanation I say that $(Fx^3 \wedge Gx^3)$ is false, for example, this is just a shorthand for saying that '$(Fx \wedge Gx)$' is false if 'x' is temporarily treated as referring to the number 3.

Example 2 $\exists xFx \wedge \exists xGx$

Notice that the main operator in this is $\wedge$, i.e. it's a conjunction. So to determine its truth value, we have to evaluate its two conjuncts $\exists xFx$ and $\exists xGx$ *separately*. Both of them turn out to be true, and so the conjunction is true:

▷ $\exists xFx \wedge \exists xGx$ is true, because
  ▷ $\exists xFx$ is true
    ▷ since Fx^1 is true
  ▷ and $\exists xGx$ is also true
    ▷ since Gx^2 is true

Notice that 'Gx^3' is of course also true, so I could instead have given this as my reason for why $\exists xGx$ is true. It doesn't matter whether I pick 2 or 3 — as long as I can point to one object that satisfies 'Gx', that's enough for $\exists xGx$ to be true.

What these first two examples illustrate is that it's important to pay attention to the main operator. In '$\exists x(Fx \wedge Gx)$', the main operator is the existential quantifier, so I have to find a single object to make the whole conjunction '$(Fx \wedge Gx)$' true. In '$\exists xFx \wedge \exists xGx$', by contrast, the main operator is $\wedge$, so here I do not have to find a single object to make both 'Fx' and 'Gx' true. Rather I consider '$\exists xFx$' and '$\exists xGx$' *separately*, and check first whether any object satisfies 'Fx', and second whether any object (perhaps a different one) satisfies 'Gx'

Things get more complicated with sentences that contain nested quantifiers, like $\exists x\forall yLxy$. Let's determine the truth value of this on the following interpretation:

Interpretation B

Domain: 1, 2, 3
L: $\langle 1{,}2\rangle$, $\langle 1{,}2\rangle$, $\langle 1{,}3\rangle$

$$\begin{tikzcd} 1 & 2 \arrow[l] \arrow[loop right] \arrow[d] \\ & 3 \end{tikzcd}$$

Example 3 $\exists x\forall yLxy$

If we paraphrase this in English, it says that there is some object x which bears the relation L to *every* object y. And indeed there is such an object in Interpretation B, namely the number 2: this number bears L to 1, and to itself, and also to 3, that is, to every object in the domain.

To put the same point formally: in order for '$\exists x\forall yLxy$' to be true, there has to be some object such that '$\forall yLxy$' is true when 'x' is treated as referring to that object. And there is such an object, namely the number 2. So '$\exists x\forall yLxy$' is true because '$\forall yLx^2y$' is true. And why is '$\forall yLx^2y$' true? That's because no matter which object o in our domain we pick, 'Lx^2y^o' comes out true. That is, all of the following are true: Lx^2y^1, Lx^2y^2, and Lx^2y^3. Our official explanation in other words looks like this:

- ▷ $\exists x\forall yLxy$ is true because:
  - ▷ $\forall yLx^2y$ is true. And this is because:
    - ▷ Lx^2y^1 is true, and
    - ▷ Lx^2y^2 is true, and
    - ▷ Lx^2y^3 is true

Although sentences with nested quantifiers occur most commonly with many-place predicates, nested quantifiers can occur in combination with just one-place predicates too. Let's return to Interpretation A, and consider a case like this for our final example.

Example 4 $\exists x(Fx \rightarrow \forall y(Hy \leftrightarrow Gx))$

The main operator is the existential quantifier, so we have to consider whether there is any object o in the domain such that the conditional $(Fx^o \rightarrow \forall y(Hy \leftrightarrow Gx^o))$ comes out true. In fact, an assignment of 1 to 'x' will do the trick. That's because both 'Fx^1' and '$\forall y(Hy \leftrightarrow Gx^1)$' are true, thus making the conditional true.

Why is '$\forall y(Hy \leftrightarrow Gx^1)$' true? Well, this has a universal quantifier as its main operator, and it is true because no matter what object o in the domain we pick, $Hy^o \leftrightarrow Gx^1$ is true. That is: $Hy^1 \leftrightarrow Gx^1$ is true, and $Hy^2 \leftrightarrow Gx^1$ is true, and $Hy^3 \leftrightarrow Gx^1$ is true. The complete explanation for why Example 4 is true on Interpretation A then looks like this:

- ▷ $\exists x(Fx \rightarrow \forall y(Hy \leftrightarrow Gx))$ is true because:

- ▷ $Fx^1 \rightarrow \forall y(Hy \leftrightarrow Gx^1)$ is true. And this is because:
  - ▷ Fx^1 is true, and
  - ▷ $\forall y(Hy \leftrightarrow Gx^1)$ is also true. This in turn is because:
    - ▷ $(Hy^1 \leftrightarrow Gx^1)$ is true (since Hy^1 and Gx^1 are both false) and
    - ▷ $(Hy^2 \leftrightarrow Gx^1)$ is true (since Hy^2 and Gx^1 are both false) and
    - ▷ $(Hy^3 \leftrightarrow Gx^1)$ is true (since Hy^3 and Gx^1 are both false)

An explanation of this sort is called a SEMANTIC DEMONSTRATION of the truth or falsity of a given sentence (in an interpretation). In the exercises below, you should give semantic demonstrations like this to justify your claims about the truth values of sentences.

■ Exercises 7.5

A. Consider the following interpretation:[3]

Domain:	1, 2
c:	1
A:	1, 2
B:	2
N:	

		A	B	N
c	1	+	−	−
	2	+	+	−

Determine whether each of the following sentences is true or false in this interpretation, and give a semantic demonstration to justify your answer.

1. Bc
2. $Ac \leftrightarrow \neg Nc$
3. $Nc \rightarrow (Ac \vee Bc)$
4. $\forall xAx$
5. $\forall x \neg Bx$
6. $\exists x(Ax \wedge Bx)$
7. $\exists x(Ax \rightarrow Nx)$
8. $\forall x(Nx \vee \neg Nx)$
9. $\exists xBx \rightarrow \forall xAx$

B. Consider the following interpretation:

Domain:	1, 2, 3
c:	2
e:	3
G:	1, 2, 3
H:	2
M:	1, 3

		G	H	M
	1	+	−	+
c	2	+	+	−
e	3	+	−	+

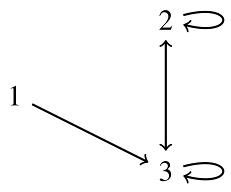
Determine whether each of the following sentences is true or false in this interpretation, and give a semantic demonstration to justify your answer.

[3]See §7.2 on how diagrams like the one to the right are used to represent interpretations. In a matrix diagram like this, a lower-case letter to the left of a domain-object indicates that it is a name of that object.

