forallχ

An Introduction to Formal Logic

For use in PHILOSOPHY 101

Initially written by
P. D. Magnus, **Tim Button**

This version rewritten and revised for the University of Auckland by
Patrick Girard, **Andrew Withy**, **Jeremy Seligman**

with additions by
J. Robert Loftis, **Robert Trueman**, **Aaron Thomas-Bolduc**, **Richard Zach**, **Jonathan Jenkins Ichikawa**, **Isabella McAllister**

forallχ contents

I Key Notions of Logic **4**

1 What is Logic? **5**
1.1 Statements . . . 6
1.2 Statements and Possibilities . . . 7
1.3 What is Truth? . . . 8
Practice Exercises . . . 9

2 The scope of logic **10**
2.1 Arguments . . . 10
2.2 Introducing validity . . . 11
2.3 Cases and types of Validity . . . 12
2.4 Cases and types of possibility . . . 14
2.5 Validity and Truth . . . 16
Practice Exercises . . . 17

3 Other logical notions **18**
3.1 Consistency . . . 18
3.2 Logical Terms . . . 19
Practice Exercises . . . 21

II Truth-Functional Logic **23**

4 Symbolising **24**
4.1 Formal Validity . . . 24
4.2 English v Formal Validity . . . 27
4.3 Atomic Statements . . . 28
Practice Exercises . . . 29

5 Connectives **30**
5.1 Negation 31
5.2 Conjunction 33
5.3 Disjunction 35
5.4 Conditionals 36
Practice Exercises 37

6 Connective Complications **38**
6.1 Interpreting Language 38
6.2 Exclusive Disjunction 39
6.3 Conditional Complications 40
6.4 Only Conditionals 41
6.5 Biconditional 42
6.6 Unless 44
Practice Exercises 45

III Truth Tables **49**

7 Introducing Truth Tables **50**
7.1 Characteristic Truth Tables 50
7.2 Truth-functional Connectives 52
7.3 Symbolising vs. Translating 53
Practice Exercises 54

8 Complete Truth Tables **55**
8.1 Our First Complete Truth Table 55
8.2 Building Complete Truth Tables 57
8.3 Dropping Brackets 58
Practice Exercises 60

9 Semantic Notions **61**
9.1 Logical Truths and Falsehoods 61
9.2 Consistency 63
9.3 Validity 64
9.4 Equivalent and Contradictory 65
9.5 Relationships between Logical Notions 66
9.6 Some Challenges with TFL 67
Practice Exercises 69

10 Partial Truth Tables **72**
10.1 Going Backwards to Move Forwards 72
10.2 Testing our Logical Notions 73
Practice Exercises 81

IV TFL Truth Trees **84**

11 Truth Trees **85**
11.1 Truth Tables aren't Enough 85
11.2 Mutual Consistency 85
Practice Exercises 90

12 Planting your Tree **91**
12.1 Selecting the Right Root 91
12.2 Growing your Tree 92
12.3 Completing your Tree 94
Practice Exercises 95

13 TFL Tree Rules **96**
13.1 Decomposition Rules 96
13.2 Truth Tree Complexities 100
13.3 Order of Decomposition 102
Practice Exercises 104

V Predicate Logic **108**

14 The Language of PL **109**
14.1 Decomposing Statements 109
14.2 Names 112
14.3 Predicates 113
Practice Exercises 115

15 Quantifiers **117**
15.1 Introducing Quantifiers 118
15.2 Quantifiers and Scope 120
15.3 The Power of Paraphrase 122
Practice Exercises 124

16 Single Quantifier Statements **126**
16.1 Common Quantifier Phrases 126
16.2 Ambiguous Predicates 129
16.3 Aristotelian Syllogisms 132
Practice Exercises 134

17 Multiple generality **137**
17.1 Many-place Predicates 137
17.2 Quantifier Order 139
17.3 Stepping-stones to Symbolisation 140
17.4 De-Symbolising 142
17.5 Suppressed Quantifiers 144
Practice Exercises 146

18 Identity **150**
18.1 Adding Identity 151
18.2 Using Identity 153
18.3 Counting on Identity 156
Practice Exercises 159

VI Models **162**

19 Predicate Models **163**
19.1 Models 163
19.2 The Domain 163
19.3 Predicate tables 165
19.4 Many-place Predicates 168
Practice Exercises 171

20 Models with Identity **174**
20.1 Identity tables 174
20.2 Identity & Predicate tables 176
20.3 Identity & two-place predicate tables 180
Practice Exercises 183

21 Models with Quantifiers **186**
21.1 Quantified predicates 186
21.2 Single Quantifiers 188
21.3 Multiple Quantifiers 191
Practice Exercises 197

VII PL Truth Trees **200**

22 Reintroducing Truth Trees **201**
22.1 Unquantified Trees 201
22.2 Quantified Trees 203
22.3 Completing your Tree 206
22.4 More PL Tree Examples 207

23 Tree Strategies **212**
23.1 Order of Decomposition 212
23.2 Truth Tree Complexities 214
23.3 Common Student Errors 218
Practice Exercises 220

24 Truth Trees With Identity **221**
Practice Exercises 222
24.1 Logical Truths of Identity 222
24.2 Arguments with Identity 224
Practice Exercises 228

VIII Appendix **229**

A Solutions to selected exercises **230**

Part I

Key Notions of Logic

Chapter 1

What is Logic?

Logic studies which claims can go together without causing problems, and which claims have to be true whenever other ones are. It also studies the methods we use for checking for problems and truth, and their limitations.

That's vague. Let's try again:

- Formal Logic, which is the topic of this textbook, uses formal, symbolic methods for manipulating representations of claims to demonstrate relationships between them. The relationships that we are interested are solely those related to the truth or falsity of the claims, and not what might be known, believed, implied, inferred, or contextually relevant.
- Logic does not study how humans use language to communicate claims, or the cultural or social significance of exactly how a claim is phrased. It is not dependent on which language is used, or the social or moral standing of the logician, or the knowledge, power, or skill of the logician. Logic is person-independent. Or so we would like to think.
- The logics we will introduce in this book are formal, algebraic, and central to the Western philosophical tradition of the ancient Greeks, Indians, and Romans, the Medieval Islamic and Christian scholars, and the modern mathematicians, computer scientists, and linguists. They are the two core, primary logics that underpin formal reasoning today.
- Logic is not a study of how we think or reason, or even a very good story about how we should reason. It is not psychology, although it is used in psychology.

One of the key attributes of Logic is that it is precise, and that everything we talk about can be defined. So we will attempt to describe our terms more carefully, in preparation for providing full and precise definitions for our logical notions.

1.1 Statements

Logic is a study of relationships between claims. These claims are something like thoughts, and something like sentences. But all we are interested in is the truth or falsity of the claims, not the rest of their properties. Thoughts are hard to examine, so we will just consider sentences. The type of sentence that is used to directly make truth claims is a STATEMENT.

Philosophy Corner

If we were being careful, we would talk solely of *propositions*. A proposition is whatever is in common between different statements that 'mean the same thing', whether written or spoken, whether in English, Chinese or Serbian, and even if only thought, or expressed in language. This is because we can apply logic to our thoughts, as well as to sentences, and logic is constant regardless of which language (if any) we use. However, we will continue to talk of statements.

You should not confuse the idea of a statement with the difference between fact and opinion. Often, statements in logic will express things that would count as facts – such as 'Kierkegaard was a hunchback' or 'Kierkegaard liked almonds.' They can also express things that you might think of as matters of opinion – such as, 'Almonds are tasty.' In other words, a statement is not disqualified because we don't know if it is true or false, or because its truth or falsity is a matter of opinion. We only need it to be either true or false, and not anything else.

There are many types of sentences that are not statements, and we will usually ignore them, or re-write them as statements. It's worth considering some of the major classes of non-statements, just so we don't get tripped up.

Questions 'Are you sleepy yet?' is a question. Although your answer might be a statement, the question itself is neither true nor false. e.g., 'What is Logic?' is not a statement, but 'Logic is mysterious' is a statement.

Commands Sentences like 'Wake up!', 'Sit up straight', and so on are commands. Although it might be good for you to wake up, the command is neither true nor false.

Exclamations 'Ouch!' is an exclamation; it is neither true nor false. We will treat 'Ouch, I hurt my toe!' as meaning the same thing as 'I hurt my toe.' The 'ouch' does not explicitly add any information to the claim.

1.2 Statements and Possibilities

Most statements can be true if the world is one way, and false if the world is another way. This seems obvious, and even uninteresting. But if we reflect on the possible relationships between statements, we might end up asking ourselves questions like these:

1. Are there any statements that are always true, or always false?
2. Are there collections of statements which can't all be true together?
3. Are there some statements that have to be true if other ones are?
4. Are there pairs of opposite statements, where one is true exactly when the other is false?
5. Are there pairs of similar statements which are always true or false together?

Exploring techniques for investigating these five questions, particularly #2 and #3, will occupy us for the rest of this book.

You might be wondering what we mean when we say that a statement is not always true. We don't mean that it's true now, and false tomorrow, or true for me and false for you. Instead, we are talking about POSSIBILITIES – ways reality might be, or might have been. We will use logic to represent possibilities by giving truth values to various symbols by means of VALUATIONS. And then we can systematically check these valuations. We can re-frame the above questions in terms of valuations:

1. Is there a valuation where this statement is true and one where it is false?
2. Is there a valuation for which this set of statements are all true?
3. Is there no valuation where the first statement is true and the second statement false?
4. Is there no valuation where these two statements have the same truth value?
5. Is there no valuation where these two statements have different truth values?

Now we need an algorithm or set of rules to systematically search through each possible valuation, and determine which symbols are true for that valuation. And we'll also need a way to turn statements into symbols, so we can check their truth value for each valuation. And we should have a clear idea of what we mean by truth, and possibility.

Lets start at the end – with examining truth, and possibility.

1.3 What is Truth?

The only property of statements that logic needs is their truth value. So, what values of truth are there? To avoid a lot of controversial philosophy, we will make some assumptions:

First, that in every possibility, every statement is either true or not true. So any scenario that leaves the truth-value of a statement in our argument undetermined will not be considered. For instance, a scenario where it is neither true nor false that I like chocolate is not a possibility we would consider. When dealing with predictions, we might say instead that every statement will eventually either be true or not true.

Second, that if a statement isn't true in a possibility, it is FALSE; that is, every statement is either true or false in a possibility.

Finally, that it is impossible that a statement is both true and false.

We will use the symbol '1' for the truth value TRUE and the symbol '0' for the truth value FALSE. We assume that these are the only two truth values, and that every statement has exactly one of these truth values.

Philosophy Corner

Even if these assumptions seem common-sense to you, they are controversial among philosophers of logic. First of all, there are logicians who want to consider cases where statements are neither true nor false, but have some kind of intermediate level of truth or probability. More controversially, some logicians think we should allow for the possibility of statements being both true and false at the same time.

It's important not to be confused between a statement being true or false, and us knowing whether a statement is true or false. We will often not know the truth value of a statement. But there will be an actual truth of the matter; we merely don't happen to know it. If this occurs, we'll use '?' to represent that we haven't got the information to determine a truth value. This is NOT a third truth value. We will be using just two truth values – true and false – in this book.

Finally, we will assume COMPOSITIONALITY – that the truth value of complex statements can be calculated by only considering the truth values of the simpler statements they contain, and how these statements are connected. So, if we know the truth-values of each part of a statement, we will know the truth-value of the whole statement.

Practice Exercises

At the end of most chapters there are exercises that review and explore the material covered in the chapter. Actually working through some problems is really important, because learning logic is more about developing a way of thinking than it is about memorising facts. Learning it is a surprising practical, hands-on activity.

⋆ **Exercise A:**
Select the statement expressing the conclusion of each argument:

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. 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 pipe. Recall, after all, that the gun had not been fired.

Exercise B: Which of the following sentences are statements?

1. The trains are always late.
2. Welcome to the University of Auckland.
3. Tailgating is a major cause of car accidents.
4. How can I stop tailgating?
5. The reason that I like bananas is that they have no bones.
6. When the car ahead passes an object, make sure you can count to four crocodiles before you pass the same object.
7. Just leave them in the bathroom and I'll deal with them on Wednesday afternoon.
8. Logic is the study of deductive arguments.
9. Never engage in light treason.
10. All great things are the cause of their own self-destruction.
11. Only 5% of North Americans can locate New Zealand on a world map.
12. Do at least a few of the exercises from the textbook every day.
13. An apple a day keeps the doctor away.
14. Ask me if I'm a policeman.
15. Are you a policeman?
16. The Mayor said bus passengers should be belted.

Chapter 2
The scope of logic

2.1 Arguments

One important use for Logic is evaluating arguments; sorting the good from the bad. People sometimes use the word 'argument' to talk about belligerent shouting matches. Logic is not concerned with such teeth-gnashing and hair-pulling. For logicians, an argument is not a disagreement, it is a reasoned series of statements culminating in a conclusion.

To be perfectly general, we can define an ARGUMENT as a series of statements. All the statements are premises, except for the final sentence, which is the conclusion. If the premises are true and the argument is a good one, then you should accept the conclusion is true.

An argument, as we will understand it, is something more like this:

Either the butler or the gardener did it.
The butler didn't do it.
∴ The gardener did it.

We have here a series of sentences. The three dots on the final line of the argument are read as 'therefore'. They indicate that this sentence is the *conclusion* of the argument. The two sentences before that are the *premises* of the argument. If you believe the premises and you think the conclusion follows from the premises, then you have a reason to believe the conclusion.

We will say that an argument is any collection of premises, together with a conclusion. An argument whose conclusion follows from its premises will be called a VALID argument.

We will discuss some further concepts that apply to arguments in a natural language like English, so that we begin with a clear understanding of what arguments are and what it means for an argument to be valid. Later we will 'translate' arguments from English into a formal language, explore some of the limitations of this process, then finally move to relying predominantly on arguments in the formal language. We will approach logic in this order as we want validity, as defined in the formal language, to have at least some of the important features of natural-language validity.

In the example argument above, we used individual sentences to express both of the argument's premises, and we used a third sentence to express the argument's conclusion. Many arguments are expressed in this way, but a single sentence can contain a complete argument. Consider:

> The butler has an alibi; so they cannot have done it.

This argument has one premise followed by a conclusion.

Many arguments start with premises, and end with a conclusion, but not all of them. An argument can also have its conclusion at the beginning:

> The gardener did it. After all, it was either the butler or the gardener. And the gardener didn't do it.

Equally, the conclusion might have been in the middle:

> The butler didn't do it. Accordingly, it was the gardener, given that it was either the gardener or the butler.

In logic, the most important thing we want to know about almost any argument is whether or not the conclusion follows from the premises. So the first thing to do is to separate out the conclusion from the premises. The following words are often used to indicate an argument's conclusion:

so, therefore, hence, thus, accordingly, consequently

For this reason, they are sometimes called conclusion indicator words.

By contrast, these expressions are premise indicator words, as they often indicate that we are dealing with a premise, rather than a conclusion:

since, because, given that

But in analysing an argument, there is no substitute for practice.

2.2 Introducing validity

We have talked about arguments, i.e., a single statement (the conclusion) that *follows from* a collection of statements (the premises). Logic develops theories and tools that tell us when a statement follows from some others.

Let's review the first argument we discussed:

Either the butler or the gardener did it.
The butler didn't do it.
∴ The gardener did it.

We don't have any context for what the statements in this argument refer to. Perhaps you suspect that 'did it' here means 'was the perpetrator' of some unspecified crime. You might imagine that the argument occurs in a mystery novel or TV show, perhaps spoken by a detective working through the evidence. But even without having any of this information, you probably agree that the argument is a good one in the sense that if both the premises are true (whatever they refer to), the conclusion must be true as well. We call arguments that have this property VALID.

By way of contrast, consider the following argument:

If the driver did it, the maid didn't do it.
The maid didn't do it.
∴ The driver did it.

We still have no idea what is being talked about here. But this argument is different from the previous one in an important respect. If the premises are true, the conclusion need not also be true. In this argument, the conclusion does not follow from the premises alone. So we say it is INVALID.

2.3 Cases and types of Validity

How did we determine that the second argument is invalid? We pointed to a case where neither the driver nor maid did it, but some other person did. Here, the premises are true and the conclusion is not. We call such a case a COUNTEREXAMPLE to the argument. If there is a counterexample to an argument, the conclusion cannot follow from the premises. For the conclusion to follow from the premises, the truth of the premises must *guarantee* the truth of the conclusion. If there is a counterexample, the truth of the premises does not guarantee the truth of the conclusion.

There can be more than one counterexample to an argument. For instance, perhaps in the above argument, neither the maid nor the driver 'did it', and in fact no one 'did it' at all. Perhaps there was a misunderstanding or deception. This serves equally well as a counterexample. But whichever counterexample you find, you only need one. One failure is enough to show than an argument is invalid; after that, we are just being mean.

As logicians, we want to be able to determine when the conclusion of an argument follows from the premises. The conclusion follows from the premises if there is no counterexample, a case where the premises are all true but the conclusion is not. This motivates a definition:

> A statement A FOLLOWS FROM statements $B_1, \ldots, B_n$ if and only if there is no case where $B_1, \ldots, B_n$ are all true and A is not true.)

This 'definition' is incomplete: it does not tell us what a 'case' is or what it means to be 'true in a case.' So far we've only seen an example: a hypothetical scenario involving three people, including a driver and a maid. In this scenario, as described above, the driver didn't do it, and so it is a case in which 'the driver did it' is not true. The premises of our second argument are true but the conclusion is not true: the scenario is a counterexample.

We said that arguments where the conclusion follows from the premises are called valid, and those where the conclusion does not follow from the premises are invalid. Since we now have at least a first stab at a definition of 'follows from', we'll record this:

> An argument is VALID if and only if the conclusion follows from the premises.

> An argument is INVALID if and only if it is not valid.

Philosophy Corner

In the logics we'll cover in this course, an argument is invalid exactly when it has a counterexample; that is, a conclusion follows from premises unless you can describe a situation where the premises are true and the conclusion is not. Now, suppose you have two premises that disagree. Then there is no case where they are both true and the conclusion false, so there can't be any counterexamples; every conclusion follows from these premises. Weird but true!

Logicians are in the business of making the concept of 'case' more precise, and investigating which arguments are valid under various readings of 'case'. If we take 'case' to mean 'hypothetical scenario' like the counterexample to the second argument, then the first argument counts as valid. If we imagine a scenario in which either the butler or the gardener did it, and also the butler didn't do it, we are automatically imagining a scenario in which the gardener did it. So any hypothetical scenario in which the premises of our first argument are true automatically makes the conclusion of our first argument true. This makes the first argument valid.

However, there are other ways of unpacking the concept of 'case'. We can look at cases as being scenarios where we have incomplete information, and anything that is compatible with the information *we already have* is possible. We might also consider some inconsistent situations, in case our evidence conflicts with itself in a few areas. Both of these distinctions lead to different types of logic – roughly Intuitionistic or Constructive logics in the former case, and Relevant or Dialetheic logics in the latter. Some of these logics do not guarantee that the gardener did it! We will not study these logics in this course. They are great fun, but we think they are only appropriate for more advanced students. Let's learn to walk, first…

2.4 Cases and types of possibility

Making 'case' more specific by interpreting it as 'hypothetical scenario' is not the end of the story. We still don't know what to count as a hypothetical scenario. If we limit ourselves to our existing beliefs, then we can't use reason to change them, so our hypotheses must be able to include statements we would ordinarily reject. But how far do we go? Are our hypotheses limited by the laws of nature? Or by our conceptual and linguistic connections? Or neither? What answers we give to these questions determine which arguments we count as valid.

Suppose our logic is limited by the current laws of physics. Then the following argument is valid:

The spaceship *Rocinante* took six hours to reach Jupiter from Tycho.
∴ The distance between Tycho and Jupiter is less than $6.5 \times 10^9 km$.

A counterexample to this argument would be a scenario in which the *Rocinante* makes a trip of over 7 billion kilometres in 6 hours, exceeding the speed of light. This scenario isn't compatible with current physics, which does not allow faster than light travel. So it can't be a counterexample; but it would have been a counterexample under our theories before Einstein. We are in the awkward position of saying the argument used to be invalid, but now is valid. Worse still, we can't reason about alternative scientific theories.

Limiting ourselves to reasoning without our best theories of physics restricts us unduly. Worse still, if we accepted this, shouldn't we also be restricted by all our best theories of nature – chemistry, biology, psychology, sociology, etc.? We'd not be able to use reason to challenge any part of the scientific consensus.

Suppose we are instead limited by what we can conceive or communicate. Then the following is valid:

Priya is an ophtalmologist.
∴ Priya is an eye doctor.

If you imagine Priya being an ophtalmologist, and you know what the word means, you also imagine Priya being an eye doctor. That's just what 'opthalmologist' and 'eye doctor' mean. A scenario where Priya is an ophtalmologist but not an eye doctor is ruled out by the conceptual connection between these words. But we know that word meanings change over time, and so does what we can conceive. Our logic would be culturally specific, rather than universal.

Depending on what kinds of cases we consider as potential counterexamples, we arrive at different notions of validity. We could require our counterexamples to conform to the laws of nature (nomologic validity). Or we could require they don't violate our conceptual worldview. Or even our moral standards. But for all these notions of validity, the laws of nature or thought or meaning or morality are part of whether an argument is valid. And that's going to be messy.

Instead of these forms, we will only be considering whether an argument is LOGICALLY VALID. So if you ever start taking into account natural laws, or the meanings of different words, or conceivability or morality, you can be sure that you have shifted away from doing Logic. We have taught you about these ways of reasoning solely to make you aware of them so you can avoid them (in this course).

2.5 Validity and Truth

If an argument is valid, and all its premises are true, then its conclusion must be true. That's why we value validity so highly. But validity *by itself* doesn't require the premises (or conclusion) to be true. Consider this example:

Oranges are either fruit or musical instruments.
Oranges are not fruit.
∴ Oranges are musical instruments.

The conclusion of this argument is ridiculous. Nevertheless, it follows from the premises. *If* both premises are true, *then* the conclusion just has to be true. So the argument is valid. Don't let the meanings of the words distract you, nor whether the premises are true, or whether they even make sense. None of that is part of Logic.

Conversely, having true premises and a true conclusion is not enough to make an argument valid. Consider this example:

London is in England.
Beijing is in China.
∴ Paris is in France.

The premises and conclusion of this argument are all true, but the argument is invalid. If Paris were to declare independence from France, then the conclusion would be false, while the premises would remain true. This is a counter-example to the argument, so the argument is invalid.

The important thing to remember is that validity is not about the truth or falsity of the statements in the argument. It is about whether it is *possible* for all the premises to be true and the conclusion to be not true at the same time (in some hypothetical case). What is actually true does not determine whether an argument is valid or not. Logic doesn't care about anything except mere validity.

There are also good arguments that are less than valid. Consider this one:

Every winter so far, it has rained in Auckland.
∴ It will rain in Auckland this coming winter.

The seems like good reasoning, but the argument is invalid. Even if it has rained in Auckland every winter thus far, it remains *possible* that Auckland will stay dry all through the coming winter. There are many good, reliable arguments that don't guarantee their conclusions. But determining what makes them good or reliable is again a matter of actual facts. For the same reason (we aren't interested in actual facts) that we don't care about stronger notions than validity, we don't care about weaker notions either.

Practice Exercises

Exercise A: Which of the following arguments are logically valid? Which are invalid?

1. Socrates is a man.
2. All men are carrots.

∴ Socrates is a carrot.

1. Jacinda Ardern was either born in Berlin or she was once a nurse.
2. Jacinda Ardern was never a nurse.

∴ Jacinda Ardern was born in Berlin.

1. If I light the match, the paper will burn.
2. I do not light the match.

∴ The paper will not burn.

1. Confucius was either from France or from Luxembourg.
2. Confucius was not from Luxembourg.

∴ Confucius was from France.

1. If the world ends today, then I will not need to get up tomorrow morning.
2. I will need to get up tomorrow morning.

∴ The world will not end today.

1. Joe is now 19 years old.
2. Joe is now 87 years old.

∴ Bob is now 20 years old.

⋆ **Exercise 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?
3. A valid argument with only false premises and a false conclusion?
4. An invalid argument that can be made valid by adding a new premise?
5. A valid argument that can be made invalid by adding a new premise?

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

Chapter 3
Other logical notions

In Chapter 2, we introduced the concept of a valid argument. This is perhaps the most important concept in logic. In this section, we will introduce some related concepts. They all rely, as did validity, on the idea that statements are true (or not) in cases.

3.1 Consistency

Consistency is the other central logical notion that we'll be working with, along with validity. We highly value those of our ideas and beliefs that could all be true together, and tend to reject one or more of a set of statements if they can't all be true.

Consider these two statements:

B1. Jane's only brother is shorter than her.
B2. Jane's only brother is taller than her.

Logic alone cannot tell us which, if either, of these statements is true. Yet we can say that *if* the first statement (B1) is true, *then* the second statement (B2) must be false. And if B2 is true, then B1 must be false. There is no case where both statements are true together. This motivates the following definition:

> Statements are MUTUALLY CONSISTENT if and only if there is a case where they are all true together.

B1 and B2 are *mutually inconsistent*, while the following two statements are mutually consistent:

B1. Jane's only brother is shorter than her.
B2. Jane's only brother is younger than her.

We can ask about the mutual consistency of any number of statements. For example, consider the following four statements:

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.

G1 and G4 together tell us there are at least four martian giraffes at the park. This conflicts with G3, which implies that there are no more than two martian giraffes there. So the statements G1–G4 are mutually inconsistent. They cannot all be true together.

3.2 Logical Terms

We can also consider the consistency of single statements. Consider these statements:

1. It is raining.
2. It is both raining here and not raining here.
3. Either it is raining here, or it is not.

In order to know if statement 1 is true, we would need to look outside or check the weather channel. It might be true; it might be false. A statement which might be true and might be false is called CONTINGENT.

But we do not need to check the weather to determine whether or not statement 2 is true. It must be false; it's not consistent with itself. It is a LOGICAL FALSEHOOD.

Statement 3 is different again; it's always true. Regardless of what the weather is like, it is either raining or it is not. Statement 3 is mutually consistent with every contingent statement. It is a LOGICAL TRUTH. This is also sometimes known as a tautology. Tautologies are important in logic, although primarily for historical reasons.

Philosophy Corner

Something might *always* be true and still be contingent. For instance, if there never were a time when the universe contained fewer than two things, then the statement 'At least two things exist' would always be true. Yet the statement is contingent: we can imagine that the universe only contains a single photon, and then the statement would have been false.

We can also ask about the logical relations *between* two statements. For example:

> John went to the store after he washed the dishes.
> John washed the dishes before he went to the store.

These two statements are both contingent, since John might not have gone to the store or washed dishes at all. But if either of the statements is true, then they both are; and if either of the statements is false, then they both are. When two statements are true in exactly the same cases, we say that they are LOGICALLY EQUIVALENT.

Let's change that example slightly:

> John went to the store after he washed the dishes.
> John washed the dishes after he went to the store.

Now, if either of those statements is true, the other is false (assuming that John washed the dishes exactly once, and went to the store exactly once). And if either of them is false, the other is true! When two statements are true in exactly the opposite cases, we say that they are LOGICALLY CONTRADICTORY.

Summary of Logical Terms

- An argument is VALID if there is no case where the premises are all true and the conclusion is false; it is INVALID otherwise.
- A collection of statements is MUTUALLY CONSISTENT if there is a case where they are all true; otherwise it is MUTUALLY INCONSISTENT.
- A LOGICAL TRUTH is a statement that is true in every case.
- A LOGICAL FALSEHOOD is a statement that is false in every case.
- A CONTINGENT statement is neither a logical truth nor a logical falsehood; it is true in some case and false in some other case.
- Two statements are LOGICALLY EQUIVALENT if, in each case, they are both true or both false.
- Two statements are LOGICALLY CONTRADICTORY if, in each case, one is true and the other false.

Practice Exercises

Exercise A: Are these logical truths, logical falsehoods, or contingent?

1. Rangi Topeora, a Ngāti Toa leader, signed the Treaty of Waitangi.
2. A woman signed the Treaty of Waitangi.
3. No woman has ever signed the Treaty of Waitangi.
4. If Rangi Topeora signed the Treaty of Waitangi, then someone has.
5. If anyone has ever signed the Treaty of Waitangi, it was Rangi Topeora.

⋆ **Exercise B:** Are these logical truths, logical falsehoods, or contingent?

1. Elephants dissolve in water.
2. Wood is a light, durable substance useful for building things.
3. If wood were a good building material, it would be good for building.
4. I live in a three story building that is two stories tall.
5. If gerbils were mammals they would nurse their young.

Exercise C: Which pairs of statements are logically equivalent?

1. Elephants dissolve in water.
   If you put an elephant in water, it will disintegrate.
2. All mammals dissolve in water.
   If you put an elephant in water, it will disintegrate.
3. John Key is the 39th Prime Minister of New Zealand.
   Jacinda Arden is the 40th Prime Minister of New Zealand.
4. Jacinda Arden is the 40th Prime Minister of New Zealand.
   Jacinda Arden is the Prime Minister of New Zealand immediately after the 39th Prime Minister.
5. Elephants dissolve in water.
   All mammals dissolve in water.

⋆ **Exercise D:** Which pairs of statements are logically equivalent?

1. Sarah Murphy is a biathlete.
   Valerie Adams is a shot putter.
2. Sarah Murphy represented New Zealand at the Olympics.
   Valerie Adams represented New Zealand at the Olympics.
3. All biathletes are cross-country skiers.
   Sarah Murphy is a cross-country skier.
4. Sarah Murphy won more medals at the Olympics than Valerie Adams.
   Valerie Adams won fewer medals at the Olympics than Sarah Murphy.
5. Sarah Murphy represented New Zealand at the Olympics, or she didn't.
   Valerie Adam represented New Zealand at the Olympics, or she didn't.

Exercise E: Consider the following statements:

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.

Which combinations of statements are mutually consistent?

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

⋆ **Exercise F:** Consider the following statements.

M1 All people are mortal.
M2 Hypatia is a person.
M3 Hypatia will never die.
M4 Hypatia is mortal.

Which combinations of statements are mutually consistent?

1. Statements M2 and M3
2. Statements M1 and M4
3. Statements M1, M2, and M3
4. Statements M1, M2, and M4
5. Statements M1, M3, and M4
6. Statements M2, M3, and M4
7. Statements M1, M2, M3, and M4

⋆ **Exercise G:** Which of the following is possible?
If it is possible, give an example. If it is not possible, explain why.

1. A valid argument, the conclusion of which is a logical falsehood
2. An invalid argument, the conclusion of which is a logical truth
3. A valid argument, whose premises are all logical truths, and whose conclusion is contingent
4. A logical truth that is contingent
5. Two logically equivalent statements, both of which are logical truths
6. Two logically equivalent statements, one of which is a logical truth and one of which is contingent
7. Two logically equivalent statements that are mutually inconsistent
8. A mutually consistent collection that contains a logical falsehood
9. A mutually inconsistent set of statements that contains a logical truth

Part II

Truth-Functional Logic Symbolisation

Chapter 4
Symbolising

4.1 Formal Validity

One key feature of *logical* validity is that it should not depend on the content of the premises and conclusion, but only on their logical form. Appealing to logical form will allow us to make finer-grained distinctions between arguments. For instance, both

Either Priya is an ophtalmologist or a dentist.
Priya isn't a dentist.
∴ Priya is an eye doctor.

and

Either Priya is an ophtalmologist or a dentist.
Priya isn't a dentist.
∴ Priya is an ophtalmologist.

may seem to be valid arguments. But while the validity of the first depends on the meanings of 'ophtalmologist' and 'eye doctor', the second does not. The second argument is FORMALLY VALID; the *form* of the argument is sufficient. We can describe the form of this argument as a pattern, something like this:

Either A is an X or a Y.
A isn't a Y.
∴ A is an X.

Here, A, X, and Y are placeholders for appropriate phrases that, when substituted for A, X, and Y, turn the pattern into an argument consisting of statements.

For instance,

Either Mei is a mathematician or a botanist.
Mei isn't a botanist.
∴ Mei is a mathematician.

is an argument of the same form, but the first argument of this chapter is not. In fact, it is not formally valid. Its form is:

Either A is an X or a Y.
A isn't a Y.
∴ A is a Z.

In this pattern we can replace X by 'ophtalmologist', Y by 'dentist', and Z by 'eye doctor' to obtain the original argument. But here is another argument of the same form:

Either Mei is a mathematician or a botanist.
Mei isn't a botanist.
∴ Mei is an acrobat.

This argument is clearly not valid, since we can create a counter-example of a mathematician named Mei who is neither an acrobat nor botanist.

Our strategy as logicians will be to come up with a notion of 'possibility' on which an argument turns out to be valid if it is formally valid. Clearly such a notion of 'possibility' will have to violate not just some laws of nature but some rules of English. Since we want our first argument to be invalid, we must allow a counterexample where Priya is an ophtalmologist but not an eye doctor, despite the meanings of 'ophtalmologist' and 'eye doctor.' But we can't ignore English entirely. You'll notice that the form keeps some English terms; those that make the argument valid. We'll start in this chapter by focusing on 'either-or', 'not', 'if-then', and 'both-and', which we will take to be *logical* terms.

History Corner

Why those four terms? We can blame the Stoic logicians of Ancient Greece. They developed an early logic system using the Greek equivalents of 'either-or', 'not', 'if-then', and 'both-and'. The Stoics flourished in the third century BCE. The Stoic logicians were the first to create a logic based on statements, a tradition that we are still following in this course.

We'll look at other logical terms such as 'is the same as', 'none', 'some', and 'all' later in this book.

Consider for instance this argument, which uses the logical term 'if-then':

It is raining outside.
If it is raining outside, then Dipan is miserable.
∴ Dipan is miserable.

and another argument:

Talia is an anarcho-syndicalist.
If Talia is an anarcho-syndicalist, then Ilya is a fan of Tolstoy.
∴ Ilya is a fan of Tolstoy.

Both arguments are valid, and there is a straightforward sense in which we can say that they share a common form. We might express the form thus:

A
If A, then C
∴ C

This is a simple argument *form*, and it is formally valid. Note also that the symbols are standing for statements, which is simpler than our earlier attempt at a form where they stood for a mixture of names and adjectives. Let's re-analyse our earlier example:

Either Priya is an ophtalmologist or a dentist.
Priya isn't a dentist.
∴ Priya is an ophtalmologist.

With symbols standing for statements, its argument form is:

Either A or B
Not A
∴ B

This argument will also turn out to be formally valid.

These examples illustrate an important idea. The validity of these arguments has little to do with the meanings of English expressions like 'Priya is an ophtalmologist', 'Ilya is a fan of Tolstoy', or 'Dipan is miserable'. If it has to do with meanings at all, it is with the meanings of phrases like 'either-or', 'not,' 'if-then', and 'both-and'.

We will shortly introduce a formal language which allows us to symbolise many arguments in such a way as to show that they are formally valid. That language will be *truth-functional logic*, or TFL.

4.2 English v Formal Validity

There are plenty of arguments that are valid in English, but not for reasons relating to their argument form. Take an example:

  ‘Mint Sauce’ is my pet ewe
∴ ‘Mint Sauce’ is my pet sheep

Because a ewe is a female sheep, it seems impossible for the premise to be true and the conclusion false. So the English argument is valid. However, this validity is not related to the form of the argument. Here is an invalid argument with the same form:

  ‘Mint Sauce’ is my pet ewe
∴ ‘Mint Sauce’ is my pet pukeko

In the first argument, the conclusion follows from the premise due to the relationship between the meanings of the English words ‘ewe’ and ‘sheep’. Similarly, consider the argument:

  The sculpture is green all over.
∴ The sculpture is not red all over.

Again, it seems impossible for the premise to be true and the conclusion false, because nothing *physical* can be both green all over and red all over. But here is an invalid argument with the same logical form:

  The sculpture is green all over.
∴ The sculpture is not shiny all over.

This argument is invalid, since it is possible for something to be green all over and shiny all over. (We might paint our nails an elegant shiny green.) Again, the validity of the first of these two English arguments is due to the relationships between colours (or colour-words), not the *form* of the argument.

Arguments that are valid in English but not formally valid demonstrate both the power and limitations of formal systems.

TFL only helps us with checking if arguments are formally valid.

In addition, TFL is based on only four logical terms. It doesn’t include ‘all’, ‘most’, ‘is the same as’, ‘exists’, ‘possibly’, ‘afterwards’, or many other potential logical terms. So while an argument that is formally valid in TFL will be valid generally, many arguments will be valid, even formally valid, but not valid in TFL.

4.3 Atomic Statements

We started isolating the form of an argument, in §4.1, by replacing the simplest *statements* with individual letters. For example, 'it is raining outside' is a statement within 'If it is raining outside, then Jenny is miserable'. There are no simpler statements within 'it is raining outside', it cannot be broken down further. It is an atomic statement, and we might symbolise it with the letter 'A'.

Our artificial language, TFL, pursues this idea ruthlessly. We start with some *statement symbols*. These symbols will be the basic formulas out of which more complex formulas are built to express more complex English statements. We will use single uppercase letters as statement symbols in TFL, such as A or P. There are only twenty-six letters in the Latin alphabet, and there is no limit to the number of statements that we might want to consider. But in practice we won't need that many symbols.

We will use statement symbols to represent, or *symbolise*, our atomic English statements. These symbols may stand for different things in each argument. So that we know which symbol stands for which statement, we will need a SYMBOLISATION KEY, such as the following:

A:	It is raining outside
F:	Jenny is miserable

This symbolisation key will remain constant for an entire argument (or piece of prose). In doing this, we are not fixing this symbolisation *once and for all*. We are just saying that, for the time being, we will think of the statement symbol 'A' of TFL as symbolising the English statement 'It is raining outside', and the statement symbol 'C' of TFL as symbolising the English statement 'Jenny is miserable'. Later, when we are dealing with different statements or different arguments, we can provide a new symbolisation key; perhaps:

A:	Talia is an anarcho-syndicalist
F:	Ilya is a fan of Tolstoy

It is important to understand that whatever structure an English statement might have is lost when it is symbolised by a statement symbol of TFL. From the point of view of TFL, a statement symbol is just an atomic formula. It can be used to build more complex formulas, but it cannot be taken apart. We cannot split the atom (at least, not yet).