1. Hc
2. He
3. $Mc \vee Me$
4. $Gc \vee \neg Gc$
5. $Mc \rightarrow Gc$
6. $\exists x Hx$
7. $\forall x Hx$
8. $\exists x \neg Mx$
9. $\exists x(Hx \wedge Gx)$
10. $\exists x(Mx \wedge Gx)$
11. $\forall x(Hx \vee Mx)$
12. $\exists x Hx \wedge \exists x Mx$
13. $\forall x(Hx \leftrightarrow \neg Mx)$
14. $\exists x Gx \wedge \exists x \neg Gx$
15. $\forall x \exists y(Gx \wedge Hy)$

C. Consider the following interpretation:

Domain: 1, 2, 3
L: $\langle 1,2\rangle$, $\langle 2,2\rangle$, $\langle 2,3\rangle$, $\langle 3, 2\rangle$ $\langle 3,3\rangle$



Determine whether each of the following sentences is true or false in this interpretation, and give a semantic demonstration to justify your answer.

1. $\exists x Rxx$
2. $\forall x Rxx$
3. $\forall x \forall y Rxy$
4. $\forall x \exists y Rxy$
5. $\exists x \forall y Rxy$
6. $\exists x \forall y Ryx$
7. $\exists x \forall y \neg Ryx$
8. $\forall y \exists x \neg Ryx$
9. $\forall x \forall y(Rxy \rightarrow Ryx)$
10. $\exists x \forall y(Rxy \rightarrow Ryx)$
11. $\forall x(\exists y Rxy \rightarrow \exists y Ryx)$
12. $\exists x \forall y(Rxy \leftrightarrow x = y)$
13. $\exists x \forall y(Ryx \leftrightarrow x = y)$
14. $\forall x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$
15. $\exists x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$

7.6 Semantic Concepts

Defining truth in FOL was a bit tricky, due to the presence of quantifiers. But now that we know what determines the truth value of an FOL sentence in an interpretation, we can use that to define various other central logical notions. As you can see, these definitions are basically the same as those from TFL, except that we here use the notion of truth in an *interpretation* rather than truth on a *valuation*. In all the definitions below, the metavariables $\mathscr{A}_1 \ldots \mathscr{A}_n$ and $\mathscr{C}$ range over arbitrary *sentences* of FOL:[4]

$\mathscr{A}_1, \ldots, \mathscr{A}_n$ LOGICALLY ENTAIL $\mathscr{C}$, written $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \mathscr{C}$, iff no interpretation makes all of $\mathscr{A}_1, \ldots, \mathscr{A}_n$ true but $\mathscr{C}$ false.

$\mathscr{A}$ is a LOGICAL TRUTH, written $\vDash \mathscr{A}$, iff it is true in every interpretation.

$\mathscr{A}$ is a CONTRADICTION iff it is false in every interpretation.

$\mathscr{A}$ and $\mathscr{B}$ are LOGICALLY EQUIVALENT, written $\mathscr{A} \Dashv\vDash \mathscr{B}$ iff they have the same truth value in every interpretation.

$\mathscr{A}_1, \ldots, \mathscr{A}_n$ are JOINTLY CONSISTENT iff there is at least one interpretation which makes them all true. They are JOINTLY INCONSISTENT iff there is no such interpretation.

As in TFL, perhaps the most important of these concepts is that of logical entailment, since it is closely related to the notion of a valid argument. We will say that an FOL argument:

$$\mathscr{A}_1, \ldots, \mathscr{A}_n \therefore \mathscr{C}$$

is VALID just in case its premises $\mathscr{A}_1, \ldots, \mathscr{A}_n$ entail its conclusion $\mathscr{C}$. Since entailment is defined in terms of the concept of truth-in-an-interpretation, we can use interpretations to investigate the validity of FOL arguments.

In particular, we can use interpretations to show that an argument is *not* valid, or that the corresponding entailment *fails*. To show that a given argument is not valid, it suffices to construct a single interpretation that simultaneously makes all of the premises true but still makes the conclusion false. Such an interpretation is again called a *counterexample* to the validity of the argument, or a COUNTERMODEL.

In the next two sections, we'll look at how to construct countermodels. We'll first look at arguments that involve only one-place predicates, and then at ones that involve many-place predicates. Of course, we can use interpretations to investigate the other logical concepts we've defined above too. We'll return to that once we've looked at our central concept of entailment, or validity.

7.7 Countermodels with One-Place Predicates

Example 1 Let's begin with one of the examples we looked at way back in §1.1.

(1) All rabbits are mammals

[4]Our logical concepts are in other words only defined for FOL sentences. By contrast, the official explanation of truth in FOL, given in the Appendix, covers formulas in general, not just sentences.

Bugs Bunny is a mammal.
∴ Bugs Bunny is a rabbit.

As we noted at the time, this argument is intuitively not valid. We are now in a position to demonstrate this formally. First, we can offer the following FOL symbolization:

$\forall x(Rx \to Mx)$
Mb
$\therefore Rb$

Next, we can construct an interpretation which makes the premises true and the conclusion false. One such interpretation would be the following:

Domain:	all people
b:	Lady Gaga
R:	____ is an opera singer
M:	____ is a vocalist

In this interpretation, '$\forall x(Rx \to Mx)$' is true, since every opera singer is a vocalist, and 'Mb' is also true, since Lady Gaga is a vocalist. But '*Rb*' is false, since she is not an opera singer, but a pop singer. So the argument isn't valid. However, using this kind of countermodel has the drawback that it requires us to appeal to real-world knowledge. After all, who knows, maybe, Lady Gaga secretly moonlights as an opera singer, in which case '*Rb*' is true in this interpretation, and it doesn't constitute a countermodel after all.

To avoid these kinds of issues, and to give some uniformity to our countermodels, we will again use interpretations with domains that contain just positive integers, and give *direct* specifications of the extensions of predicates, by listing the objects they are true of. Furthermore, since we may have to explain why universal sentences like '$\forall x(Rx \to Mx)$' are true on our countermodels, and since the number of objects we have to consider to do this increases the larger the domain is, we will try to construct countermodels with the smallest possible domains.

Let's begin by seeing if we can construct a countermodel that has just one object in its domain, say the number 1. First off, let's make sure our conclusion '*Rb*' is false. To that end, we can let '*b*' refer to 1, and let the extension of the predicate '*R*' remain empty. And to make the second premise, 'Mb' true, we have to put 1 (the referent of '*b*') into the extension of '*M*'. So we have an interpretation like this:

Domain:	1
b:	1
R:	
M:	1

		R	M
b	1	−	+

And luckily this also makes the first premise '$\forall x(Rx \to Mx)$' true! After all, $(Rx^1 \to Mx^1)$ is true (since Rx^1 is false), and since 1 is the only object in our domain, that suffices for '$\forall x(Rx \to Mx)$' to be true.