Practice Exercises

⋆ **Exercise A:** Using the symbolisation key below, identify the atomic statements, and then symbolise each sentence using TFL:

G:	The grass is wet
R:	It is raining
S:	The sun is shining
W:	The ground is wet

1. It's both raining and the ground is wet.
2. It's raining and the sun is not shining.
3. It's not both raining and the sun shining.
4. It's neither raining nor is the sun shining.
5. If it's raining, then the ground and the grass are wet.
6. The grass is wet but the ground isn't.
7. If it's raining, then the ground is wet and the sun is not shining.
8. If it's raining then the ground is wet, and the sun is not shining.

Exercise B: The last two sentences in Exercise A are identical apart from the location of the comma. And their symbolisations appear almost identical. Add brackets (like in maths) to your symbolisation of each sentence, to help differentiate them. What does this tell you about one role of commas?

⋆ **Exercise C:** For each argument below, construct a symbolisation key, and then symbolise each sentence in the argument. Pay especial attention to commas, using what you have learned in the previous exercises:

1. I will not be conscripted into the army. The reasons are that I won't be conscripted unless there is a Zombie Apocalypse. But if there is a Zombie Apocalypse then it will be over in ten days. If a Zombie Apocalypse is over in ten days then they won't conscript me.
2. If Kim runs for the bus, then she is stressed. If Kim runs for the bus, then she gets more exercise. So if Kim does not get more exercise, she need not run for the bus and will not be stressed.
3. If minds are wholly private, then I cannot know about any individual other than myself that they have a mind. Having a mind is needed for being a person. If this is true, then if I cannot know that any individual besides myself has a mind, then I also cannot know that any other individual is a person. So, if I can know that even one individual other than myself is a person, then it is false that minds are completely private.

Chapter 5
Connectives

In the previous chapter, we considered symbolising fairly basic English sentences with statement symbols of TFL. This leaves us wanting to deal with the English terms ‘and’, ‘or’, ‘not’, and so forth. These are *connectives* – they can be used to form new statements out of old ones. In TFL, we will make use of logical connectives to build complex formulas from atomic symbols. There are four basic logical connectives in TFL, and one more that is commonly used. This table summarises them, and they are explained throughout this section.

symbol	**what it is called**	**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 …’

These are not the only connectives of English of interest. Others include ‘unless’, ‘neither … nor …’, ‘except if’, ‘only if’, ‘but’, ‘although’, ‘since’, and ‘because’. We will see that the first few of these can be at least partially symbolised by the logical connectives we will discuss, while the last few cannot. ‘Since’ and ‘because’, although they are very useful in arguments, cannot be symbolised in TFL as they are not *truth functional.*

There are only 16 distinct truth functions that take two truth-values and return a truth-value. (Why?) This sharply limits the expressive power of TFL; most English connectives cannot be symbolised in TFL. However the four basic truth functions can be used to define all 16 truth functions. So our four connectives have the full power of TFL. They are all we need.

5.1 Negation

Consider how we might symbolise these statements:

1. Mary is in Barcelona.
2. It is not the case that Mary is in Barcelona.
3. Mary is not in Barcelona.
4. Mary has left Barcelona.

Sentence 1 is an atomic statement, so we'll add it to our symbolisation key:

B: Mary is in Barcelona.

The other statements are related to Statement 1, so to capture that relationship, we want to use the same atomic symbol. Now, Statement 2 is something like 'It is not the case that B', so we can symbolise it as '$\neg B$'.

An English statement can be symbolised as $\neg\mathcal{A}$ if it can be paraphrased as 'It is not the case that $\mathcal{A}$'.

Statement 3 also contains the word 'not', and on the surface it appears roughly similar to Statement 2 so we will also symbolise it as '$\neg B$'. Statement 4 doesn't contain 'not', but Statement 3 expresses part of its meaning. We might be tempted to also symbolise that she *had been* in Barcelona, but at a minimum '$\neg B$' is part of the symbolisation.

Linguistics Corner

Statements 2, 3, and 4 aren't interchangable in conversation. They are in different registers – 2 is a lot more formal – but 4 (and perhaps 3) also presupposes that Mary exists, while 2 does not. Try re-reading these statements while thinking of Mary as your imaginary aunt. However, TFL is simple enough that we can't make these kinds of distinctions. Keep an eye out for what other aspects of meaning are being discarded when we symbolise.

Logic Corner

You might be wondering what $\mathcal{A}$ is. It is a variable that stands for any formula, rather than being a specific atomic symbol, like A or B. We'll be using these curly letters when we are *talking about* formulas rather than using them. They form a meta-language – a set of formulas to stand for other formulas. This distinction will only become important in later courses, so don't worry too much for now.

It may help to examine a few more examples:

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

Let us use the following symbolisation key:

> R: The widget is replaceable.

Statement 5 can be symbolised by 'R'. Statement 6 says the widget is irreplaceable, which roughly means that it is not the case that (the widget is replaceable). So we will symbolise it as '$\neg R$'. Statement 7 can be paraphrased as 'It is not the case that (the widget is irreplaceable).' Which is '$\neg$' then Statement 6. So we will symbolise it as '$\neg\neg R$'. You might be tempted to just symbolise it as R. But we don't know (yet) if R is the same as $\neg\neg R$.

But not all negations are created equal. Consider:

8. Jane is happy.
9. Jane is unhappy.

Suppose we symbolise Statement 8 as 'H'. It would then be tempting to symbolise Statement 9 as '$\neg H$'. This isn't terrible, as if Jane is unhappy, then she is not happy. However Statement 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 experiencing very different emotions. In order to fully symbolise Statement 9, then, we would need a new statement symbol. This might become important if we are expressing the subtleties of a statement like 'Jane is not *un*happy'.

Finally, some apparent negations are not negations at all:

10. This textbook is flammable.
11. This textbook is inflammable.
12. This textbook is valuable.
13. This textbook is invaluable.

Statements 10 and 11 have the same meaning, so would share the same symbolisation. Even more confusingly, Statement 13 means something like 'This textbook is very valuable', which is an intensified version of Statement 12. We don't have a 'very' connective. (We'll discuss why we can't have this type of connective many chapters hence.)

Not all 'not's are created equal. You will need to think about the meaning of each statement, rather than turn every 'not ...', '__n't ...', and 'un...' into a '$\neg$...'. The same caution will be required for our other connectives.

5.2 Conjunction

Here's a key for some simple atomic statements:

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

These complex statements combine some of the above atomic statements:

14. Adam is athletic, and Barbara is also athletic.
15. Barbara is athletic and energetic.
16. Barbara and Adam are both athletic.
17. Although Barbara is energetic, she is not athletic.
18. Adam is athletic, but Barbara is more athletic than him.

Statement 14 roughly says 'A and B', so we will symbolise it as '$(A \wedge B)$'. This connective is called CONJUNCTION. We also say that 'A' and 'B' are the two CONJUNCTS of the conjunction '$(A \wedge B)$'.

An English statement can be symbolised as $(\mathcal{A} \wedge \mathcal{B})$ if it can be paraphrased as 'Both $\mathcal{A}$ and $\mathcal{B}$'.

Notice that we don't attempt to symbolise the word 'also' in Statement 14. Words like 'both' and 'also' function to draw our attention to the fact that two things are being conjoined. Maybe they affect the emphasis of a statement, but we will not (and cannot) symbolise such things in TFL. 'Both' also serves to tell us where the conjunction starts, which can help us to understand ambiguous statements. The brackets '(' and ')' around the formula serve this role in TFL. We'll see an example of this shortly.

We often need to expand or paraphrase English sentences before symbolising them, to ensure that our connectives are conjoining *statements*. For example, we might first try to symbolise Statement 15 as 'B and energetic'. This would be a mistake; 'energetic' isn't a statement. Instead, we can paraphrase it as 'Barbara is athletic and Barbara is energetic', and then symbolise it as '$(B \wedge E)$'. Similarly, Statement 16 can be paraphrased as 'Barbara is athletic and Adam is athletic', and symbolised as '$(B \wedge A)$'.

Statement 17 is more complicated. The word 'although' sets up a contrast between the first part of the statement and the second part, but TFL can't express this, so we'll ignore it. We will paraphrase Statement 17 as '*Both* Barbara is energetic *and* Barbara is *not* athletic'. This can be symbolised with the TFL statement '$(E \wedge \neg B)$'. Note that we have lost all sorts of nuance in this symbolisation.

Logic Corner

You can now see that TFL cannot really translate sentences, because it misses out on nuances in the forms and modes of expressions. That's why we talk about 'symbolisation' instead. You can think of symbolisation as a way of extracting just the information from sentences that is required for logic.

Statement 18 raises similar issues. 'But' is contrastive, which is also beyond TFL's expressive powers. So we paraphrase the statement as '*Both* Adam is athletic, *and* Barbara is more athletic than Adam'. We could symbolise this as '$(A \wedge B)$', but this logical form loses the information about Barbara's superior athleticism. If that information is important to the argument, we need a new statement symbol in our key:

R: Barbara is more athletic than Adam.

Now we can symbolise Statement 18 by '$(A \wedge R)$'.

You might be wondering why we put brackets around the conjunctions. This is for the same reason we use brackets in mathematics; it's needed for more complicated formulas. We can illustrate this by symbolising some statements containing both negation and conjunction:

19. You won't get both soup and salad.
20. You won't get soup but you'll get salad.

We'll use this symbolisation key:

P: You will get soup.
S: You will get salad.

Statement 19 can be paraphrased as 'It is not the case that (both you will get soup and you will get salad)'. We would symbolise 'both you will get soup and you will get salad' as '$(P \wedge S)$'. To symbolise Statement 19, we negate the whole statement: '$\neg(P \wedge S)$'.

Statement 20 is a conjunction of 'You will *not* get soup', and 'You will get salad'. 'You will not get soup' is symbolised by '$\neg P$'. So we symbolise Statement 20 as '$(\neg P \wedge S)$'.

These English statements have different meanings, and so their symbolisations differ. 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 – which part of the formula it applies to.

5.3 Disjunction

Here's another symbolisation key:

V:	Adara will play videogames.
M:	Adara will watch movies.
O:	Omar will play videogames.

And some English disjunctions using the above statements:

21. Either Adara will play videogames, or she will watch movies.
22. Either Adara or Omar will play videogames.

Statement 21 can be symbolised by '$(V \vee M)$'. We call 'V' and 'M' the DISJUNCTS of the disjunction '$(V \vee M)$'.
Statement 22 can be paraphrased as 'Either Adara will play videogames, or Omar will play videogames'. Now we can symbolise it by '$(V \vee O)$'.

> An English statement can be symbolised as $(\mathcal{A} \vee \mathcal{B})$ if it can be paraphrased as 'Either $\mathcal{A}$ or $\mathcal{B}$.'

Like conjunction, disjunction interacts with negation. Consider:

23. Either you will not have soup, or you will not have salad.
24. You will have neither soup nor salad.
25. You will not have both soup and salad.

Statement 23 can be paraphrased as: 'Either it is not the case that (you get soup), or it is not the case that (you get salad)'. To symbolise this in TFL, we need both disjunction and negation. 'It is not the case that you get soup' is symbolised by '$\neg P$'. 'It is not the case that you get salad' is symbolised by '$\neg S$'. So Statement 23 itself is symbolised by '$(\neg P \vee \neg S)$'.

Statement 24 can be paraphrased as: 'It is not the case that (either you get soup or you get salad)'. The '*n*either ... *n*or ...' tells us that the whole disjunction (the 'or' in 'nor' tells us it's a disjunction) is being *n*egated. We symbolise Statement 24 with '$\neg(P \vee S)$'. The different symbolisations of statements 23 and 24 allow us to express their different meanings.

Statement 25 is just a paraphrase of Statement 19, so it has the same symbolisation: '$\neg(P \wedge S)$'. Now, if you don't have both soup and salad, then either you don't have soup or you don't have salad. But that's just Statement 23. And if either you don't have soup or you don't have salad, then you can't have both. Statements 23 and 25 seem to say the same thing, even though they have different symbolisations! We are going to need methods to determine if different formulas are logically equivalent. Developing these methods will be one of the main themes of the rest of this book.

5.4 Conditionals

Our next symbolisation key is:

> A: Aroha is in Aotearoa.
> T: Aroha is in Waikato.

And here's some English conditionals using the above statements:

26. If Aroha is in Waikato then Aroha is in Aotearoa.
27. Aroha is in Waikato if Aroha is in Aotearoa.

Statement 26 is roughly of this form: 'if T then A'. We symbolise it as '$(T \to A)$'. The connective $\to$ is called a CONDITIONAL. Here, 'T' is called the ANTECEDENT of the conditional '$(T \to A)$', and 'A' is called the CONSEQUENT.

> An English statement can be symbolised as $(\mathcal{A} \to \mathcal{B})$ if it can be paraphrased as 'If A, then B'.

Statement 27 is also a conditional. You might think to symbolise this in the same way as Statement 26. That would be a mistake. The location of the term 'if' is important. Your knowledge of geography tells you that Statement 26 is true: there is no way for Aroha to be in Waikato that doesn't involve her being in Aotearoa (given current geography – we aren't saying this is logically true!). But Statement 27 is different: suppose Aroha is in Taranaki or Taupo; then she is in Aotearoa without being in Waikato, and Statement 27 is false. Since geography alone dictates the truth of Statement 26, whereas Aroha's location affects the truth of Statement 27, they must be distinct.

Tip: It's helpful to always paraphrase conditional statements to use an 'if', and to do so with the 'if' (antecedent) before the 'then' (consequent). This paraphrasing of Statement 27 gives 'If Aroha is in Aotearoa, then Aroha is in Waikato'. So we can symbolise it by '$(A \to T)$'.

Many English expressions can be paraphrased using conditionals.

28. Aroha's being in Waikato informs us that she is in Aotearoa.
29. For Aroha to be in Aotearoa, it is enough that she is in Waikato.
30. Aroha's being in Aotearoa guarantees that she is in Waikato.

These statements can all be paraphrased as 'If Aroha is in Waikato, then Aroha is in Aotearoa'. So they can all be symbolised by '$(A \to T)$'.

But symbolising conditionals isn't as straight-forward as it looks. We'll have more to say about some potential problems with them shortly.

Practice Exercises

Exercise A: Symbolise each of the following statements, being careful with negation. Create your own key as needed. If a statement is ambiguous, symbolise each possible meaning.

1. No one has read my novel.
2. Your assignment was unintelligible, and I don't need to hear any excuses.
3. He doesn't pray at the University mosque.
4. I was not unaware of the time but I am now.
5. I cannot say that I do not disagree with you.

Exercise B: Symbolise each of the following statements, being careful with conjunction and disjunction. Create your own key as needed. If a statement is ambiguous, symbolise each possible meaning.

1. **

Exercise C: Symbolise each of the following statements, being careful with conditionals. Create your own key as needed. If a statement is ambiguous, symbolise each possible meaning.

1. **

Chapter 6
Connective Complications

6.1 Interpreting Language

Many students struggle with figuring out which connectives to use when symbolising English statements. One reason for this is that we are all trained to interpret the meaning of a statement in its context, from the moment we are born. And we tend to construct our own statements within its context.

For example, we tend not to say 'or' when we can say 'and'; instead we provide alternatives that don't overlap, due to restrictions based on natural laws and social customs. These choices of words exclude the 'or' from allowing both options. More generally we choose our words to convey and imply and connote a lot of information – the probable, the expected, and the default – as well as the explicit content.

Then a logician comes along and says "you are interpreting 'or' incorrectly – it's never exclusive". But you've been treating it as excluding the possibility of both options for decades, and so has everyone else. What is the logician thinking? Don't they understand language?

Yes, we do. Linguists and philosophers of language divide communication into semantics (the literal truth-based meaning of sentences), and pragmatics (how we use the sentences in a continual, fragile, dynamic negotiation for agreement). If you've ever listened to a recording of a real conversation between friends, you'll know that what is said is fragmentary and incoherent – a lot of the communication comes from shared history, understanding key phrases, body language, etc.

The pragmatics of a statement is not part of our symbolisation, because it's all culture and context-dependent. This means that people will disagree on what is being said. We will stick to what is the agreed core of literal meaning (semantics).

This chapter contains advice on how to do this well.

6.2 Exclusive Disjunction

Sometimes in English, the word 'or' is used in a way that excludes the possibility that both disjuncts are true. This is called an EXCLUSIVE OR. An *exclusive or* is clearly intended when it says, on a restaurant menu, 'the main course comes with either soup or salad': you may have soup; you may have salad; but if you want *both* soup *and* salad you will have to pay extra.

Other times, the word 'or' allows for the possibility that both disjuncts might be true. For example, 'Adara or Omar will come to the party' allows that Adara might come alone, Omar might come alone, or they might both come to the party. The statement merely says that *at least* one of them will come to the party. This is called an INCLUSIVE OR. The TFL symbol '$\vee$' always symbolises an *inclusive or*.

But what about the *exclusive or*? Consider:

1. You get either soup or salad, but not both.

Statement 1 expresses the exclusivity of the options; it is an *exclusive* 'or'. We can break the statement into two parts. The first part says that you get soup or salad. We symbolise this as '$(P \vee S)$'. The second part says that you do not get both. Using both negation and conjunction, we symbolise this with '$\neg(P \wedge S)$'. Now we just need to put the two parts together. As we saw previously, 'but' can usually be symbolised as '$\wedge$'. Statement 1 can thus be symbolised as '$((P \vee S) \wedge \neg(P \wedge S))$'.

This last example shows something important. Although the TFL symbol '$\vee$' always symbolises *inclusive or*, we can symbolise an *exclusive or* in TFL. We just have to combine a few of our symbols. However, if we wanted to use a single symbol for exclusive disjunction, we could introduce one, such as $(P \veebar S)$. We won't be using this exclusive disjunction symbol, partially because it can be defined by the other existing symbols, but mainly because we don't think logical exclusive disjunction is actually that common in English.

Linguistics Corner

Many uses of 'or' in English, such as Statement 1, seem exclusive. But most linguists think that exclusivity isn't due to the connective. Instead, the exclusive restriction is from our knowledge of the world. That is, almost all exclusivity is due to the laws of nature, or social customs, or regulations, and not the laws of logic. As we are interested solely in logical validity, this physical exclusivity is not part of disjunction.

6.3 Conditional Complications

Conditionals are much more complex than we have indicated above. There are dozens of books written by philosophers, linguists, and logicians simply titled 'Conditionals', and hundreds more on the topic.

A common approach taken by logicians is to dismiss many uses of 'if' (and similar subordinating connectives) as not being real conditionals. These uses generally can't be represented by the TFL connective '$\rightarrow$'. This is because '$\rightarrow$', like all connectives in TFL, is truth functional. This means that '$(A \rightarrow B)$' only tells us that if A is true then B is also true. For instance, it says nothing about a *causal* or *temporal* connection or any *relevance* between A and B. In fact, we lose a huge amount of information when we use '$\rightarrow$' to symbolise English conditionals. Sometimes what we lose is exactly what the conditional is trying to convey, and so we can't use '$\rightarrow$'. Here are a few examples of the wide range of conditionals, which show the limited expressivity of TFL.

2. If Logic is the gold standard of thought, then Philosophy is iron pyrites.
3. If only there was a simple way to treat conditionals.
4. If I don't see you before Monday, have a great weekend.
5. If you don't mind, I think I'll leave now.
6. Even if you don't enjoy logic, you'll find it useful.
7. Since gold is a soft metal, it is easy to work.
8. If I were a cat, I'd like to chase mice.
9. Be quiet, and I'll give you a cookie.
10. If I get funding, I'll go to the conference.

Statement 2 is a metaphor. Statement 3 is a wish – note there is no consequent. Statements 4-6 are really *unconditional* statements. Statement 4 wishes you a great weekend, whether you see them before Monday or not. Similarly, Statement 5 usually isn't about whether you mind or not; it's a declaration of intent. And Statement 6 says you'll find logic useful, whether you enjoy it or not. Statement 7 is a factual conditional – as well as being a conditional, it tells us the antecedent is true. This would be symbolised in TFL as $(A \wedge (A \rightarrow B))$, which we'll later see is the same as '$(A \wedge B)$'; so it's really a conjunction. Statement 8 is a counter-factual conditional; one where the antecedent is false. This would be symbolised in TFL as $(\neg A \wedge (A \rightarrow B))$, which we'll later see is the same as '$\neg A$'. Finally, Statement 10 seems to also say that 'if I don't get funding, I won't go to the conference'. So it somehow combines two different conditionals in one.

6.4 Only Conditionals

We have already seen that 'B if A' and 'if A then B' are paraphrases of each other, and are both symbolised by '$(A \rightarrow B)$'. We might feel we've discovered a general rule that the phrase that goes at the start of the conditional is the phrase that follows the 'if'. But consider this:

11. Aroha is in Waikato only if Aroha is in Aotearoa.
12. Only if Aroha is in Waikato then Aroha is in Aotearoa.

The 'only' seems to change the meaning of Statement 11 from 'If Aroha is in Aotearoa then Aroha is in Waikato' into 'If Aroha is in Waikato then Aroha is in Aotearoa'. A common rule of thumb is to say that 'only if' swaps the antecedent and consequent order.

This rule of thumb makes 'Aroha is in Waikato only if Aroha is in Aotearoa' a paraphrase of 'Aroha is in Aotearoa if Aroha is in Waikato', which is already a paraphrase of 'if Aroha is in Waikato then Aroha is in Aotearoa'. We've swapped order twice, back to the original order.

'A only if B' *can* be symbolised as '$(A \rightarrow B)$'.
'only if A , B' *can* be symbolised as '$(B \rightarrow A)$'.

But there's a better way. 'Only' has a property that applies in TFL, in the logic we'll learn later in this book, and in linguistics generally: It's actually a form of global negation. This is more complicated to explain, but it's a much more generally applicable principle. The short version is that 'only if' negates both parts of the conditional formed by 'if'.

Under this approach, Statement 11 becomes 'Aroha is not in Waikato if Aroha is not in Aotearoa', which is a paraphrase of 'if Aroha is not in Aotearoa, then Aroha is not in Waikato'. And Statement 12 becomes 'if Aroha is not in Waikato then Aroha is not in Aotearoa'.

This method avoids the confusion of swapping order twice, and it also applies to more subtle statements that we will later want to symbolise, such as 'Only you can save the world', 'There are only two solutions to this puzzle', and 'I only felt free on the dance floor'.

'A only if B' *should* be symbolised as '$(\neg B \rightarrow \neg A)$.
'only if A , B' *should* be symbolised as '$(\neg A \rightarrow \neg B)$.

We suggest that only if you want an easy option now that you will need to discard in a month would you not use this rule instead. That is, if you have other priorities than taking the option that is easiest for this week, you should use this negation-based rule for 'only if'.

6.5 Biconditional

Biconditionals aren't a common part of English, and they can always be paraphrased away. But they are a common part of Philosophical and Mathematical writing, and are very common in Logic. So we are introducing a symbol for biconditionals for our convenience, not because standard English speakers need them.

A biconditional is two conditionals combined – an 'if' conditional, and an 'only if' conditional. Unsurprisingly, the biconditional is often written 'if and only if' in English; this is sometimes abbreviated as 'iff'.

Consider these statements:

13. If Laika has a heart then she has a kidney
14. Laika has a kidney if she has a heart
15. Laika has a kidney only if she has a heart
16. Laika has a kidney if and only if she has a heart

We will use the following symbolisation key:

H: Laika has a heart
K: Laika has a kidney

Statement 13 can be symbolised by '$(H \rightarrow K)$'. Statement 14 is a paraphrase of Statement 13, so has the same symbolisation. Statement 15 can be symbolised by '$(\neg H \rightarrow \neg K)$'. The conjunction of 14 and 15 is Statement 16, so we can symbolise it as '$((H \rightarrow K) \wedge (\neg H \rightarrow \neg K))$'. If you prefer the simpler form of symbolising 'only if', it would be '$((H \rightarrow K) \wedge (K \rightarrow H))$'.

We could treat every biconditional this way. 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. Because the biconditional occurs so often, however, we will use the symbol '$\leftrightarrow$' for it. We can then symbolise Statement 16 with the TFL formula '$(H \leftrightarrow K)$'.

An English statement can be symbolised as $\mathcal{A} \leftrightarrow \mathcal{B}$ if it can be paraphrased as 'A if and only if B'.

We often read ordinary conditionals as biconditionals. Suppose we say 'if it rains, the cricket will be cancelled tomorrow'; then we expect that if it doesn't rain, the cricket will not be cancelled. But this doesn't follow from the statement; after all snowfall, a plague of frogs, or an earthquake might also cause cricket to be cancelled. It's our knowledge of physical reality, not the logical content of the sentence, that tempts us to add this extra thought. Biconditionals are rare in nature; let's keep them that way.

You might have thought that the previous paragraph railing against biconditionals was too strong, or even just plain wrong. If so, here's an argument that you might use as an example:

> Suppose your parents told you, when you were a child: 'if you don't eat your greens, you won't get any dessert'. Now imagine that you ate your greens, but that your parents refused to give you any dessert, on the grounds that they were only committed to the *conditional*, rather than the biconditional ('you get dessert iff you eat your greens'). A tantrum would rightly ensue.

It seems reasonable on the surface to translate 'if you don't eat your greens, you won't get any dessert' as a biconditional. Perhaps you think of 'if you don't eat your greens, you won't get any dessert' as a threat, and its inverse 'if you eat your greens, you will get dessert' as a promise. Now if you think that threats always come paired with promises as a matter of logic, go ahead and use a biconditional. And if you think that your parents meant both conditionals, add both.

But here's a reason you might not want to do that: your parents' next statement could be 'if you don't sit up straight, you won't get any dessert'. Conditionals can be added to each other; you need to sit up straight and eat your greens to have a hope of dessert. But you can't do that if you treat them as biconditionals. (We must wait until a later chapter to explain this completely.)

So what are biconditionals for? We mainly use biconditionals in definitions, in giving exact conditions for something to occur, and when stating that things are identical, interchangeable or equivalent.

17. A polygon is a square iff it has four equal sides and four equal angles.
18. An animal is human iff it is a featherless biped.
19. Fire occurs iff there is oxygen, fuel, and enough heat.

Statement 17 is the sort of definition you'll see throughout mathematics and some other formal disciplines such as physics or computer science. Statement 18 was Plato's attempt at defining a human (in response, a rival philosopher Diogenes plucked a chicken and let it loose in Plato's academy). Statement 19 describes the necessary and sufficient conditions for fire – all these conditions are needed, and if they are all met, there will be fire.

6.6 Unless

The English 'unless' is another tricky connective. Consider:

20. *Unless* you wear a jacket, you will catch a cold.
21. You will catch a cold *unless* you wear a jacket.

We will use the symbolisation key:

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

Both statements are paraphrases of 'if you do not wear a jacket, then you will catch a cold'. So we could symbolise them as '$(\neg J \to D)$'. But both statements are also paraphrases of 'if you do not catch a cold, then you must have worn a jacket', which is symbolised as '$(\neg D \to J)$'. We might also claim that 'either you will wear a jacket or you will catch a cold' is a paraphrase, and symbolise them as '$(J \lor D)$'.

All three are *adequate* symbolisations. Indeed, we will shortly be able to prove that all three symbolisations are equivalent in TFL. However, we don't typically treat disjunctions and conditionals interchangeably in English. The differences between these connectives in English are not captured by the truth functions of TFL. But your intuitions as a speaker of English will cause you to make mistakes, or doubt yourself, if you don't use the paraphrase (and symbolisation) that best fits how you use 'unless'.

Most people use 'unless' as a conditional, with the statement that comes earlier in time as the antecedent. In the example above, that's '$(\neg J \to D)$'. But if in doubt, the simplest approach is to replace the word 'unless' with 'if not'.

If a statement can be paraphrased as 'Unless A, B,' then it can be symbolised as '$(\neg \mathcal{A} \to \mathcal{B})$'.

Warning: there is a trap. 'Unless' is a type of conditional, and people often use a conditional when the physical or social constraints make it seem like a biconditional. And if we treat it as a disjunction, the same physical or social constraints make it seem like an exclusive disjunction. Suppose someone says: 'I will go running unless it rains'. You might think they mean the biconditional 'I will go running iff it does not rain'. As we discussed in the section on the biconditional, don't assume that. If they go for a run in the rain, they wouldn't have lied; you would have just made a wrong assumption.

Don't treat a disjunction as exclusive, or a conditional as a biconditional, if they aren't explicitly treated that way in the text.

Practice Exercises

⋆ **Exercise A:** Symbolise each English statement in TFL, using the key.

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.

Exercise B: Symbolise each English statement in TFL, using the key.

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

1. Either Ace or Emma was murdered.
2. If Ace was murdered, then the cook did it.
3. If Emma 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. Ace was murdered if and only if Emma was not murdered.
9. The Duchess is lying, unless it was Emma who was murdered.
10. If 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!

⋆ **Exercise C:** Symbolise each English statement in TFL, using the key.

A:	Ava is an electrician.
B:	Harrison is an electrician.
C:	Ava is a firefighter.
D:	Harrison is a firefighter.
E:	Ava is satisfied with her career.
F:	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.

Exercise D: Symbolise each English statement in TFL, using the key.

T:	John Coltrane played tenor sax.
S:	John Coltrane played soprano sax.
U:	John Coltrane played tuba
R:	Miles Davis played trumpet
B:	Miles Davis played tuba

1. John Coltrane played tenor and soprano sax.
2. Neither Miles Davis nor John Coltrane played tuba.
3. John Coltrane did not play both tenor sax and tuba.
4. John Coltrane did not play tenor sax unless he also played soprano sax.
5. John Coltrane did not play tuba, but Miles Davis did.
6. Miles Davis played trumpet only if he also played tuba.
7. If Miles Davis played trumpet, then John Coltrane played at least one of these three instruments: tenor sax, soprano sax, or tuba.
8. If Coltrane played tuba then Davis played neither trumpet nor tuba.

⋆ **Exercise E:** Create a key, and symbolise each English statement 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 American 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 American embassy will be in an uproar regardless.
6. Either Alice or Bob is a spy, but not both.

Exercise F: Create a key, and symbolise each English statement 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.

⋆ **Exercise G:** For each argument, create a key and symbolise all of the statements of the argument in TFL.

1. If Anna plays in the snow in the morning, then Elsa wakes up cranky. Anna plays in the snow in the morning unless she is distracted. So if Elsa does not wake up cranky, then Anna will be distracted.
2. It will either rain or snow on Tuesday. If it rains, Xinyu will be sad. If it snows, Xinyu will be cold. Therefore, Xinyu will either be sad or cold on Tuesday.
3. If Badr 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.
4. Jane will see exactly one of her parents at her 21st party. If her mum doesn't go, her father will definitely attend; and if her dad can't make it, her mother will make an appearance. But they simply refuse to be in the same room any more.

Exercise H: For each argument, create a key and symbolise the argument as well as possible in TFL. Don't symbolise the part of the passage in italics; it is there only to provide context for the argument.

1. It is going to rain soon. I know because my leg is hurting, and my leg hurts if it's going to rain.
2. *Spider-man tries to figure out the bad guy's plan.* If Doctor Octopus gets the uranium, he will blackmail the city. I am certain of this because if Doctor Octopus gets the uranium, he can make a dirty bomb, and if he can make a dirty bomb, he will blackmail the city.
3. *A westerner tries to predict the policies of the Chinese government.* If the Chinese government cannot solve the water shortages in Beijing, they will have to move the capital. They don't want to move the capital. Therefore they must solve the water shortage. But the only way to solve the water shortage is to divert almost all the water from the Yangzi river northward. Therefore the Chinese government will go with the project to divert water from the south to the north.

Exercise I: We symbolised an *exclusive or* using '$\lor$', '$\land$', and '$\neg$'. How could you symbolise an *exclusive or* using only two connectives? What about symbolising *exclusive or* using only one connective (repeatedly)?

Part III

Truth Tables

Chapter 7
Introducing Truth Tables

7.1 Characteristic Truth Tables

Any formula of TFL is composed of stomic statement symbols, possibly combined using sentential connectives. The truth value of a complex formula depends only on the truth value of the statement symbols that it contains. In order to know the truth value of '$(D \wedge E)$', for instance, you only need to know the truth value of 'D' and the truth value of 'E', and how '$\wedge$' works. That's what makes the logic truth-functional.

We will need to define the connectives we introduced in chapter 5. We will abbreviate *True* with '1' and *False* with '0'. (But just to be clear, the truth values are *True* and *False*; truth values are not numbers!)

Negation. If $\mathcal{A}$ is true, then $\neg\mathcal{A}$ is false. If $\neg\mathcal{A}$ is true, then $\mathcal{A}$ is false. We can summarise this in the *characteristic truth table* for negation:

$\mathcal{A}$	$\neg\mathcal{A}$
1	0
0	1

Conjunction. $(\mathcal{A} \wedge \mathcal{B})$ is true if and only if $\mathcal{A}$ is true and $\mathcal{B}$ is true. Here is the characteristic truth table for conjunction:

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \wedge \mathcal{B})$
1	1	1
1	0	0
0	1	0
0	0	0

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

Disjunction. $(\mathcal{A} \vee \mathcal{B})$ is true if and only if at least one of $\mathcal{A}$ and $\mathcal{B}$ is true. Here is the characteristic truth table for disjunction:

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \vee \mathcal{B})$
1	1	1
1	0	1
0	1	1
0	0	0

Like conjunction, disjunction is symmetrical.

Conditional. Conditionals are complex and TFL treats them in a simplified manner: $(\mathcal{A} \rightarrow \mathcal{B})$ is false if and only if $\mathcal{A}$ is true and $\mathcal{B}$ is false; otherwise it is true. Here is the characteristic truth table for the conditional.

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \rightarrow \mathcal{B})$
1	1	1
1	0	0
0	1	1
0	0	1

The conditional is *asymmetrical*. You cannot swap the antecedent and consequent without changing the meaning of the formula, because $(\mathcal{A} \rightarrow \mathcal{B})$ has a very different truth table from $(\mathcal{B} \rightarrow \mathcal{A})$.

Biconditional. Since a biconditional is to be the same as the conjunction of a conditional running in each direction, the characteristic truth table for the biconditional must be:

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \leftrightarrow \mathcal{B})$
1	1	1
1	0	0
0	1	0
0	0	1

Unsurprisingly, the biconditional is symmetrical.

Exclusive Disjunction. We won't be using exclusive disjunction very much. However, it's a worthwhile exercise for you to devise your own characteristic truth table for exclusive disjunction, or for any other connective you would like to add to TFL.

7.2 Truth-functional Connectives

Let's formalise an idea we've talked about earlier.

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

Every connective in TFL is truth-functional. The truth value of a negation is uniquely determined by the truth value of the unnegated formula. The truth value of a conjunction is uniquely determined by the truth value of both conjuncts. The truth value of a disjunction is uniquely determined by the truth value of both disjuncts, and so on. To determine the truth value of some TFL formulas, we only need to know the truth value of its component formulas.

Truth-functionality is actually quite rare. Almost no connectives and operators in English are truth-functional.

For example the connectives 'although' and 'because' seem very 'logical', but aren't truth-functional. Think about how the components of each of these statements contribute to the truth of the overall statement:

1. Sara is trustworthy and she is a lawyer.
2. Sara is trustworthy because she is a lawyer.
3. Sara is trustworthy although she is a lawyer.

To be true, each sentence needs the statements 'Sara is trustworthy' and 'Sara is a lawyer' to be true. Truth is all that TFL knows about, so the TFL symbolisation of these three statements would be identical. However, you can see that they do convey very different ideas about the relationship between being trustworthy and a lawyer, so they aren't interchangeable in English. 'Although' and 'because' are not *truth-functional.*

More abstractly, we can form a new statement from any simpler statement by prefixing it with, for example: 'Audrey Hepburn thought 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. Beyoncé's middle name is Giselle.

As both these sentences are true, adding our Hepburn operator should make them both true (or both false). But while Audrey Hepburn thought that $2 + 2 = 4$, she certainly didn't think Beyoncé's middle name was Giselle. So 'Audrey Hepburn thought that. . . ' is an English operator, but it is not a *truth-functional* or logical operator.

7.3 Symbolising vs. Translating

All of the connectives of TFL are truth-functional, but more than that: they really do *nothing* except map between truth values.

When we symbolise a sentence or an argument in TFL, we ignore everything *besides* the contribution that the truth values of a component might make to the truth value of the whole. There are subtleties to our ordinary claims that far outstrip their mere truth values, such as sarcasm; poetry; tension; surprise; causation; temporality; emotional connotations; emphasis; and many others. These are important parts of everyday discourse, which are not expressible in TFL. As remarked in Chapter 5, TFL cannot capture the subtle differences between the following English sentences:

1. Dana is a logician and Dana is a nice person
2. Although Dana is a logician, Dana is a nice person
3. Dana is a logician despite being a nice person
4. Dana is a nice person, since they are a logician
5. Dana is a logician; co-incidentally they are also a nice person

All of the above sentences would be symbolised with the same TFL formula, perhaps '$(L \land N)$'.

We keep saying that we use TFL formulas to *symbolise* English statements. Many other textbooks talk about *translating* English sentences into TFL. However, a translation must preserve certain facets of meaning, and – as we have just pointed out – TFL cannot do that. This is why we will speak of *symbolising* English statements, rather than of *translating* them.

This affects how we should understand our symbolisation keys. Consider a key like:

L: Dana is a logician.
N: Dana is a nice person.

Other textbooks will understand this as a stipulation that the TFL formula 'L' should *mean* that Dana is a logician, and that the TFL formula 'N' should *mean* that Dana is a nice person, but TFL is totally unequipped to deal with *meaning*. The preceding symbolisation key is only stipulating that the TFL formula 'L' must take the same truth value as the English statement 'Dana is a logician', and that the TFL formula 'N' must take the same truth value as the English statement 'Dana is a nice person'.

We should not expect that a symbolisation of a statement's truth value would also contain all the richness of communication. Truth is only a very small (if central) element of communication.