Example 2 Let's look at a slightly more complex example:

$$\exists x(Fx \to Gx) \therefore \exists xFx \to \exists xGx$$

We will again begin with the smallest possible domain, containing just 1, and think about what's needed to make the conclusion, '$\exists xFx \to \exists xGx$' false. Since this is a conditional, we have to make its antecedent '$\exists xFx$' true and its consequent '$\exists xGx$' false. Using just the number 1, we can do that as follows:

	F	G
1	+	−

The trouble is that this interpretation will also make our premise '$\exists x(Fx \to Gx)$' false. After all, $(Fx^1 \to Gx^1)$ is false, and there's no other object in the domain we could assign to 'x' to make '$(Fx \to Gx)$' true.

So let's expand our domain by adding a second object:

	F	G
1	+	−
2		

We don't yet know whether our two predicates should be true or false of this new object, so I've left the cells next to it blank. Let's begin with our conclusion '$\exists xFx \to \exists xGx$' again: we want it to be false, so '$\exists xFx$' has to be true and '$\exists xGx$' has to be false. Of course, '$\exists xFx$' remains true because Fx^1 is still true, so no change is needed here. But to keep '$\exists xGx$' false, we need to make sure that our new object 2 is not in the extension of 'G' either:

	F	G
1	+	−
2		−

Now, can we make the premise '$\exists x(Fx \to Gx)$' true? In fact we can, by making sure that 'F' is not true of 2. For in that case, $(Fx^2 \to Gx^2)$ will be true, because the antecedent Fx^2 will be false. And that suffices for the truth of '$\exists x(Fx \to Gx)$'. Our final countermodel, and the accompanying semantic demonstrations showing that our premise is true and our conclusion false, then look like this:

Domain: 1, 2
F: 1
G:

	F	G
1	+	−
2	−	−

- $\exists x(Fx \to Gx)$ is true because:
  - ▷ $(Fx^2 \to Gx^2)$ is true (since Fx^2 is false)

- $\exists xFx \to \exists xGx$ is false because:
  - ▷ $\exists xFx$ is true
    - ▷ since Fx^1 is true
  - ▷ but $\exists xGx$ is false. This is because

▷ Gx^1 is false, and

▷ Gx^2 is also false

Whereas we only needed a single object in the domain of our first countermodel, our second countermodel ended up requiring two objects to simultaneously make the premises true and the conclusion false. Other examples might require you to use three objects, or even four, or more. Is there any upper bound on the number of objects that might be needed to produce a countermodel? It turns out that for arguments that only involve one-place predicates, the answer is 'yes'. The logician Leopold Löwenheim showed that if an FOL argument contains n one-place predicates (and no other predicates), then if the argument is invalid, a countermodel with a domain of at most 2^n objects exists. So for something like Example 2, which involves just two predicates, we can be sure that we won't need more than four objects (though we managed to do it with just two). As we'll see, there is no such upper bound for arguments with many-place predicates.

Before we turn to many-place predicates, though, one more general observation is in order. Consider the following English argument:

All foxes are mortal.
∴ Every vixen is mortal.

This argument is valid in the sense that it's impossible for its premise to be true but its conclusion to be false: a vixen is just a female fox, so any possible world where every fox is mortal has to be one where every vixen is mortal. But if we symbolize this, the resulting FOL argument is *not* valid:

$$\forall x(Fx \to Mx)$$
$$\therefore\ \forall x(Vx \to Mx)$$

It's easy to construct an interpretation that makes the premise true and the conclusion false. So from the fact that the symbolization of an English argument in FOL is not valid, we can't invariably conclude that the original English argument is not valid.

What we can conclude is just that the original English argument is not *formally* valid, or more specifically, that it isn't valid in virtue of the kind of logical form captured in its FOL symbolization. But it might still be valid for other reasons — in this case, because of the connection between the meanings of 'fox' and 'vixen'. As we discussed in §1.4, logic doesn't aim to capture the validity of arguments like this; it only cares about formally valid arguments.[5]

7.8 Countermodels with Many-Place Predicates

Let's next look at how to construct countermodels for arguments with many-place predicates. First, though, there's some new terminology that will come in handy. Two-place predicates like 'loves', 'respects', 'admires' etc. express RELATIONS between objects (the loving relation, the respecting relation, and so on). And there are some important characteristics that relations like this can have:

[5]Of course we can make the above argument formally valid by adding a second premise: 'Every vixen is a fox', which we'd symbolize as '$\forall x(Vx \to Fx)$'. But this is now a *different* argument, one with two premises.

A relation R is SERIAL iff $\forall x \exists y Rxy$

A relation R is REFLEXIVE iff $\forall x Rxx$

A relation R is SYMMETRIC iff $\forall x \forall y (Rxy \to Ryx)$

A relation R is TRANSITIVE iff $\forall x \forall y \forall z ((Rxy \land Ryz) \to Rxz)$

One interesting thing to do, then, is to show that a relation's having certain of these characteristics does, or does not, imply its having some other characteristic. For example, being reflexive implies being serial. After all: if every object bears R to itself, then every object bears R to *something*, namely itself! You could do a natural deduction proof to show that $\forall x Rxx \vdash \forall x \exists y Rxy$. However, the implication does not hold in the other direction, i.e. being serial does not imply being reflexive. Showing this will be our first example.

Example 1 $\forall x \exists y Rxy \nvDash \forall x Rxx$

You can perpahs already think of a countermodel that would make '$\forall x \exists y Rxy$' true but '$\forall x Rxx$' false, but again, we're going to try to find the smallest countermodel to do the job. Let's start with a domain that just contains the number 1. For '$\forall x \exists y Rxy$' to be true, every object in the domain has to bear the relation R to at least one object. So since we only have one object, that means 1 has to bear R to itself:

$$\begin{tikzcd} 1 \arrow[loop above] \end{tikzcd}$$

However, this now also makes '$\forall x Rxx$' true, whereas our goal is to make this false.

So let's expand our domain to include two objects, 1 and 2. In order for '$\forall x Rxx$' to be false, at least one of these objects must not bear R to itself, i.e. must not have an arrow looping back to itself. And for '$\forall x \exists y Rxy$' to be true, every object has to bear R to something, i.e. must have at least one outgoing arrow. One easy way to achieve both is to expand our earlier model to look like this:

Domain: 1, 2
R: $\langle 1{,}1 \rangle$, $\langle 1{,}2 \rangle$

$$\begin{tikzcd} 1 \arrow[loop above] & 2 \arrow[l] \end{tikzcd}$$

The accompanying semantic demonstration runs like this:

- $\forall x \exists y Rxy$ is true because:
  - ▷ $\exists y Rx^1 y$ is true
    - ▷ since $Rx^1 y^1$ is true
  - ▷ and $\exists y Rx^2 y$ is also true
    - ▷ since $Rx^2 y^1$
- $\forall x Rxx$ is false because:
  - ▷ $Rx^2 x^2$ is false

There are of course many other countermodels that would do the job just as well. But what we have in any case discovered is that *at least two* objects are necessary to show that the entailment from seriality to reflexivity fails.