Practice Exercises

Exercise A: Write characteristic truth tables for each of the following connectives:

1. Exclusive disjunction (Either A nor B, but not both).
2. Nor (Neither A nor B).
3. Only if (Only if A then B).
4. Another connective that you think might be interesting.

Exercise B: One major limitation on TFL connectives is our choice of having two truth values (True and False). What third truth value might you add (e.g. ‘unknown’, ‘both true and false’, ‘irrelevant’, ‘unproven’, ‘future’ (i.e. not yet true or false), ‘partially true’? Create characteristic truth tables for conjunction and negation using True, False, and your third truth value.

Chapter 8
Complete Truth Tables

We have two main uses for truth values. First, we can assign truth values to a TFL formula based on the truth value of the statement it symbolises. For example, 'B' could take the same truth value as the English sentence 'Big Ben is in London'. Second – and more interesting for logicians – we can stipulate the truth value of some of our formulas, and check if this informs us of the truth values of others. Usually we only provide values for the atomic statement symbols in TFL.

> A VALUATION in TFL is any assignment of truth values to individual TFL statement symbols.

The power of truth tables lies in the following: Each row of a truth table represents a possible valuation. The entire truth table represents all possible valuations; thus the truth table provides us with a means to calculate the truth values of complex formulas, over all possibilities. This is easiest to explain with examples.

8.1 Our First Complete Truth Table

Consider the formula '$((H \vee I) \rightarrow H)$'. There are four possible ways to assign True and False to the statement symbols 'H' and 'I' – four possible valuations – which we can represent on four rows, as follows:

H	I	$((H$	$\vee$	$I)$	$\rightarrow$	$H)$
1	1					
1	0					
0	1					
0	0					

To calculate the truth value of the entire formula '$((H \vee I) \rightarrow H)$', we first copy the truth values underneath their symbols in the formula:

H	I	$((H$	$\vee$	$I)$	$\rightarrow$	$H)$
1	1	1		1		1
1	0	1		0		1
0	1	0		1		0
0	0	0		0		0

Now consider the subformula '$(H \vee I)$'. This is a disjunction $(\mathcal{A} \vee \mathcal{B})$ with 'H' as $\mathcal{A}$ and with 'I' as $\mathcal{B}$. A disjunction is true iff one of its disjuncts is true. In this case, our disjuncts are just 'H' and 'I'.

		$(\mathcal{A}$	$\vee$	$\mathcal{B})$		
H	I	$((H$	$\vee$	$I)$	$\rightarrow$	$H)$
1	1	1	1	1		1
1	0	1	1	0		1
0	1	0	1	1		0
0	0	0	0	0		0

Tip: When we show that a column is used to to calculate another, we will often grey that column out once it's used, as we'll never need to use that column again. You could cross out, tick, or highlight them instead.

The next connective is a conditional $(\mathcal{A} \rightarrow \mathcal{B})$ with '$(H \vee I)$' as $\mathcal{A}$ and with 'H' as $\mathcal{B}$. On the first row, for example, '$(H \vee I)$' is true and 'H' is true, so '$((H \vee I) \rightarrow I)$' is true. We write '1' on the row:

			$(\mathcal{A}$		$\rightarrow$	$\mathcal{B})$
H	I	$((H$	$\vee$	$I)$	$\rightarrow$	$H)$
1	1	1	1	1	1	1
1	0	1	1	0		1
0	1	0	1	1		0
0	0	0	0	0		0

The second row is similar. In the third row, however, '$(H \vee I)$ is true and 'H' is false, so we write '0'. Finally, the antecedent of the conditional is false on the last row, so we write '1'. The completed table looks like this:

H	I	$((H$	$\vee$	$I)$	$\rightarrow$	$H)$
1	1	1	1	1	**1**	1
1	0	1	1	0	**1**	1
0	1	0	1	1	**0**	0
0	0	0	0	0	**1**	0

The conditional is the main logical operator of the formula. It is true on three rows, and false on one row.

8.2 Building Complete Truth Tables

A complete truth table has a row for every possible valuation; that is, every possible combination of *True* and *False* for the relevant statement symbols. To make sure you include every possibility, it's best to be orderly when building your truth table.

The size of the complete truth table depends on the number of different statement symbols in the table. Two rows are needed for the complete truth table for a formula that contains only one atomic statement symbol, such as the characteristic truth table for negation. Four rows are needed for the complete truth table for formulas that contain two atomic statement symbols, such as the characteristic truth tables for our binary connectives, and the truth table for '$((H \wedge I) \rightarrow H)$'.

Eight rows are needed for the complete truth table for formulas that contain three atomic statement symbols. Sixteen rows for four different symbols, 32 rows for five symbols, 64 rows for six symbols, etc., In general: A complete truth table with n atomic statement symbols has 2^n rows.

In order to write the columns of a complete truth table, begin with the right-most atomic symbol; write alternately '1' and '0'. In the next column to the left, write two '1's, then two '0's, and repeat. For the third atomic symbol, write four '1's then four '0's. This yields an eight line truth table like the one below:

M	N	P	$(M$	$\wedge$	$(N$	$\vee$	$P))$
1	1	1					
1	1	0					
1	0	1					
1	0	0					
0	1	1					
0	1	0					
0	0	1					
0	0	0					

For a 16 line truth table, the next column of statement symbols should have eight '1's followed by eight '0's. Then 16 '1's and 16 '0's, and so on.

But what if the same atomic symbol occurs many times, as in the formula '$(((C \leftrightarrow C) \rightarrow C) \wedge \neg(C \rightarrow C))$'? Only two lines are required because there is only one atomic symbol C, and so only two possibilities: C is true or false. The complete truth table for this formula is:

C	$(((C$	$\leftrightarrow$	$C)$	$\rightarrow$	$C)$	$\wedge$	$\neg$	$(C$	$\rightarrow$	$C))$
1	1	1	1	1	1	**0**	0	1	1	1
0	0	1	0	0	0	**0**	0	0	1	0

8.3 Dropping Brackets

Our formulas are starting to accumulate a lot of brackets. You might think that not all these brackets are necessary. And you'd be right. But it can be tricky to know which brackets are needed and which are not. One pair more than you need is annoying. One pair less is disastrous. Given that warning, there *are* times you can safely drop brackets.

(1) You can drop the outermost set of brackets, if they are the very first and very last symbols.

$$\begin{array}{lcl}
(A \vee B) & \Rightarrow & A \vee B \\
((A \wedge B) \rightarrow (C \vee D)) & \Rightarrow & (A \wedge B) \rightarrow (C \vee D) \\
(A \wedge B) \rightarrow (C \vee D) & \not\Rightarrow & A \wedge B) \rightarrow (C \vee D \\
\neg(A \vee B) & \not\Rightarrow & \neg A \vee B
\end{array}$$

Don't forget to add those brackets again though, if you need to negate your formula, or join it to another formula.

(2) If you have several conjuncts in a row, or several disjuncts in a row, you can drop the pairs of brackets separating them. However, be careful, and if in doubt, leave the brackets alone.

$$\begin{array}{lcl}
((A \wedge B) \wedge C) & \Rightarrow & (A \wedge B \wedge C) \\
(A \vee (B \vee C)) & \Rightarrow & (A \vee B \vee C) \\
((A \rightarrow B) \rightarrow C) & \not\Rightarrow & (A \rightarrow B \rightarrow C) \\
((A \wedge B) \vee C) & \not\Rightarrow & (A \wedge B \vee C)
\end{array}$$

You cannot drop brackets if there are several conditionals in a row. Nor can you drop brackets if there are a mixture of conjuncts and disjuncts. And even if it's permissible to drop brackets, you'll find it is sometimes useful to use them to group some of the subformulas.

You might be wondering, why only conjunctions or disjuncts? The answer is that $((A \wedge B) \wedge C)$ and $(A \wedge (B \wedge C))$ have the same truth tables:

A	B	C	$((A$	$\wedge$	$B)$	$\wedge$	$C)$	$(A$	$\wedge$	$(B$	$\wedge$	$C))$
1	1	1	1	1	1	**1**	1	1	**1**	1	1	1
1	1	0	1	1	1	**0**	0	1	**0**	1	0	0
1	0	1	1	0	0	**0**	1	1	**0**	0	0	1
1	0	0	1	0	0	**0**	0	1	**0**	0	0	0
0	1	1	0	0	1	**0**	1	0	**0**	1	1	1
0	1	0	0	0	1	**0**	0	0	**0**	1	0	0
0	0	1	0	0	0	**0**	1	0	**0**	0	0	1
0	0	0	0	0	0	**0**	0	0	**0**	0	0	0

Because these two conjunction formulas have the same truth table, the precise bracketting doesn't matter. We can then write this formula simply as: $(A \wedge B \wedge C)$. The same story holds for disjunction: $((A \vee B) \vee C)$ and $(A \vee (B \vee C))$ have the same truth tables, so we can simply write $(A \vee B \vee C)$.

But $((A \to B) \to C)$ and $(A \to (B \to C))$ have different truth tables:

A	B	C	$((A$	$\to$	$B)$	$\to$	$C)$	$(A$	$\to$	$(B$	$\to$	$C))$
1	1	1	1	1	1	**1**	1	1	**1**	1	1	1
1	1	0	1	1	1	**0**	0	1	**0**	1	0	0
1	0	1	1	0	0	**1**	1	1	**1**	0	1	1
1	0	0	1	0	0	**1**	0	1	**1**	0	1	0
0	1	1	0	1	1	**1**	1	0	**1**	1	1	1
0	1	0	0	1	1	**0**	0	0	**1**	1	0	0
0	0	1	0	1	0	**1**	1	0	**1**	0	1	1
0	0	0	0	1	0	**0**	0	0	**1**	0	1	0

Because their truth tables are different, we need to distinguish between them, so we can't simplify the formula to: $(A \to B \to C)$. Similarly $((A \wedge B) \vee C)$ and $(A \wedge (B \vee C))$ have different truth tables, so we can't write: $(A \wedge B \vee C)$.

This tension between having enough brackets, and avoiding confusing clutter, will reoccur throughout this course, as we learn more techniques and create more complicated formulas. If in doubt, use brackets.

Importantly, always make sure you have the same number of opening '('and closing ')' brackets. There are several useful tricks for this. One way is to use different sized or shaped pairs of brackets, such as

$$[\big((p \wedge q) \to q\big) \to \neg\{p \vee q\}]$$

However, we are going to use [square brackets] for a particular type of bracket in the second half of this book, so you might find that using different colours, or even numbering your brackets, might work better:

$$({}^1({}^2({}^3 p \wedge q)^3 \to q)^2 \to \neg({}^4 p \vee q)^4)^1$$

With a little practice, you'll be keeping track of your brackets, and dropping the ones you don't need, without causing yourself any confusion.

Finally, don't drop all your brackets when writing a complex formula. Writing $((A \wedge B) \wedge C)$ as $A \wedge B \wedge C$ isn't wrong, but it leaves the formula completely naked. Have *some* decency.

Practice Exercises

Exercise A: Write complete truth tables for each of the following:

1. $(A \to A)$
2. $(C \to \neg C)$
3. $((A \leftrightarrow B) \leftrightarrow \neg(A \leftrightarrow \neg B))$
4. $((A \to B) \lor (B \to A))$
5. $((A \land B) \to (B \lor A))$
6. $(\neg(A \lor B) \leftrightarrow (\neg A \land \neg B))$
7. $(((A \land B) \land \neg(A \land B)) \land C)$
8. $(((A \land B) \land C) \to B)$
9. $\neg((C \lor A) \lor B)$

Exercise B: Check all the claims made in §8.3 Dropping Brackets, i.e. show:

1. '$((A \land B) \land C)$' and '$(A \land (B \land C))$' have the same truth table
2. '$((A \lor B) \lor C)$' and '$(A \lor (B \lor C))$' have the same truth table
3. '$((A \lor B) \land C)$' and '$(A \lor (B \land C))$' have different truth tables
4. '$((A \to B) \to C)$' and '$(A \to (B \to C))$' have different truth tables

Also, check whether:

5. '$((A \leftrightarrow B) \leftrightarrow C)$' and '$(A \leftrightarrow (B \leftrightarrow C))$' have the same truth table

⋆ **Exercise C:** Write complete truth tables for the following formulas and mark the column that represents the truth values for the whole formula.

1. $\neg(S \leftrightarrow (P \to S))$
2. $\neg((X \land Y) \lor (X \lor Y))$
3. $((\neg P \lor \neg M) \leftrightarrow M)$
4. $((A \to B) \leftrightarrow (\neg B \leftrightarrow \neg A))$
5. $\neg\neg(\neg A \land \neg B)$
6. $(((D \land R) \to I) \to \neg(D \lor R))$
7. $((C \leftrightarrow (D \lor E)) \land \neg C)$
8. $(\neg(G \land (B \land H)) \leftrightarrow (G \lor \neg(B \lor H)))$
9. $(\neg((D \leftrightarrow O) \leftrightarrow A) \to (\neg D \land O))$

Exercise D: Write out each of the formulas in this set of exercises, using as few pairs of brackets as possible.

Exercise E: If you want additional practice, you can construct truth tables for any of the formulas in the exercises for the previous chapter.

Chapter 9
Semantic Notions

Now we know how to determine the truth value of any TFL formula for any possible valuation by using a truth table, we can put this to good use. We have discussed several logical notions such as validity and consistency for English statements. We can now create tests for these same notions in TFL.

9.1 Logical Truths and Falsehoods

In Chapter 3, we explained *logical truth* and *logical falsity*. Both notions have analogues in TFL. Here is the definition of logical truth for TFL:

$\mathcal{A}$ is a LOGICAL TRUTH (in TFL) iff it is true in every valuation.

We can determine whether a formula is a logical truth just by using truth tables. If the formula is true on every line of a complete truth table, then it is true in every valuation, so it is a logical truth. For example, the formula '$(H \land I) \to H$' is a logical truth, as is shown in the completed truth table:

H	I	$((H$	$\land$	$I)$	$\to$	$H)$
1	1	1	1	1	**1**	1
1	0	1	0	0	**1**	1
0	1	0	0	1	**1**	0
0	0	0	0	0	**1**	0

The column of '1's underneath the conditional shows that the TFL formula '$((H \land I) \to H)$' is a logical truth: it is true in all cases. 'H' and 'I' can be true or false in any combination, and the conditional formula still comes out true. Since we have considered all possibilities for the truth of 'H' and 'I' – since, that is, we have considered all their *valuations* – we can say that '$((H \land I) \to H)$' is true on every valuation, so is a logical truth.

A completed truth-table can also show that $(A \to B) \to C$ isn't a logical truth, because some rows of its main connective are false (shaded pink):

A	B	C	$((A$	$\to$	$B)$	$\to$	$C)$
1	1	1	1	1	1	**1**	1
1	1	0	1	1	1	**0**	0
1	0	1	1	0	0	**1**	1
1	0	0	1	0	0	**1**	0
0	1	1	0	1	1	**1**	1
0	1	0	0	1	1	**0**	0
0	0	1	0	1	0	**1**	1
0	0	0	0	1	0	**0**	0

This methods identifies all logical truths that can be captured by TFL. There are some logical truths that we cannot adequately symbolise in TFL. For example 'Every cat is a cat' *must* always be true, but the best symbolisation we can offer is an atomic symbol, and no atomic symbol is a logical truth. That's because any atomic symbol has the possibility of being true, and of being false. Still, any English statement that is symbolised by a logical truth in TFL will also be a logical truth in English.

We have a similarly restricted definition for logical falsity:

$\mathcal{A}$ is a LOGICAL FALSEHOOD iff it is false in every valuation.

We can determine whether a formula is a falsehood just by using truth tables. If the formula is false on every line of a complete truth table, then it is false on every valuation, so it is a falsehood. This formula from Chapter 8, '$((C \leftrightarrow C) \to C) \land \neg(C \to C)$', is a logical falsehood:

C	$(((C$	$\leftrightarrow$	$C)$	$\to$	$C)$	$\land$	$\neg$	$(C$	$\to$	$C))$
1	1	1	1	1	1	**0**	0	1	1	1
0	0	1	0	0	0	**0**	0	0	1	0

Finally, most statements are neither logical truths nor falsehoods:

$\mathcal{A}$ is CONTINGENT iff it is not a logical truth or falsehood.

Contingent statements are those that are useful in the world; one's that can affect our actions, because they can be either true or false, and we need to engage in observation to determine which they are. For logicians, that's just too much work.

9.2 Consistency

In Chapter 3, we said that statements are mutually consistent iff it is possible for all of them to be true at once. We can adapt this to TFL too:

> $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$ are MUTUALLY CONSISTENT (in TFL) iff there is some valuation which makes them all true.

To test for mutually consistency using complete truth tables, we look for a truth table row where all the formulas are true. For instance, the formulas '$(P \to Q)$' and '$(\neg P \land \neg Q)$' are mutually consistent:

P	Q	$(P$	$\to$	$Q)$	$(\neg$	P	$\land$	$\neg$	$Q)$
1	1	1	**1**	1	0	1	**0**	0	1
1	0	1	**0**	0	0	1	**0**	1	0
0	1	0	**1**	1	1	0	**0**	0	1
0	0	0	**1**	0	1	0	**1**	1	0

Both formulas get the value '1' (true) on the last row in the table, which shows it is possible for the formulas to be true together.

A set of formulas is MUTUALLY INCONSISTENT iff it isn't mutually consistent.

> $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$ are MUTUALLY INCONSISTENT (in TFL) iff there is no valuation which makes them all true.

We can show that '$(P \to Q)$' and '$(P \land \neg Q)$' are mutually inconsistent:

P	Q	$(P$	$\to$	$Q)$	$(P$	$\land$	$\neg$	$Q)$
1	1	1	**1**	1	1	**0**	0	1
1	0	1	**0**	0	1	**1**	1	0
0	1	0	**1**	1	0	**0**	0	1
0	0	0	**1**	0	0	**0**	1	0

To show that a set of formulas is mutually inconsistent, we must check every row, as we did for checking logical truth and falsehood.

Consistency (and inconsistency) need not apply just to pairs of formulas. In fact, it's possible to have a set of formulas, each pair of which is consistent, while the whole set is inconsistent:

P	Q	$(P$	$\lor$	$Q)$	$(\neg$	P	$\lor$	$\neg$	$Q)$	$(P$	$\leftrightarrow$	$Q)$
1	1	1	**1**	1	0	1	**0**	0	1	1	**1**	1
1	0	1	**1**	0	0	1	**1**	1	0	1	**0**	0
0	1	0	**1**	1	1	0	**1**	0	1	0	**0**	1
0	0	0	**0**	0	1	0	**1**	1	0	0	**1**	0

So be careful when you specify which set of formulas is consistent!

9.3 Validity

The validity of an argument is closely related to mutual consistency of its premises and negated conclusion:

$\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n \therefore \mathcal{C}$ is valid iff
$\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n, \neg\mathcal{C}$ are mutually inconsistent.

Restating this relationship in English, an argument is valid if and only if there is no possibility that every premise and the negated conclusion can all be true. And this holds iff the conclusion can't be false when the premises are true.

We can use a complete truth table to test for validity. In a truth table, we indicate the start of each premise with a vertical line, and the start of the conclusion with a double vertical line. For example, to test whether '$\neg L \to (J \vee L), \neg L \therefore J$' is a valid argument, we check if every row where the premises '$\neg L \to (J \vee L)$' and '$\neg L$' are true also has 'J' true:

J	L	$\neg$	L	$\to$	$(J$	$\vee$	$L)$	$\neg$	L	J
1	1	0	1	**1**	1	1	1	**0**	1	**1**
1	0	1	0	**1**	1	1	0	**1**	0	**1**
0	1	0	1	**1**	0	1	1	**0**	1	**0**
0	0	1	0	**0**	0	0	0	**1**	0	**0**

The above argument is valid, because the conclusion is true on every row where the premises are true. Note that had there been *no* rows where the premises were true, the argument would automatically be valid, because there are no rows where it could go wrong.

The test for invalidity is similar. For example, to test whether '$\neg L \to (J \vee L), \neg L \therefore \neg J$' is invalid argument, we check if there is any valuation that makes both '$\neg L \to (J \vee L)$' and '$\neg L$' true and '$\neg J$' false:

J	L	$\neg$	L	$\to$	$(J$	$\vee$	$L)$	$\neg$	L	$\neg$	J
1	1	0	1	**1**	1	1	1	**0**	1	**0**	1
1	0	1	0	**1**	1	1	0	**1**	0	**0**	1
0	1	0	1	**1**	0	1	1	**0**	1	**1**	0
0	0	1	0	**0**	0	0	0	**1**	0	**1**	0

Only the second row has both '$\neg L \to (J \vee L)$' and '$\neg L$' true, and on that row '$\neg J$' is false. We have found one row where it failed (no matter how many rows it might pass the test for validity. So '$\neg L \to (J \vee L), \neg L \therefore \neg J$' is invalid.

9.4 Equivalent and Contradictory

Only pairs of formulas can be Equivalent or Contradictory. We start with Equivalence in TFL:

> $\mathcal{A}$ and $\mathcal{B}$ are EQUIVALENT (in TFL) iff their truth values agree for each valuation.

We have already made use of this notion in §8.3 when we showed that '$(A \wedge B) \wedge C$' and '$A \wedge (B \wedge C)$' are equivalent. To test for equivalence using truth tables, we complete the truth tables and check the columns are identical. For example, are the formulas '$\neg(P \vee Q)$' and '$(\neg P \wedge \neg Q)$' equivalent?

P	Q	$\neg$	$(P$	$\vee$	$Q)$	$(\neg$	P	$\wedge$	$\neg$	$Q)$
1	1	**0**	1	1	1	0	1	**0**	0	1
1	0	**0**	1	1	0	0	1	**0**	1	0
0	1	**0**	0	1	1	1	0	**0**	0	1
0	0	**1**	0	0	0	1	0	**1**	1	0

Both formulas are false on the first three rows, and true on the final row. Since they have identical truth values on every row, the two formulas are equivalent.

The definition of Contradictory formulas in TFL is similar:

> $\mathcal{A}$ and $\mathcal{B}$ are CONTRADICTORY (in TFL) iff their truth values differ for each valuation.

We can test if the formulas '$(P \to Q)$' and '$(P \wedge \neg Q)$' are contradictory:

P	Q	$(P$	$\to$	$Q)$	$(P$	$\wedge$	$\neg$	$Q)$
1	1	1	**1**	1	1	**0**	0	1
1	0	1	**0**	0	1	**1**	1	0
0	1	0	**1**	1	0	**0**	0	1
0	0	0	**1**	0	0	**0**	1	0

The first formula is false only on the second row, while the second formula is true on the second row, and false elsewhere. Since they have different truth values on every row, the two formulas are contradictory.

Being equivalent and contradictory are negations of each other, in the sense that $\mathcal{A}$ and $\mathcal{B}$ are equivalent iff $\mathcal{A}$ and $\neg\mathcal{B}$ are contradictory.

9.5 Relationships between Logical Notions

We can observe a number of relationships between our logical terms.

1. The negation of a logical truth is a logical falsehood.
2. The negation of a logical falsehood is a logical truth.

3. The conjunction of inconsistent formulas is a logical falsehood.
4. A single formula is inconsistent iff it is a logical falsehood.
5. A set of formulas is inconsistent iff their conjunction is a logical falsehood.
6. A formula's negation is inconsistent iff it is a logical truth.
7. A set of formulas is inconsistent iff the disjunction of their negations is a logical truth.

8. An argument is valid iff the conditional from the conjunction of its premises to its conclusion is a logical truth.
9. An argument is valid iff its premises plus its negated conclusion is inconsistent.
10. An argument is invalid iff its premises plus its negated conclusion is consistent.
11. Any argument whose conclusion is a logical truth is valid.
12. Any argument whose premises are inconsistent is valid.

13. All logical truths are equivalent to each other.
14. All logical falsehoods are equivalent to each other.
15. Any logical truth and falsehood are contradictory.
16. A pair of formulas is equivalent iff one paired with the negation of the other is contradictory.
17. A pair of formulas is equivalent iff a formula joining them with a biconditional is a logical truth.
18. A pair of formulas is equivalent iff each is mutually inconsistent with the other's negation.
19. A pair of formulas is contradictory iff a formula joining them with a biconditional is a logical falsehood.
20. A pair of formulas is contradictory iff a formula joining them with an exclusive disjunction is a logical truth.
21. A pair of formulas is contradictory iff they are mutually inconsistent and so are their negations.
22. A pair of formulas is equivalent iff the arguments from each to the other are both valid.

9.6 Some Challenges with TFL

We can now test for validity of arguments in TFL! Sadly, this means that we can also describe some of TFL's limitations. We will illustrate some of these using three examples.

(1) The expressive paucity of TFL affects our analysis of argument validity. Consider the argument:

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

This argument is valid in English. But as there are no logical connectives that TFL recognises, any symbolisation of this argument in TFL would be of the form '$A \therefore B$', which is not formally valid:

A	B	A	B
1	1	**1**	**1**
1	0	**1**	**0**
0	1	**1**	**1**
0	0	**0**	**0**

On the second row the premise is true and the conclusion is false. So TFL hasn't preserved whatever made the argument valid. The logical link between the English premise and conclusion is that both are about the number of legs that Daisy has. But TFL doesn't recognise the subject of a sentence. Later in this book we will introduce a logic that can represent this link, and show that this argument is valid in that logic.

(2) When reasoning with vague concepts, using binary categories such as truth/falsity imposes substantial limitations. Consider the statement:

2. Jan is neither completely bald nor completely not-bald.

A symbolisation of this statement in TFL would be roughly like '$\neg B \wedge \neg\neg B$'. This is a contradiction:

B	$\neg$	B	$\wedge$	$\neg$	$\neg$	B
1	0	1	**0**	1	0	1
0	1	0	**0**	0	1	0

But Statement 2 doesn't seem to be a contradiction; we might say something like 'Jan isn't really bald, but he's also not really not-bald. Jan is somewhere in-between.' Or 'Jan is kinda bald', or 'Jan is bald-ish'.

(3) Conditionals aren't represented adequately in TFL. For example:

P1 If God doesn't exist, then it is not the case that if I pray, then God answers my prayers.
P2 I don't pray.
∴ God exists.

We will use the following symbolisation key:

A: God answers my prayers.
G: God exists.
P: I pray.

Symbolising the argument in TFL, we obtain:

P1 $\neg G \to \neg(P \to A)$
P2 $\neg P$
∴ G

But this argument is valid! Look at the truth-tables:

A	G	P	$\neg$	G	$\to$	$\neg$	$(P$	$\to$	$A)$	$\neg$	P	G
1	1	1	0	1	**1**	0	1	1	1	**0**	1	**1**
1	1	0	0	1	**1**	0	0	1	1	**1**	0	**1**
1	0	1	1	0	**0**	0	1	1	1	**0**	1	**0**
1	0	0	1	0	**0**	0	0	1	1	**1**	0	**0**
0	1	1	0	1	**1**	1	1	0	0	**0**	1	**1**
0	1	0	0	1	**1**	0	0	1	0	**1**	0	**1**
0	0	1	1	0	**1**	1	1	0	0	**0**	1	**0**
0	0	0	1	0	**0**	0	0	1	0	**1**	0	**0**

All rows in which both premises are true (the second and sixth row) are rows in which the conclusion is also true. So the argument is valid. But the premises could be true, and in that case, the conclusion must also be true. Hence, I can prove that God exists simply by not praying.

That escalated quickly!

In different ways, these examples highlight some of the limits of working with a language (like TFL) that can *only* handle truth-functional connectives. Moreover, these limits give rise to some interesting questions in philosophical logic. The case of Jan's quasi-baldness raises the general question of how logic should deal with *vague* discourse. The case of the existence of God raises the question of how logic should deal with negated conditionals such as 'it is not the case that if I pray, then God answers my prayers.' Both are discussed in PHIL 216. The expressive paucity of TFL is addressed in the second half of this book.

Practice Exercises

Exercise A: Revisit your answers to Chapter 8 Exercise **A**. Determine which formulas were tautologies (logical truths), which were contradictions (logical falsehoods), and which were contingent.

$\star$ **Exercise B:** Use truth tables to determine whether these formulas are mutually consistent, or mutually inconsistent:

1. $A \to A$, $\neg A \to \neg A$, $A \land A$, $A \lor A$
2. $A \lor B$, $A \to C$, $B \to C$
3. $B \land (C \lor A)$, $A \to B$, $\neg(B \lor C)$
4. $A \leftrightarrow (B \lor C)$, $C \to \neg A$, $A \to \neg B$

$\star$ **Exercise C:** Use truth tables to determine whether each argument is valid or invalid.

1. $A \to A. \therefore A$
2. $A \to (A \land \neg A). \therefore \neg A$
3. $A \lor (B \to A). \therefore \neg A \to \neg B$
4. $A \lor B, B \lor C, \neg A. \therefore B \land C$
5. $(B \land A) \to C, (C \land A) \to B. \therefore (C \land B) \to A$

Exercise D: Determine whether each formula is a logical truth, a logical falsehood, or a contingent formula, using a complete truth table.

1. $\neg B \land B$
2. $\neg D \lor D$
3. $(A \land B) \lor (B \land A)$
4. $\neg[A \to (B \to A)]$
5. $A \leftrightarrow [A \to (B \land \neg B)]$
6. $[(A \land B) \leftrightarrow B] \to (A \to B)$

Exercise E: Determine whether each pair of formulas are logically equivalent using complete truth tables.

1. A and $\neg A$
2. $A \land \neg A$ and $\neg B \leftrightarrow B$
3. $[(A \lor B) \lor C]$ and $[A \lor (B \lor C)]$
4. $A \lor (B \land C)$ and $(A \lor B) \land (A \lor C)$
5. $[A \land (A \lor B)] \to B$ and $A \to B$

Exercise F: Determine whether each pair of formulas are logically equivalent using complete truth tables.

1. $A \to A$ and $A \leftrightarrow A$
2. $\neg(A \to B)$ and $\neg A \to \neg B$
3. $A \lor B$ and $\neg A \to B$
4. $(A \to B) \to C$ and $A \to (B \to C)$
5. $A \leftrightarrow (B \leftrightarrow C)$ and $A \land (B \land C)$

Exercise G: Determine whether each list of formulas is mutually consistent or mutually inconsistent using a complete truth table.

1. $A \land \neg B$, $\neg(A \to B)$, $B \to A$
2. $A \lor B$, $A \to \neg A$, $B \to \neg B$
3. $\neg(\neg A \lor B)$, $A \to \neg C$, $A \to (B \to C)$
4. $A \to B$, $A \land \neg B$
5. $A \to (B \to C)$, $(A \to B) \to C$, $A \to C$

Exercise H: Determine whether each collection of formulas is mutually consistent or mutually inconsistent, using a complete truth table.

1. $\neg B$, $A \to B$, A
2. $\neg(A \lor B)$, $A \leftrightarrow B$, $B \to A$
3. $A \lor B$, $\neg B$, $\neg B \to \neg A$
4. $A \leftrightarrow B$, $\neg B \lor \neg A$, $A \to B$
5. $(A \lor B) \lor C$, $\neg A \lor \neg B$, $\neg C \lor \neg B$

Exercise I: Determine whether each argument is valid or invalid, using a complete truth table.

1. $A \to B$, $B . \because A$
2. $A \leftrightarrow B$, $B \leftrightarrow C . \because A \leftrightarrow C$
3. $A \to B$, $A \to C . \because B \to C$
4. $A \to B$, $B \to A . \because A \leftrightarrow B$

Exercise J: Determine whether each argument is valid or invalid, using a complete truth table.

1. $A \vee \big[A \to (A \leftrightarrow A)\big] \therefore A$
2. $A \vee B$, $B \vee C$, $\neg B \therefore A \wedge C$
3. $A \to B$, $\neg A \therefore \neg B$
4. A, $B \therefore \neg(A \to \neg B)$
5. $\neg(A \wedge B)$, $A \vee B$, $A \leftrightarrow B \therefore C$

⋆ **Exercise K:** Give reasons for your answers to these questions.

1. Suppose that $\mathcal{A}$ and $\mathcal{B}$ are logically equivalent. What can you say about $\mathcal{A} \leftrightarrow \mathcal{B}$?
2. Suppose that $(\mathcal{A} \wedge \mathcal{B}) \to \mathcal{C}$ is neither a logical truth nor a contradiction. What can you say about whether $\mathcal{A}, \mathcal{B} \therefore \mathcal{C}$ is valid?
3. Suppose that $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ are mutually inconsistent. What can you say about $(\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{C})$?
4. Suppose that $\mathcal{A}$ is a contradiction. What can you say about $\mathcal{A}, \mathcal{B} \therefore \mathcal{C}$?
5. Suppose that $\mathcal{C}$ is a logical truth. What can you say about $\mathcal{A}, \mathcal{B} \therefore \mathcal{C}$?
6. Suppose that $\mathcal{A}$ and $\mathcal{B}$ are logically equivalent. What can you say about $(\mathcal{A} \vee \mathcal{B})$?
7. Suppose that $\mathcal{A}$ and $\mathcal{B}$ are *not* logically equivalent. What can you say about $(\mathcal{A} \vee \mathcal{B})$?

Exercise L: Consider the following possible principle:

- Suppose $\mathcal{A}$ and $\mathcal{B}$ are logically equivalent. The validity of any argument containing $\mathcal{A}$ would be unaffected, if we replaced $\mathcal{A}$ with $\mathcal{B}$.

Is this principle correct? Does it matter if we replace a premise or the conclusion? What about replacing a subformula? Explain your answer.

Chapter 10
Partial Truth Tables

10.1 Going Backwards to Move Forwards

When we use truth tables to test for mutual consistency, we check for a good row – one where all the formulas are true. Similarly when testing for validity we check for a *bad* row – one where the premises are all true and the conclusion is false. To test logical truth we check for a bad row – one where the formula is false. To test logical falsity we check for a bad row – one where the formula is true. And finally to test for equivalence, we test for a bad row – one where the formulas have different truth values.

Since each test for a semantic notion *only* looks for a single row, we could try to create a partial truth table with only the valuation we are seeking. This should save a lot fo work, particualry for long truth tables.

Here's our general approach for testing: we will assume we've found the row we want, and work backwards to identify what row we are on. If we get contradictory information, the row is impossible. If not, we've described our row. Either way, we'll have a result for our test. All we need to do is use our truth tables backwards:

1. If a negated formula is true, the formula is false.
2. If a negated formula is false, the formula is true.
3. If a conjunction is true, both its conjuncts are true.
4. If a conjunction is false, at least one of its conjuncts is false.
5. If a disjunction is false, both its disjuncts are false.
6. If a disjunction is true, at least one of its disjuncts is true.
7. If a conditional is false, its antecedent is true and consequent false.
8. If a conditional is true, its antecedent is false or consequent true.
9. If a biconditional is true, its subformulas have the same truth value.
10. If a biconditional is false, its subformulas have different truth values.

10.2 Testing our Logical Notions

Consistency We will use this approach to test if the set of formulas $\{\ (P \wedge Q), (P \to R)\ \}$ is mutually consistent. If the formulas are consistent, there is a valuation where they are both true:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
				1			1	

We first analyse the first formula. As $(P \wedge Q)$ is true, so are P and Q:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
			1	1	1		1	

We record this information in the valuation for P and Q:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
1	**1**		1	1	1		1	

Next, we consider the second formula, and we start by copying the information from the valuation, in this case that P is true:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
1	**1**		1	1	1	1	1	

As $(P \to R)$ and P are true, R must be true as well:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
1	**1**		1	1	1	1	1	1

Finally, we record the information in the valuation for R:

P	Q	R	$(P$	$\wedge$	$Q)$	$(P$	$\to$	$R)$
1	**1**	**1**	1	1	1	1	1	1

We've found the truth values for all the subformulas and atomic symbols. There has been no contradictory information, and we've identified a valuation where the formulas are true, so they are mutually consistent.

By working backwards from the values requires for the desired row, we can fill in the rest of the truth values, and then check if this row is possible. While each step requires a little more thought, the ultimate process is faster than the laborious task of filling in a large complete truth table. And, with the number of steps drastically reduced, the chance of a silly error can be reduced with practice.

Now let's use the same method on another set of formulas:

$$\{ (P \wedge Q), (\neg P \vee \neg Q) \}$$

If the formulas are consistent, there is a valuation where they are both true:

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
			1				1		

We start with the first formula. As $(P \wedge Q)$ is true, so are P and Q:

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
		1	1	1			1		

We record this information in the valuation for P and Q:

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
1	**1**	1	1	1			1		

We can copy the values of P and Q in the second formula:

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
1	**1**	1	1	1		1	1		1

We can now find $\neg P$ and $\neg Q$:

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
1	**1**	1	1	1	0	1	1	0	1

But as$\neg P$ and $\neg Q$ are false, so is $(\neg P \vee \neg Q)$. But we already made that true. This is impossible!

P	Q	$(P$	$\wedge$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$
1	**1**	1	1	1	0	1	**1/0**	0	1

We have tried to find an example of a row where the formulas are all true, that is, proof of the claim that the formulas are mutually consistent. However, our attempt has not produced a possible valuation. We can then confidentially say that there is no row where all the premises are true, and so the set if mutually inconsistent.

Contradiction. Showing that a formula is a contradiction requires us to show that there is no valuation which makes the formula true. However, to show that a formula is *not* a contradiction, we only need find a single valuation which makes the formula true. The formula $((P \to Q) \to P) \land \neg P$ seems like it might be contradictory. Let's try to make it true:

P	Q	$((P$	$\to$	$Q)$	$\to$	$P)$	$\land$	$\neg$	P
							1		

As the conjunction is true, so are $(P \to Q) \to P$ and $\neg P$:

P	Q	$((P$	$\to$	$Q)$	$\to$	$P)$	$\land$	$\neg$	P
					1		1	1	

Since $\neg P$ is true, P must be false, which we record in the valuation and copy over under each occurence of P:

P	Q	$((P$	$\to$	$Q)$	$\to$	$P)$	$\land$	$\neg$	P
0		0			1	0	1	1	0

As $(P \to Q) \to P$ is true, and P is false, $(P \to Q)$ must be false:

P	Q	$((P$	$\to$	$Q)$	$\to$	$P)$	$\land$	$\neg$	P
0		0	0		1	0	1	1	0

Now, $(P \to Q)$ is false, as is P. But this isn't possible; a false antecedent always gives a true conditional:

P	Q	$((P$	$\to$	$Q)$	$\to$	$P)$	$\land$	$\neg$	P
0		0	**1/0**		1	0	1	1	0

So there's no valuation where the formula is true; it's a contradiction.