Example 2 Next, let's show the following:

$$\forall xLxx, \forall x\forall y(Lxy \to Lyx) \nvDash \exists x\forall yLxy$$

That is: a relation L's being both reflexive and symmetric does not imply that that there is some object x that bears L to everything (there's no official name for this latter characteristic). If we begin with a domain containing just 1, then to make '$\forall xLxx$' true, we would have to do the following:

$$\begin{tikzcd} 1 \arrow[loop above] \end{tikzcd}$$

However, since we want to make '$\exists x\forall yLxy$' *false*, we need to make sure that for every x, there is at least one object it *doesn't* bear L to, i.e. we want the following to hold: $\forall x\exists y\neg Lxy$. And the trouble is that as things stand, every object *does* bear L to something.

So let's try it with two objects. Again, to make '$\forall xLxx$' true, they both need to bear L to themselves:

$$\begin{tikzcd} 1 \arrow[loop above] & 2 \arrow[loop above] \end{tikzcd}$$

Notice that at this point, '$\forall x\exists y\neg Lxy$' holds. For every object we can find at least one thing it doesn't bear L to: 1 doesn't bear L to 2, and 2 doesn't bear L to 1. We still have to make sure that symmetry holds, i.e. that '$\forall x\forall y(Lxy \to Lyx)$' is true. But if you think about it, symmetry does hold in our diagram: everything that 1 bears L to — which is just itself — also bears L back to 1, and similarly, everything that 2 bears L to — which is just itself — also bears L back to 2. So we're done! Our countermodel and accompanying semantic demonstration (which gets pretty involved in the case of symmetry!) looks like this:

Domain: 1, 2
R: $\langle 1,1\rangle$, $\langle 2,2\rangle$

$$\begin{tikzcd} 1 \arrow[loop above] & 2 \arrow[loop above] \end{tikzcd}$$

- $\forall xLxx$ is true because:
  - ▷ Lx^1x^1 is true, and
  - ▷ Lx^2x^2 is also true

- $\forall x\forall y(Lxy \to Lyx)$ is true because:
  - ▷ $\forall y(Lx^1y \to Lyx^1)$ is true, since:
    - ▷ $(Lx^1y^1 \to Ly^1x^1)$ is true (because Lx^1y^1 and Ly^1x^1 are boths true), and
    - ▷ $(Lx^1y^2 \to Ly^2x^1)$ is also true (because Lx^1y^2 is false)
  - ▷ and $\forall y(Lx^2y \to Lyx^2)$ is true, since:
    - ▷ $(Lx^2y^1 \to Ly^1x^2)$ is true (because Lx^2y^1 is false), and
    - ▷ $(Lx^2y^2 \to Ly^2x^2)$ is also true (because Lx^2y^2 and Ly^2x^2 are both true)

- $\exists x\forall yLxy$ is false because:
  - ▷ $\forall yLx^1y$ is false, since:

  - ▷ Lx^1y^2 is false
- ▷ and $\forall yLx^2y$ is also false, since
  - ▷ Lx^2y^1 is false

Again, the countermodel we arrived at is not the only one possible. The following would work too, for example, and might be more intuitive:

$$\begin{tikzcd}
1 \arrow[loop above] \arrow[r, leftrightarrow] & 2 \arrow[loop above] \arrow[r, leftrightarrow] & 3 \arrow[loop above]
\end{tikzcd}$$

But with three objects in the domain, giving a semantic demonstration would require more work. To show that '$\forall x\forall y(Lxy \rightarrow Lyx)$' holds, for example, we'd have to show that '$\forall y(Lxy \rightarrow Lyx)$' is true on an assignment of each of our three objects to 'x'; and then relative to each choice for 'x', we'd have to show that '$(Lxy \rightarrow Lyx)$' is true no matter what we assign to 'y'. So we'd have to consider nine different assignments of objects to 'x' and 'y' in total, whereas the demonstration above only required us to look at four assignments.

7.9 Validity and Decidability

Let's investigate one more argument involving a two-place predicate:

$$\forall x\exists yRxy, \forall x\forall y\forall z((Rxy \wedge Ryz) \rightarrow Rxz) \therefore \exists xRxx$$

For this to be valid, the seriality and transitivity of R would together have to imply that some object bears R to itself (again, there's no official name for this latter characteristic). Let's see if we can construct a countermodel.

To make the conclusion '$\exists xRxx$' false, we have to make sure that *no* object in our interpretation bears R to itself, i.e. there can be no arrows that loop from an object onto itself. And as we already noticed at the hand of Example 1 in the previous section, for seriality to hold in a domain of just one object, that object would have to bear the relation to itself. So we cannot make seriality true and '$\exists xRxx$' false with just one object.

So let's move to a domain with two objects. We can make seriality hold while avoiding any self-looping arrows like this:

$$\begin{tikzcd}
1 \arrow[r, leftrightarrow] & 2
\end{tikzcd}$$

But now consider how to make transitivity, i.e. '$\forall x\forall y\forall z((Rxy \wedge Ryz) \rightarrow Rxz)$', hold. For this to be true, the conditional '$(Rxy \wedge Ryz) \rightarrow Rxz$' has to be true no matter which objects we assign to each of the variables 'x', 'y', and 'z'. So suppose we assign 2 to 'y', and 1 to both 'x' and 'z': $(Rx^1y^2 \wedge Ry^2z^1) \rightarrow Rx^1z^1$. Both Rx^1y^2 and Ry^2z^1 are true, so for the latter conditional to be true, the consequent Rx^1z^1 has to be true. But this would now require a self-looping arrow from 1 back to 1! So we can't get what we want with just two objects.

Let's try three objects. What we've just seen is that we can't have any double-headed arrows, since that would introduce a self-looping arrow by transitivity. With three objects, we can secure seriality (making sure every object has an arrow going to some object) while avoiding both double-headed arrows and self-looping arrows like this:

$$\begin{tikzcd} 1 \arrow[r] & 2 \arrow[r] & 3 \arrow[ll, bend left] \end{tikzcd}$$

But again, transitivity causes a problem. Let's assign 2 to 'x', 3 to 'y', and 1 to 'z', giving us: $(Rx^2y^3 \wedge Ry^3z^1) \rightarrow Rx^2z^1$. Since both Rx^2y^3 and Ry^3z^1 are true, for the conditional to be true, the consequent Rx^2z^1 has to be true. That would mean adding an arrow going from 2 to 1. But since we already have an arrow going from 1 to 2, that would reintroduce a double-headed arrow!

What this shows is that we can't have any arrows that "go backward" in our interpretation, to an earlier number. We could add a fourth object so that 3's arrow can move forward:

$$\begin{tikzcd} 1 \arrow[r] & 2 \arrow[r] & 3 \arrow[r] & 4 \arrow[d] \\ & & & ? \end{tikzcd}$$

But 4 *also* has to have an arrow going to some object (for seriality). And that arrow can't go backward to 1, 2, or 3, and can't go from 4 to itself, so what's to be done? Does this mean it's impossible to construct a countermodel, and that our argument is therefore valid?

No, it does not: the argument is invalid, and a countermodel exists, but it requires *infinitely many objects* in its domain. The following is a countermodel, for example:

Domain: all positive integers
R: ____ is less than ____

We can't specify this interpreation "directly" by listing all the objects in the domain and all the pairs in the extension of 'R', since there are too many of them. But we can specify the extension of 'R' indirectly, via an English predicate. This interpretation does what we want:

- '$\forall x \exists y Rxy$' is true, since for every positive integer x we can find a y that x is less than..
- '$\forall x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$' is true, since if x is less and y and y is less than z, it follows that x must be less than z.
- '$\exists x Rxx$' is false, since there exist no positive integer that is less than itself.

What we've seen, then, is that showing an FOL argument to be invalid may require an interpretation with infinitely many objects. As mentioned in §7.7, this is not the case for FOL arguments that contain only *one-place* predicates. If such an argument is invalid, then there exists a countermodel with a domain of at most 2^n objects (where n is the number of one-place predicates the argument contains). Since there are finitely many interpretations for n predicates using at most 2^n objects, this means that a computer could in principle crunch through all of those interpretations and determine if the argument is valid: if it finds a countermodel among them, the argument is invalid, if it doesn't, the argument is valid.

The same holds for TFL. Any TFL argument can be tested for validity by constructing its joint truth table and checking wether any line (i.e. valuation) makes all the premises true and the conclusion false. So a computer can be programmed to mechanically test any TFL argument for validity.

What we've seen is that this does not hold for FOL arguments that contain two-place predicates. Here there is no finite number of interpretations we could program a computer to check such that, if it fails to find a countermodel among them, the argument is guaranteed to be valid. And in fact, as the logicians Alan Turing and Alonzo Church independently proved in 1936, there exists *no* mechanical test for validity in FOL. Validity in FOL is therefore said to be UNDECIDABLE, in contrast to TFL, where validity is DECIDABLE via truth-tables.

Demonstrating FOL arguments to be valid therefore invariably requires some ingenuity and insight — it's not the kind of thing a computer can do. This is why we use natural deduction proofs to demonstrate validity in FOL: if methods that require ingenuity are going to be required anyhow, we might as well use the method of natural deduction.[6] To be sure, one *can* also give semantic proofs to demonstrate the validity of FOL arguments. For example to show that the following entailment holds:

$$\exists x(Fx \wedge Gx) \vDash \exists xFx \wedge \exists xGx$$

we could give the following semantic proof in English:

> *Semantic Proof:* consider some arbitrary interpretation $\mathscr{I}$, and suppose $\exists x(Fx \wedge Gx)$ is true in $\mathscr{I}$. This means there is some object in the domain of $\mathscr{I}$, let's call it a, such that $(Fx^a \wedge Gx^a)$ is true in $\mathscr{I}$. But then since Fx^a is true in $\mathscr{I}$, this means $\exists xFx$ is true in $\mathscr{I}$. And similarly, since Gx^a is true in $\mathscr{I}$, it follows that $\exists xGx$ must be true in $\mathscr{I}$. Therefore $\exists xFx \wedge \exists xGx$ is true in $\mathscr{I}$. Since $\mathscr{I}$ was an arbitrary interpretation, we can conclude that *every* interpretation that makes $\exists x(Fx \wedge Gx)$ true must also make $\exists xFx \wedge \exists xGx$ true, meaning that the entailment above holds.

However, we can also just give a natural deduction proof, which in fact goes through a very similar line of reasoning:

$$
\begin{array}{r|l|ll}
1 & \exists x(Fx \wedge Gx) & & \\
\hline
2 & & Fa \wedge Ga & \\
\cline{3-3}
3 & & Fa & \wedge\text{E } 2 \\
4 & & \exists xFx & \exists\text{I } 3 \\
5 & & Ga & \wedge\text{E } 2 \\
6 & & \exists xFx & \exists\text{I } 5 \\
7 & & \exists xFx \wedge \exists xGx & \wedge\text{I } 4,\ 6 \\
8 & \exists xFx \wedge \exists xGx & & \exists\text{E } 1,\ 2\text{–}8
\end{array}
$$

Natural deduction proofs do not, however, allow use to demonstrate that arguments are *invalid*. To do this, we have to rely on the method of constructing countermodels. And that's what we have been doing in the last several sections.

[6]Of course, by using natural deductions to prove arguments valid, we are relying on the fact that our proof system is sound, as well as complete. See the beginning of Chapter 6 if you need to review these concepts.

7.10 Working with Other Semantic Concepts

So far, we've just been focusing on validity, or entailment. But we can use interpretations, as well as natural deduction proofs, in connection with other semantic concepts too.

Take the concept of logical truth first. To show that some FOL sentence $\mathscr{A}$ is *not* a logical truth, it suffices to construct an interpretation that makes it false. And to show that it *is* a logical truth, we can give a natural deduction that proves $\mathscr{A}$ as a theorem. This gives us a way to approach contradictions as well: since $\mathscr{A}$ is a contradiction iff $\neg\mathscr{A}$ is a logical truth, we can show that $\mathscr{A}$ is a contradiction by proving $\neg\mathscr{A}$ as a theorem. And to show that $\mathscr{A}$ is *not* a contradiction, it suffices to construct an interpretation in which it is true.

Next consider logical equivalence. To show that $\mathscr{A}$ and $\mathscr{B}$ are *not* logically equivalent, all we need to do is construct an interpretation on which one of them is true and the other is false. And to show that $\mathscr{A}$ and $\mathscr{B}$ *are* equivalent, we can give a natural deduction that proves $\mathscr{A} \leftrightarrow \mathscr{B}$ as a theorem, since $\mathscr{A}$ and $\mathscr{B}$ are equivalent just in case $\mathscr{A} \leftrightarrow \mathscr{B}$ is a logical truth.