The formula $(P \to Q) \land (P \to \neg Q)$ also seems contradictory. Let's try to make it true. As it's a conjunction, its conjuncts will also be true:

P	Q	$(P$	$\to$	$Q)$	$\land$	$(P$	$\to$	$\neg$	$Q)$
			1		1		1		

But there are lots of valuations where each of these formulas can be true. We will have to try several valuations; let's start with those for P:

P	Q	$(P$	$\to$	$Q)$	$\land$	$(P$	$\to$	$\neg$	$Q)$
1		1	1		1	1	1		
0		0	1		1	0	1		

On the first row, as P and $P \to Q$ are true, Q must be true:

P	Q	$(P$	$\to$	$Q)$	$\wedge$	$(P$	$\to$	$\neg$	$Q)$
1	1	1	1	1	1	1	1		1
0		0	1		1	0	1		

If Q is true, then $\neg Q$ must be false:

P	Q	$(P$	$\to$	$Q)$	$\wedge$	$(P$	$\to$	$\neg$	$Q)$
1	**1**	1	1	1	1	1	1	0	1
0		0	1		1	0	1		

Now we have a conflict:

P	Q	$(P$	$\to$	$Q)$	$\wedge$	$(P$	$\to$	$\neg$	$Q)$
1	**1**	1	1	1	1	1	**1/0**	0	1
0		0	1		1	0	1		

So we discard the first attempt at a valuation as it is impossible:

P	Q	$(P$	$\to$	$Q)$	$\wedge$	$(P$	$\to$	$\neg$	$Q)$
0		0	1		1	0	1		

For the remaining valuation, notice that both conjuncts are conditionals with a false antecedent, which is always true in TFL. We are thus free to pick any value for Q. Let's make it false:

P	Q	$(P$	$\to$	$Q)$	$\wedge$	$(P$	$\to$	$\neg$	$Q)$
0	**0**	0	1	0	1	0	1	1	0

We've found truth values for all the subformulas and atomic symbols. There has been no contradictory information, and we've identified a valuation where the formula is true, so it is not contradictory.

Logic Corner

When there is only one truth value for the subformulas that gives the desired truth value for the overall formula (such as a true conjunction), this process is straight-forward. No choices need to be made. When there are several options (such as a true disjunction), you need to list all the options, and the advantages of this method start to fade away. Either subtle strategy (or computational brute force) starts to play an important role, or we can revert to using complete truth tables.

Tautology. Showing that a formula is a logical truth requires us to show that there is no valuation which makes the formula false. Thus, to show that a formula is *not* a logical truth, we only need find a single valuation which makes the formula false. Let's test the formula $(P \to Q) \lor (Q \to P)$:

P	Q	$(P$	$\to$	$Q)$	$\lor$	$(Q$	$\to$	$P)$
					0			

If a disjunction is false, so are both its disjuncts:

P	Q	$(P$	$\to$	$Q)$	$\lor$	$(Q$	$\to$	$P)$
			0		0		0	

Each disjunct is a conditional, and a false conditional has a true antecedent and false consequent:

P	Q	$(P$	$\to$	$Q)$	$\lor$	$(Q$	$\to$	$P)$
		1	0	0	0	1	0	0

This means that P (and Q) is both true and false:

P	Q	$(P$	$\to$	$Q)$	$\lor$	$(Q$	$\to$	$P)$
1/0	**1/0**	1	0	0	0	1	0	0

This is an impossible valuation. There is thus no valuation where the formula is false; it's a tautology.

Let's try a different formula. Is $(P \to Q) \to (P \lor Q)$ a tautology?

P	Q	$(P$	$\to$	$Q)$	$\to$	$(P$	$\lor$	$Q)$
					0			

A false conditional has a true antecedent and false consequent:

P	Q	$(P$	$\to$	$Q)$	$\to$	$(P$	$\lor$	$Q)$
			1		0		0	

If a disjunction is false, so are both its disjuncts:

P	Q	$(P$	$\to$	$Q)$	$\to$	$(P$	$\lor$	$Q)$
			1		0	0	0	0

We record this information in the valuation for P and Q and compute the rest of the table:

P	Q	$(P$	$\to$	$Q)$	$\to$	$(P$	$\lor$	$Q)$
0	**0**	0	1	0	0	0	0	0

We've found the truth values for all the subformulas and atomic symbols. There has been no contradictory information, and we've identified a valuation where the formula is false, so it is not a tautology.

Equivalence. Showing that two formulas are equivalent requires that we show there *is no* row where the formulas have different truth values. To show that two formulas are *not* equivalent, we need to show that there *is* a row where they have different truth values. Either way, we need to check if the first formula is true and second is false, or *vice versa*. Let's test the formulas $(P \wedge \neg Q)$ and $\neg(\neg P \vee Q)$:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
			0			1				
			1			0				

We'll start with the second formula. If a negation is true, the subformula is false, and *vice versa*:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
			0			1			0	
			1			0			1	

Now we work row by row. In the first row, we have a false conjunction and a false disjunction. The latter is more informative, because a false disjunction tells us that each disjunct is false:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
			0			1	0		0	0
			1			0			1	

This tells us that P is true (because $\neg P$ is false) and that Q is false. We record this in the valuation and compute the rest of the row, if possible:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
1	**0**	1	0	1	0	1	0	1	0	0
			1			0			1	

Unfortunately, that gives an impossible answer for the conjunction:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
1	**0**	1	1/0	1	0	1	0	1	0	0
			1			0			1	

This attempt at finding a valuation has failed, so we try the second row:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
			1			0			1	

This time, we start with the first formula, a true conjunction, which tells us that P and $\neg Q$ must be true:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
1		1	1	1		0			1	

Since $\neg Q$ is true, Q must be false. We record this in the valuation, copy the values over to the second formula and compute as much as we can:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
1	**0**	1	1	1	0	0	0	1	1	0

Now the disjunction has an impossible answer:

P	Q	$(P$	$\wedge$	$\neg$	$Q)$	$\neg$	$(\neg$	P	$\vee$	$Q)$
1	**0**	1	1	1	0	0	0	1	1/0	0

We are unable to find a valuation that makes the two formulas have different truth values. Therefore, the formulas are equivalent.

Validity. To show an argument is invalid argument requires that we find a valuation which makes all of the premises true and the conclusion false. So to show that an argument is valid we show there is no valuation which makes all of the premises true and the conclusion false. Consider this argument:

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

We look for a valuation where $\neg L \to (J \vee L)$ and $\neg L$ are true and J false:

J	L	$\neg$	L	$\to$	$(J$	$\vee$	$L)$	$\neg$	L	J
				1				1		0

If $\neg L$ is true, L is false. We know J is false. We record this information in the valuation and copy the values over to the first formula:

J	L	$\neg$	L	$\to$	$(J$	$\vee$	$L)$	$\neg$	L	J
0	**0**	0		1	0		0	1	0	0

Next we compute as much as we can:

J	L	$\neg$	L	$\to$	$(J$	$\vee$	$L)$	$\neg$	L	J
0	**0**	0	1	1/0	0	0	0	1	0	0

We've reached an impossible situation, so there's no counter-example; this means that the argument is valid.

We will test one last argument:

$$A,\ (\neg A \to \neg B)\ \therefore\ B$$

As before, we assign the truth values required for a counter-example:

A	B	A	$(\neg$	A	$\to$	$\neg$	$B)$	B
		1			1			0

We immediately get valuations for A and B, which we record in the valuation, and copy over in the second formula:

A	B	A	$(\neg$	A	$\to$	$\neg$	$B)$	B
1	**0**	1		1	1		0	0

We compute further, looking for conflict:

A	B	A	$(\neg$	A	$\to$	$\neg$	$B)$	B
1	**0**	1	0	1	1	1	0	0

But everything calculates fine. We've found truth values for all the subformulas and atomic symbols. There has been no contradictory information, and we've identified a valuation where the premises are true and conclusion false. Therefore, the argument is invalid.

Every logical notion can be tested for (perhaps the trickiest is contingency, which requires that a formula not be a logical truth nor a logical falsehood). While the process can be messy, the essential points that we hope to have established are:

(1) You can use truth tables 'backwards' to derive the truth value of simpler formulas from more complex ones.

(2) By searching for a single row that serves as a counter-example (or a confirming instance, for consistency), you can avoid the time-consuming process of filling in a complete truth table. For the small examples we've given so far, this is not really a savings, but as formulas get more complicated, the savings can become substantial.

(3) When there's more than one option to consider, this method doesn't work as well. For complex cases, it can be worse than a complete truth table.

We still value the key insights of (1) and (2), but would like to avoid the problems with (3). To do this, we will have to create a new proof method from scratch. This new method, Truth Trees, is the topic of the next Part of this textbook. Until then, practice the reasoning-backwards mindset, and become confident in how to use the definition of a logical notion to know which row you are looking for in your partial truth table.

Practice Exercises

⋆ **Exercise A:** Use partial truth tables to test for logical equivalence:

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

Exercise B: Use partial truth tables to test for a tautology or contradiction.

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. $(B \wedge D) \leftrightarrow (A \leftrightarrow (A \vee C))$

Exercise C: Use partial truth tables to test for mutual consistency:

1. $A \wedge B$, $C \to \neg B$, C
2. $A \to B$, $B \to C$, A, $\neg C$
3. $A \vee B$, $B \vee C$, $C \to \neg A$
4. A, B, C, $\neg D$, $\neg E$, 0
5. $A \wedge (B \vee C)$, $\neg(A \wedge C)$, $\neg(B \wedge C)$
6. $A \to B$, $B \to C$, $\neg(A \to C)$

⋆ **Exercise D:** Use partial truth tables to test for validity:

1. $A \vee \big(A \to (A \leftrightarrow A)\big) \therefore A$
2. $A \leftrightarrow \neg(B \leftrightarrow A) \therefore A$
3. $A \to B, B \therefore A$
4. $A \vee B, B \vee C, \neg B \therefore A \wedge C$
5. $A \leftrightarrow B, B \leftrightarrow C \therefore A \leftrightarrow C$

Exercise E: Use partial truth tables to test for a tautology or contradiction.

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

Exercise F: Use partial truth tables to test for a tautology or contradiction.

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

Exercise G: Use complete truth tables to test for logical equivalence. Then attempt this exercise again using partial truth tables.

1. A and $A \vee A$
2. A and $A \wedge A$
3. $A \vee \neg B$ and $A \rightarrow B$
4. $(A \rightarrow B)$ and $(\neg B \rightarrow \neg A)$
5. $\neg(A \wedge B)$ and $\neg A \vee \neg B$
6. $((U \rightarrow (X \vee X)) \vee U)$ and $\neg(X \wedge (X \wedge U))$
7. $((C \wedge (N \leftrightarrow C)) \leftrightarrow C)$ and $(\neg\neg\neg N \rightarrow C)$
8. $((A \vee B) \wedge C)$ and $(A \vee (B \wedge C))$
9. $((L \wedge C) \wedge I)$ and $L \vee C$

Exercise H: Test for mutual consistency. Use either a complete or partial truth table.

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

Exercise I: Test for validity. Use either a complete or partial truth table.

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

Exercise J: Test for validity. Use either a complete or partial truth table.

1. $A \leftrightarrow \neg(B \leftrightarrow A) \therefore A$
2. $A \vee B$, $B \vee C$, $\neg A \therefore B \wedge C$
3. $A \to C$, $E \to (D \vee B)$, $B \to \neg D \therefore (A \vee C) \vee (B \to (E \wedge D))$
4. $A \vee B$, $C \to A$, $C \to B \therefore A \to (B \to C)$
5. $A \to B$, $\neg B \vee A \therefore A \leftrightarrow B$

Part IV

TFL Truth Trees

Chapter 11
Truth Trees

11.1 Truth Tables aren't Enough

We've learned two ways of using truth tables to test for consistency, equivalence, validity, etc. Both of them have their uses and their drawbacks. A complete truth table always gives the right answer by following a mechanical process. But it can be long and tedious; and the complexity and length increases exponentially with the number of variables. A 2 variable table only has 4 rows, but a 20 variable table has over 1 million.

A partial truth table, on the other hand, requires a little bit of cunning to see which step should be taken next, and to spot contradictions. And it can be difficult to tell which steps were done in which order, unless you make many copies of your truth table.

So our first method takes too long for complex formulas, while our second method is too unstructured both when creating and reviewing. In this part, we will introduce a third method that is intrinsically more complex than either, but does not grow as fast as complete truth tables, and is more structured, rule-driven and checkable than partial truth tables. It will also work for the logic we introduce in the second half of this book, unlike truth tables.

The structures produced by this new method are called TRUTH TREES, because of the way the diagram of formulas can 'branch' out to represent possibilities.

11.2 Mutual Consistency

The method of trees is most directly a way to test for the consistency of a set of TFL formulas. Since all our other logical notions can be defined in terms of consistency (or lack of consistency), it indirectly gives a way to test for equivalence, validity, and so forth. For example, you may recall that one way to test if an argument is valid, is to test if the premises and the negation of the conclusion are mutually consistent – that is, if there is a counter-example, and thus the argument is invalid.

The basic idea of the tree method is that we'll write down the formulas we are testing for consistency, and then we'll break these formulas down into their component subformulas, like with did with our partial truth tables, in order to see whether it's possible to find a valuation where they are all true.

Our first Truth Tree We will begin by working through an example that illustrates the general method, then come back to give precise rules for the tree method. We want to know whether the following formulas are all mutually consistent in TFL:

$$A \wedge B, \neg(C \vee D), (\neg B \vee C) \vee E, \neg E$$

We begin by writing down the formulas we will test for consistency:

1.	$A \wedge B$	Root
2.	$\neg(C \vee D)$	Root
3.	$(\neg B \vee C) \vee E$	Root
4.	$\neg E$	Root

The formulas we write down at the beginning of the tree are called the ROOT. Trees are designed to show whether the root is consistent, and if so, provide a valuation. We will assume the formulas in the root are all true. We then write down formulas which our previous formulas tell us must also be true, and repeat this process until we have found the information we need.

Consider line (1), $A \wedge B$. For this conjunction to be true, both conjuncts must be true. The truth of each conjunct follows from what is already written down. We add them to the tree:

1.	$A \wedge B$ ✓	Root
2.	$\neg(C \vee D)$	Root
3.	$(\neg B \vee C) \vee E$	Root
4.	$\neg E$	Root
5.	A	$1 \wedge$
6.	B	$1 \wedge$

When we add lines (5) and (6), we note that they came from line (1), and that we considered conjunction ($\wedge$) to produce them. This is useful for you when revising your work, and to help us to understand what you thought you were doing when we see some unexpected formulas in the truth tree. We also add a check mark on (1) once we've decomposed it, to avoid repeating our actions.

Now consider line (2) $\neg(C \vee D)$. This is a negated disjunction. A negated disjunction is true when a disjunction is false. Disjunctions are false iff *both* disjuncts are false. We only write down formulas that are true, so if a formula is false, we write down its negation, which will be true. So we include new lines for $\neg C$ and $\neg D$, adding a check mark on line (2):

1.	$A \wedge B$ ✓	Root
2.	$\neg(C \vee D)$ ✓	Root
3.	$(\neg B \vee C) \vee E$	Root
4.	$\neg E$	Root
5.	A	1 $\wedge$
6.	B	1 $\wedge$
7.	$\neg C$	2 $\neg\vee$
8.	$\neg D$	2 $\neg\vee$

Line (3) is a disjunction. Unlike the previous two cases, it doesn't tell us what *must* be the case; it says that (at least) one of the disjuncts must be true. We represent this by *branching* our tree :

1.	$A \wedge B$ ✓		Root
2.	$\neg(C \vee D)$ ✓		Root
3.	$(\neg B \vee C) \vee E$ ✓		Root
4.	$\neg E$		Root
5.	A		1 $\wedge$
6.	B		1 $\wedge$
7.	$\neg C$		2 $\neg\vee$
8.	$\neg D$		2 $\neg\vee$
9.	$\neg B \vee C$	E	3 $\vee$

These two branches represent different potential valuations that we are considering. We always read a branch from the bottom up. So both branches contain all the root formulas. Another way of thinking about branches in a truth tree is that we have several lists, and we've been too lazy to write the formulas in common more than once.

So now we have to consider our branches in turn. We start by examining the right branch, just because it's simpler – it just contains an atomic formula,

E. Notice, however, that line (4) was $\neg E$. So this branch's valuation requires both E and $\neg E$ to be true; that's impossible. If a branch contains any formula and its negation, we know that branch represents an impossible valuation. We'll call this branch *closed*, and mark it with an '$\times$'.

The left branch on (9) is another disjunction $\neg B \vee C$. It too branches out into its two disjuncts:

1.	$A \wedge B$ ✓		Root
2.	$\neg(C \vee D)$ ✓		Root
3.	$(\neg B \vee C) \vee E$ ✓		Root
4.	$\neg E$		Root
5.	A		$1 \wedge$
6.	B		$1 \wedge$
7.	$\neg C$		$2 \neg\vee$
8.	$\neg D$		$2 \neg\vee$
9.	$\neg B \vee C$ ✓	E $\times$	$3 \vee$
10.	$\neg B$ $\times$ C $\times$		$9 \vee$

Both of these disjuncts also result in closed branches. The left branch at (10) is the negation of (6), and the right branch is the negation of (7). Now every branch in this tree is closed. This corresponds to the idea that every potential way to make the formulas consistent ended up requiring a contradiction, so are impossible. There is no possible valuation that makes (1)-(4) all true. In other words, the formulas are mutually inconsistent.

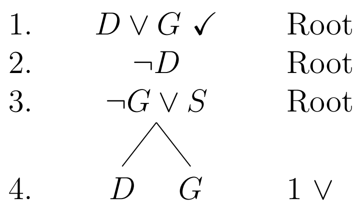
Our second Truth Tree Let's work through another consistency example. We've seen most of the elements of a truth tree, so we can make this one simpler. we want to test whether the following formulas are mutually consistent in TFL:

$$D \vee G, \neg D, \neg G \vee S$$

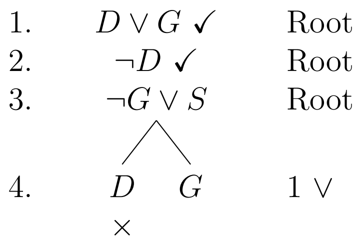
We begin by writing down the root:

1.	$D \vee G$	Root
2.	$\neg D$	Root
3.	$\neg G \vee S$	Root

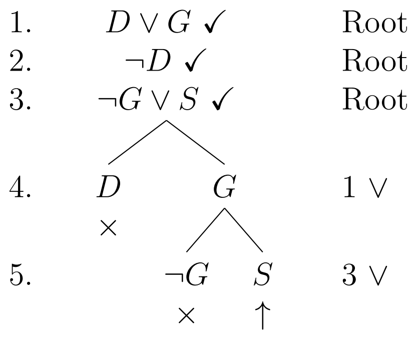
Line (1) is a disjunction, so it will produce a branch:



Line (2) is already atomic, and can't be decomposed. But it does help us to close one of the branches:



Line (3) is another disjunction, and will branch again. Note that if we hadn't closed one of the branches, we would have to add branches to both of them, leading to four open branches! It is important to try to close branches as soon as you can, to stop the tree from getting too complex.



We've closed one of these branches, but we can see that the last branch is still open, and that all the formulas above it either have a check mark, or are atomic. There is nothing more we can do; our tree is complete. We mark the open branch with an $\uparrow$. This represents the valuation we were seeking.

To read off the valuation (or truth table row, if you prefer), start at the arrow and read all the atomic and negated atomic symbols from the tree. In this case, we get: $S, G, \neg D$. So when $G = 1$, $S = 1$ and $D = 0$, all the formulas are true; they are mutually consistent.

Practice Exercises

⋆ **Exercise A:** How well do you understand the idea of a truth tree? Try to explain each of the following concepts:

1. What is the Root of a truth tree?
2. When should you tick (✓) a formula, and why?
3. What does a branch of a truth tree represent?
4. When should you create new branches of your tree?
5. When should you close a branch, and why?
6. When is an open branch complete? Why is this important?
7. What does the collection of atomic formulas in an open branch mean?

Chapter 12

Planting your Tree

12.1 Selecting the Right Root

Truth trees are used to indicate whether the root set of formulas is consistent. To use a truth tree to answer a question, the first step is always to convert the question into a question about whether some formula, or set of formulas, is mutually consistent.

Consider how this would apply to validity. An argument in TFL is invalid iff the set of its premises and the negation of its conclusion are mutually consistent. It is valid iff this set of formulas is mutually inconsistent. So we can test for the validity of a TFL argument using a truth tree. If at least one branch of the tree is open, then the set is consistent, and we've found a valuation that makes the premises true and the conclusion false. The argument is invalid, and the valuation is a counter-example. If the tree closes, the argument is valid.

Consistency and validity are the two main notions we are interested in. But we've already looked at several other TFL properties, including tautologies (logical truths), contradictions (logical falsehoods), and equivalences. These can all be tested with truth trees too.

For example, suppose you want to test if a formula is a tautology. To test this with a truth tree, we'll check if the formula can be false – that is whether it is consistent for its *negation* to be true. So we will put the negation of the formula in the root. If this truth tree remains open, it shows us how to make the negation true, so the formula is *not* a tautology. If it closes, the negation is inconsistent (can't be true), so the formula *is* a tautology.

You should be able to devise methods for testing for contradictions and equivalences from our complete and partial truth table methods, and the definitions of these notions. Give this a go, and then test your trees by trying some of the Exercises from previous chapters.

12.2 Growing your Tree

If you have been wondering how to decompose each formula in the tree, we'll introduce the formal rules for TFL truth trees in §13. In the meantime, we are just reading our truth tables backwards, as we did for partial truth tables. We figure out what subformulas have to be true for the current formula to be true, and write those sub-formulas down. If either of two subformulas being true is enough to make the current formula true, then we have a choice. Instead of making that choice, we branch the tree and put one option on each branch. This allows us to delay making our choice until we have more information.

Let's now test an argument for validity. Consider this argument form:

$$(D \vee A) \wedge \neg N, N \vee \neg A \therefore \neg N \wedge A$$

We are looking to see if a counter-example is possible; that is, if the premises are consistent with the conclusion being false. So the root of our tree will have the premises and the negation of the conclusion. We'll be looking for a valuation that makes all these formulas true. It's also useful to note what each part of the root comes from. We'll label the premises 'Premise' and the negated conclusion 'Neg Conc'. This also serves as a reminder to negate the conclusion!

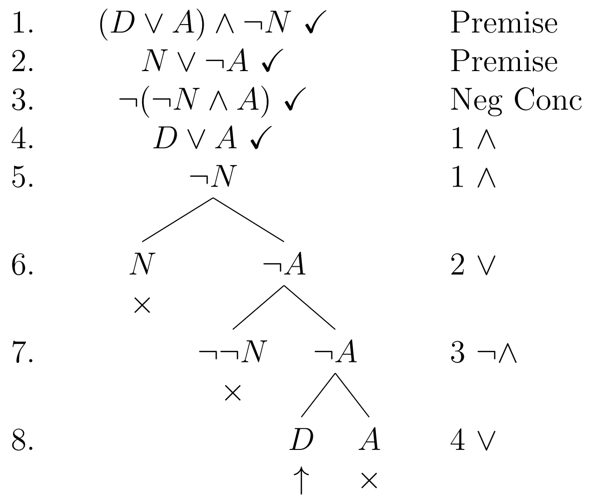
1.	$(D \vee A) \wedge \neg N$	Premise
2.	$N \vee \neg A$	Premise
3.	$\neg(\neg N \wedge A)$	Neg Conc

Line (1) is a conjunction; it decomposes into its conjuncts on (4) and (5). Line (2) is a disjunction; it branches into its disjuncts on line (6), and then the left branch closes as N and $\neg N$ cannot both be true.

1.	$(D \vee A) \wedge \neg N$ ✓		Premise
2.	$N \vee \neg A$ ✓		Premise
3.	$\neg(\neg N \wedge A)$		Neg Conc
4.	$D \vee A$		$1 \wedge$
5.	$\neg N$		$1 \wedge$
	/	\	
6.	N	$\neg A$	$2 \vee$
	$\times$		

Line (3) is a negated conjunction. A conjunction is true iff both conjuncts are true, so it is false if at least one conjunct is false. As we don't know which conjunct has to be false, the tree branches on line (7), with one negated conjunct on each branch: $\neg\neg N$ and $\neg A$. The first of these branches then closes because it is the negation of line (5). The final formula to decompose is line (4)'s disjunction; it branches on line (8), and one branch closes as it is the negation of line (7).

The remaining formulas are all atomic, so cannot be decomposed.


1. $(D \vee A) \wedge \neg N$ ✓ Premise
2. $N \vee \neg A$ ✓ Premise
3. $\neg(\neg N \wedge A)$ ✓ Neg Conc
4. $D \vee A$ ✓ 1 $\wedge$
5. $\neg N$ 1 $\wedge$
6. N $\times$ | $\neg A$ 2 $\vee$
7. $\neg\neg N$ $\times$ | $\neg A$ 3 $\neg\wedge$
8. D $\uparrow$ | A $\times$ 4 $\vee$

The $\uparrow$ indicates that the open branch ending in D is *completed.* (We'll define 'complete' in §13.) This branch represents the valuation we were seeking. This valuation is a counter-example, and so this argument is *not* valid in TFL. We can read the counter-example off the tree by starting at the arrow, reading upwards, and listing the atomic symbols: $\neg A$, D, and $\neg N$.

Our counter-example is: $A = 0, D = 1, N = 0$. A partial truth table can prove the mutual consistency of the formulas in the root:

A	D	N	$(D$	$\vee$	$A)$	$\wedge$	$\neg$	N	N	$\vee$	$\neg$	A	$\neg$	$(\neg$	N	$\wedge$	$A)$
1	1	0	1	1	0	1	1	0	0	1	1	0	1	1	0	0	0

All three formulas are true – the counter-example is possible.

On line (7) we wrote $\neg\neg N$. Some students want to write N instead. But that's not what our rules say. We also wrote $\neg A$ on both lines (6) and (7). That's absolutely fine. There's more than one piece of information that led us to the same formula. Again, don't try to out-think the rules and leave out seemingly redundant steps. It will just cause you confusion and error.

12.3 Completing your Tree

A tree is complete when there are no more formulas to decompose on any of its branches. This occurs when each branch has either been closed, or all its complex formulas have already been decomposed. By a complex formula, we mean any formula which is not just an atomic symbol or negated atomic symbol.

Closing a Branch

We *close* a branch when it contains a formula along with its negation:

A branch is CLOSED iff it contains both some formula $\mathcal{A}$ and its negation $\neg\mathcal{A}$. We mark closed branches with the '$\times$' symbol.
A branch is OPEN iff it is not closed.
A tree is CLOSED iff every branch in that tree is closed.
A tree is OPEN iff it is not closed.

If a tree is closed, we have shown that its root is inconsistent.

Completing an Open Branch

An open branch is only complete when we've decomposed all its complex formulas. We then write an '$\uparrow$' at the bottom of that branch.

An open branch is COMPLETE iff all its complex formulas have been decomposed, as indicated by a check mark.
A tree is COMPLETE iff all its branches are closed or complete.

If there is at least one open branch in a completed tree, then each open branch represents a valuation that makes all the formulas in the root true.

Reading the Tree Leaves

To read off the valuation from an open branch on a completed tree, start at the '$\uparrow$' arrow and follow the tree upwards, writing down all the atomic and negated atomic symbols from the tree. An atomic symbol means that atom is true; a negated atomic symbol means that atom is false. If you have both an atomic symbol and its negation in your list, close the branch.

This valuation will correspond to a case where all the formulas in the root are true. The mutual consistency of these formulas will mean different things depending on whether you are testing for consistency, validity, logical truth, logical falsehood, or equivalence, as we've already discussed.

Practice Exercises

⋆ **Exercise A:** List the formulas in the root of your tree when testing whether:

1. Are $(p \to q), (p \land r), (\neg r \lor \neg q)$ consistent?
2. Are $(p \lor \neg q), (q \lor \neg r), (r \lor \neg p)$ consistent?
3. Is $(\neg p \to p)$ a logical falsehood?
4. Is $(((p \to q) \to p) \land \neg q)$ a logical falsehood?
5. Is $(((p \to q) \to p) \to p)$ a logical truth?
6. Is $((q \lor \neg q) \to p)$ a logical truth?
7. Are $(p \leftrightarrow q), (\neg(p \land q) \leftrightarrow (\neg p \land \neg q))$ equivalent?
8. Are $((p \lor q) \land (q \lor r) \land (r \lor p)), ((p \land q) \lor (q \land r) \lor (r \land p))$ equivalent?
9. Are $(p \leftrightarrow q), (p \leftrightarrow \neg q)$ contradictory?
10. Are $(p \land q), (\neg p \land \neg q)$ contradictory?
11. Is $(p \to r), (q \to r) \therefore (p \to q) \to r)$ valid?
12. Is $(p \to r), (q \to r) \therefore (p \lor q) \to r)$ valid?

Exercise B: Complete the trees from Exercise A. For each closed tree, state what you have proved.

Exercise C: For each open tree from Exercise B, complete the partial truth table, and state what you have proved.

Chapter 13
TFL Tree Rules

The examples we've worked through should have given you an overview of the tree method for TFL. But we haven't precisely defined how to decompose each type of main connective. We are now ready to give formal rules for trees. You should be able to recognize the following rules as a generalization of the steps of the proofs given above.

Our DECOMPOSITION rules describe how the tree can be extended by decomposing formulas. The rules depend on the main connective of the formula. (If the main connective is a negation, then they also depend on the main connective inside that negation.)

13.1 Decomposition Rules

The decomposition rule for conjunction is that if you have a conjunction in a branch, you may make a linear extension of the branch that includes each conjunct, adding a check mark next to the conjunction. So, any time you have a conjunction $\mathcal{A} \wedge \mathcal{B}$ on line (i) you may extend that branch of the tree by writing:

$$\begin{array}{ll} \mathcal{A} & i \wedge \\ \mathcal{B} & i \wedge \end{array}$$

It is important to remember once again that $\mathcal{A}$ and $\mathcal{B}$ here can stand for *any* formula of TFL, including complex ones. The formulas must be copied exactly as written, including any negation signs. Also, write the line # of the decomposed formula, and which decomposition rule you used, on the new line(s). This will help you when you check your work. You do check your work, right?

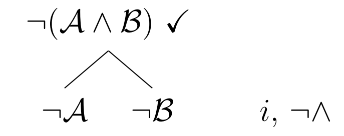
Conjunction

Our conjunction decomposition rule is:

$$\begin{array}{cl} \mathcal{A} \wedge \mathcal{B} \checkmark & \\ \mathcal{A} & i, \wedge \\ \mathcal{B} & i, \wedge \end{array}$$

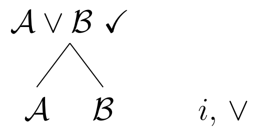
Negated conjunction

A negated conjunction, branches into the negation of each conjunct. This is because there are two ways for a negated conjunct to be true – either conjunct can be false:



Disjunction

Disjunctions branch into each disjunct:



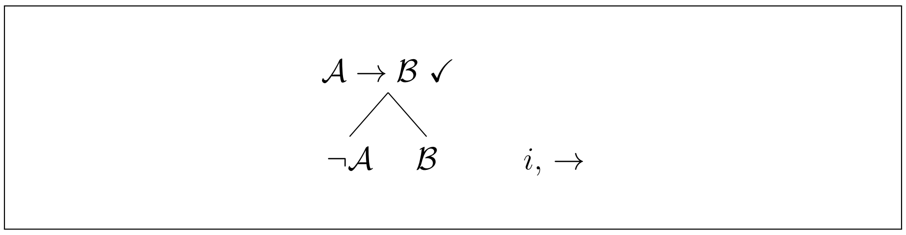
Negated disjunction

Since disjunctions are true any time either disjunct is true, they are only false if both disjuncts are false; that is, their negations are both true:

$$\begin{array}{cl} \neg(\mathcal{A} \vee \mathcal{B}) \checkmark & \\ \neg\mathcal{A} & i, \neg\vee \\ \neg\mathcal{B} & i, \neg\vee \end{array}$$

Conditional

Conditionals are true when the antecedent is false *or* the consequent is true. This means that conditionals in TFL are treated similarly to disjunctions:



Negated conditional

The negation of a conditional is true when the conditional is false. That is, when its antecedent is true *and* its consequent is false:

$$\begin{array}{cl} \neg(\mathcal{A} \to \mathcal{B}) \checkmark & \\ \mathcal{A} & i, \neg\to \\ \neg\mathcal{B} & i, \neg\to \end{array}$$

Double negation

A doubly-negated formula is true iff the singly-negated formula is false iff the original formula is true. So we can simply remove a double negation. However, this has to be done as a separate step:

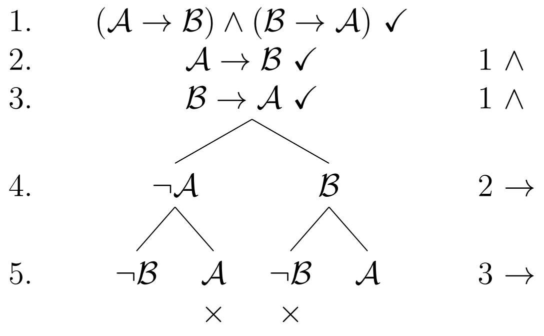
$$\begin{array}{cl} \neg\neg\mathcal{A} \checkmark & \\ \mathcal{A} & i, \neg\neg \end{array}$$

Logic Corner

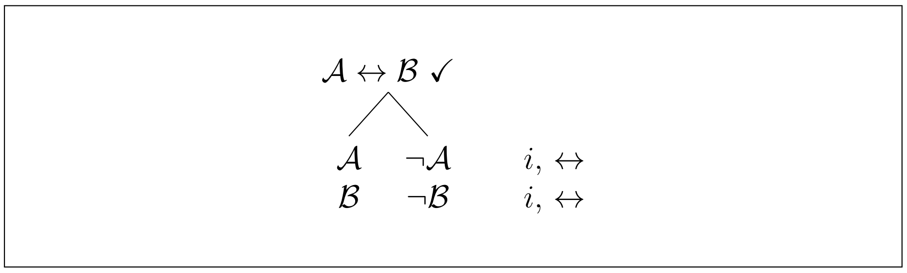
You may have noticed that the decomposition rules for the standard connectives and their negations are duals of each other – if one branches, the other doesn't, if one negates $\mathcal{A}$, the other does not, and so forth. This means that if you know the decomposition rule for a connective, you also know the rule for its negation. Just do everything the opposite way!

Biconditional

Biconditionals are really just a combination of two conditionals. That is, $\mathcal{A} \leftrightarrow \mathcal{B}$ iff $(\mathcal{A} \to \mathcal{B}) \wedge (\mathcal{B} \to \mathcal{A})$. If we decompose this formula, we get:

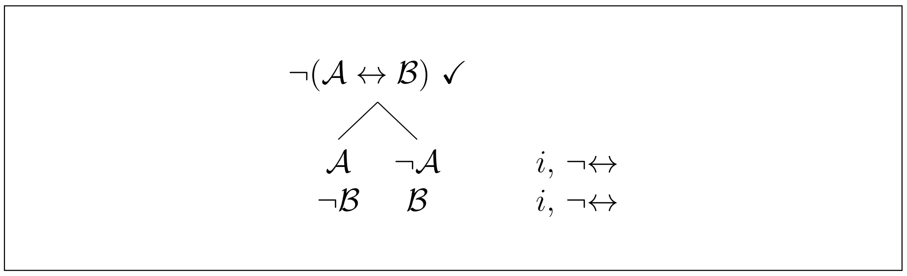

1. $(\mathcal{A} \to \mathcal{B}) \wedge (\mathcal{B} \to \mathcal{A})$ ✓
2. $\mathcal{A} \to \mathcal{B}$ ✓ 1 $\wedge$
3. $\mathcal{B} \to \mathcal{A}$ ✓ 1 $\wedge$
4. $\neg\mathcal{A}$ $\mathcal{B}$ 2 $\to$
5. $\neg\mathcal{B}$ $\mathcal{A}$ $\neg\mathcal{B}$ $\mathcal{A}$ 3 $\to$
× ×

One open branch has $\mathcal{A}$ and $\mathcal{B}$; the other $\neg\mathcal{A}$ and$\neg\mathcal{B}$. This might seem odd, but recall that a biconditional is true when its sub formulas have the same truth value: either both are true, or both are false. And that's exactly what this rule states:


$\mathcal{A} \leftrightarrow \mathcal{B}$ ✓
$\mathcal{A}$ $\neg\mathcal{A}$ $i, \leftrightarrow$
$\mathcal{B}$ $\neg\mathcal{B}$ $i, \leftrightarrow$

Negated biconditional

Negated biconditionals are true when the subformulas have different truth values. The tree rule looks remarkably similar to the standard biconditional:


$\neg(\mathcal{A} \leftrightarrow \mathcal{B})$ ✓
$\mathcal{A}$ $\neg\mathcal{A}$ $i, \neg\leftrightarrow$
$\neg\mathcal{B}$ $\mathcal{B}$ $i, \neg\leftrightarrow$

You can derive this rule too, by creating a tree for the negation of a conjunction of conditionals: $\neg((\mathcal{A} \to \mathcal{B}) \wedge (\mathcal{B} \to \mathcal{A}))$.

13.2 Truth Tree Complexities

Incomplete Valuations

Sometimes our open branch will not contain every atom in our set of formulas. That is, our valuation does not specify the truth value of some atomic symbols. When this occurs, the truth value of the remaining atomic symbols does not affect the consistency of the formulas. However, a valuation must specify the truth value of every symbol, so you can arbitrarily pick some truth value for each, or indicate its truth value is unknown by a '?'.

For example, suppose an open branch for a set of formulas using the symbols $\{p, q, r\}$ contains only p and $\neg q$. This valuation would be: p is true, q is false, r is unknown; or in symbols: $p = 1, q = 0, r =?$.

Early Closing

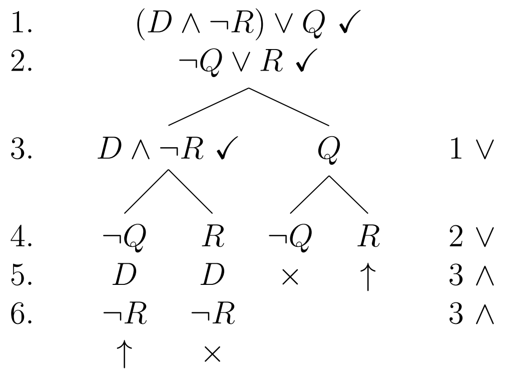
Note that we close a branch when it contains *any* formula and its negation. We don't need an atomic formula and its negation. We saw this in an early example when we closed a branch containing $\neg\neg N$ and $\neg A$. Here is another example:

$$\begin{array}{lcc}
1. & \multicolumn{2}{c}{(\neg S \wedge T) \rightarrow (\neg P \leftrightarrow (Q \vee R)) \checkmark} \\
2. & \multicolumn{2}{c}{\neg S \wedge T} \\
3. & \multicolumn{2}{c}{\neg(\neg P \leftrightarrow (Q \vee R))} \\
 & \swarrow & \searrow \\
4. & \neg(\neg S \wedge T) & \neg P \leftrightarrow (Q \vee R) \\
 & \times & \times
\end{array}$$

In this tree, we only decomposed the first formula – using the conditional rule – and then the tree immediately closed. The left branch of (4) is the negation of (2), and the right branch is the negation of (3). Notice that we *could* have ignored (or missed) this, and decomposed lines (2) and (3) using the conjunction or negated biconditional rules, respectively. However, this would have resulted in a more complicated tree that would also have eventually closed. Missing an early closing opportunity will never affect the final result of a truth tree, just create more work and a higher chance of error. You can occasionally save yourself a lot of effort by noticing when branches are ready to mark as closed. However, if you don't notice this early closing, no real harm will be done.

Multiple Open Branches

If you are decomposing a formula after the tree has already branched, you must add the new formulas under *each* open branch that descends from that formula. Here is a tree illustrating how to do this:


1. $(D \wedge \neg R) \vee Q$ ✓
2. $\neg Q \vee R$ ✓
3. $D \wedge \neg R$ ✓ Q 1 ∨
4. $\neg Q$ R $\neg Q$ R 2 ∨
5. D D × ↑ 3 ∧
6. $\neg R$ $\neg R$ 3 ∧
↑ ×

This tree has two disjunctions in the root, so it will branch multiple times. We decomposed line (1) first, branching into the two disjuncts at line (3). When we decomposed (2), both branches descending from (2) were still open, so we added line (4) to both these branches. That's why our two branches split into four at line (4). The third branch contains Q and $\neg Q$, and so closes. At line (5), the conjunction $D \wedge \neg R$ at line (3) is decomposed. Only the left-most 2 of the 3 open branches descend from this conjunction. So the conjuncts need to be added to both branches that descend from that conjunction, but not the right-most branch, which does not.

Multiple Valuations

A tree can have more than one completed open branch. The tree above is an example of this. These completed open branches can have different valuations. This is because there can be more than one counter-example, or row that doesn't satisfy the property you are testing. It is fine to use any completed open branch. However, be careful.

First, don't combine your open branches. In the tree above, one valuation has $Q = 1, R = 1$, and the other $Q = 0, R = 0$. Any combination of these would fail to be a valuation where all the formulas in the root are true.

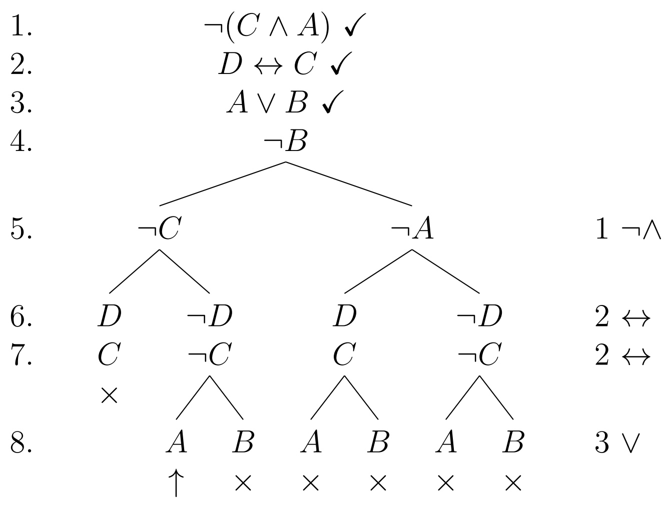
Second, indicate clearly which open branch your counter-example comes from. We suggest that you only use the symbol ↑ on the completed branch which you are reading your valuation from. Leave the other open branches blank.

13.3 Order of Decomposition

You do *not* need to decompose formulas in strict order from the top of the tree. You can decompose formulas in whichever order you find convenient. Some ways of decomposing trees are more efficient than others. The most basic principle is to decompose formulas that won't branch before those that will. When all the remaining complex formulas will branch, try to look a step ahead, and select a formula that will have a branch that closes immediately. This reduces the likelihood of having multiple open branches. Next, decompose simple formulas before complex ones, as you are less likely to muck this up. Finally, I suggest you decompose biconditionals last, as they have the most complex rules of all.

1. Non-branching rules before branching.
2. Close branches quickly.
3. Simple formulas before complex.
4. Biconditionals last.

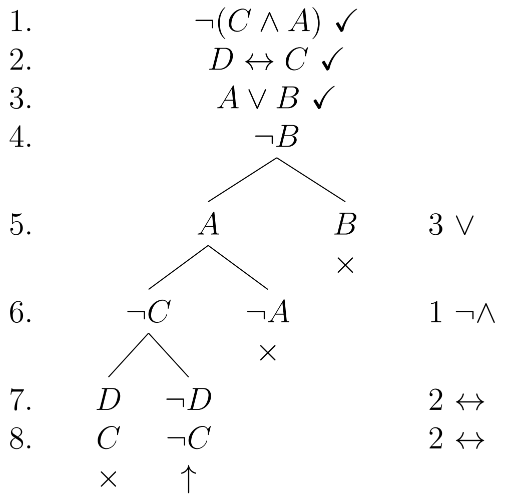
Suppose we want to test whether $\neg(C \wedge A), D \leftrightarrow C, A \vee B, \neg B$ are mutually consistent. These formulas would then form the root of our tree. Here's the tree formed if we decompose the formulas in the order in which they're listed:


1. $\neg(C \wedge A)$ ✓
2. $D \leftrightarrow C$ ✓
3. $A \vee B$ ✓
4. $\neg B$
5. $\neg C$ $\neg A$ 1 $\neg\wedge$
6. D $\neg D$ D $\neg D$ 2 $\leftrightarrow$
7. C $\neg C$ C $\neg C$ 2 $\leftrightarrow$
×
8. A B A B A B 3 $\vee$
↑ × × × × ×

This completed tree is open. We can read off the valuation from its one open branch. The atomic and negated atomic symbols on this open branch are: A, $\neg C$, $\neg D$, $\neg C$, $\neg B$. This tells us that a valuation where all the formulas are true is $A = 1, B = 0, C = 0, D = 0$.

This tree suffers from the problem of having multiple open branches during its construction. That's why, for example, decomposing line (3) at line (8) requires three different new branchings. So when there are more open branches, decomposing formulas requires more work on the tree, more repetition, and more chance of errors.

The tree above is a perfectly fine completed tree, and it gives a correct answer. However, it's possible to get there with less work, by choosing to decompose formulas that will close off branches right away. Here is another tree with the same root as in the previous example, but which decomposes formulas in a more efficient order:


1. $\neg(C \wedge A)$ ✓
2. $D \leftrightarrow C$ ✓
3. $A \vee B$ ✓
4. $\neg B$
5. A B $3\ \vee$
$\times$
6. $\neg C$ $\neg A$ $1\ \neg\wedge$
$\times$
7. D $\neg D$ $2 \leftrightarrow$
8. C $\neg C$ $2 \leftrightarrow$
$\times$ $\uparrow$

We can use a partial truth table to confirm our answer:

A	B	C	D	$\neg$	$(C$	$\wedge$	$A)$	$(D$	$\leftrightarrow$	$C)$	A	$\vee$	B	$\neg$	B
1	0	0	0	1	0	0	1	0	1	0	1	1	0	1	0

This tree gets us to the same result much more quickly, and generates the same valuation. Here's a description of the reasoning for deciding which formula to decompose at each step:

5. Line (4) contains a $\neg B$. Line (3) would generate a branch with B.
6. Line (5) contains an A. Line (1) would generate a branch with $\neg A$.
7. Line (6) contains a C. Line (2) would generate a branch with $\neg C$.

You usually don't need to think more than one line ahead to select branches that create a much simpler tree.

Practice Exercises

$\star$ **Exercise A:** To evaluate each of the following claims with a tree, (a) what would you put in the root of the tree?, and (b) if the tree closes, does that show that the claim is true or false?

1. $P, P \to Q, Q \to \neg P$ is mutually consistent
2. $(P \to Q) \leftrightarrow (Q \to P)$ is a tautology.
3. The following argument is valid: $P \wedge Q, \neg R \to \neg Q \ \therefore\ P \wedge R$
4. Every valuation making P, $P \to Q$, and $\neg Q$ true also makes A true.
5. There is no valuation that makes $A \vee B$, $B \to C$, and $A \leftrightarrow C$ all true but C false.
6. $A \leftrightarrow \neg A$ is a contradiction.
7. There is at least one valuation that makes $P \to Q$, $\neg P \vee \neg Q$, and $Q \to P$ true.

$\star$ **Exercise B:** Evaluate each claim from Exercise A by constructing a tree. If applicable, give the valuation that demonstrates the claim true or false, and use a partial truth table to confirm this.

$\star$ **Exercise C:** Evaluate the argument

$$A \leftrightarrow B, \neg B \to (C \vee D), E \to \neg C, (\neg D \wedge F) \vee G, \neg A \wedge E, \ \therefore\ H \vee G$$

by constructing a tree. If the tree is open, prove your counter-example is correct using a partial truth table.

Exercise D: The Sheffer stroke $\mathcal{A} \mid \mathcal{B}$ is a rarely-used connective of TFL. It has the following characteristic truth table:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A} \mid \mathcal{B}$
1	1	0
1	0	1
0	1	1
0	0	1

Create appropriate tree decomposition rules for the Sheffer stroke, and for the negated Sheffer stroke.

Exercise E: Construct trees for any of the exercises from the complete and partial truth tables chapters. Check the valuations from any open branches are correct using partial truth tables.

Interlude

Well done! You've finished the first half of the book, where we learn the basics of TFL. For the rest of the book, we will work through the same steps (symbolisation, tables, trees) with another logic, PL. But let's just pause for a short reflection first.

The Limits of Validity

The logic we've taught you so far is quite limited, but *any* formal method has some limitations. The first limitation is with validity. If an argument is valid, then whenever its premises are true, its conclusion is guaranteed to be valid. That's great, but it's not enough.

False Premises If even one of the premises is false, validity tells us nothing about the conclusion. Here's a valid argument with false premises and a true conclusion:

> I just ran 10km in 5 minutes, and swam to the moon and back. And there'll be 12 solar eclipses before lunch tomorrow. So, like it or not, there'll either be those eclipses or I was born on Earth.

Relevance Even if all the premises are true, many of them may not be relevant to the conclusion. Here's an argument where most of the premises don't contribute at all:

> Well I'm feeling great this morning and feel like a good meal. I'll eat a peach. At any rate, I'll eat a peach or the whole of Mars.

Circularity The conclusion may be one of the premises, making the reasoning valid but question-begging. See what you make of this valid argument:

> I say that all experts tell the truth, and you can trust me not to lie because I'm an expert. So I'm telling the truth.

Explosion If the premises are inconsistent, then the argument will be valid, no matter what conclusion you choose. Here's a valid argument that's slightly bananas:

> Whangarei is in the north, and so is Taihape if Whangarei is. But Taihape is not in the north, so you'll be immortal if you eat a banana every day.

The Limits of Symbolisation

Another major issue is that formalisation is inherently asymmetric. I'll explain what I mean via an example. Suppose that we have an argument in English, and we want to know if it is valid. We can symbolise the argument in TFL (or any other logic we know), then test it for validity.

If the truth table, truth tree, or whatever, says that the formal symbolisation of the argument is valid, and we didn't make any mistakes, then the original argument is valid. We are done.

However, if the formal symbolisation is invalid, then we need to check if the counter-example is applicable to the original English argument. By considering what our counter-example means when desymbolised, we can see if its relevant. For instance, perhaps our counter-example requires faster-than-light travel, or someone being in two places at the same time, or being both a cat and a dog. None of these are relevant counter-examples, but they do show an implicit hole in our original argument, just because we often don't write down all the basic premises.

In this case, we should adapt our original argument by adding another premise that rules out the crazy counter-example, symbolise the new argument, and test it. If it's not valid, check if your counter-example is relevant, and keep repeating this loop until either you have a relevant counter-example that shows your argument is not good, or your new argument is valid.

If you have a relevant counter-example, not even the improved argument is valid. But if the new argument is valid, what does that say about your original one? You could say your argument was valid, if you did believe all the extra added premises required to make it valid. But if you had to add something you didn't initially believe, then perhaps it wasn't valid; it might have been missing some key information that you hadn't realised.

On the other hand, maybe you just didn't work with a powerful enough logic. Consider this argument:

Every horse likes carrots.
Jasper is a horse
∴ Jasper likes carrots.

This is a valid argument in English. But TFL can't capture what makes it valid. The logic you are about to learn does make it valid, but there can *never* be a logic that can capture all the aspects of English that make valid arguments. And we can prove that with a valid argument.

Enjoy !!

Part V

Predicate Logic Symbolisation

Chapter 14
The Language of PL

14.1 Decomposing Statements

Consider the following argument:

Phoenix is a logician.
All logicians are kind.
∴ Phoenix is kind.

This argument is valid. To see this, take any situation in which Phoenix is a logician and all logicians are kind. We do not know whether there are other logicians in this case, but we know that Phoenix is one of them. We further know from the second premise that anyone who is a logician is someone who is kind. So it must be that Phoenix is kind. Therefore, the argument is valid, because every case in which the premises are true is a case in which the conclusion is also true.

Logic Corner

Logicians sometimes like to work with counterexamples to demonstrate that arguments are valid. Here's how this might go with the Phoenix argument:

> Imagine there were a counterexample, i.e., a case in which the premises are true but the conclusion is false. From the truth of the first premise and the falsity of the conclusion, this would be a case in which Phoenix is a logician that isn't kind. If that were so, we would have at least one logician who isn't kind, which would make the second premise false. So our case would fail to be a counterexample. Hence, there cannot be a counterexample to the argument. The argument is therefore valid.

Because the argument is quite simple, appealing to counterexamples might seem to you like an overkill. But remember this strategy when you face arguments in the wild and wonder if they are valid: assume there's a counterexample, i.e., a case in which the premises are true and the conclusion is false, and try to get a contradiction.

Mathematics Corner

Working with counterexamples can be very efficient in mathematics. For those of you who have to sit in mathematics exams and have to provide proofs, try the counterexample methodology: assume that there is a case in which conditions set out in the problem hold, but the result you've been asked to proved doesn't. Use this information to derive a contradiction. If you succeed, you know that the result follows *validly* from the conditions of the problem, and so you have an efficient proof.

Let's try now to symbolise it in TFL, and see if we get the same diagnosis. We might use this symbolisation key:

L:	Phoenix is a logician.
A:	All logicians are kind.
F:	Phoenix is kind.

And the argument itself becomes:

	Phoenix is a logician.		L
	All logicians are kind.		A
$\therefore$	Phoenix is kind.	$\therefore$	F

This is *invalid* in TFL, as can be see with a valuation that assigns 1 to L and A, and 0 to F. Something has gone wrong. We have a valid argument whose symbolisation tell us that it's invalid. What's going on?

The problem is not that we have made a mistake while symbolising the argument. This is the best symbolisation we can give *in* TFL. The problem lies with TFL itself. 'All logicians are kind' is about two kinds of things: logicians and things that are kind. Our symbolisation in TFL fails to make this distinction, so we lose the connection between Phoenix being a logician and Phoenix being kind. In other words, connectives do not allow us to decompose statements like 'All logicians are kind' into simpler components. This is why we need a more powerful formal system.

To symbolise arguments like the preceding one, we will have to develop a new logical language which will allow us to *split the atom*. We will call this language Predicate Logic (PL). The details of PL will be explained throughout this chapter, but here is the basic idea. First, we have *names*. Names allow us to refer directly to things, like Phoenix. In PL, we indicate names with lowercase italic letters. For instance, we might let 'p' be a name for Phoenix, or let 'b' be a name for SpongeBob.

Second, we have predicates, which we use to symbolise properties of things, like *being a logician* or *wearing square pants*. In PL, we indicate predicates with uppercase italic letters. For instance, we might use the letter 'L' to symbolise the property of *being a logician*, or the letter 'P' to symbolise the property of *wearing squarepants*. We can then combine names and predicates to symbolise statements: $L(p)$ symbolises 'Phoenix is a logician' and $P(b)$ symbolises 'SpongeBob wears squarepants'.

Finally, we have symbols that allow us to symbolise the part of statements that talk about *all* or *some* things. We call those quantifiers. In this course, we will work with two quantifiers: the existential quantifier '$\exists$' and the universal quantifier '$\forall$'. We will use quantifiers to symbolise statements like 'All logicians are kind' as $\forall x\,[Lx \rightarrow Kx]$'. Combining all ingredients together, we symbolise the Phoenix argument in PL like this:

$$\begin{array}{ll} & L(p) \\ & \forall x\,[Lx \rightarrow Kx] \\ \therefore & K(p) \end{array}$$

That is the general idea, but PL is significantly more subtle than TFL, so we will come at it slowly.

14.2 Names

The tallest building in Auckland is 328 metres high. This is a true fact. Here's another true fact: the Sky Tower is 328 metres high. The two sentences express the same thing, because 'the tallest building in Auckland' and 'Sky Tower' are two ways to refer to the same thing. The Sky Tower *is* the tallest building in Auckland. Although they refer to the same thing, however, 'the tallest building in Auckland' and 'Sky Tower' perform different linguistic roles. 'The tallest building in Auckland' is a description, whereas 'Sky Tower' is a name. A description refers to a thing by listing some of its properties. A name refers to a thing without describing it. A thing is given a name, and the name refers to that thing, by stipulation. As people, we all have a name, which is given to us at birth, but we can also refer to people by describing them. Thus, 'Emily Parke' and 'the philosopher of science at the University of Auckland' refer to the same person.

In PL, names are by default lower-case letters 'a' through to 't'. On some occasions, we might also use letters u to z if the context makes it nicer, for instance if we wanted to use w for Willi or z for Zeus. You can think of these names along the lines of the proper names that all of us have, but with one difference. More than one person may be given the name 'Emily Parke', and usually context makes it clear which person the name refers to. We have to live with this type of ambiguity, allowing context to individuate the fact that 'Emily Parke' refers to a philosopher of science at the University of Auckland, and not some other Emily. In PL, 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 can provide symbolisation keys. These indicate, temporarily, what a name will pick out, for example:

e:	Emily
g:	Gregor
s:	the Sky Tower

14.3 Predicates

The simplest predicates are about properties of individuals, such as 'being a dog', 'being a square' or 'being a member of the Avengers'.

In general, you can think of predicates as things which combine with names to to make statements. For example you can combine 'Hulk' and 'is a member of The Avengers' to make the statement 'Hulk is a member of the Avengers'.

In PL, PREDICATES are capital letters A through Z. When we include predicates in symbolisation keys, we will use variables to make it clear how many things a predicate applies to. In PL, variables are italic lowercase letters 'u' through 'z'. Until we get to §17, the predicates we will use refer to individual properties, such as 'being a dog' or 'being a logician'. Later, we will use more complex predicates that refer to relations between things, such as 'being siblings' or 'being taller than'. We call a predicate that refers to an individual property a *one-place predicate*, a predicate that refers to a relation between two things a *two-place predicate*, a predicate that refers to a relation between three things a *three-place predicate*, and so on. The use of variables makes it clear what type of predicates we are using, so Hx is a one-place predicate, Rxy is a two-place predicate, and $S(xyz)$ is a three-place predicate.

Notation Corner

There are various conventions about the use of brackets in writing predicates. Some write $H(x)$ instead of Hx for one-place predicates, and some write $R(x, y)$ or xRy for two-place predicates. We write Hx and Rxy in this textbook.

For now, let's focus on one-place predicates. Combining names and predicates, a symbolisation key looks like this:

e:	Emily Parke
h:	Hulk
Ax:	x is angry.
Px:	x is a philosopher of science.

With this symbolisation key, we can symbolise statements that use these names and predicates in combination. For example, consider these statements:

1. Hulk is angry.
2. Emily isn't angry.
3. Hulk is angry, but Emily isn't.

Statement 1 is straightforward. We symbolise it thus:

$$Ah$$

Statement 2 expresses that Emily doesn't have the property of being angry, so is the negation of the statement $A(e)$. We symbolise it thus:

$$\neg A(e)$$

Statement 3 is the conjunction of Statements 1 and 2. This illustrates an important point: PL has all of the truth-functional connectives of TFL. We thus symbolise Statement 3, a conjunction, like this:

$$(A(h) \wedge \neg A(e))$$

Practice Exercises

Exercise A:

Identify all the names in the following story, and list them in a symbolisation key:

> Once upon a time a cat showed up at a house in Ponsonby. She was fluffy and lovely, but seemed to be lost. At the house was a family of four, Raz and Aki, and their two children, a boy and a girl. The kids asked if they could keep the cat, because she looked lost and hungry. Their parents said that they would first reach out to their community on Facebook and other social media, but that they could take care of her while they tried to find her home. The cat came to be known as 'The Queen of Ponsonby', because of the way she seemed to demand a lot of attention and get upset when no one would feed her. Eventually someone called Umut responded to the family's post on Facebook and claimed to be the cat's owner. When Umut came to pick her up, he told them her name was Shiro, and was very thankful that Raz and Aki's family help in reuniting them. The children were sad when Shiro had to leave, but they were also happy that she would go back to her house. When they talked about the lovely times they had with her, they always refer to her as 'The Queen of Ponsonby'.

Refer to this symbolisation key for the next two problem sets:

e:	Emily Parke
h:	Hulk
Ax:	x is an Avenger.
Mx:	x is a Marvel superhero.
Px:	x is a philosopher of science.

Exercise B:

Symbolise the following statements in PL:

1. Hulk is a Marvel superhero.
2. Emily Parke is not an Avenger or a Marvel superhero.
3. Hulk isn't a philosopher of science, but he is an Avenger and a Marvel superhero.
4. If Hulk is a philosopher of science, then he isn't an Avenger.
5. Neither Hulk nor Emily Parke are Avengers, Marvel superheroes, or philosophers of science.

Exercise C:

Express the following formulas of PL in English:

1. $\neg Ap$
2. $Pe \wedge (Mh \wedge Ah)$
3. $Me \rightarrow \neg Pe$

Exercise D:

For each statement, provide a symbolisation key and suggest a symbolisation. Pay special attention to how many places your predicates have.

1. The Lion King is a great movie.
2. Titanic is a better movie than Avatar.
3. Hamilton is south of Auckland and north of Wellington.
4. Hamilton is in between Wellington and Auckland.
5. Wellington is in the middle of New Zealand.
6. Wellington is in the middle of Auckland, Gisborne, Nelson and Christchurch.

Chapter 15
Quantifiers

Consider again the argument we started with:

Phoenix is a logician.
All logicians are kind.
$\therefore$ Phoenix is kind.

With the work we've done so far, we are able to symbolise that Phoenix is a logician with something like $L(p)$ and that Phoenix is kind with $K(p)$, but we are not able yet to symbolise the second premise 'all logicians are kind'. If we had a list of all people (dead or alive), with full information telling us who is a logician and who is kind, we might be able to express 'all logicians are kind' with a very long conjunction like this:

> 'Jeremy is a logician and is kind' and 'Emily is a logician and is kind' and 'Phoenix is a logician and is kind' and . . .

This strategy would only work if we were talking about finitely many things, which isn't always the case. We know for instance that 'all prime numbers are only divisible by 1 and themselves', but we wouldn't be able to symbolise this as a conjunction by listing all the cases. The first reason is that the conjunction would be infinite, because there are infinitely many prime numbers, and the second reason is that we do not know which numbers are prime numbers. Yet we can express true things about infinite collections of things (like prime numbers), even thought we don't know all the members of the collections). In PL, we symbolise this with QUANTIFIERS. allow us to talk about (finite or infinite) collections of things in one statement.

15.1 Introducing Quantifiers

In this book, we will focus on two types of quantifiers, the existential quantifier and the universal quantifier. We use the symbol $\exists$ for the existential quantifier and the symbol $\forall$ for the universal quantifier. The existential quantifier expresses that *at least one thing* of a collection has a certain property, for example that at least one planet has life, or that at least one philosopher at the University of Auckland specialises in philosophy of science. On the opposite end, the universal quantifiers refers to *everything* in a collection, for instance that all logicians are kind, or that all planets orbit a star.

Linguistics Corner

There are further quantifiers between these two, for instance a quantifier that expresses that *most things* in a collection have a property, such as most birds can fly, or most prime numbers are odd. We will not include those in our formal language, because they require more complexity than we want to go into in this book. As you will see, dealing with the existential and universal quantifiers is challenging enough.

When we write formulas with quantifiers, we always use variables. There's no exception to this rule: a quantifier must always be followed by a variable. Formulas with quantifiers will look like this in their simplest form, when they apply only to predicates:

$$\forall x\,[Ax] \qquad \exists y\,[Hy]$$

We always use brackets to indicate the *scope* of the quantifier. In this book we are using [square brackets], just so that you can more easily keep track of quantifier scope. However, they function just like normal (parentheses), as we can see below:

$$\forall x\,[Ax \wedge \neg Hx] \qquad \exists y\,[Hy \rightarrow \neg(Ay \vee \neg By)]$$

We will come back to the notion of the *scope* of a quantifier shortly. There is no special reason to use 'x' rather than some other variable. The statements

'$\forall x\,[Hx]$', '$\forall y\,[Hy]$', '$\forall z\,[Hz]$' use different variables, but are all logically equivalent.

Notation Corner

There are various conventions on how to use brackets in formulas with quantifiers. Some put brackets around the quantifiers and write $(\forall x\,)Hx$, others put brackets around the predicate, as in $\forall x\,(Hx)$. You could be extra-thorough and write $(\forall x\,)(H(x))$. In this textbook, we don't put brackets around the quantifier, and we always put square brackets around the scope of the quantifier, as in $\forall x\,[Ax \wedge Bx]$. You can drop the scope brackets if it contains a single predicate, as in $\forall x\,Ax$.

Let's look at some examples, based on this symbolisation key:

p:	Phoenix
Lx:	x is a logician.
Kx:	x is kind.

How do we express the following statements?

1. Some logicians are kind.
2. Phoenix is kind, but some logician isn't.
3. All logicians are kind.

Statement 1 expresses that there is at least one logician who is kind. We can express this with this formula:

$$\exists x\,[Lx \wedge Kx]$$

This formula says that there is something that is both a logician and is kind. We can express statement 2 as a conjunction of two statements, one expressing that Phoenix is kind, the other that at least one logician isn't kind:

$$K(p) \wedge \exists x\,[Lx \wedge \neg Kx]$$

For the last statement, let's first see how you *don't* express that all logicians are kind:

$$\forall x\,[Lx \wedge Kx]$$

This formula not only says that all logicians are kind, but that *everything is a kind logician*. Obviously, there are things that are neither logicians nor kind, like chairs or rocks. What we want is to express not that everything is a kind logician, but that *everything that happens to be a logician is also kind*. We need to restrict the reach of the quantifier to talk about logicians only. We achieve this by using a conditional:

$$\forall x\,[Lx \to Kx]$$

This says of each thing that *if* it is a logician (some things aren't, but if it is…) then it is kind. The universal quantifier talks about everything, and we use the conditional to restrict it to talk about logicians only. With the restriction as the antecedent of the conditional, we can then express that everything in that restriction is kind.

Logic Corner

Let us use the opportunity to introduce two important rules of thumb for the use of quantifiers in PL. The first is that the existential quantifier $\exists$ is usually accompanied by a conjunction $\wedge$, the second that the universal quantifier $\forall$ is usually accompanied by a conditional $\to$. The existential quantifier typically expresses a conjunction of properties that hold of at least one thing. The universal quantifier typically expresses that things which meet some conditions also have other properties. This will become second nature for you before too long, but for now remember this: $\exists$ goes with $\wedge$, and $\forall$ goes with $\to$.

Putting everything together, we can now express the Phoenix argument we saw at the start of this section in PL:

	Phoenix is a logician.		$L(p)$
	All logicians are kind.		$\forall x\,[Lx \to Kx]$
$\therefore$	Phoenix is kind.	$\therefore$	$K(p)$

15.2 Quantifiers and Scope

Let's expand our symbolisation key:

p:	Phoenix
Lx:	x is a logician.
Kx:	x is kind.
Px:	x is a person.

How would you symbolise this statement?

4. If someone is a logician, then someone is kind.

There are two different quantifiers in the statement, so we expect to use two different variables, one for each quantifier. The first thing to notice is that the statement is a conditional:

$$\text{someone is a logician} \quad \rightarrow \quad \text{someone is kind}$$

Now that we've identified the form of the statement and isolated the quantified bits, we can complete the symbolisation:

5. $\exists x\,[Px \wedge Lx] \rightarrow \exists y\,[Py \wedge Ky]$

It would be a mistake to symbolise 4 as:

$$\exists x\,[(Px \wedge Lx) \rightarrow Kx]$$

This says that if someone is a logician, then *they* are kind. Statement 4, however, didn't specify of the *same* person that they are kind if they are a logician. All 4 says is that if *someone* is a logician, then *someone* (maybe the same person, but not necessarily the same) is kind. There are two quantifiers in 5, and each has a different *scope*. The scope of $\exists x$ is $[Px \wedge Lx]$, and the scope of $\exists y$ is $[Py \wedge Ky]$.

It would also be a mistake to symbolise 4 as:

$$\exists x\,[Px \wedge Lx] \rightarrow Kx$$

This is an incomplete statement, that would read, literally, as: 'If someone is a logician, then x is kind.' The variable x in the Kx is left dangling; it is outside the scope of the quantifier $\forall x$.

The SCOPE OF A QUANTIFIER, then, is the formula that is included in the brackets attached to the quantifier: $\forall x\,[\dots]$ or $\exists x\,[\dots]$. The variable that is attached to a quantifier may only be used within the scope of the quantifier. Once the bracket indicating the scope of the quantifier closes, you can no longer use it. So you can write:

$$\exists x\,[(Ax \wedge \neg Bx) \rightarrow \neg(Cx \vee \neg Dx)]$$

or

$$\exists x\,[Ax \wedge \exists y\,[By] \rightarrow Cx]$$

but you cannot write:

$$\exists x\,[Ax] \to (Bx)$$

or

$$\exists x\,[Ax \wedge By]$$

There you have it. Those are the basic ingredients of the language of PL.

15.3 The Power of Paraphrase

When symbolising English statements in PL, it is important to understand the structure of the statements you want to symbolise. What matters is the final symbolisation in PL, and sometimes you will be able to move from an English language statement directly to a statement of PL. Other times, it helps to paraphrase the statement one or more times. Each successive paraphrase should move from the original statement to something that you can more easily symbolise directly in PL. We will use this symbolisation key:

s:	Sam Smith
Fx:	x is famous.
Px:	x is a person.
Sx:	x is a pop star.

Consider these statements:

6. If Sam Smith is a pop star, then they are famous.
7. If someone is a pop star, then they are famous.

It may look as though the two statements are quite similar, but we will see that their symbolisation reveals quite a different logical structure. First, the same pronouns appear in the consequent of statements 6 and 7 (‘they’), although they mean different things. In 6, we use the pronoun ‘they’ as a pronoun to refer specifically to Sam Smith. In 7, we use the pronoun ‘they’ as a pronoun to refer to an arbitrary pop star. To make this clear, it helps to paraphrase the original statements, removing pronouns altogether:

8. If Sam Smith is a pop star, then *Sam Smith* is famous.
9. If someone is a pop star, then *that person* is famous.

The paraphrases 8 and 9 express the same things as 6 and 7, albeit in less elegant ways. With these paraphrases, we can proceed to symbolisation in PL, starting with statement 8:

10. $Ss \to Fs$

For statement 9, we need to make a decision as to whether the statement is universal or existential. You might be tempted to think that 9 is an existential statement because it uses ‘someone’, but this would be a mistake. Don’t worry, that’s a common mistake. In fact, 9 talks about arbitrary people that are pop stars, not specific ones, so it is a universal statement. It says, of anyone, that if they are a pop star, then they are famous, which suggests this further paraphrase:

11. Any person who is a pop star is famous.

This looks more like the universal statements we’ve symbolised previously:

12. $\forall x\,[(Px \land Sx) \to Fx]$

The symbolisation makes it clear that 6 and 7, even though they look alike, have a different logical structure in PL. The structure of sentences in everyday language isn’t always the same as the structure of their symbolisation in PL. Should you use an existential or a universal quantifier? Paraphrasing with different words can help you reveal the logical structure of sentences. When you paraphrase statements, don’t worry about expressing things nicely. What is important is to reveal the logical structure, and so make your symbolisation easier.

Practice Exercises

Refer to this symbolisation key for the next two problem sets:

h:	Hulk
p:	Phoenix
Ax:	x is an Avenger.
Lx:	x is a logician.
Mx:	x angry.
Px:	x is a person.

Exercise A: Symbolise the following statements in PL:

1. Some Avengers are angry.
2. Phoenix is a logician, not an Avenger, whereas Hulk is an Avenger, but not a logician.
3. No Avenger is a logician.
4. All logicians are angry, but Phoenix isn't.
5. Everything is an Avenger, a logician, and is angry.

Exercise B:
Express the following formulas of PL in English:

1. $\exists x\,[Px \wedge Lx]$
2. $\forall x\,[Lx \rightarrow Px]$
3. $\neg\forall x\,[Ax \rightarrow Mx]$
4. $\neg\exists x\,[Ax \wedge Lx]$

⋆ **Exercise C:**
Identify the scope of each quantifier in the following formulas:

1. $\exists x\,[Fx] \wedge \exists x\,[Gx]$
2. $\exists x\,[Fx] \wedge \exists y\,[Fy]$
3. $\exists x\,[Fx \wedge \exists y\,[Gy]]$
4. $\exists x\,[Fx \rightarrow \exists y\,[(Gy \vee Hy) \wedge \neg\forall z\,[Fx \vee Gz]]]$
5. $\neg(Fa \wedge \exists x\,[Ga \wedge Hx])$

Exercise D:

Paraphrase the following statements to make their logical structure explicit (where possible):

1. Every pop star is famous
2. If everyone is a pop star, then Sam Smith is.
3. If everyone is a pop star, then they are famous.
4. If they are famous, then Sam Smith is a pop star.
5. If everyone is a pop star, then everyone is famous.
6. If Sam Smith is a pop star, then anyone is.
7. If someone is a pop star, then Sam smith is.
8. If someone is a pop star, then they are famous.
9. If Sam Smith is a pop star, then someone is famous.
10. If Sam Smith is a pop star, then anyone is famous.
11. If nobody is a pop star, then Sam Smith isn't.
12. No one is a pop star that isn't famous.
13. If Sam Smith isn't famous, then nobody is.
14. If Sam Smith isn't famous, then someone is.
15. If anyone is, Sam Smith is a pop star.
16. If they are a pop star, then they are famous.

Exercise E: Use the following symbolisation key to symbolise the above statements in PL, based on your paraphrases.

s:	Sam Smith
Fx:	x is famous.
Px:	x is a person.
Sx:	x is a pop star.

Chapter 16
Single Quantifier Statements

We now have all of the pieces of PL. Symbolising more complicated statements will only be a matter of knowing the right way to combine predicates, names, quantifiers, and connectives. There is a skill to this, and there is no substitute for practice.

16.1 Common Quantifier Phrases

Consider these statements:

1. Every animal in the paddock is a sheep.
2. Some sheep are rams.
3. Not all sheep in the paddock are ewes.
4. None of the sheep in the paddock are wethers.

We will work with this symbolisation key:

Ax:	x is an animal
Ex:	x is a ewe
Px:	x is in the paddock
Rx:	x is a ram
Sx:	x is a sheep
Wx:	x is a wether

Try to symbolise the four statements above before reading on.

The first step when symbolising in PL is to figure out if a statement is universal or existential. A statement is universal if it talks about all things of the same type, and it is existential if it talks about some of them. Universal statements often use words like 'every' or 'all', like 'all mice are afraid of cats', whereas existential statements use words like 'some' or 'there is', like 'Some cats like dogs'. Let's start with universal statements:

Statement 1 *Every animal in the paddock is a sheep.*

Statement 1 is about *every* animal in the paddock, so is a universal statement. It talks about everything that is an animal in the paddock, and says of those things that they are sheep. As per our rule of thumb from the previous section, we expect our symbolisation to use a combination of a universal quantifier $\forall$ and a conditional $\rightarrow$. In the antecedent of the conditional, we list the conditions that must be met to qualify, in this case to be an animal and to be in the paddock. In the consequent, we express the relevant properties ascribed to the things that meet the conditions set out in the antecedent, namely that they are sheep. So the symbolisation looks like this:

$$\forall x\,[(Ax \wedge Px) \rightarrow Sx]$$

This formulas says something like this, if you were to read it aloud: 'for all x, if x is an animal and x is in the paddock, then x is a sheep''. This is how the language of PL expresses that every animal in the paddock is a sheep.