Lastly, take the concept of consistency. To show that some sentences are jointly consistent, it suffices to give an interpretation on which they are all true. As for inconsistency, notice that it connects to entailment in the following way:

If $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \bot$, then $\mathscr{A}_1, \ldots, \mathscr{A}_n$ are jointly inconsistent

For suppose $\mathscr{A}_1, \ldots, \mathscr{A}_n \vDash \bot$. This means there's no interpretation that makes all of $\mathscr{A}_1, \ldots, \mathscr{A}_n$ true but $\bot$ false. However, since every interpretation makes $\bot$ false, this just means that there's no interpretation that makes all of $\mathscr{A}_1, \ldots, \mathscr{A}_n$ true. That is, $\mathscr{A}_1, \ldots, \mathscr{A}_n$ are inconsistent. So if we want to show that $\mathscr{A}_1, \ldots, \mathscr{A}_n$ are inconsistent, we can do that by giving a natural deduction proof showing that $\mathscr{A}_1, \ldots \mathscr{A}_n \vdash \bot$.

The following table summarizes what is needed to demonstrate that a given concept does or does not apply:

	Yes	**No**
logical truth?	give a proof	give an interpretation
contradiction?	give a proof	give an interpretation
equivalent?	give a proof	give an interpretationl
consistent?	give an interpretation	give a proof
valid?	give a proof	give an interpretation
entailment?	give a proof	give an interpretation

■ Exercises 7.10

A. Give counermodels to show the following:

1. $\forall x(Ax \to Bx) \nvDash \exists xBx$
2. $\forall x(Rx \to Dx), \forall x(Rx \to Fx) \nvDash \exists x(Dx \wedge Fx)$
3. $\exists x(Px \to Qx) \nvDash \exists xPx$
4. $Na \wedge Nb \wedge Nc \nvDash \forall xNx$
5. $\exists x(Ex \wedge Fx), \exists xFx \to \exists xGx \nvDash \exists x(Ex \wedge Gx)$
6. $Rde, \exists xRxd \nvDash Red$

7. $\forall x Oxc, \forall x Ocx \nvDash \forall x Oxx$
8. $\forall x \exists y Lxy \nvDash \exists x \forall y Lxy$
9. $\exists x(Jx \wedge Kx), \exists x \neg Kx, \exists x \neg Jx \nvDash \exists x(\neg Jx \wedge \neg Kx)$
10. $Lab \to \forall x Lxb, \exists x Lxb \nvDash Lbb$
11. $\forall x(Dx \to \exists y Tyx) \nvDash \exists y \exists z\, y \neq z$
12. $\forall x(Fx \vee \neg Gx) \nvDash (\forall x Fx \vee \neg \exists x Gx)$
13. $\forall x \exists y(Rxy \to Rxx) \nvDash \forall x(\exists y Rxy \to Rxx)$
14. $\forall x Fx \leftrightarrow \forall y Gy \nvDash \forall x(Fx \leftrightarrow AyGy)$
15. $\forall x \exists y Rxy, \exists x \forall y Rxy \nvDash \forall x \forall y Rxy$
16. $\nvDash \exists x \forall y((Rxy \wedge \sim Ryx) \to (Rxx \leftrightarrow Ryy))$

B. Show that each of the following is neither a logical truth nor a contradiction:

1. $Da \wedge Db$
2. $\exists x Txh$
3. $Pm \wedge \neg \forall x Px$
4. $\forall z Jz \leftrightarrow \exists y Jy$
5. $\forall x(Wxmn \vee \exists y Lxy)$
6. $\exists x(Gx \to \forall y My)$
7. $\exists x(x = h \wedge x = i)$
8. $\exists x \forall y((Rxy \wedge \neg Ryx) \to (Rxx \leftrightarrow Ryy))$

C. Show that the following pairs of sentences are not logically equivalent.

1. Ja, Ka
2. $\exists x Jx$, Jm
3. $\forall x Rxx$, $\exists x Rxx$
4. $\exists x Px \to Qc$, $\exists x(Px \to Qc)$
5. $\forall x(Px \to \neg Qx)$, $\exists x(Px \wedge \neg Qx)$
6. $\exists x(Px \wedge Qx)$, $\exists x(Px \to Qx)$
7. $\forall x(Px \to Qx)$, $\forall x(Px \wedge Qx)$
8. $\forall x \exists y Rxy$, $\exists x \forall y Rxy$
9. $\forall x \exists y Rxy$, $\forall x \exists y Ryx$

D. Show that the following sentences are jointly consistent:

1. $Ma, \neg Na, Pa, \neg Qa$
2. $Lee, Leg, \neg Lge, \neg Lgg$
3. $\neg(Ma \wedge \exists x Ax), Ma \vee Fa, \forall x(Fx \to Ax)$
4. $Ma \vee Mb, Ma \to \forall x \neg Mx$
5. $\forall y Gy, \forall x(Gx \to Hx), \exists y \neg Iy$
6. $\exists x(Bx \vee Ax), \forall x \neg Cx, \forall x\big[(Ax \wedge Bx) \to Cx\big]$
7. $\exists x Xx, \exists x Yx, \forall x(Xx \leftrightarrow \neg Yx)$
8. $\forall x(Px \vee Qx), \exists x \neg(Qx \wedge Px)$
9. $\exists z(Nz \wedge Ozz), \forall x \forall y(Oxy \to Oyx)$
10. $\neg \exists x \forall y Rxy, \forall x \exists y Rxy$
11. $\neg Raa$, $\forall x(x = a \vee Rxa)$
12. $\forall x \forall y \forall z(x = y \vee y = z \vee x = z)$, $\exists x \exists y\, \neg x = y$
13. $\exists x \exists y(Zx \wedge Zy \wedge x = y)$, $\neg Zd$, $d = e$

7.11 Appendix: Semantics with Variable Assignments

In §7.4 we explained the semantics for quantifiers using the notion of an assignment of objects to variables. However, once we introduce variable assignments to deal with quantifiers, we should really go back and re-do our semantics with variable assignments from the beginning, even when we're just looking at atomic sentences.

To see why, consider a simple example like '$\forall xFx$', and suppose our domain contains just the numbers 1 and 2. Then in order for '$\forall xFx$' to be true, the formula 'Fx' needs to be true on the assignment $[x:1]$ and also on $[x:2]$. But what is required for 'Fx' to be true on $[x:1]$, say? The answer, of course, is that 'F' has to be true of 1, the object we're treating 'x' as referring to. But the semantics we gave for atomic sentences doesn't technically tell us this! All we said in §7.3 was the following:

> When $\mathscr{R}$ is an n-place predicate and $\mathscr{a}_1, \mathscr{a}_2, \ldots, \mathscr{a}_n$ are names, the sentence $\mathscr{R}\mathscr{a}_1\mathscr{a}_2\ldots\mathscr{a}_n$ is true in a given interpretation **iff** $\mathscr{R}$ is true of the objects referred to by $\mathscr{a}_1, \mathscr{a}_2, \ldots, \mathscr{a}_n$ (in that order) in that interpretation

This doesn't mention variable assignments anywhere. And furthermore, it only deals with *sentences*, that is, expressions built by combining predicates with *names*. 'Fx' isn't built like this, because it contains a variable instead of a name. So the above explanation simply does not apply to it.