Statement 2 *Some sheep in the paddock are rams.*

Statement 2 is about *some* of the sheep in the paddock, and says that at least one of them is a ram. According to our rule of thumb, we expect to use the existential quantifier $\exists$ and conjunction $\wedge$ in our symbolisation. What we need to express is that there's at least one thing that is a sheep and is in the paddock and is *also* a ram:

$$\exists x\,[(Sx \wedge Px) \wedge Rx]$$

Remember our convention about the use of brackets in long conjunctions. The convention is useful with existential statements when they have several conjuncts, in which case we can drop brackets:

$$\exists x\,[Sx \wedge Px \wedge Rx]$$

But never drop the bracket that comes after the existential quantifier. Never write something like this:

$$\exists x\, Sx \wedge Px \wedge Rx$$

The next two common quantifier phrases are negative statements, which are either negated universal or negated existential statements. Each can be expressed in two different ways, depending on whether we take the main connective to be a negation or a quantifier. These examples will illustrate the difference:

Statement 3 *Not all sheep in the paddock are ewes.*

We have two ways of symbolising this statement. Statement 3 starts with a negation 'not', and denies that all sheep in the paddock are ewes. This is one way of approaching the symbolisation. First symbolise that all sheep in the paddock are ewes with $\forall x\,[(Sx \wedge Px) \rightarrow Ex]$, then negate it to obtain:

$$\neg\forall x\,[(Sx \wedge Px) \rightarrow Ex]$$

The second way of approaching the symbolisation is to ask what would be sufficient to make the statement true. All it takes is for one sheep in the paddock to be a ram, in which case it is fair to say that not all sheep in the paddock are ewes. This is the second way we can think of the symbolisation:

$$\exists x\,[Sx \wedge Px \wedge \neg Ex]$$

Logic Corner

Logicians are used to the equivalence between these two symbolisations, because of two logical facts. The first fact is that the pattern of quantifier and negation $\neg\forall$ is logically equivalent to $\exists\neg$. The second is that the negation of a conditional $\neg(A \rightarrow B)$ is equivalent to the conjunction of the antecedent and the negation of the consequent $(A \wedge \neg B)$. This is why logicians think there's no effective difference between these symbolisations:

$$\begin{gathered}\neg\forall x\,[(Sx \wedge Px) \rightarrow Ex]\\ \exists x\,[\neg((Sx \wedge Px) \rightarrow Ex)]\\ \exists x\,[Sx \wedge Px \wedge \neg Ex]\end{gathered}$$

Statement 4 *None of the sheep in the paddock are wethers.*

As with the previous example, there are two ways to symbolise this statement. The first way is to take it at face value with the main connective being a negation. The question to ask, then, is what statement is being negated. Are we denying that all sheep in the paddock are wethers, or that at least one them is? The second option is correct. What we are saying is that we can't find at least one sheep in the paddock that is a wether. So we start by symbolising the statement to be negated as $\exists x\,[Sx \wedge Px \wedge Wx]$. We then put a negation in front of it to obtain our first symbolisation:

$$\neg\exists x\,[Sx \wedge Px \wedge Wx]$$

The second way to approach the symbolisation is to ask this question: does the statement talk about all the sheep in the paddock? Yes! And it says of them that they are not wethers. So here's a symbolisation:

$$\forall x\,[(Sx \wedge Px) \rightarrow \neg Wx]$$

Logic Corner

Logicians are also used to the equivalence between these two symbolisations, because the pattern $\neg\exists$ is logically equivalent to $\forall\neg$. So they think that there's no difference between these symbolisations:

$$\begin{array}{l} \neg\exists x\,[Sx \wedge Px \wedge Wx] \\ \forall x\,[\neg(Sx \wedge Px \wedge Wx] \\ \forall x\,[(Sx \wedge Px) \rightarrow \neg Wx] \end{array}$$

16.2 Ambiguous Predicates

Suppose we want to symbolise this statement:

5. Adina is a skilled surgeon.

We could do it with this symbolisation key:

a:	Adina.
Kx:	x is a skilled surgeon.

The symbolisation of 5 is then:

$$Ka$$

Suppose instead that we want to symbolise this argument:

> The hospital will only hire a skilled surgeon. All surgeons are greedy. Billy is a surgeon, but is not skilled. Therefore, Billy is greedy, but the hospital will not hire him.

We need to distinguish being a *skilled surgeon* from merely being a *surgeon.* So we introduce a predicate for each in our symbolisation key:

b:	Billy
Gx:	x is greedy.
Hx:	The hospital will hire x.
Kx:	x is skilled.
Rx:	x is a surgeon.

The argument can then be symbolised in this way:

$$\begin{array}{rl} & \forall x\,[\neg(Rx \land Kx) \to \neg Hx] \\ & \forall x\,[Rx \to Gx] \\ & Rb \land \neg Kb \\ \therefore & Gb \land \neg Hb \end{array}$$

Next suppose that we want to symbolise this argument:

> Carol is a skilled surgeon and a tennis player. Therefore, Carol is a skilled tennis player.

This argument is invalid. The premise tells us that Carol is a skilled surgeon, and that she is a tennis player, but being a good surgeon doesn't make you a skilled tennis player. With this in mind, suppose we try to expand the previous symbolisation key thus:

b:	Billy
c:	Carol
Gx:	x is greedy.
Hx:	The hospital will hire x.
Kx:	x is skilled.
Rx:	x is a surgeon.
Tx:	x is a tennis player.

A symbolisation of the argument might then look like this:

$$\begin{array}{rl} & Rc \land Kc \land Tc \\ \therefore & Tc \land Kc \end{array}$$

This symbolisation is a disaster. Can you see why? It takes an invalid argument and symbolises it as a valid argument in PL. The problem is that there is a difference between being *skilled as a surgeon* and *skilled as a tennis player*. Symbolising this argument correctly requires two separate predicates, one for each type of skill. We thus use a different symbolisation key:

c:	Carol
Gx:	x is a skilled surgeon.
Px:	x is a skilled tennis player.
Rx:	x is a surgeon.
Tx:	x is a tennis player.

Now we can properly symbolise the argument in this way:

$$(Rc \wedge Gc) \wedge Tc$$
$$\therefore\ Tc \wedge Pc$$

The moral of these examples is that you need to be careful of symbolising predicates in an ambiguous way. Similar problems can arise with predicates like *good*, *bad*, *big*, and *small*. Just as skilled surgeons and skilled tennis players have different skills, big dogs, big mice, and big problems are big in different ways.

Is it enough to have a predicate that means 'x is a skilled surgeon', rather than two predicates 'x is skilled' and 'x is a surgeon'? Sometimes. As statement 5 shows, sometimes we do not need to distinguish between skilled surgeons and other surgeons.

Must we always distinguish between different ways of being skilled, good, bad, or big? No. As the argument about Billy shows, sometimes we only need to talk about one kind of skill. If you are symbolising an argument that is just about dogs, it is fine to define a predicate that means 'x is big.' If you are symbolising an argument that is about dogs and mice, however, it is probably best to use a predicate for 'x is big for a dog' and another one for 'x is big for a mouse.'

16.3 Aristotelian Syllogisms

Aristotle of Stagira (384-322 BCE) is often regarded as the founder of formal logic. Some of his observations are still part of logic today, even if we have become slightly more advanced in the subsequent 2300 years. Three of the distinctions he made when classifying simple statements are still relevant:

1. *Subject / Predicate*: Every simple statement has a subject (what we are talking textit), and a predicate (what we say about the subject).
2. *Universal / Particular*: Perhaps the simplest ways of having a subject subject are to talk about everything in that subject (the universal), or one thing in that subject (the particular).
3. *Affirmative / Negative*: A simple statement either affirms that a subject has a predicate, or denies (negates) that affirmation.

We can use these three distinctions to build up a taxonomy of simple statements, exactly as Aristotle did; and as we did at the start of this chapter:

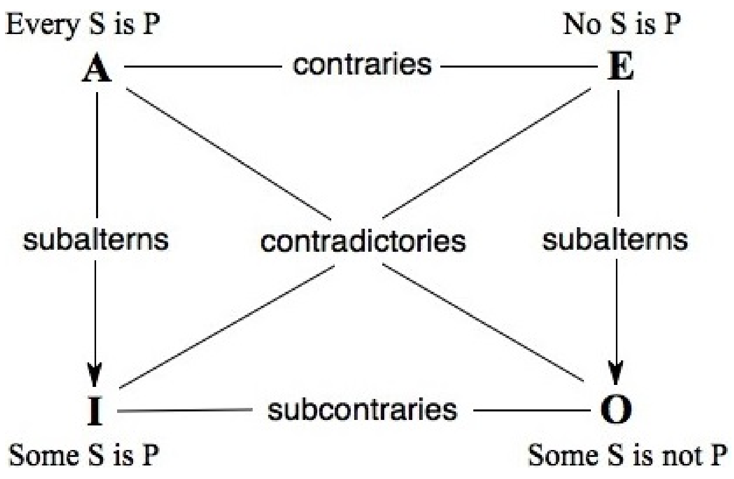
A	All Stamps are Paper (Universal Subject + Affirmative Predicate)
E	All Stamps aren't Paper (Universal Subject + Negative Predicate)
I	Some Stamps are Paper (Particular Subject + Affirmative Predicate)
O	Some Stamps aren't Paper (Particular Subject + Negative Predicate)

There is a mechanical way to convert Aristotle's four simple statement types into modern PL. If a statement has a Universal Subject, it uses a '$\forall$' quantifier, and the Subject and Predicate are connected with an '$\rightarrow$'. If a statement has a Particular Subject, it uses a '$\exists$' quantifier, and the Subject (first predicate) and Predicate (second predicate) are connected with an '$\wedge$'. The Subject (first predicate) is never negated. If a statement is Negative, we negate the Predicate (second predicate). For example:

A	All Stamps are Paper – $\forall x\,[Sx \rightarrow Px]$
E	All Stamps aren't Paper – $\forall x\,[Sx \rightarrow \neg Px]$
I	Some Stamps are Paper – $\exists x\,[Sx \wedge Px]$
O	Some Stamps aren't Paper – $\exists x\,[Sx \wedge \neg Px]$

These four statement types are all that are allowed for Aristotle. We have already seen PLallows a lot more complexity, but it took until the late 1800s before we figured out how to make that work. Having only four statement types also allowed them to describe logical relations between all the statement types, via the Square of Opposition.

The Square of Opposition



In this book, we aren't interested in the logical relations of subalternation, contrariety, or subcontrariety, just contradiction. This means that the Aristotelian Square of Opposition, the jewel of logic carefully passed down on scrolls for 2,000 years, reduces to a straight-forward observation about contradiction.

Aristotle's A-type statement 'All Stamps are Paper' can be symbolised as $\forall x\,[Sx \to Px]$, which is the same as $\neg\exists x\,[Sx \wedge \neg Px]$, which symbolises the negation of the O-type statement 'Some Stamps aren't Paper'. That is, the A and O-type statements are negations of each other, and so are contradictory. Similarly, the E and I-type statements are contradictory.

Note that Aristotelian Syllogisms are **NOT** a simple version of PL; have a different notion validity, and so are a different logic system. But for our current purposes, they do provide a ring-fenced set of highly structured statements for exploration and practice. The Universal Affirmative, Universal Negative, Particular Affirmative, and Particular Negative statement types are still the key building blocks of PL, 2,300 years later.

Logic Corner

Over the next 2,000 years, the number of valid syllogisms varied between 15 and 24, as logicians subtly changed their ideas about validity, primarily by changing their minds about whether every Frog being Green meant that there were Green Frogs. This is the infamous *Existential Import*. Actually, they didn't talk about Frogs, but properties of God; this had major theological significance, and sometimes logic changed with the election of a new Pope.

Practice Exercises

Refer to this symbolisation key for the next two problem sets:

Ax:	x is airborne.
Ix:	x is infectious.
Hx:	x is infectious to humans.
Vx:	x is a virus.

Exercise A: Symbolise the following statements in PL:

1. All viruses are infectious.
2. Not all viruses are infectious.
3. All viruses are not infectious.
4. Some viruses are infectious.
5. No virus is infectious.
6. Some viruses are not infectious.
7. Some viruses are infectious, but not to humans.
8. All viruses are infectious, but none is infectious to humans unless it's airborne.
9. If all viruses are airborne, then some are infectious to humans.
10. If no virus is airborne, then none is infectious to humans.
11. Not all viruses that are infectious to humans are airborne.
12. Some viruses that aren't infectious to humans are airborne.

Exercise B: Using the logical equivalence between negated quantifiers ($\neg\exists$ vs. $\forall\neg$; and $\neg\forall$ vs. $\exists\neg$), as well as equivalences known from TFL, link the statements from the first column to those in the second column that are logically equivalent:

$\neg\forall x\,[Fx]$	$\exists x\,[\neg\neg Fx]$
$\forall x\,[\neg Fx]$	$\exists x\,[\neg Fx]$
$\neg\forall x\,[\neg Fx]$	$\neg\exists x\,[\neg Fx]$
$\neg\neg\forall x\,[Fx]$	$\neg\exists x\,[Fx]$
$\exists x\,[\neg(Px \land Qx)]$	$\neg\exists x\,[Px] \lor \neg\exists x\,[Qx]$
$\neg\forall x\,[Px \land Qx]$	$\neg\forall x\,[Px \land \neg Qx]$
$\neg\neg\exists x\,[Px \to Qx]$	$\neg\forall x\,[Px \land Qx]$
$\exists x\,[Px] \to \forall x\,[\neg Qx]$	$\exists x\,[\neg Px \lor \neg Qx]$

Exercise C: Provide two symbolisation keys for the following argument, and discuss how the treatment of ambiguous predicates affect the validity of the argument:

> Willi is a fluffy cat, but a lousy mouser. Therefore, Willi is a lousy cat.

Exercise D: Using the following symbolisation key:

Kx:	x knows the combination to the safe		
Px:	x is a person	h:	Hofthor
Sx:	x is a spy	i:	Ingmar
Vx:	x is a vegetarian		

Symbolise the following statements in PL:

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.

⋆ **Exercise E:** Using this symbolisation key:

Ax:	x is an alligator.
Mx:	x is a monkey.
Rx:	x is a reptile.
Zx:	x lives at the zoo.
a:	Amos
b:	Bouncer
c:	Cleo

Symbolise each of the following statements in PL:

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. Anything 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.

Exercise F: For each argument, write a symbolisation key and symbolise the argument in PL.

1. Phoenix is a logician. All logicians wear funny hats. So Phoenix wears a funny hat.
2. Nothing on my desk escapes my attention. There is a computer on my desk. Thus 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.

Exercise G: These are some of the syllogistic figures identified by Aristotle and his successors, along with their medieval names.

Symbolise each argument in PL.

Barbara : All Bs are Cs. All As are Bs. ∴ All As are Cs.
Baroco : All Cs are Bs. Some A is not B. ∴ Some A is not C.
Baralipton : All Bs are Cs. All As are Bs. ∴ Some C is A.
Bocardo : Some B is not C. All As are Bs. ∴ Some A is not C.
Celantes : No Bs are Cs. All As are Bs. ∴ No Cs are As.
Celarent : No Bs are Cs. All As are Bs. ∴ No As are Cs.
Cemestres : No Cs are Bs. No As are Bs. ∴ No As are Cs.
Cesare : No Cs are Bs. All As are Bs. ∴ No As are Cs.
Dabitis : All Bs are Cs. Some A is B. ∴ Some C is A.
Darii : All Bs are Cs. Some A is B. ∴ Some A is C.
Datisi : All Bs are Cs. Some A is B. ∴ Some A is C.
Disamis : Some B is C. All As are Bs. ∴ Some A is C.
Ferison : No Bs are Cs. Some A is B. ∴ Some A is not C.
Ferio: No Bs are Cs. Some A is B. ∴ Some A is not C.
Festino : No Cs are Bs. Some A is B. ∴ Some A is not C.
Frisesomorum : Some B is C. No As are Bs. ∴ Some C is not A.

Exercise H: Bonus question. One or two of the syllogisms listed above aren't valid in PL. See if you can figure out which ones, before we introduce the formal methods for testing validity in PL.

Chapter 17
Multiple generality

So far, we have only considered statements that require one-place predicates and one quantifier. The full power of PL really becomes apparent when we start to use many-place predicates and multiple quantifiers.

17.1 Many-place Predicates

All of the predicates that we have considered so far concern properties that things might have. They are ONE-PLACE predicates. Other predicates concern the *relation* between things, such as:

1. Patrick and Sylvie are siblings.
2. Auckland is to the north of Dunedin.
3. Hamilton is in between Auckland and Tauranga.

Statement 1 is about two things, Patrick and Sylvie, and relates them by saying that they are siblings. Statement 2 also relates two things, Auckland and Dunedin, by saying that one is to the north of the other. Statement 3, however, relates three things, Hamilton, Auckland and Tauranga with a relation of *in-betweenness*. We call the relations in statements 1 and 2 TWO-PLACE predicates, and the relation in statement 3 a THREE-PLACE predicate. In principle, there is no upper limit on the number of places that our predicates may have. In practice, more than three is rare.

To keep track of the number of places in relations, as well as their order, we use variables. This is best explained by an example. Suppose we want to symbolise the following statements:

4. Karl loves Jesse.
5. Jesse is loved by Karl.
6. Jesse loves themselves.
7. Karl loves Jesse, but Jesse doesn't love Karl.

We will start with the following symbolisation key:

j:	Jesse
k:	Karl
Lxy:	x loves y

The symbolisation key has two names, j and k, and a two-place predicate Lxy. The use of the variables x and y show that L is a two-place predicate. Furthermore, the order in which they appear in Lxy and in the English key shows that it is x that loves y, and not the other way around. Statements 4 and 5 both express that Karl loves Jesse, but in different voices: 'Karl loves Jesse' is in the active voice, 'Jesse is loved by Karl' in the passive voice. This difference in voices is lost in PL; we symbolise statements 4 and 5 in the same way:

$$Lk, j$$

Statement 6 expresses that Jesse loves Jesse, which illustrates that the same name may occur more than once in a relation:

$$Ljj$$

Statement 7 is a conjunction. The first conjunct expresses that Karl loves Jesse, which we symbolise as

$$Lk, j$$

The second conjunct says that Jesses doesn't love Karl back:

$$\neg Lj, k$$

This again illustrates the importance of the order of the places in predicates: Karl loves Jesse, but Jesse doesn't love Karl. Another way of saying this is that the loving relation holds from Karl to Jesse, but not from Jesse to Karl. Putting things together, we symbolise statement 7 like this:

$$Lkj \wedge \neg Ljk$$

When we are dealing with predicates with more than one place, we need to pay careful attention to the order of the places.

17.2 Quantifier Order

Consider the ambiguous statement ‘everyone loves someone’. Why is it ambiguous? Because it could mean either of the following:

8. Everyone loves a person, though not necessarily the same person.
9. There is some particular person whom everyone loves.

The language of PL may not be the most beautiful language you’ve encountered, but it is very good at avoiding ambiguity. It avoids ambiguity not only by using the order of variables in predicates, but also the order of quantifiers in formulas. The difference between 8 and 9 is that the first expresses a $\forall\exists$ pattern of quantification whereas the second expresses a $\exists\forall$ pattern. More precisely, take this symbolisation key:

Px:	x is a person
Lxy:	x loves y

We can then symbolise statements 8 and 9 like this:

$$\forall x\,\exists y\,[(Px \wedge Py) \to Lxy]$$
$$\exists x\,\forall y\,[(Px \wedge Py) \to Lyx]$$

The point of the example is to illustrate that the order of the quantifiers matters a great deal. Indeed, to switch them around is called a *quantifier shift fallacy.* Here is an example, which comes up in various forms throughout the philosophical literature:

For every person, there is some truth they cannot know. ($\forall\exists$)
$\therefore$ There is some truth that no person can know. ($\exists\forall$)

This argument form is invalid. It’s just as bad as:

Everyday someone is struck by lightning. ($\forall\exists$)
$\therefore$ Some poor person is struck by lightning everyday. ($\exists\forall$)

This is why we ask you to take great care with the order of quantification.

17.3 Stepping-stones to Symbolisation

Once we have the possibility of multiple quantifiers and many-place predicates, symbolisation in PL can become quite tricky. However, there are several stepping stones you can make use of. These steps are best illustrated by example. Consider this symbolisation key:

g:	Geraldo
Dx:	x is a dog
Fxy:	x is a friend of y
Oxy:	x owns y

Now let's try to symbolise these statements:

10. Geraldo is a dog owner.
11. There are dog owners.
12. All of Geraldo's friends are dog owners.
13. Every dog owner is a friend of a dog owner.
14. Every dog owner's friend owns a dog of a friend.

Paraphrasing can help with identifying the quantifiers that are appropriate for symbolising a statement. Statement 10 can first be paraphrased as saying that 'Geraldo owns a dog'. This is what it means to be a dog owner. Next, is this a universal or an existential statement? The statement doesn't say that Geraldo owns every dog, but that Geraldo owns at least one dog, so we can paraphrase it as 'There is a dog that Geraldo owns'. This is easier to symbolise:

$$\exists x\,[Dx \land Ogx]$$

Statement 11 says that at least one person is the owner of a dog, so it refers to two objects (the person and the dog) that don't have names, so will have two quantifiers, and their associated variables. It may help to take them one at the time. Let's first paraphrase 11 as, 'There is some x such that x owns a dog'. This takes care of the first quantifier, so we can write, as an intermediary step, '$\exists x\,[x$ owns a dog]'. Now the fragment we have left as 'x

owns a dog' is much like statement 10, except that it is not specifically about Geraldo. So we can symbolise statement 11 by:

$$\exists x\, \exists y\, [Dy \wedge Oxy]$$

We should pause to clarify something here. In working out how to symbolise the last statement, we wrote down '$\exists x\, [x$ owns a dog]'. To be very clear: this is *neither* an PL statement *nor* an English statement, because it uses bits of PL ('$\exists$', 'x') and bits of English ('owns a dog'). It is really is *just a stepping-stone* on the way to symbolising the entire English statement with a PL statement.

Statement 12 talks about all of Geraldo's friends, so it is a universal statement. That's why we first paraphrase it as, 'Every friend of Geraldo also owns a dog'. Using our stepping-stone tactic, we might write

$$\forall x\, [Fxg \rightarrow x \text{ is a dog owner}]$$

This leaves us to deal with the fragment 'x owns a dog', which we've already encountered in statement 10. Because the variable x is already used to talk about Geraldo's friends, we need to introduce a new variable, y:

$$\forall x\, [Fxg \rightarrow \exists y\, [Dy \wedge Oxy]]$$

Speeding up the intermediate steps, statement 13 can be paraphrased as 'For any x that owns a dog, there is someone who owns a dog and whom x is a friend of'. Using our stepping-stone tactic, this becomes

$$\forall x\, [x \text{ owns a dog} \rightarrow \exists y\, [y \text{ owns a dog} \wedge Fxy]]$$

Completing the symbolisation, we end up with

$$\forall x\, [\exists z\, [Dz \wedge Oxz] \rightarrow \exists y\, [\exists w\, [Dw \wedge Oyw] \wedge Fxy]]$$

Statement 14 is the trickiest yet. First we paraphrase it as 'For any x that is a friend of someone who owns a dog, x owns a dog, and that dog is also owned by a friend of x'. Using our stepping-stone tactic, this becomes:

$$\forall x\, [x \text{ is a friend of someone who owns a dog} \rightarrow \\ x \text{ owns a dog which is also owned by a friend of } x]$$

Breaking this down a bit more:

$$\forall x\, [\exists y\, [Fxy \wedge y \text{ owns a dog}] \rightarrow \\ \exists z\, [Dz \wedge Oxz \wedge z \text{ is owned by a friend of } x]]$$

And a bit more:

$$\forall x\,[\exists y\,[Fxy \,\wedge\, \exists w\,[Dw \,\wedge\, Oyw]] \;\rightarrow\; \exists z\,[Dz \,\wedge\, Oxz \,\wedge\, \exists v\,[Fvx \,\wedge\, Ovz]]]$$

And we are done. We told you, things get tricky quickly!

Hopefully, using these stepping-stones between English and PL will help you simplifying the problem of symbolising complex statements into formulas by breaking down the tasks into simpler ones.

17.4 De-Symbolising

So far we've looked at symbolising statements, which goes from a natural language (English) to a formal language (PL). When you start working with formal languages like PL, however, you will be working directly with the symbolisation, but will still have to communicate about it in natural language. You will then need to be able to read formulas, or as we like to put it, to *de-symbolise.* Using a stepping-stone approach is very useful for that. Let's see some examples, working with this symbolisation key:

Ix:	x is an instrument.
Mx:	x is a musician.
Pxy:	x plays y.

We will desymbolise these two formulas back to English:

15. $\forall x\,[Mx \rightarrow \exists y\,[Iy \wedge Pxy]]$
16. $\exists x\,[Mx \wedge \forall y\,[Iy \rightarrow Pxy]]$

Start with formula 15. One step in desymbolising a formula is to simply read every symbol out loud, as in:

> For every x, if x is a musician, then there is a y such that y is an instrument and x plays y.

This is (nearly) a sentence of English, as you have transformed most parts of the formula into English, but it's only a stepping stone towards a naturally flowing English sentence. Our stepping-stone approach to desymbolising has as an end-goal to produce a nice sentence of English.

Let's try again, taking each part of the formula in turn. One useful way to divide up a complex formula is quantifier by quantifier. Formula 15 has two quantifiers. We will start by transforming the outermost quantifier:

> For every x, if x is a musician, then $\exists y\,[Iy \wedge Pxy]$

Now that we know what we are talking about, namely everything that is a musician, we can express this part of the formula in English with:

> Every musician . . . $\exists y\,[Iy \wedge Pxy]$

Next we transform the existential quantifier:

> Every musician . . . there is a y such that y is an instrument and Pxy.

This tells us that every musician does something to an instrument. What is it?

> Every musician . . . there is a y such that y is an instrument and x plays y.

Now we have all the information we need: for every musician there is an instrument that the musician plays, or written elegantly:

> Every musician plays an instrument.

And this is how we desymbolise formula 15 into English. For formula 16, we follow a similar procedure. We transform the outermost quantifier first:

> There is a x such that x is a musician and $\forall y\,[Iy \rightarrow Pxy]$

So we know that something is going on with a musician:

> Some musician . . . $\forall y\,[Iy \rightarrow Pxy]$

Next we transform the universal quantifier:

> Some musician . . . for all y, if y is an instrument, then Pxy.

What does the musician do to every instruments? They play it:

> Some musician . . . for all y, if y is an instrument, then x plays y.

Now we have all the information needed to express it in English:

> Some musician plays all instruments.

17.5 Suppressed Quantifiers

Logic can often help to clarify statements, especially where the quantifiers are left implicit or their order is ambiguous or unclear. The clarity of expression and thinking afforded by PL can give you a significant advantage in argument, as can be seen in the following takedown by British political philosopher Mary Astell (1666–1731) of her contemporary, the theologian William Nicholls. In Discourse IV: The Duty of Wives to their Husbands of his *The Duty of Inferiors towards their Superiors, in Five Practical Discourses* (London 1701), Nicholls argued that women are naturally inferior to men. In the preface to the 3rd edition of her treatise *Some Reflections upon Marriage, Occasion'd by the Duke and Duchess of Mazarine's Case; which is also considered,* Astell responded as follows:

> 'Tis true, thro' Want of Learning, and of that Superior Genius which Men as Men lay claim to, she [Astell] was ignorant of the *Natural Inferiority* of our Sex, which our Masters lay down as a Self-Evident and Fundamental Truth. She saw nothing in the Reason of Things, to make this either a Principle or a Conclusion, but much to the contrary; it being Sedition at least, if not Treason to assert it in this Reign.
>
> For if by the Natural Superiority of their Sex, they mean that *every* Man is by Nature superior to *every* Woman, which is the obvious meaning, and that which must be stuck to if they would speak Sense, it wou'd be a Sin in *any* Woman to have Dominion over *any* Man, and the greatest Queen ought not to command but to obey her Footman, because no Municipal Laws can supersede or change the Law of Nature; so that if the Dominion of the Men be such, the *Salique Law,*[1] as unjust as *English Men* have ever thought it, ought to take place over all the Earth, and the most glorious Reigns in the *English, Danish, Castilian*, and other Annals, were wicked Violations of the Law of Nature!
>
> If they mean that *some* Men are superior to *some* Women this is no great Discovery; had they turn'd the Tables they might have seen that *some* Women are Superior to *some* Men. Or had they been pleased to remember their Oaths of Allegiance and Supremacy, they might have known that *One* Woman is superior to *All* the Men in these Nations, or else they have sworn to very

[1]The Salique law was the common law of France which prohibited the crown be passed on to female heirs.

> little purpose.[2] And it must not be suppos'd, that their Reason and Religion wou'd suffer them to take Oaths, contrary to the Laws of Nature and Reason of things.[3]

We can symbolise the different interpretations Astell offers of Nicholls' claim that men are superior to women: He either meant that every man is superior to every woman, i.e.,

$$\forall x\,[Mx \to \forall y\,[Wy \to Sxy]]$$

or that some men are superior to some women,

$$\exists x\,[Mx \land \exists y\,[Wy \land Sxy]].$$

The latter is true, but then so is

$$\exists y\,[Wy \land \exists x\,[Mx \land Syx]].$$

(some women are superior to some men), so that, Astell says, would be "no great discovery". In fact, since the Queen is superior to all her subjects, it's even true that some woman is superior to every man, i.e.,

$$\exists y\,[Wy \land \forall x\,[Mx \to Syx]].$$

But this is incompatible with the "obvious meaning" of Nicholls' claim, i.e., the first reading. So what Nicholls claims amounts to treason against the Queen!

[2]In 1706, England was ruled by Queen Anne.

[3]Mary Astell, *Reflections upon Marriage*, 1706 Preface, iii–iv, and Mary Astell, *Political Writings*, ed. Patricia Springborg, Cambridge University Press, 1996, 9–10.

Practice Exercises

Exercise A: Using the following symbolisation key:

i:	me
Px:	x is a person.
Wx:	x is a winner.
Tx:	x is a time.
Lxy:	x loves y.
$Hxyz$:	x loves y at z.

symbolise the following statements in PL:

1. Everybody loves a winner.
2. Nobody loves me.
3. Everybody loves somebody sometime.

Exercise B: Using the following symbolisation key:

i:	me
Px:	x is a person.
Wx:	x is a winner.
Tx:	x is a time.
Lxy:	x loves y.
$Hxyz$:	x loves y at z.

Desymbolise the following formulas of PL into English:

1. $\exists x\,[Px \wedge \forall y\,[Wy \rightarrow Lxy]]$
2. $\neg\forall x\,[Lxi]$
3. $\exists x\,[Px \wedge \forall y\,[Py \rightarrow \exists z\,[Tz \wedge Hxyz]]]$

⋆ **Exercise C:** Using this symbolisation key:

a:	Amos
b:	Bouncer
c:	Cleo
Ax:	x is an alligator
Mx:	x is a monkey
Rx:	x is a reptile
Zx:	x lives at the zoo
Lxy:	x loves y

symbolise each of the following statements in PL:

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.

Exercise D: Using this symbolisation key:

r:	Rave
h:	Shane
d:	Daisy
Ax:	x is an animal
Dx:	x is a dog
Sx:	x likes samurai movies
Lxy:	x is larger than y

symbolise the following statements in PL:

1. Rave is a dog who likes samurai movies.
2. Rave, Shane, and Daisy are all dogs.
3. Shane is larger than Rave, and Daisy is larger than Shane.
4. All dogs like samurai movies.
5. Only dogs like samurai movies.
6. There is a dog that is larger than Shane.
7. If there is a dog larger than Daisy, then there is a dog larger than Shane.
8. No animal that likes samurai movies is larger than Shane.
9. No dog is larger than Daisy.
10. Any animal that dislikes samurai movies is larger than Rave.
11. There is an animal that is between Rave and Shane in size.
12. There is no dog that is between Rave and Shane 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. Any animal that is larger than any dog does not like samurai movies.

Exercise E: Using this symbolisation key:

Cx:	x is a candy.
Mx:	x has marzipan in it.

Sx: x has sugar in it.
Tx: Boris has tried x.
Bxy: x is better than y.

symbolise the following statements in PL:

1. Boris has never tried any candy.
2. Marzipan is always made with sugar.
3. Some candy is sugar-free.
4. No candy is better than itself.
5. Boris has never tried sugar-free candies.
6. Any candy with sugar is better than any candy without it.

Exercise F: Using this symbolisation key:

e: Eli
f: Francesca
g: the guacamole
Dx: x is a dish.
Fx: x is food.
Px: x is a person.
Rx: x has run out.
Tx: x is on the table.
Lxy: x likes y.

symbolise the following English statements in PL:

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.

⋆ **Exercise G:** Using this symbolisation key:

e: Elmer
j: Jane
p: Patrick
Dx: x dances ballet.

Fx:	x is female.
Mx:	x is male.
Px:	x is a person.
Cxy:	x is a child of y.
Sxy:	x is a sibling of y.

symbolise the following statements in PL:

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.

Chapter 18
Identity

Consider this statement:

1. Aristotle is more philosophical than anyone on YouTube.

Using the symbolisation key:

a:	Aristotle
Yx:	x is a person on Youtube
Pxy:	x is more philosophical than y

we could attempt to symbolise statement 1 with:

$$\forall x\,[Yx \to Pax]$$

This symbolisation isn't accurate, however. Can you see why? The problem is that Aristotle is also on YouTube, and if Aristotle is more philosophical than anyone on YouTube, then Aristotle is more philosophical than themself, which is absurd. What statement 1 is meant to say is rather something like this:

2. Aristotle is more philosophical than *anyone else* on YouTube.

but we do not have the resources in PL to express *anyone else*. What we have in PL are two quantifiers, the universal quantifier $\forall$ that talks about everything, and the existential quantifier $\exists$ that talks about at least one thing. Neither quantifier allows us to talk about everything *except* Aristotle. You might be tempted by this symbolisation:

$$\forall x\,[Yx \to Pax] \land \neg Paa$$

This says that Aristotle is more philosophical than anyone on YouTube (including Aristotle) and that Aristotle isn't more philosophical than Aristotle. But this formula is a contradiction, and that won't do. The solution is to add identity to PL. Identity is the subject of this chapter.

18.1 Adding Identity

An IDENTITY between terms expresses that we have two ways of talking about one thing. For instance, identity holds between Diana Prince and Wonder Woman, because they are the same person. In PL, we treat identity as a special predicate and we add a special symbol for it, the symbol '='. We *always* adopt the following symbolisation key for identity:

$x = y$: x is identical to y

To say that x and y are identical does not mean *merely* that the objects they refer to are indistinguishable, or that all of the same things are true of them. Rather, it means that x and y refer to *the very same* object.

For the next examples, we will use this symbolisation key:

d:	Diana Prince
p:	Peter Parker
w:	Wonder Woman
$x = y$:	x is identical to y

We use identity to symbolise statements like this:

3. Diana Prince is Wonder Woman.

We can then symbolise statement 3 in this way:

$$d = w$$

Simple! This means that the names 'd' and 'w' both refer to (name) the same thing. One notational difference about identity is that we write $d = w$ instead of $= (d, w)$ like other predicates. This makes it easier to read.

Computer Science Corner

This way of writing identity is what is an *infix* notation, because the identity symbol is written in between its two arguments x and y. In contrast, we have been using a *prefix* notation for our predicates. In terms of expressivity, the difference in notation doesn't make a difference. For some coding purposes, however, prefix notation is taken to be easier to work with, especially with complex codes.

Like other predicates, we can express that identity doesn't hold between two things with a negation. For instance, we could symbolise that Peter Parker isn't Wonder Woman like this:

$$\neg(p = w)$$

but this is not very pretty. We prefer to use the following notation:

$$p \neq w$$

which expresses that Peter Parker isn't (identical to) Wonder Woman.

With this new symbol added to PL, we can now come back to our example about Aristotle. We now use this symbolisation key:

a:	Aristotle
Yx:	x is a person on Youtube
Pxy:	x is more philosophical than y
$x = y$:	x is identical to y

To express that Aristotle is more philosophical than *anyone else* on YouTube, we use identity to exclude Aristotle:

$$\forall x\,[(Yx \wedge x \neq a) \rightarrow Pax]$$

This formula says that Aristotle is more philosophical than anyone on Youtube, except themself. We can also use identity to symbolise something more complex, like:

4. Aristotle is the most philosophical person on YouTube

How might we express this in PL? Here's one reasonable symbolisation:

$$Ya \wedge \forall x\,[Yx \rightarrow (\forall z\,[(Yx \wedge z \neq x) \rightarrow Pxz] \leftrightarrow x = a)]$$

This says that Aristotle is on Youtube, that if anyone is more philosophical than anyone else on Youtube then that person is Aristotle (to avoid ties), and that he is more philosophical than anyone else on Youtube.
Hint To read this formula: break down the $\leftrightarrow$ into an $\rightarrow$ and a $\rightarrow$, and pay attention to the bracketing, and the quantifier scope for z.

Having identity allows us to express more nuanced concepts. But we are also going to have to be more careful, and to accustom ourselves to reading and creating longer formulas.

18.2 Using Identity

We have already seen a couple of ways to use identity. It's time to go through some common applications a little more systematically.

We can state that two English names are identical, as we did when declaring Diana Prince to be Wonder Woman. We symbolised this in PL by $d = w$.

We can also do this with name-like definite descriptions. With this key:

k:	The cutest kitten in Auckland.
b:	Belinda.
a:	Auckland.
$c(x)$:	The cutest kitten in x.

'Belinda is the cutest kitten in Auckland' can be symbolised as $b = k$. Or if we allow functions in our language, we could symbolise this statement as $b = c(a)$. It would also be true that $k = c(a)$.

We can also state that a name satisfies a quantifier. With this key:

j:	Jeremy.
Lx:	x is your lecturer.

'One of your lecturers is Jeremy' can be symbolised by $\exists x\, [Lx \wedge x = j]$. You could also symbolise this simply as Lj, although that might be more suited to symbolising 'Jeremy is your lecturer', a statement with a similar meaning but that we would use in different contexts. The opportunities for subtlety continue to grow!

We can also claim that names are non-identical, such as when we symbolised that Peter Parker was not Wonder Woman, by $p \neq w$. There are a number of common words whose meaning is a kind of non-identity.

Other To be 'other' is to not be the same as. With this key:

i:	Me.
u:	You.
Px:	x is a person.
Txy:	x is thinking of y.

'I was thinking of someone other than you' can be symbolised by the formula $\exists x\, [Px \wedge Tix \wedge x \neq u]$. We might want to add $(u \neq i)$ if we think that grammatical truth is important to the argument.

Another / Else To be ‘another’ is something like being ‘other’, but usually in addition to. ‘I was thinking of another person, not just of you’ can be symbolised using the above key by the formula $\exists x\,[Px \land Tix \land x \neq u] \land Tiu$. We could also symbolise ‘I was not just thinking of you’ similarly, although we don’t know I’m thinking of another *person*: $\exists x\,[Tix \land x \neq u] \land Tiu$. And ‘I wasn’t thinking of you at all’ could be symbolised as $\neg Tiu$, but if we interpret this ambiguous statement as ‘I was thinking of someone else’ it could be symbolised as $\exists x\,[Px \land Tix \land x \neq u] \land \neg Tiu$.

No one else / No one but The phrase ‘no one else’ is a negation of ‘someone else’. Suppose we stated ‘I was thinking of you and no one else’, or ‘I was thinking of no one but you’, or ‘All I thought of was you’. Then using the above key, we can symbolise these sweet whispers as $Tiu \land \forall x\,[Tix \to x = u]$ or more compactly $\forall x\,[Tix \leftrightarrow x = u]$. Isn’t logic romantic!

Also ‘Also’ is something like ‘and’, but the two conjuncts must refer to distinct (non-identical) objects. With this key:

i:	Me.
u:	You.
Fxy:	x is a friend of y.
Hxy:	x is hanging with y.

We can symbolise ‘I hang out with all my friends, and also with you’ as $\forall x\,[Fix \to (Hix \land x \neq u)] \land Hiu$, which is pretty brutal. Contrast that with ‘I hang out with all my friends, including you’ which we would symbolise as $\forall x\,[Fix \to Hix] \land Fiu$.

Except / Unless ‘Except’ works roughly like ‘unless’, except that it can be used for excluding individuals. We can express most ‘except’ statements using an ‘unless’, replacing individuals in the exception with a statement. For example, ‘I hang out with all my friends, except you’ means roughly the same as ‘I hang out with all my friends, unless they are you’, or ‘I hang out with all my friends, if they are not you’.

We can symbolise any of these statements using the above key as $\forall x\,[(Fix \land x \neq u) \to Hix] \land Fiu$. Compare this with the similar ‘also’ sentence above. Here you are a friend, and we leave it open whether I hang out with you or not. If you think that the English sentence logically excludes that I hang out with you, we can change the formula from an ‘$\to$’ to a ‘$\leftrightarrow$’: $\forall x\,[(Fix \land x \neq u) \leftrightarrow Hix] \land Fiu$.

Only As promised in Chapter 6, we are finally returning to discuss 'only' in more detail. It's worth have another read of §6.4 first. With this key:

j:	Janice.
Hx:	x can talk.

We can symbolise 'Only humans talk' as $\forall x\,[\neg Hx \to \neg Tx]$, or $\forall x\,[Tx \to Hx]$. But it doesn't mean that all humans talk $\forall x\,[Hx \to Tx]$, and so it can't be symbolised as $\forall x\,[Hx \leftrightarrow Tx]$. This is just like what we said when symbolising 'only' in TFL. But with identity, we can also consider individuals. And our rule of thumb of negating both the antecedent and consequent of our conditional still applies when we are making statements about individuals. The statement 'Only Janice can talk' would be symbolised as $\forall x\,[x \neq j \to \neg Tx]$; we do not include Tj, as it does not automatically also mean that in all contexts.

The One and Only / The Only The expression 'The one and only' is the name-level analogue to the propositional 'if and only if'. It is often shortened to 'the only'. This expression does include both halves of the logical content. The expression 'Janice is the (one and) only person who can talk' is roughly 'Janice is one person who can talk AND only Janice can talk', which can be would be symbolised as $Tj \wedge \forall x\,[x \neq j \to \neg Tx]$, or as the equivalent but more compact $\forall x\,[Tx \leftrightarrow x = j]$.

The This now gives us another approach to understanding and symbolising definite descriptions. To say 'the A' is to say that there is at least one thing that is A, and that only that thing is A. With this key:

a:	The accountant.
Ax:	x is an accountant.
Sx:	x smirked.

We can symbolise 'The accountant smirked' as either: Sa by treating 'the accountant' as standing for a name, or as $\exists x\,[Ax \wedge \forall y\,[Ay \to x = y] \wedge Sx]$. This latter approach would naturally lead us to consider statements where a definite description does not refer to exactly one object as being false. Considerations from our previous discussion on definite descriptions then help us decide whether we think this is an adequate symbolisation or not.

18.3 Counting on Identity

At least. . .

We can also use identity to say how many things there are of a particular kind. For example, consider these statements:

5. There is at least one apple
6. There are at least two apples
7. There are at least three apples

We will use the symbolisation key:

Ax: x is an apple

Statement 5 does not require identity. It can be adequately symbolised by '$\exists x\,[Ax]$': There is an apple; perhaps many, but at least one.

It might be tempting to also symbolise statement 6 without identity. Yet consider the statement '$\exists x\,\exists y\,[Ax \wedge Ay]$'. Roughly, this says that there is something x that is an apple and something y that is an apple. But since nothing prevents these from being the same apple, this would be true even if there were only one apple. In order to make sure that we are dealing with *different* apples, we need an identity predicate. Statement 6 needs to say that the two apples that exist are not identical, and so we symbolise it by:

$$\exists x\,\exists y\,[(Ax \wedge Ay) \wedge x \neq y].$$

Statement 7 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:

$$\exists x\,\exists y\,\exists z\,[(Ax \wedge Ay \wedge Az) \wedge (x \neq y \wedge y \neq z \wedge x \neq z)].$$

Note that it is *not* enough to use '$x \neq y \wedge y \neq z$' to symbolise 'x, y, and z are all different.' For that would be true if x and y were different but $x = z$. Nor can we use the mathematical short-hand '$x \neq y \neq z$' As logicians, we aspire to avoid the casual notational abuse and general sloppiness of mathematicians.

At most... Now consider these statements:

8. There is at most one apple
9. There are at most two apples

Statement 8 can be paraphrased as 'It is not the case that there are at least *two* apples'. This is just the negation of statement 6:

$$\neg\exists x\,\exists y\,[(Ax \land Ay) \land x \neq y]$$

But statement 8 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 symbolised by

$$\forall x\,\forall y\,[(Ax \land Ay) \to x = y]$$

The two statements will turn out to be logically equivalent.

In a similar way, statement 9 can be approached in two equivalent ways. It can be paraphrased as, 'It is not the case that there are *three* or more distinct apples', so we can write:

$$\neg\exists x\,\exists y\,\exists z\,[Ax \land Ay \land Az \land x \neq y \land y \neq z \land x \neq z]$$

Alternatively we can read it as saying that if you pick out an apple, and an apple, and an apple, then you will have picked out (at least) one of these objects more than once. Thus:

$$\forall x\,\forall y\,\forall z\,[(Ax \land Ay \land Az) \to (x = y \lor x = z \lor y = z)]$$

Another way of saying that is 'if you are showing me three apples, then you must have shown me at least one apple twice'.

This method of counting can get tiresome if we are dealing with complex predicates. For example, consider the statement 'at least 2 cats chased a mouse or two into someone's room'. We might symbolise this as:

$$\begin{aligned}\exists x\,\exists y\,\exists z\,\exists t\,\exists u\,[(&Cat(x) \land Cat(y) \land Person(z) \land Room(t) \land Owns(zt)\\ &\land Mouse(u) \land Chased(xut) \land Chased(yut) \land (x \neq y)\\ &\land \neg\exists v\,\exists w\,[Mouse(v) \land Mouse(w) \land Chased(xvt) \land Chased(yvt)\\ &\land Chased(xwt) \land Chased(ywt) \land (u \neq v \land u \neq w \land v \neq w)]].\end{aligned}$$

Exactly. . . We can now consider precise statements, like:

10. There is exactly one apple.
11. There are exactly two apples.
12. There are exactly three apples.

Statement 10 can be paraphrased as, 'There is *at least* one apple and there is *at most* one apple'. This is just the conjunction of statement 5 and statement 8. So we can write:

$$\exists x\,[Ax] \land \forall x\,\forall y\,[(Ax \land Ay) \to x = y]$$

But it is perhaps more straightforward to paraphrase statement 10 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 write:

$$\exists x\,[Ax \land \forall y\,[Ay \to x = y]]$$

Similarly, statement 11 may be paraphrased as, 'There are *at least* two apples, and there are *at most* two apples'. Thus we could write

$$\begin{aligned}&\exists x\,\exists y\,[(Ax \land Ay) \land x \neq y] \land \\ &\qquad\qquad \forall x\,\forall y\,\forall z\,[(Ax \land Ay \land Az) \to (x = y \lor x = z \lor y = z)]\end{aligned}$$

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 write:

$$\exists x\,\exists y\,[x \neq y \land \forall z\,[Az \leftrightarrow (x = z \lor y = z)]]$$

Finally, consider these statements:

13. There are exactly two things
14. There are exactly two objects

It might be tempting to add a predicate to our symbolisation key, to symbolise the English predicate '_____ is a thing' or '_____ is an object', but this is unnecessary. Words like 'thing' and 'object' do not sort wheat from chaff: they apply trivially to everything, which is to say, they apply trivially to every thing. So we can symbolise Statement 13 or 14 with either of:

$$\exists x\,\exists y\,[x \neq y] \land \neg\exists x\,\exists y\,\exists z\,[x \neq y \land y \neq z \land x \neq z]$$
$$\exists x\,\exists y\,[x \neq y \land \forall z\,[x = z \lor y = z]]$$

Practice Exercises

⋆ **Exercise A:** Using this symbolisation key:

a:	Andrew
k:	Kim
Px:	x is a person
Lx:	x loves y
$x = y$:	x is identical to y

symbolise each of the following statements in PL:

1. Everyone loves someone else.
2. Kim loves no one but herself.
3. Andrew is the only one who doesn't love Kim.
4. Only Kim loves Andrew.

Exercise B: Explain why:

- '$\exists x \, \forall y \, [Ay \leftrightarrow x = y]$' is a good symbolisation of 'there is exactly one apple'.
- '$\exists x \, \exists y \, [x \neq y \wedge \forall z \, [Az \leftrightarrow (x = z \vee y = z)]]$' is a good symbolisation of 'there are exactly two apples'.

Exercise C: Using the following symbolisation key:

h:	Hofthor
i:	Ingmar
Kx:	x knows the combination to the safe.
Px:	x is a person.
Sx:	x is a spy.
Vx:	x is a vegetarian.
Txy:	x trusts y.

symbolise the following statements in PL:

1. Hofthor trusts a vegetarian.
2. Everyone who trusts Ingmar trusts a vegetarian.
3. Everyone who trusts Ingmar trusts someone who trusts a vegetarian.
4. Only Ingmar knows the combination to the safe.
5. Ingmar trusts Hofthor, but no one else.
6. The person who knows the combination to the safe is a vegetarian.
7. The person who knows the combination to the safe is not a spy.

⋆ **Exercise D:** Using the following symbolisation key:

Bx:	x is black.
Cx:	x is a club.
Dx:	x is a deuce.
Jx:	x is a jack.
Mx:	x is a man with an axe.
Ox:	x is one-eyed.
Sx:	x is a card.
Wx:	x is wild.

symbolise each statement in PL:

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. The deuce of clubs is a black card.
9. One-eyed jacks and the man with the axe are wild.
10. If the deuce of clubs is wild, then there is exactly one wild card.
11. The man with the axe is not a jack.
12. The deuce of clubs is not the man with the axe.

Exercise E: Using the following symbolisation key:

Ax:	x is an animal.
Bx:	x is in Farmer Brown's field.
Hx:	x is a horse.
Px:	x is a Pegasus.
Wx:	x has wings.

symbolise the following statements in PL:

1. There are at least three horses in the world.
2. There are at least three animals in the world.
3. There is more than one horse in Farmer Brown's field.
4. There are three horses in Farmer Brown's field.
5. There is a single winged creature in Farmer Brown's field; any other creatures in the field must be wingless.

6. The Pegasus is a winged horse.
7. The animal in Farmer Brown's field is not a horse.
8. The horse in Farmer Brown's field does not have wings.

Exercise F: In this chapter, we symbolised 'Nick is the traitor' by '$\exists x\, [Tx \wedge \forall y\, [Ty \rightarrow x = y] \wedge x = n]$'. Two equally good symbolisations would be:

- $Tn \wedge \forall y\, [Ty \rightarrow n = y]$
- $\forall y\, [Ty \leftrightarrow y = n]$

Explain why these would be equally good symbolisations.

Part VI

Models

Chapter 19
Predicate Models

19.1 Models

In TFL we use truth tables to calculate truth values of complex formulas based on the values of simpler ones, and see how the truth-value of formulas impact on that of others. In PL, because we are dealing with a more complex language, we need something more complex than truth tables: *models*. Models have a domain, predicate tables, relation tables, and identity tables. Models, like truth tables, are based on stipulations about the simplest bits of our language, namely names and predicates. They give possible ways information about predicates and names could be combined. With this basic information provided by models, we can use logic to find the truth value of more complex formulas. Like in TFL, we can also use models to figure out semantic notions that hold between formulas (consistency, validity, etc).

19.2 The Domain

The *domain* is the collection of things that we are talking about. So if we want to talk about people in Auckland, we define the domain to be people in Auckland. The quantifiers *range over* the domain. Given this domain, '$\forall x$' can be read roughly as 'Every person in Auckland is such that. . . ' and '$\exists x$' as 'Some person in Auckland is such that. . . '.

In PL, the domain must always include at least one thing. We insist that each name must pick out exactly one thing in the domain. If we want to name people in places beside Auckland, then we need to include those people in the domain.

A domain must have *at least* one member. Each member of the domain must be named. A name must pick out *exactly* one member of the domain, but a member of the domain may be picked out by more than one name.

Because of the close relationship between names and objects in the domain, we use a notation that makes it clear which item of the domain names refer to. If a is a name in a symbolisation key, we will write $\texttt{a}$ for the object of the domain it refers to. Take for instance this symbolisation key:

a:	Andrew
e:	Emily
k:	Kim
Px:	x is a philosopher of science.
Lxy:	x works with y.
$x = y$:	x is identical to y.

Models for this key will have at most three items, because each item in the domain must be named, and the key only has three names. In a model, we would write a domain as:

$$\texttt{domain} = \{\texttt{a}, \texttt{e}, \texttt{k}\}$$

The name a, e and k respectively refer to items $\texttt{a}, \texttt{e}$ and $\texttt{k}$ of the domain. We could have a different model with a different domain in which two names refer to the same item:

$$\texttt{domain} = \{\texttt{a}, \texttt{e}(= \texttt{k})\}$$

This is a domain with two items $\texttt{a}$ and $\texttt{e}$, and the names e and k from the symbolisation key refer to the same item: $\texttt{e}$.

Logic Corner

When you start to do advanced work in logic, the assumption that every member of the domain must be named is dropped. If you think about it, that makes sense, because we don't have names for everything. If we did, every rock, every leaf of grass, indeed every subatomic particle would have a name. In this course, however, we will make our lives much simpler by requiring everything in the domain to have a name.

Mathematics Corner

We acknowledge mathematical reasons against the assumption that everything in a domain has a name. These go back to the work of Georg Cantor in the Nineteenth century about sizes of infinity. If a domain has an uncountable number of items (for instance the domain that contains all real numbers), then it is *impossible* to name them all! This follows from what some of you might know as the *pigeon-hole principle*. Such mathematical sophistication goes well beyond the practical purposes of this book.

Numbers and numerals provide an excellent way to appreciate the distinction between items and names. You may have wondered about the difference between the number 1 and the numeral 'one'. Consider the following domain and symbolisation key:

$$\texttt{domain} = \{1, 2, 3, 4\}$$

i:	One
ii:	Two
iii:	Three
iv:	Four

There are many ways we can assign the names 'i', 'ii', 'iii', and 'iv' to our domain. The Romans decided to give the item 1 the name 'i', 2 the name 'ii', 3 the name 'iii', and 4 the name 'iv'. In English we use the names 'one', 'two', 'three', and 'four'. Numerals (in any language) are merely names that refer to numbers. Numbers are the actual items in the mathematical domain.

19.3 Predicate tables

Carol has a small dog called Benji, and a Maine Coon cat called Louie. Maine Coon cats are big! We will analyse this case with this symbolisation key:

b:	Benji
l:	Louie
Bx:	x is big.
Dx:	x is a dog.
Mx:	x is a Maine Coon cat.

We record information about Carol's pets in a model using a *predicate table*:

`domain` $= \{$`b`$,$`l`$\}$

	Bx	Dx	Mx
b	0	1	0
l	1	0	1

The model has a domain with two items, `b` (Benji) and `l` (Louie), and a predicate table that records information about them. Once we have a predicate table, we no longer need to work with the domain. We've collated the data in our table, so we can focus on how to treat the information it contains logically. We will no longer list the domain unless it is important to do so, although every model has a domain at its foundation. So let's focus on the predicate table:

	Bx	Dx	Mx
b	0	1	0
l	1	0	1

In the leftmost column, we list all the names of items in the domain, in this case the items `b` and `l`. The remaining columns, one for each predicate of the symbolisation key, tell us which items they apply to. For example, the second column tells us that b isn't a B ($B(b) = 0$), whereas l is a B ($B(l) = 1$). Hence, this predicate table models a case in which Benji is big, and Louie is big. We can use PL to read off all the basic information of this case:

$$\begin{array}{ccc} \neg Bb & Db & \neg Mb \\ Bl & \neg Dl & Ml \end{array}$$

We can use PL formulas to express the complex information contained in the table:

$$\begin{array}{cc} Db \land \neg Bb & Bl \land Ml \\ Bb \to \neg Db & Ml \lor \neg Bl \end{array}$$

Given a predicate table that record information about Benji and Louie, we can test the truth of complex statements, such as:

1. If Benji is a dog or a Maine Coon cat, then Benji is big.
2. Louie is not a big dog or a Maine Coon cat.

First we symbolise the statements in PL:

3. $(Db \lor Mb) \to Bb$
4. $\neg((Bl \land Dl) \lor Ml)$

Then we use the information from the predicate table to find the truth-values. We start by reading off the values from the predicate table:

$(Db$	$\vee$	$Mb)$	$\to$	Bb
1		0		0

After this step, we no longer need the predicate table. The rest is found by logic, using truth-tables. We start with the disjunction:

$(Db$	$\vee$	$Mb)$	$\to$	Bb
1	1	0		0

Finally, we find the value of the conditional:

$(Db$	$\vee$	$Mb)$	$\to$	Bb
1	1	0	0	0

Our logical analysis of Statement 1 tells us that it is false in the given model. We can similarly use PL to analyse Statement 2, resulting in this truth-table:

$\neg$	$((Bl$	$\wedge$	$Dl)$	$\vee$	$Ml)$
0	1	0	0	1	1

Our analysis of Statement 2 reveals that it is also false in this model.

We can now see the primary role of predicate tables in models: they lay out the basic information of the case being modelled, so that we can truth-functionally compose it into more complicated formulas. The predicate tables for a model in PL serve the same role as the valuation of the propositional variables in TFL.

This means that our counter-example will be represented by a model with predicate tables in PL rather than a simple valuation. Our methods for testing consistency, validity, etc will need to allow for this extra complexity.

Different cases will be modelled by different tables, which give us different information. Consider for instance this case:

	Bx	Dx	Mx
b	1	0	0
l	0	0	1

This predicate table models a case in which Benji, not Louie, is a big Maine Coon cat, and Louie is neither a dog nor a Maine Coon cat, and is not big. With this new predicate table, we again start by reading off the atomic information and plugging it in a truth table symbolising Statement 1:

$(Db$	$\vee$	$Mb)$	$\rightarrow$	Bb
0		0		1

Using logic to find the truth-value of the whole formula, we get:

$(Db$	$\vee$	$Mb)$	$\rightarrow$	Bb
0	0	0	1	1

Again, we use PL to analyse the symbolisation of Statement 2 for this case:

$\neg$	$((Bl$	$\wedge$	$Dl)$	$\vee$	$Ml)$
0	0	0	0	1	1

In this case, Statement 1 is true and statement 2 is false.

Predicate tables list all the possible ways information about predicates and names can be combined. Some are realistic, some aren't, but this is not something logic can assess. What logic can do is to calculate the information of complex formulas given the simple information contained in a predicate table. Logic even allows for weird cases that don't match up with reality, for instance:

	Bx	Dx	Mx
b	1	1	1
l	0	0	0

This is a predicate table that represents a case in which Benji is big, is a dog, and is a Maine Coon cat. It is not realistic, because dogs aren't cats, but logic is not concerned about that. What matters to logic is how the truth and falsity ascribed to simple facts combine into complex logical formulas.

19.4 Many-place Predicates

Predicate tables aren't too bad to understand when they only contain one-place predicates, but it gets messier when we consider two-place predicates. Consider a symbolisation key like:

Lxy: x loves y

Given what we said above, this symbolisation key should be read as saying:

- 'Lxy' and 'x loves y' are true of exactly the same things

So, in particular:

- 'Lxy' is to be true of x and y (in that order) iff x loves y.

It is important that we insist upon the order here, since love – famously – is not always reciprocated. The way we represent information in predicate tables will make the order explicit.

Suppose we are using the following names for people:

b:	de Beauvoir
l:	Lenin
m:	Marx
s:	Sartre

We can use a predicate table to represent who loves who in different cases, such as:

Lxy	b	l	m	s
b	1	0	0	0
l	1	1	1	1
m	0	0	1	0
s	0	0	0	0

Given this table, we read the information in a specific way. The top corner tells us what predicate we are considering, in this case the love predicate Lxy:

Lxy	b	l	m	s
b	1	0	0	0
l	1	1	1	1
m	0	0	1	0
s	0	0	0	0

A entry in the table tells us whether the name on the left column is related by the predicate to the name on the top row, in that order.

Lxy	b	l	m	s
b	1	0	0	0
l	1	1	1	1
m	0	0	1	0
s	1	0	0	0

The table tells us that Ll, m is true, because of the value 1 highlighted in green, which tells us that l (in the left column) is related to m (in the top row) by L. The table also tells us that Lm, l is false, because of the value 0 highlighted in magenta, which tells us that m is not related to l by L. Simply put, the predicate table models a case in which Lenin loves Marx, but Marx doesn't love Lenin. Further information can be read in the same way, for instance that Sartre loves de Beauvoir and no-one else, and that de Beauvoir loves herself and no-one else.

A model and its predicate tables can only tell us of one possible case amongst many others. Here's for instance a case in which everyone (in the domain) loves everyone:

Lxy	b	l	m	s
b	1	1	1	1
l	1	1	1	1
m	1	1	1	1
s	1	1	1	1

This simple table can tell many stories – that of mutual love, unfulfilled yearning, jealousy, betrayal, polyamory, and self-love. Combining it with predicates for sex, hate, and death, and you have the key elements for a heart-wrenching best-seller: 'PL: A Logical Romantic Tragedy'.

Logic Corner

Tables work well for one-place and two-place predicates. With three-place predicates, we would need three-dimensional tables, and for four-place predicates, we would need to step in the fourth-dimension. There are general ways to represent information for cases that are about for complex predicates, but you will only learn about those in upper stage logic.

Practice Exercises

Exercise A: Four superheroes walk into a bar...
This symbolisation key has four names:

b:	Batman
j:	Jessica Jones
s:	Spiderman
w:	Wonder Woman

Provide four models, with one, two, three, and four items in the domain.
Exercise B: Given this predicate table:

	Fx	Gx	Hx
a	1	0	1
b	0	1	0
c	1	1	1
d	0	0	0

What are the truth-values of the following formulas:

1. $Fa, Gb, Gd.$
2. $\neg Fb, \neg Gb, \neg Hb.$
3. $Fa \wedge Ga, Fa \vee Ga, Ga \wedge (Fa \wedge Ha).$
4. $Fc \rightarrow Hc, Fc \rightarrow Gd, Gd \vee \neg Hb.$
5. $Ga \rightarrow (Hc \rightarrow Hd), Ha \rightarrow (Hc \rightarrow Hd).$

Exercise C: Given this predicate table:

	Fx	Gx	Hx
a	1	1	0
b	0	1	1
c	1	0	0
d	0	0	1

What are the truth-values of the formulas from Exercise B?

Exercise D: For each group of formulas, provide a predicate table that makes them all true. If you cannot, explain why.

1. $Fb, Gc, Gd, Pa.$
2. $Fa, \neg Fc, \neg Gb, Gc, \neg He.$
3. $\neg Fa \wedge \neg Fb, Fa \vee Gc, Gc \rightarrow Fb.$
4. $Fa, Fa \rightarrow (\neg Ga \vee Gb), \neg(Gb \vee Fb).$

5. $Ga \to (Hc \to Hd), Ha \to (Hc \to Hd)$.

Exercise E: For each group of formulas, provide a predicate table that makes them all false. If you cannot, explain why.

1. Fb, Gc, Gd, Pa.
2. $Fa, \neg Fc, \neg Gb, Gc, \neg He$.
3. $\neg Fa \wedge \neg Ga, Fa \vee Gc, Gc \to Fb$.
4. $Fa, Fa \to (\neg Ga \vee Gb), \neg(Gb \vee Fb)$.
5. $Ga \to (Hc \to Hd), Ha \to (Hc \to Hd)$.

Exercise F:
Given this two-place predicate table:

Lxy	b	l	m	s
b	1	0	0	0
l	1	1	1	1
m	0	0	1	0
s	0	0	0	0

What is the truth-value of the following formulas:

1. $Lb, b, Lb, s, Ls, s, Lm, l$.
2. $\neg Lb, b, \neg Lm, s, \neg Ll, m$.
3. $Ll, b \wedge Lm, m, Ls, s \vee Ls, l, Lm, s \wedge (Lm, s \vee Ll, l)$.
4. $Ll, m \to Lm, l, Lm, l \to Ll, m$.
5. $Ll, s \to (Lm, m \to (Ll, b \wedge Lm, s))$.

Exercise G: Given these predicate tables:

	Px	Qx
a	0	0
b	1	0
c	0	1
d	1	1

Rxy	a	b	c	d
a	1	0	0	1
b	0	1	1	1
c	0	0	1	0
d	0	1	0	0

Sxy	a	b	c	d
a	0	0	1	0
b	1	1	0	0
c	1	1	1	0
d	0	0	0	0

What is the truth-value of the following formulas:

1. $Pa, \neg Qc, Pa \vee \neg Qd$
2. $Pd \vee Ra, b, \neg Pa \wedge Sc, c, Sd, b \vee Ra, c$.
3. $Qb \to (Ra, c \wedge Sd, d), Pd \to \neg(Rc, c \to Sc, d)$.
4. $\neg((Ra, b \to Rc, a), Pd \wedge Qd, \neg Sc, b \to (Pc \wedge Qa)$.

Exercise H: For each group of formulas, provide a model that makes them all true. If you cannot, explain why.

1. $Pa, \neg Qc, Ra, b, \neg Sb, c$
2. $Pb \wedge Ra, b, \neg Pa \vee Ra, a, Sb, a \vee Rc, a.$
3. $Fa, Fa \rightarrow Gb, La, b \wedge \neg Gb.$
4. $\neg((Ra, b \rightarrow Rc, a), Pd \wedge Qd, \neg Sc, b \rightarrow (Pc \wedge Qa).$

Exercise I: For each group of formulas in Exercise H above, provide a model that makes them all false. If you cannot, explain why.

Chapter 20

Models with Identity

20.1 Identity tables

We talked of identity as a special predicate of PL. We write it a bit differently than other two-place predicates: ‘$x = y$’ instead of ‘Ixy’ (for example). That it is a special predicate is most apparent by looking at *identity tables*. Identity tables are predicate tables that obey the specific principles that govern identity. These principles are:

Reflexivity	Everything is identical to itself.
Symmetry	If a is identical to b, then b is identical to a.
Transitivity	If a is identical to b, and b is identical to c, then a is identical to c.

Mathematics Corner

A relation that is reflexive, symmetric and transitive is called an *equivalence relation.*

Reflexivity That everything is identical to itself is represented in identity tables by (always) having a diagonal of 1’s:

$=$	a	b	c
a	1		
b		1	
c			1

The values highlighted in green represent that $a = a$, $b = b$ and $c = c$. From now on, we will always include a diagonal of 1’s in identity tables.

Symmetry That the order of names in an identity predicate does not matter is represented in identity tables by always copying the value for one ordering to the other. This has an interesting geometric effect on the identity table. You can imagine a mirror being placed over the main diagonal of the truth-table, with the values identical on both sides of the mirror.

$=$	a	b	c
a	1	1	0
b	1	1	
c	0		1

The values highlighted in green represent the symmetry between $a = b$ and $b = a$; both are true. The values highlighted in magenta represent the symmetry of $a = c$ and $c = a$; both are false.

Transitivity Transitivity isn't as easy to spot in identity tables as reflexivity or symmetry, but is just as important.

$=$	a	b	c
a	1	1	1
b		1	1
c			1

The values highlighted in green tell us that $a = b$, and that $b = c$. Transitivity of identity then tells us that $a = c$, which is highlighted in blue. Transitivity could manifest in other configurations of identity tables, such as:

$=$	a	b	c
a	1	1	1
b		1	
c		1	1

$a = c$
$c = b$
$\therefore a = b$

$=$	a	b	c
a	1		1
b	1	1	1
c			1

$b = a$
$a = c$
$\therefore b = c$

$=$	a	b	c
a	1	1	
b		1	
c	1	1	1

$c = a$
$a = b$
$\therefore c = b$

To keep track of transitivity, we recommend you write identity statements under the tables, as we just illustrated.

20.2 Identity & Predicate tables

There's a fundamental principle that describes how identity and predicate tables interact. It is attributed to Wilhelm Gottfried Leibniz in his *Discourse on Metaphysics*, and is called the *indiscernibility of identicals*:

$\boxed{\text{Identical things have exactly the same properties.}}$

For example, Spiderman can shoot webs, and Spiderman is identical to Peter Parker, so Peter Parker can also shoot webs. This principle tells us that the these tables are incompatible:

$=$	p	s
p	1	1
s	1	1

	Wx
p	0
s	1

They are incompatible because the identity table tells us that $p = s$, but the predicate table tells us that $Wp = 0$ and $Ws = 1$.

Identical implies Indiscernible

The principle of *indiscernibility of identicals* helps us to complete information described by partial predicate and identity tables. The principle of *indiscernibility of identicals* tells us that identical things have the same properties. This means these things have the same rows for all their predicate tables. Consider these partial tables:

$=$	a	b	c
a			1
b			
c			

	Px
a	1
b	
c	

We start by using reflexivity, symmetry, transitivity, and the information that $a = c$ to complete as much of the identity table as we can:

$=$	a	b	c
a	1		1
b		1	
c	1		1

	Px
a	1
b	
c	

Next, we apply the *indiscernibility of identicals* to complete the predicate table. We will make sure identical names have the same predicate entries.

Because $a = c$, the rows in the predicate tables for a and for c must be the same, we add a 1 for c under Px:

$=$	a	b	c
a	1		1
b		1	
c	1		1

	Px
a	1
b	
c	1

And this is all that we can do. As we don't have enough information for the remaining entries, we write '?' in the gaps:

$=$	a	b	c
a	1	?	1
b	?	1	?
c	1	?	1

	Px
a	1
b	?
c	1

Philosophy Corner

Leibniz also believed the reverse of the *indiscernibility of identicals*: if two things have the same properties, then they are identical. This is called the *identity of indiscernibles*. Our logic does not follow this principle; we allow that two *distinct* objects might have exactly the same properties.

WARNING We'd really like you to avoid a common mistake when completing tables. If you know that $a = b$ and Pa, then you also know that Pb. But you *can't* reason from $a \neq b$ and Pa to $\neg Pb$. For example John Key and Helen Clark are not the same person, but they are both past prime ministers of New Zealand. Suppose you are given those tables:

$=$	j	h
j		0
h		

	Px
j	1
h	

You can use reflexivity and symmetry to complete the identity table, but that's it. All you have is that $j \neq h$ and that Pj, which leaves open whether Ph or not:

$=$	j	h
j	1	0
h	0	1

	Px
j	1
h	?

Discernible implies Non-identical

When completing tables, you can also use negative information with the *converse indiscernibility of identicals* principle:

$\boxed{\text{Things that have different properties are non-identical.}}$

For example, Serena Williams has won 23 grand slams, and Roger Federer has won 20, so Serena Williams is not identical to Roger Federer. To apply the *converse indiscernibility of identicals* principle, we look for rows that have different entries. For example, consider these partial tables:

$=$	a	b	c
a			
b			
c			

	Px
a	1
b	
c	0

We start with the identity table, using reflexivity, which we can always apply:

$=$	a	b	c
a	1		
b		1	
c			1

	Px
a	1
b	
c	0

Next, we apply *converse indiscernibility of identicals.* We look for rows with different entries:

$=$	a	b	c
a	1		
b		1	
c			1

	Px
a	1
b	
c	0

Because Pa is true, but not Pc, we know that $a \neq c$, and we can include this information in the identity table:

$=$	a	b	c
a	1		0
b		1	
c	0		1

	Px
a	1
b	
c	0

And this is all that we can do. As we don't have enough information for any other entries, we put '?' in the remaining spaces:

$=$	a	b	c
a	1	?	0
b	?	1	?
c	0	?	1

	Px
a	1
b	?
c	0

WARNING We'd also like you to avoid a second common mistake when completing tables. If you know that Pa and $\neg Pb$, then you also know that $a \neq b$. But you can't reason from Pa and Pb to $a = b$. It's quite common for two different things to share a property. For example, John Key and Helen Clark are both past prime ministers of New Zealand, but they are not the same person! Suppose you are given these tables:

$=$	j	h
j		
h		

	Px
j	1
h	1

You can use reflexivity to enter values in the identity table, but that's it. You are given that Pj and Ph, but that doesn't inform you whether j is identical to h:

$=$	j	h
j	1	?
h	?	1

	Px
j	1
h	1

Now, before you accuse us of being conspiracy mongers claiming that John Key might be Helen Clark, let us assure you that's not what we are claiming. We are merely claiming that we haven't been provided enough information to tell them apart, yet.

So what would we need? Well, how do we ever tell people apart – we find something that's different or distinctive about them. For instance, Helen Clark is a member of the Labour party, and John Key is not. Let's add that information to our model:

$=$	j	h
j	1	
h		1

	Px	Lx
j	1	0
h	1	1

You will have noticed that we removed the '?' symbols. When we add any information, we always remove any '?' symbols, because the new information might help us to fill in the blanks. And in this case it does. By the *converse indiscernibiility of identical* principle, as $Lh = 1$ but $Lj = 0$, we know that $h \neq j$. With a little application of symmetry, we have:

$=$	j	h
j	1	0
h	0	1

	Px	Lx
j	1	0
h	1	1

And we've shown that Helen Clark and John Key were different (non-identical) prime ministers of New Zealand!

20.3 Identity & two-place predicate tables

Identity and two-place predicate tables interact in the same way as simple predicate tables. They are regulated by the *indiscernibility of identicals* principle and its converse. The *indiscernibility of identicals* principle tells us that identical things have the same properties and relations. For two-place predicate tables, this means that they have the same rows *and* columns. Consider this example, with the identity table already partially completed:

$=$	a	b	c
a			1
b		1	
c	1		1

Rxy	a	b	c
a	1		
b	0		
c		1	1

As we know that $a = c$, we ensure that a's and c's rows are identical:

$=$	a	b	c
a			1
b		1	
c	1		1

Rxy	a	b	c
a	1	1	1
b	0		
c	1	1	1

Next, we ensure that a's and c's columns are identical:

$=$	a	b	c
a			1
b		1	
c	1		1

Rxy	a	b	c
a	1	1	1
b	0		0
c	1	1	1

To apply the *converse indiscernibility of identicals* principle, we look for rows or columns that have different entries, as in:

$=$	a	b	c
a			
b			
c			

Rxy	a	b	c
a	1		
b	0		
c			

$a \neq b$

Sxy	a	b	c
a			
b	0		1
c			

$a \neq c$

With reflexivity and symmetry, we can complete our identity table:

$=$	a	b	c
a	1	0	0
b	0	1	?
c	0	?	1

Putting everything together, we can complete the following set of tables:

$=$	a	b	c
a		1	
b			
c			

	Fx	Gx
a		1
b	0	
c	0	

Rxy	a	b	c
a		0	1
b	0		
c		0	

We start with the identity table, completing as much as we can:

$=$	a	b	c
a	1	1	
b	1	1	
c			1

$a = b$

	Fx	Gx
a		1
b	0	
c	0	

Rxy	a	b	c
a		0	1
b	0		
c		0	

We gathered the information that $a = b$, which allows us to apply the *indiscernibility of identicals* principle. We ensure that a's and b's rows are the same:

$=$	a	b	c
a	1	1	
b	1	1	
c			1

$a = b$

	Fx	Gx
a	0	1
b	0	1
c	0	

Rxy	a	b	c
a	0	0	1
b	0	0	1
c		0	

Next we ensure that a's and b's columns are the same:

$=$	a	b	c
a	1	1	
b	1	1	
c			1

$a = b$

	Fx	Gx
a	0	1
b	0	1
c	0	

Rxy	a	b	c
a	0	0	1
b	0	0	1
c	0	0	

This is as much information as we can extract from the *indiscernibility of identicals* principle.

You might be tempted to reason that because both b and c have 0 for predicate F, they must be identical. This is one of the errors we just warned you about. It's particularly tempting in this example, as the b and c rows agree for several predicates, including the two-place predicate. However, B and C might just be similar; only the identity table can tell us that two items are identical.