To remedy this situation, we need to return to the official explanation of the syntax, or grammar, of FOL that we gave way back in §5.9. Recall that we there defined a general class of *formulas* of FOL, among which we then singled out the smaller class of FOL *sentences*. We did this in four steps. First, we stipulated that a TERM is to be any name *or any variable*. Second, we said that an ATOMIC FORMULA is anything that results from combining any predicate (including the identity symbol) with an appropriate number of *terms*. Third, we said that a FORMULA, more generally, is anything that can be built up from atomic formulas using truth functional operators and quantifiers. And lastly, we then said that a SENTENCE is any formula that contains no free, or unbound, variables.

We here have to do something similar. We will first have to give a general explanation of truth that covers all *formulas*, such as 'Fx', and then extract from that a notion of truth for just *sentences*. In particular, what we'll do is to explain what is required for any formula whatsoever to be true in an interpretation *relative to a variable assignment*. In what follows, let's use $\mathscr{I}$ for an arbitrary interpretation, and $[]$ for an arbitrary variable assignment. So we will give a general explanation of what is required for any formula to be *true on* $[]$ *in* $\mathscr{I}$.

To begin, we first give an expanded notion of reference that covers both names (like $a, b, c\ldots$) and variables (like $x, y, z\ldots$). Of course, a variable doesn't actually have a referent, since it isn't a name. But *relative to a variable assignment*, variables can be construed as referring to things. So:

> For any term t, the REFERENT OF t ON $[]$ IN $\mathscr{I}$ is:
>
> ▷ whatever object the interpretation $\mathscr{I}$ assigns to t, if t is a name, and
>
> ▷ whatever object the assignment $[]$ assigns to t, if t is a variable

We can now use this expanded notion of reference to explain truth for all atomic formulas, whether they contain names or variables:

> When $\mathscr{R}$ is an n-place predicate and $t_1, \ldots, t_n$ are terms, the formula $\mathscr{R}t_1 \ldots t_n$ is true on [] in $\mathscr{I}$ **iff** $\mathscr{R}$ is true of the objects referred to by $t_1, \ldots, t_n$ (in that order) on [] in $\mathscr{I}$.
>
> And for any terms t_1 and t_2, the formula $t_1 = t_2$ is true on [] in $\mathscr{I}$ iff t_1 and t_2 have the same referent on [] in $\mathscr{I}$.

This now *does* specify what's required for e.g. 'Fx' to be true on $[x : 1]$. What's required is that 'F' be true of the referent of 'x' on $[x : 1]$. And the referent of 'x' on this assignment is of course just 1, give our expanded notion of reference. So 'F' has to be true of 1. With atomic formulas covered, we can go on to explain truth for all complex formulas:

> $\neg\mathscr{A}$ is true on [] in $\mathscr{I}$ **iff** $\mathscr{A}$ false on [] in $\mathscr{I}$
>
> $\mathscr{A} \wedge \mathscr{B}$ is true on [] in $\mathscr{I}$ **iff** both $\mathscr{A}$ and $\mathscr{B}$ are true on [] in $\mathscr{I}$
>
> $\mathscr{A} \vee \mathscr{B}$ is true on [] in $\mathscr{I}$ **iff** either $\mathscr{A}$ or $\mathscr{B}$ are true on [] in $\mathscr{I}$
>
> $\mathscr{A} \to \mathscr{B}$ is true on [] in $\mathscr{I}$ **iff** either $\mathscr{A}$ is false or $\mathscr{B}$ is true on [] in $\mathscr{I}$
>
> $\mathscr{A} \leftrightarrow \mathscr{B}$ is true on [] in $\mathscr{I}$ **iff** $\mathscr{A}$ and $\mathscr{B}$ have the same truth value on [] in $\mathscr{I}$
>
> $\forall \mathscr{v} \mathscr{A}(\ldots \mathscr{v} \ldots)$ is true on [] in $\mathscr{I}$ **iff** for *every object o* in the domain, $\mathscr{A}(\ldots \mathscr{v} \ldots)$ is true on [$\mathscr{v}$:o] in $\mathscr{I}$
>
> $\exists \mathscr{v} \mathscr{A}(\ldots \mathscr{v} \ldots)$ is true on [] in $\mathscr{I}$ **iff** for *at least one object o* in the domain, $\mathscr{A}(\ldots \mathscr{v} \ldots)$ is true on [$\mathscr{v}$:o] in $\mathscr{I}$.

Again, in the quantifier clauses [$\mathscr{v}$:o] is the assignment that's just like our arbitrary assignment [] in every respect, except that it is stipulated to assign the object o to the variable $\mathscr{v}$. The idea, again, being that we go through each object o in the domain, and check whether the formula $\mathscr{A}(\ldots \mathscr{v} \ldots)$ is true when o is assigned to the free variable $\mathscr{v}$. If so, $\forall \mathscr{v} \mathscr{A}(\ldots \mathscr{v} \ldots)$ is true, if not, it is false.

Given all of this, we can now say what is required for any *sentence*, i.e. any formula with no free variables, to be true in an interpretation $\mathscr{I}$:

> For any FOL sentence $\mathscr{A}$ and any interpretation $\mathscr{I}$, $\mathscr{A}$ is true in $\mathscr{I}$ iff $\mathscr{A}$ is true in $\mathscr{I}$ on any assignment whatsoever.

With all these pieces in place, the definition of the various logical concepts (like entailment) now proceeds as it did in §7.6.

Quick Reference 8

Truth Functional Operators

$\mathscr{A}$	$\mathscr{B}$	$\neg\mathscr{A}$	$\mathscr{A}\wedge\mathscr{B}$	$\mathscr{A}\vee\mathscr{B}$	$\mathscr{A}\rightarrow\mathscr{B}$	$\mathscr{A}\leftrightarrow\mathscr{B}$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Deduction Rules for TFL

Conjunction Introduction

m	$\mathscr{A}$	
n	$\mathscr{B}$	
	$\mathscr{A}\wedge\mathscr{B}$	$\wedge$I m, n

Conjunction Elimination

m	$\mathscr{A}\wedge\mathscr{B}$	
	$\mathscr{A}$	$\wedge$E m

m	$\mathscr{A}\wedge\mathscr{B}$	
	$\mathscr{B}$	$\wedge$E m

Conditional Introduction

i	$\quad\vert\ \mathscr{A}$	
j	$\quad\vert\ \mathscr{B}$	
	$\mathscr{A}\rightarrow\mathscr{B}$	$\rightarrow$I i–j

Conditional Elimination

m	$\mathscr{A}\rightarrow\mathscr{B}$	
n	$\mathscr{A}$	
	$\mathscr{B}$	$\rightarrow$E m, n

Biconditional Introduction

i	$\quad\vert\ \mathscr{A}$	
j	$\quad\vert\ \mathscr{B}$	
k	$\quad\vert\ \mathscr{B}$	
l	$\quad\vert\ \mathscr{A}$	
	$\mathscr{A}\leftrightarrow\mathscr{B}$	$\leftrightarrow$I i–j, k–l

Biconditional Elimination

m	$\mathscr{A}\leftrightarrow\mathscr{B}$	
n	$\mathscr{A}$	
	$\mathscr{B}$	$\leftrightarrow$E m, n

$$\begin{array}{r|ll} m & \mathscr{A} \leftrightarrow \mathscr{B} & \\ n & \mathscr{B} & \\ & \mathscr{A} & \leftrightarrow\text{E } m, n \end{array}$$

Disjunction Introduction

$$\begin{array}{r|ll} m & \mathscr{A} & \\ & \mathscr{A} \lor \mathscr{B} & \lor\text{I } m \end{array}$$

$$\begin{array}{r|ll} m & \mathscr{A} & \\ & \mathscr{B} \lor \mathscr{A} & \lor\text{I } m \end{array}$$

Disjunction Elimination

$$\begin{array}{r|ll} m & \mathscr{A} \lor \mathscr{B} & \\ i & \quad \mid \underline{\mathscr{A}} & \\ j & \quad \mid \mathscr{C} & \\ k & \quad \mid \underline{\mathscr{B}} & \\ l & \quad \mid \mathscr{C} & \\ & \mathscr{C} & \lor\text{E } m, i\text{–}j, k\text{–}l \end{array}$$

Negation Introduction

$$\begin{array}{r|ll} m & \quad \mid \underline{\mathscr{A}} & \\ n & \quad \mid \bot & \\ & \neg\mathscr{A} & \neg\text{I } m\text{–}n \end{array}$$

Negation Elimination

$$\begin{array}{r|ll} m & \mathscr{A} & \\ n & \neg\mathscr{A} & \\ & \bot & \neg\text{E } m, n \end{array}$$

Indirect Proof

$$\begin{array}{r|ll} m & \quad \mid \underline{\neg\mathscr{A}} & \\ n & \quad \mid \bot & \\ & \mathscr{A} & \text{IP } m\text{–}n \end{array}$$

Derived Rules for TFL

Sequent:	**Derived Rule:**
$\mathscr{A} \lor \mathscr{B}, \neg\mathscr{B} \vdash \mathscr{A}$	DS
$\mathscr{A} \lor \mathscr{B}, \neg\mathscr{A} \vdash \mathscr{B}$	DS
$\mathscr{A} \to \mathscr{B}, \neg\mathscr{B} \vdash \neg\mathscr{A}$	MT
$\mathscr{A} \vdash \mathscr{B} \to \mathscr{A}$	PMI
$\neg\mathscr{A} \vdash \mathscr{A} \to \mathscr{B}$	PMI
$\mathscr{A} \to \mathscr{B} \dashv\vdash \neg\mathscr{A} \lor \mathscr{B}$	Imp
$\neg(\mathscr{A} \to \mathscr{B}) \dashv\vdash \mathscr{A} \land \neg\mathscr{B}$	NegImp
$\neg(\mathscr{A} \land \mathscr{B}) \dashv\vdash \neg\mathscr{A} \lor \neg\mathscr{B}$	DeM
$\neg(\mathscr{A} \lor \mathscr{B}) \dashv\vdash \neg\mathscr{A} \land \neg\mathscr{B}$	DeM
$\mathscr{A} \dashv\vdash \neg\neg\mathscr{A}$	DN
$\mathscr{A} @ \mathscr{B} \vdash \mathscr{B} @ \mathscr{A}$	Com
$\bot \vdash \mathscr{A}$	EX
$\vdash \mathscr{A} \lor \neg\mathscr{A}$	LEM

Deduction Rules for FOL

Universal Elimination

$$\begin{array}{r|ll}
m & \forall \mathscr{v}\,\mathscr{A}(\ldots \mathscr{v} \ldots) & \\
 & \mathscr{A}(\ldots \mathscr{c} \ldots) & \forall\text{E } m
\end{array}$$

Existential Introduction

$$\begin{array}{r|ll}
m & \mathscr{A}(\ldots \mathscr{c} \ldots) & \\
 & \exists \mathscr{x}\,\mathscr{A}(\ldots \mathscr{x} \ldots) & \exists\text{I } m
\end{array}$$

Universal Introduction

$$\begin{array}{r|l|ll}
m & & \mathscr{c} & \text{Flag} \\ \cline{3-3}
 & & \vdots & \\
n & & \mathscr{A}(\ldots \mathscr{c} \ldots) & \\
 & \multicolumn{2}{l}{\forall \mathscr{v}\,\mathscr{A}(\ldots \mathscr{v} \ldots)} & \forall\text{I } m\text{–}n
\end{array}$$

The Flag-ed name $\mathscr{c}$ may not occur outside the subproof.

Existential Elimination

$$\begin{array}{r|l|ll}
m & \multicolumn{2}{l}{\exists \mathscr{x}\,\mathscr{A}(\ldots \mathscr{x} \ldots)} & \\
i & & \mathscr{A}(\ldots \mathscr{c} \ldots) & \text{Flag } \mathscr{c} \\ \cline{3-3}
j & & \mathscr{B} & \\
 & \multicolumn{2}{l}{\mathscr{B}} & \exists\text{E } m,\ i\text{–}j
\end{array}$$

The Flag-ed name $\mathscr{c}$ may not occur outside the subproof.

Identity Elimination

$$\begin{array}{r|ll}
m & \mathscr{a} = \mathscr{b} & \\
n & \mathscr{A}(\ldots \mathscr{a} \ldots \mathscr{a} \ldots) & \\
 & \mathscr{A}(\ldots \mathscr{b} \ldots \mathscr{a} \ldots) & =\text{E } m,\ n
\end{array}$$

$$\begin{array}{r|ll}
m & \mathscr{a} = \mathscr{b} & \\
n & \mathscr{A}(\ldots \mathscr{b} \ldots \mathscr{b} \ldots) & \\
 & \mathscr{A}(\ldots \mathscr{a} \ldots \mathscr{b} \ldots) & =\text{E } m,\ n
\end{array}$$

Identity Introduction

$$\begin{array}{|ll}
\mathscr{c} = \mathscr{c} & =\text{I}
\end{array}$$