Next we turn to the *converse* principle, looking for entries in tables that allow us to differentiate items:

$=$	a	b	c
a	1	1	
b	1	1	
c			1

	Fx	Gx
a	0	1
b	0	1
c	0	

Rxy	a	b	c
a	0	0	1
b	0	0	1
c	0	0	

$a = b$ $b \neq c$

We see that b and c do not have the same columns, which tells us that they are not equal. We add that information to the identity table, and apply the principle of symmetry:

$=$	a	b	c
a	1	1	
b	1	1	0
c		0	1

	Fx	Gx
a	0	1
b	0	1
c	0	

Rxy	a	b	c
a	0	0	1
b	0	0	1
c	0	0	

$a = b$ $b \neq c$

Now, we see from transitivity that $c \neq a$. Why? If it was true that $c = a$, we would get from transitivity that $c = b$, because $c = a$ and $a = b$ guarantees that $c = b$. But we already know that $c \neq b$. So it must be that $c \neq a$. (We also could have got this information by applying the *converse* principle to columns a and c, as we did for b and c.) This allows us to complete the identity table:

$=$	a	b	c
a	1	1	0
b	1	1	0
c	0	0	1

	Fx	Gx
a	0	1
b	0	1
c	0	

Rxy	a	b	c
a	0	0	1
b	0	0	1
c	0	0	

$a = b$ $b \neq c$, $a \neq c$

That's it! We get no further information, and we complete our tables by putting '?' in the remaining entries:

$=$	a	b	c
a	1	1	0
b	1	1	0
c	0	0	1

	Fx	Gx
a	0	1
b	0	1
c	0	?

Rxy	a	b	c
a	0	0	1
b	0	0	1
c	0	0	?

Practice Exercises

Exercise A: The following are partially-completed identity tables. For each identity table, fill in all the blank spaces that you can, using the principles of reflexivity, symmetry and transitivity. Put a '?' in spaces for which you don't have enough information.

1.

$=$	a	b
a		
b	0	1

2.

$=$	a	b
a	1	
b		

3.

$=$	a	b
a		
b	0	

4.

$=$	a	b
a		1
b		

5.

$=$	a	b	c
a		1	
b			
c	1		

6.

$=$	a	b	c
a	1		
b			
c		1	

7.

$=$	a	b	c
a			0
b			
c		0	

8.

$=$	a	b	c
a			0
b	1		
c			

Exercise B:

The following are partial identity and predicate tables. Fill in all the blank spaces in the tables using only the information already available. Put a '?' in entries for which you don't have enough information.

1.

$=$	a	b
a		
b		

	Fx	Gx
a		
b		

2.

$=$	a	b
a	1	0
b		

	Fx	Gx
a	1	0
b		1

3.

$=$	a	b	c
a			
b			
c			

	Fx	Gx	Hx
a			
b			
c			

4.

$=$	a	b	c
a			1
b			
c			

	Fx	Gx	Hx
a	1		0
b	0	1	
c		0	

5.

$=$	d	e	f
d			
e	1		
f			

	Ax	Bx	Cx
d		0	
e	0		1
f	1	0	

6.

$=$	f	j	n
f			
j			
n			

	Gx	Wx	Qx
f	1	0	0
j			1
n		1	1

7.

$=$	a	b	c
a		1	
b			1
c			

	Fx	Gx	Hx
a	1		
b		0	
c			1

Exercise C:

The following are partial identity and predicate tables. Fill in all the blank spaces in the tables using only the information already available. Put a '?' in entries for which you don't have enough information.

1.

$=$	a	b	c
a			
b			1
c			

	Fx	Gx
a		
b		0
c		

Rxy	a	b	c
a	0		
b			
c		1	

2.

$=$	a	b	c
a		1	
b			
c			

	Fx	Gx
a	0	
b		1
c		0

Rxy	a	b	c
a	0		1
b			
c	1		

3.

$=$	a	b	c
a			1
b			
c		1	

	Fx	Gx
a	0	
b		1
c		

Rxy	a	b	c
a	1		
b			
c			

4.

$=$	a	b	c
a			
b			
c			

	Fx	Gx
a	1	1
b	1	1
c	0	0

Rxy	a	b	c
a	1	1	0
b	0	1	0
c	0	1	1

5.

$=$	a	b	c
a			
b			
c		1	

	Fx	Gx
a		0
b	1	
c		0

Rxy	a	b	c
a	0	0	
b			
c			0

Sxy	a	b	c
a	1	0	
b			
c	0		1

6.

$=$	a	b	c
a			1
b			
c			

	Fx	Gx
a	0	
b	1	1
c		0

Rxy	a	b	c
a	1		
b		1	0
c		1	

Sxy	a	b	c
a	1		
b			0
c		1	

7.

$=$	a	b	c	d
a			1	
b				0
c		0		
d	1			

	Fx	Gx
a	0	
b	1	
c		
d		0

Lxy	a	b	c	d
a	0			
b		1	0	
c				
d		1		

Chapter 21

Models with Quantifiers

21.1 Quantified predicates

We finally come to the most intricate part of PL: quantifiers. What makes calculating the truth-value of quantifiers more difficult is that the result depends on multiple rows of a predicate table. A simple existentially quantified ($\exists$) formula is true if the unquantified formula is true on at least one row of the table, and a simple universally quantified ($\forall$) formula is true if the unquantified formula is true on all the rows. Let's start with an example:

x	Fx	Gx
a	0	1
b	1	1
c	0	1

This table represents a case in which b is F, but not a or c, whereas a, b, and c are all G. Because $Fb = 1$, we know that at least one item of the domain has the property F:

x	Fx	Gx
a	0	1
b	1	1
c	0	1

So $\exists x\,[Fx] = 1$. However, not all items in the domain have property F:

x	Fx	Gx
a	0	1
b	1	1
c	0	1

So we know $\forall x\,[Fx] = 0$.

What about G? The domain of the model has three items: a, b, and c. The predicate G is true of each of them:

x	Fx	Gx
a	0	1
b	1	1
c	0	1

Everything in the domain has the property G. Hence, $\forall x\,[Gx] = 1$.

Let's consider another model, in which everything is an F, but nothing is a G. In this model $\forall x\,[Fx] = 1$ and $\exists x\,[Gx] = 0$, just as we'd expect.

x	Fx	Gx
a	1	0
b	1	0
c	1	0

The truth-value of quantified formulas in models depends on everything in the domain. A simple universal formula $\forall x\,[Fx]$ is true in a model if everything is an F, and it is false if at least one thing is not an F:

$\forall x\,[Fx] = 1$

x	Fx
a	1
b	1
c	1

$\forall x\,[Fx] = 0$

x	Fx
a	1
b	0
c	1

A simple existential formula $\exists x\,[Fx]$ is true if at least one thing in the domain is an F, and it is false if everything is not an F:

$\exists x\,[F(x)] = 1$

x	Fx
a	0
b	1
c	0

$\exists x\,[F(x)] = 0$

x	Fx
a	0
b	0
c	0

But what about complex formulas like $\forall x\,[\neg Fx]$ or $\exists x\,[Fx \wedge Gx]$, or the even more complex $\forall x\,\exists y\,[\neg Lxy]$? We will need a more orderly process, which will be an extension of the complete truth tables that we used for TFL.

21.2 Single Quantifiers

Here is the first model we will use in this section:

x	Fx	Gx
a	0	1
b	1	1
c	0	1

To find the truth value of $\exists x\,[\neg Fx]$, we need to find the truth value of $\neg Fx$ for each item in the domain. We can use something like our familiar TFL truth-table. The initial columns are a little different, however. We have one column for each variable, and one row for each name it can take.

x	$\exists x$	$[\neg$	$Fx]$
a			
b			
c			

We start with the values for Fx for each item. From the model, we know $Fa = 0$, $Fb = 1$, and $Fc = 0$, which we enter in the truth-table:

x	$\exists x$	$[\neg$	$Fx]$
a			0
b			1
c			0

With these values recorded, we can calculate the values of $\neg Fx$ for each item, just as we would calculate negation in TFL:

x	$\exists x$	$[\neg$	$Fx]$
a		1	0
b		0	1
c		1	0

To find the value of the existential quantifier, we need to look for (at least) one row with a 1, which there is:

x	$\exists x$	$[\neg$	$Fx]$
a		1	0
b		0	1
c		1	0

We indicate that the existential quantifier is true like this:

	$\exists x$	$[\neg$	$Fx]$
a		1	0
b	1 {	0	1
c		1	0

The cell of the table highlighted in green is the item that makes the existential quantifier true, and the left brace '{' with the value 1 says that the formula is true of (at least) one of the bracketed items.

A similar table shows that $\forall x\,[\neg Fx] = 0$:

	$\forall x$	$[\neg$	$Fx]$
a		1	0
b	0 {	0	1
c		1	0

The cell highlighted in magenta indicates which item makes the quantified formula false. (The magenta and green cells are just for your reference – the large curly bracket and the truth value are all that you need to write).

Now that we've got the basics, we can look at some more interesting examples. We'll use a simpler model, but add a symbolisation key:

x	Cx
p	0
w	1

p:	Patrick
w:	Willi
Cx:	x is a cat.

These are the statements that we will be evaluating in our model:

1. Not everyone is a cat.
   $\neg\forall x\,[Cx]$
2. Willi is a cat, but not everyone is.
   $Cw \wedge \neg\forall x\,[Cx]$
3. Not everyone is a cat, but Willi is.
   $\neg\forall x\,[Cx] \wedge Cw$
4. Some creatures are cats and some aren't.
   $\exists x\,[Cx] \wedge \exists y\,[\neg Cy]$.

The PL truth table for Statement 1 looks like this:

x	$\neg$	$\forall x$	$[Cx]$
p	1	0 $\{$	0
w			1

We can see that it's not just the quantifier column whose truth value is on a single row. The negation is also not dependent on which item the variable x is referring to, and so it does not need to be repeated on each row. This might be clearer in the PL truth table for Statement 2:

x	Cq	$\wedge$	$\neg$	$\forall x$	$[Cx]$
p	1	1	1	0 $\{$	0
w					1

The rows in our PL truth tables represent the different items that each variable could be referring to, and so when a column is outside the scope of a quantified variable, it only needs to be calculated once across all those items. The braces '{' helps us to see that the information held across several rows is combined by the quantifier into a single row. However, sometimes there is a single row on both sides of a quantified sub-formula, as in Statement 3, and we need a new symbol to indicate the end of a quantifier, the ']':

x	$\neg$	$\forall x$	$[Cx]$	$\wedge$	Cq
p	1	0 $\{$	0	$]$ 1	1
w			1		

Finally, we will introduce a common mistake, and a useful logic technique for avoiding it. Statement 4 has two quantifiers, and we might think its PL table should look something like:

x	y	$\exists x$	$[Cx]$	$\wedge$	$\exists y$	$[\neg$	$Cy]$
p	p		0			1	0
p	w		0			0	1
w	p		1			1	0
w	w		1			0	1

However, we can't easily combine rows 1 and 3, nor rows 2 and 4, for our $\exists y$ quantifier. Instead, *because the two quantifiers are not nested*, we can rename the y variable to x, breaking our rule-of-thumb that every new quantifier gets a new variable.

x	$\exists x$	$[Cx]$	$\wedge$	$\exists x$	$[\neg$	$Cx]$
p	1 $\{$	0	$]$ 1	1 $\{$	1	0
w		1			0	1

Now we have all the tools we need to consider multiple quantifiers.

21.3 Multiple Quantifiers

Let's start with a simple model with one predicate table:

Rxy	a	b
a	0	1
b	1	1

What is the value of $\forall x\,\forall y\,[Rxy \to \forall z\,[Ryz \to Rxz]]$ in this model?
The first step is to figure out how many rows we will need in our table. The difference with tables from the previous section is that we have quantifiers with overlapping scopes. This means that we must consider all possible combinations of items that the variables x, y, and z can take amongst a and b. There's a case in which all variables represent the item a, so we need to find the value of $Ra, a \to (Ra, a \to Ra, a)$; a case in which x represents a, y represents b and z represents a, so we need to find the value of $Ra, b \to (Rb, a \to Ra, a)$; and so on... In total, there are eight possible combinations, which means our empty PL table looks like this:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a								
a	a	b								
a	b	a								
a	b	b								
b	a	a								
b	a	b								
b	b	a								
b	b	b								

Now we can use our predicate table to record the values for the predicates, and then the connectives, in the innermost brackets:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a						0	1	0
a	a	b						1	1	1
a	b	a						1	0	0
a	b	b						1	1	1
b	a	a						0	1	1
b	a	b						1	1	1
b	b	a						1	1	1
b	b	b						1	1	1

Now, we need to find the truth-value for the innermost quantifier $\forall z$. For a universal quantifier to be true, as in the previous section, the formula must take the value 1 for each item in the domain *in which the value of the other variables is fixed.* This means that the universal quantifier will be true (for a set of rows where x and y are both fixed and z varies) if the value is 1 on all of those rows. Both x and y are fixed over the first two rows, so we group the first two rows of the table with a curly bracket, and put a line under it to demarcate it from the other cases. We repeat this for the remaining rows:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a					1	0	1	0
a	a	b						1	1	1
a	b	a					0	1	0	0
a	b	b						1	1	1
b	a	a					1	0	1	1
b	a	b						1	1	1
b	b	a					1	1	1	1
b	b	b						1	1	1

Next, we can add the values for the predicates, and then connectives, in the next layer of brackets:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a			0	1	1	0	1	0
a	a	b						1	1	1
a	b	a			1	0	0	1	0	0
a	b	b						1	1	1
b	a	a			1	1	1	0	1	1
b	a	b						1	1	1
b	b	a			1	1	1	1	1	1
b	b	b						1	1	1

For the quantifier $\forall y$, only the x remains constant, so we will group the first four rows of the table with a curly bracket, and the next four:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a		0	0	1	1	0	1	0
a	a	b						1	1	1
a	b	a			1	0	0	1	0	0
a	b	b						1	1	1
b	a	a		1	1	1	1	0	1	1
b	a	b						1	1	1
b	b	a			1	1	1	1	1	1
b	b	b						1	1	1

Finally, we find the value of $\forall x$, which needs to be true in all cases:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a	0	0	0	1	1	0	1	0
a	a	b						1	1	1
a	b	a			1	0	0	1	0	0
a	b	b						1	1	1
b	a	a		1	1	1	1	0	1	1
b	a	b						1	1	1
b	b	a			1	1	1	1	1	1
b	b	b						1	1	1

The formula $\forall x \forall y [Rxy \to \forall z [Ryz \to Rxz]]$ is therefore false in our model. The case in which x and z are set to a and y is set to b is a case in which we have $Ra, b = 1$ and $Rb, a = 1$, but $Ra, a = 0$, and this is enough to make the whole formula false in our model.

However, in a different model, it may be true. And we don't need to add or remove items from the domain. A small change to the entries in the predicate table should suffice:

Rxy	a	b
a	1	0
b	0	1

Go through the same process that we did above, with this new valuation. Once you've done that, check it against our result:

x	y	z	$\forall x$	$\forall y$	$[Rxy$	$\to$	$\forall z$	$[Ryz$	$\to$	$Rxz]]$
a	a	a	1	1	1	1	1	1	1	1
a	a	b						0	1	0
a	b	a			0	1	0	0	1	1
a	b	b						1	0	0
b	a	a		1	0	1	0	1	0	0
b	a	b						0	1	1
b	b	a			1	1	1	0	1	0
b	b	b						1	1	1

One important point, that we didn't emphasise when we set up the table, is that the order of the variables on the left hand side of the table must be the same order as the quantifiers; the variable from the first quantifier should be first, and so on. This ensures that the rows are grouped in the right order.

For our final example, we'll work with this model:

	Dx	Fx
a	0	0
b	1	1
c	1	1

Kxy	a	b	c
a	1	0	0
b	0	0	0
c	1	1	1

Is the formula $\forall x\,[Dx \to (\exists y\,[Kxy] \wedge Fx]$ true in our model? First, we need to figure out how many rows we will need in our table. As the formula has two variables that can each represent any of the three items in the model, we will need nine rows. We record the values for the innermost predicates. At this point we need to be careful, because only the predicate Kxy is in the scope of $\exists y$:

x	y	$\forall x$	$[Dx$	$\to$	$(\exists y$	$[Kxy]$	$\wedge$	$Fx]$
a	a					1		
a	b					0		
a	c					0		
b	a					0		
b	b					0		
b	c					0		
c	a					1		
c	b					1		
c	c					1		

We can now find the value of $\exists y$:

x	y	$\forall x$	$[Dx$	$\rightarrow$	$(\exists y$	$[Kxy]$	$\land$	$Fx]$
a	a					1		
a	b				1	0		
a	c					0		
b	a					0		
b	b				0	0		
b	c					0		
c	a					1		
c	b				1	1		
c	c					1		

The table tells us that for $x = a$, it is true that $\exists y\,[Kxy]$, and again for $x = c$, but for $x = b$ there is no y such that Kxy. Now we have exited the scope of the $\exists y$ quantifier, and so we will only be interested in the three distinct cases where $x = a$, $x = b$, and $x = c$. So the rest of the table will only use these three rows.
We can now complete the value for the conjunction:

x	y	$\forall x$	$[Dx$	$\rightarrow$	$(\exists y$	$[Kxy]$	$\land$	$Fx]$
a	a					1		
a	b				1	0	0	0
a	c					0		
b	a					0		
b	b				0	0	0	1
b	c					0		
c	a					1		
c	b				1	1	1	1
c	c					1		

The next step is the Dx predicate, and the conditional:

x	y	$\forall x$	$[Dx$	$\rightarrow$	$(\exists y$	$[Kxy]$	$\land$	$Fx]$
a	a					1		
a	b		0	1	1	0	0	0
a	c					0		
b	a					0		
b	b		1	0	0	0	0	1
b	c					0		
c	a					1		
c	b		1	1	1	1	1	1
c	c					1		

Finally, the truth-value of the universal quantifier $\forall x$, and the formula:

x	y	$\forall x$	$[Dx$	$\rightarrow$	$(\exists y$	$[Kxy]$	$\wedge$	$Fx]$
a	a					1		
a	b		0	1	1	0	0	0
a	c					0		
b	a					0		
b	b	0	1	0	0	0	0	1
b	c					0		
c	a					1		
c	b		1	1	1	1	1	1
c	c					1		

The formula $\forall x\,[Dx \rightarrow (\exists y\,[Kxy]\wedge Fx)]$ turns out to be false in our model. And the table even allows us to trace why. When $x = b$, the conditional is false because the conjunction is false, and that's because the formula $\exists y\,[Kxy]$ is false, and that's because there is no y such that Kby.

Practice Exercises

⋆ **Exercise A:** Consider the following model:

	Ax	Bx	Nx
a	0	1	1
b	1	1	0
c	0	1	0

Determine the truth value of the following formulas in this model:

1. Bc
2. $Ac \leftrightarrow \neg Nc$
3. $Nc \to (Ac \lor Bc)$
4. $\forall x\,[Ax]$
5. $\forall x\,[\neg Bx]$
6. $\exists x\,[Ax \land Bx]$
7. $\exists x\,[Ax \to Nx]$
8. $\forall x\,[Nx \lor \neg Nx]$
9. $\forall x\,[Nx \lor \neg Ax]$
10. $\exists x\,[Nx] \lor \neg\exists y\,[Ny]$
11. $\forall x\,[Ax] \lor \neg\forall y\,[Ay]$
12. $\forall x\,[Ax] \lor \forall y\,[\neg Ay]$
13. $\exists x\,[Nx] \lor \exists y\,[\neg Ny]$
14. $\exists x\,[Bx] \to \forall y\,[Ay]$
15. $\forall x\,[Ax \lor Nx] \to \neg\forall y\,[\neg By]$

Exercise B:
Given this model:

	Gx	Hx	Mx
c	0	0	1
e	0	1	0

Determine the truth value of the following formulas in this model:

1. Hc
2. He
3. $Mc \lor Me$
4. $Gc \lor \neg Gc$
5. $Mc \to Gc$
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. $\exists x\,[Hx] \wedge \exists y\,[My]$
13. $\forall x\,[Hx \leftrightarrow \neg Mx]$
14. $\exists x\,[Gx] \wedge \exists y\,[\neg Gy]$
15. $\forall x\,\exists y\,[Gx \wedge Hy]$
16. $\exists x\,[\forall y\,[Gx \wedge Hy]]$
17. $\forall x\,\forall y\,[Gx \rightarrow \neg(Hy \wedge My)]$

Exercise C: Consider the following model:

Rxy	a	b	c
a	1	1	0
b	0	0	0
c	1	0	1

Sxy	a	b	c
a	0	1	0
b	1	1	1
c	0	1	0

Determine the truth value of the following formulas in this model:

1. $\exists x\,[Rxx]$
2. $\forall x\,[Rxx]$
3. $\exists x\,[Sxy]$
4. $\forall x\,[Sxy]$
5. $\exists x\,\forall y\,[Rxy]$
6. $\exists x\,\forall y\,[Rxy]$
7. $\forall x\,\exists y\,[Rxy]$
8. $\forall x\,\exists y\,[Sxy]$
9. $\exists x\,\forall y\,[Rxy]$
10. $\exists x\,\forall y\,[Ryx]$
11. $\forall x\,\forall y\,[(Rxy \wedge Ryx) \rightarrow Rxx]$
12. $\forall x\,\forall y\,[Rxy \rightarrow Sxy]$
13. $\exists x\,\forall y\,[Rxy \rightarrow Syx]$
14. $\exists x\,[Rxx \vee \neg Sxx]$
15. $\exists x\,[Rxx \vee \exists y\,[\neg Syy]]$
16. $\exists x\,\forall y\,[\neg Rxy]$
17. $\forall x\,[\exists y\,[Rxy] \rightarrow \exists z\,[Rxz]]$.
    NB This one requires a big table and we'll understand if you don't complete it. But how many rows do you need?

Exercise D:

Consider the following model:

Rxy	a	b
a	0	1
b	1	0

Sxy	a	b
a	1	0
b	0	1

Determine the truth value of the following formulas in this model:

1. $\exists x\,[Rxx]$
2. $\exists y\,[Syy]$
3. $\exists x\,[\neg Sxx] \vee \exists y\,[\neg Ryy]$
4. $\forall x\,[Rxx] \vee \forall y\,[Syy]$
5. $\forall x\,[\exists y\,[Rxy] \to \exists z\,[Rxz]]$.
6. $\forall x\,\forall y\,[Rxy \to \forall z\,[Szz]]$
7. $\exists x\,\exists y\,\exists z\,[Rxz \wedge (Sxy \vee Syx)]$
8. $\exists x\,\forall y\,[\exists z\,[Rxy \wedge (Ryx \to Rzz)]]$
9. $\exists x\,\exists y\,[Rxy \wedge \exists z\,[Sxx \vee Sxz]]$
10. $\forall x\,\forall y\,[Rxy \to (\exists z\,[Szx] \vee Syx)]$

Part VII

PL Truth Trees

Chapter 22
Reintroducing Truth Trees

In TFL a model was just an assignment of truth values to atomic formulas, so reasoning about all models was relatively straightforward: we could simply list them all in a complete truth table. In PL there isn't a simple method like complete truth tables for reasoning about consistency and validity. So we must use other methods, such as truth trees.

In this Part, we will extend our TFL truth tree method to allow for the extra complexity and symbols in PL. The tree method still tests whether a set of formulas is consistent, and we can still use it to test for properties like consistency and argument validity. If the root formulas are consistent, we will still generate a valuation from the truth tree; it will merely be a valuation for a model containing objects and predicates.

Notation Corner

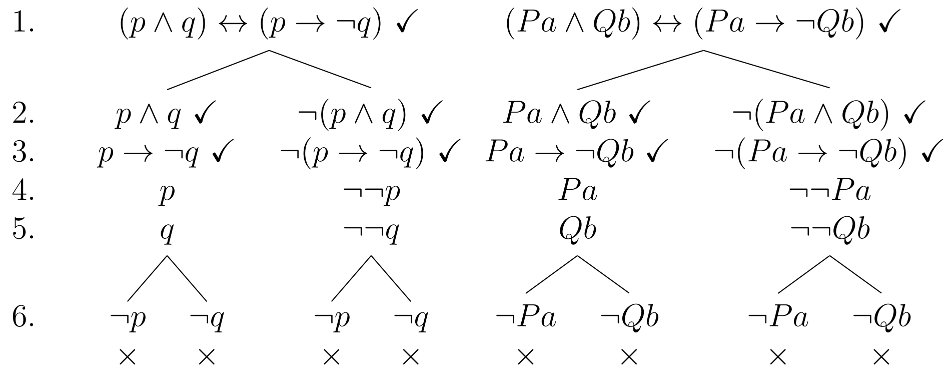
We having been representing predicates as $P(a)$ or $R(x, y)$. But trees require us to write formulas repeatedly, and brackets can be confusing. It's fine to drop predicate brackets now and just write Pa or Rxy.

22.1 Unquantified Trees

PL without quantifiers behaves a lot like TFL. We have the same set of connectives, and atomic formulas such as Pa and Rbc behave exactly like atomic formulas such as p and q in TFL. It doesn't matter whether they are one-place predicates like Pa or many-place like Rbc or even $Skdq$. These atoms are true or false, and that's all the tree method cares about. So, if the root of our PL truth tree has no quantifiers, we can use exactly the same techniques as we did with TFL truth trees.

Closed Trees

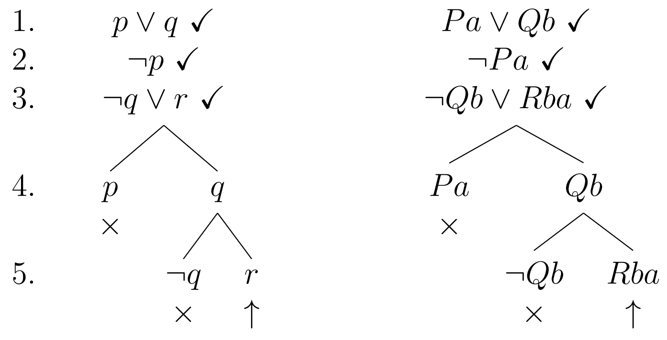
Compare the truth trees that test if the TFL formula $(p \wedge q) \leftrightarrow (p \rightarrow \neg q)$ and the PL formula $(Pa \wedge Qb) \leftrightarrow (Pa \rightarrow \neg Qb)$ are logical falsehoods:


1. $(p \wedge q) \leftrightarrow (p \rightarrow \neg q)$ ✓ $(Pa \wedge Qb) \leftrightarrow (Pa \rightarrow \neg Qb)$ ✓
2. $p \wedge q$ ✓ $\neg(p \wedge q)$ ✓ $Pa \wedge Qb$ ✓ $\neg(Pa \wedge Qb)$ ✓
3. $p \rightarrow \neg q$ ✓ $\neg(p \rightarrow \neg q)$ ✓ $Pa \rightarrow \neg Qb$ ✓ $\neg(Pa \rightarrow \neg Qb)$ ✓
4. p $\neg\neg p$ Pa $\neg\neg Pa$
5. q $\neg\neg q$ Qb $\neg\neg Qb$
6. $\neg p$ $\neg q$ $\neg p$ $\neg q$ $\neg Pa$ $\neg Qb$ $\neg Pa$ $\neg Qb$
× × × × × × × ×

These trees have parallel structures. The decomposition rules for conjunction, disjunction, conditional, biconditional, double negation, and the rules for creating a root, branching, and closing a branch, are identical.

Open Trees

Similarly, compare the truth trees that test if the TFL formulas $p \vee q$, $\neg p$, $\neg q \vee r$ and the PL formulas $Pa \vee Qb$, $\neg Pa$, $\neg Qb \vee Rba$ are mutually consistent:


1. $p \vee q$ ✓ $Pa \vee Qb$ ✓
2. $\neg p$ ✓ $\neg Pa$ ✓
3. $\neg q \vee r$ ✓ $\neg Qb \vee Rba$ ✓
4. p q Pa Qb
× ×
5. $\neg q$ r $\neg Qb$ Rba
× ↑ × ↑

Again, the trees have parallel structures. When we read off the valuation which makes all the formulas true, we use the same process, but the valuations will look different.

The valuation for the TFL tree is $p = 0, q = 1, r = 1$.

The valuation for the PL tree is:

Domain = $\{a, b\}$

	P	Q
a	0	?
b	?	1

Rxy	a	b
a	?	?
b	1	?

We can use tables to check that the valuations make all formulas true. In TFL:

p	q	r	p	$\vee$	q	$\neg$	p	$\neg$	q	$\vee$	r
0	1	1	0	1	1	1	0	0	1	1	0

In PL:

Pa	$\vee$	Qb	$\neg$	Pa	$\neg$	Qb	$\vee$	Rba
0	1	1	1	0	0	1	1	0

22.2 Quantified Trees

If we can remove all the quantifiers from a PL tree, we can simply use the existing TFL tree rules, as we have seen. This will be our goal. The problem is that we don't know which names (or objects) each variable should stand for in our valuations.

We will play it safe by assuming that an existential quantifier *might* be referring to a new object, and so we will always use a new name. (Remember that an object can have more than one name, so if the new name needs to apply to an object we've already named, that's fine). This means that our existential quantifier rule will add new information to our tree, unlike every other rule.

A universal quantifier will always be referring to all existing objects. But we don't know which objects exist, so we will confine ourselves to using old names; names that we know refer to existing objects. As a universal quantifier applies to all existing objects, it might be used several times. This means that our universal quantifier rule will be reusable, unlike every other rule.

We will also need to decompose negated quantifiers, in the same way that we decomposed negated conjunctions and so forth. Here, we will take a short-cut, to simplify our PL trees.

Negated Quantifiers

Negated existential claims are quite similar to universals. They say that no instance is true, which is another way of saying that every instance is false. And in fact:

$$\neg\exists x\ \mathcal{A} \leftrightarrow \forall x\ \neg\mathcal{A}$$

Similarly, a negated universal is similar to an existential. It says that not all instances are true, so at least one instance is false:

$$\neg\forall x\ \mathcal{A} \leftrightarrow \exists x\ \neg\mathcal{A}$$

We will take advantage of these two equivalences to transform all negated quantified formulas into non-negated quantified formulas.

$\neg\exists x\ \mathcal{A}$		$\neg\forall x\ \mathcal{A}$	
$\forall x\ \neg\mathcal{A}$	$i\ \neg\exists$	$\exists x\ \neg\mathcal{A}$	$i\ \neg\forall$

For example, if $\neg\exists x\,[Fx]$ is in the tree, we can decompose it to $\forall x\,[\neg Fx]$. And likewise we can decompose $\neg\forall z\,[Raz \rightarrow Pz]$ to $\exists z\,\neg(Raz \rightarrow Pz)$.

Existential Quantifiers

Our tree rule for decomposing existential quantifiers replaces all instances of the existential variable in the formula with our new object name:

$\exists x\mathcal{A}\ \checkmark\boldsymbol{a}$	
$\mathcal{A}\langle\chi \Rightarrow \boldsymbol{a}\rangle$	$i\ \exists\boldsymbol{a}$
	(where $\boldsymbol{a}$ is new)

Just as $\mathcal{A}$ stands in for any formula of PL, the χ stands in for any variable, and $\boldsymbol{a}$ stands in for any name. The requirement that the name be *new* means that the name chosen for the substitution instance must be a name that has not appeared *anywhere* in the tree so far.

For example, consider this tree:

1.	Fa	Root
2.	$\exists x\,[\neg Fx]\ \checkmark b$	Root
3.	$\neg Fb$	$2\ \exists b$

On line (2), we replace x with b (rather than a), because a is a name that is already in use. We want to say that *something* is F, not necessarily that a is F. Of course, Fa could also be true: there could be more than one object that is F; and it is also possible for a and b to be names for the same object ($a = b$).

There is also a b next to the check mark in line (2). When we mark the existential quantifier as decomposed, we add the name that we've used to replace the quantified variable, so we can trace our work more easily.

Consider this more complex example:

$$\begin{array}{lccl}
1. & \multicolumn{2}{c}{\exists x\,[\exists y\,[Rxy] \to Rbx] \checkmark a} & \text{Root} \\
2. & \multicolumn{2}{c}{\exists y\,[Ray] \to Rba \checkmark} & 1\ \exists a \\
 & \diagup & \diagdown & \\
3. & \neg\exists y\,[Ray] \checkmark & Rba & 2 \to \\
4. & \forall y\,[\neg Ray] & & 3\ \neg\exists
\end{array}$$

On line (2), we decompose line (1) and can replace x with any letter except b, which is already used on line (1). On line (2), the main connective is not $\exists$ but $\to$, so we can't decompose it using our existential tree rule. When we finally have a quantifier as the main connective on line (4), it's now a universal quantifier. We need another rule to decompose this formula.

Universal Quantifiers

Our tree rule for decomposing universal quantifiers replaces all instances of the existential variable in the formula with an old object name:

$$\boxed{\begin{array}{ll}
\forall x\mathcal{A} \setminus \boldsymbol{a} & \\
\mathcal{A}\langle \chi \Rightarrow \boldsymbol{a}\rangle & i\ \forall \boldsymbol{a} \\
 & \text{(where } \boldsymbol{a} \text{ is old)}
\end{array}}$$

For a name to be 'old' means that it has appeared *somewhere* in the tree.

Unlike all the other tree rules, we don't decompose a universally quantified formula and then check it off, never to use it again. Because more than one name can appear in a tree, each universal formula can be used more than once. We mark this by writing a $\setminus$ instead of a $\checkmark$ next to the universal formula.

Here is an example illustrating the importance of decomposing a universally quantified formula multiple times:

1.	$\forall x\,[Fx \wedge Gx]\ \backslash a, b$		Root
2.	$\neg Fa \vee \neg Gb$ ✓		Root
3.	$Fa \wedge Ga$ ✓		$1\ \forall a$
4.	Fa		$3\ \wedge$
5.	Ga		$3\ \wedge$
6.	$\neg Fa$	$\neg Gb$	$2\ \vee$
7.	$\times$	$Fb \wedge Gb$ ✓	$1\ \forall b$
8.		Fb	$7\ \wedge$
9.		Gb	$7\ \wedge$
		$\times$	

On line (3) we replace the universally quantified variable with *a* at line (1), and this helps to close one branch. But the other branch won't close until we take the *b* instance as well on line (7). Of course, we might have foreseen that we would need both a formula containing *a* and one containing *b* to close the tree, and replaced the quantified variable with *b* on line (4); the tree would have worked just as well. But in general, we want to delay acting on universal quantifiers for as long as possible, for reasons we'll explain shortly.

22.3 Completing your Tree

Although the connective, branching, and branch closing rules are identical to those in TFL, we will need to change our completion rule slightly.

The TFL tree completion rule, given in §12.3, said that a branch was complete if every complex formula has been decomposed, as indicated by a check mark. But universally quantified formulas may need to be decomposed several times – once for each name in the tree. We need our completion condition to ensure that we've taken *enough* instances.

For example, this should not count as a completed tree:

1.	$\forall x\,[Fx \to Ga]\ \backslash a$		Root
2.	$\neg Ga$		Root
3.	Fb		Root
4.	$Fa \to Ga$ ✓		$1\ \forall a$
5.	$\neg Fa$	Ga	$4\ \to$
		$\times$	

The universal quantifier on line (1) can be decomposed using a and using b. If we did this, the tree would close:

1.	$\forall x\,[Fx \to Ga] \setminus a, b$	Root
2.	$\neg Ga$	Root
3.	Fb	Root
4.	$Fa \to Ga$ ✓	1 $\forall a$
5.	$\neg Fa \qquad Ga$	4 $\to$
6.	$Fb \to Ga$ ✓ $\qquad \times$	1 $\forall b$
7.	$\neg Fb \quad Ga$	6 $\to$
	$\times \qquad \times$	

We don't want this tree to count as complete until it has decomposed line (1) using both a and b. A true universal formula is true of *every* object; we can't test that formula properly if we've only checked *some* objects.

For an open branch to be complete, every universally quantified formula in that branch must be decomposed using *every* name in the branch.

22.4 More PL Tree Examples

A tautology can never be false. A tree can test if a formula can be false. Is $(\neg\exists x\,[Ax \wedge \neg Bx] \to \forall y\,[Ay \to By])$ a tautology?

1.	$\neg(\neg\exists x\,[Ax \wedge \neg Bx] \to \forall y\,[Ay \to By])$ ✓	Root
2.	$\neg\exists x\,[Ax \wedge \neg Bx]$ ✓	1 $\neg \to$
3.	$\neg\forall y\,[Ay \to By]$ ✓	1 $\neg \to$
4.	$\exists y\,\neg(Ay \to By)$ ✓a	3 $\neg\forall$
5.	$\neg(Aa \to Ba)$ ✓	4 $\exists a$
6.	Aa	5 $\neg \to$
7.	$\neg Ba$	5 $\neg \to$
8.	$\forall x\,\neg(Ax \wedge \neg Bx) \setminus a$	2 $\neg\exists$
9.	$\neg(Aa \wedge \neg Ba)$	8 $\forall a$
10.	$\neg Aa \qquad \neg\neg Ba$	9 $\neg\wedge$
	$\times \qquad \times$	

The tree shows that the negated formula can never be true. This means that the formula can never be false – it is a tautology.

Is the argument 'No flasks are not made of glass so all flasks are made of glass' valid?

$$\neg\exists x\,[Fx \wedge \neg Gx] \therefore\ \forall y\,[Fy \rightarrow Gy]$$

We check if the premises can be true while the conclusion is false:

1.	$\neg\exists x\,[Fx \wedge \neg Gx]$ ✓	Premise
2.	$\neg\forall y\,[Fy \rightarrow Gy]$ ✓	Neg Conc
3.	$\forall x\,[\neg(Fx \wedge \neg Gx)]\ \backslash a$	1 $\neg\exists$
4.	$\exists y\,[\neg(Fy \rightarrow Gy)]$ ✓a	2 $\neg\forall$
5.	$\neg(Fa \rightarrow Ga)$ ✓	4 $\exists a$
6.	Fa	5 $\neg \rightarrow$
7.	$\neg Ga$	5 $\neg \rightarrow$
8.	$\neg(Fa \wedge \neg Ga)$ ✓	3 $\forall a$
9.	$\neg Fa \qquad \neg\neg Ga$	8 $\neg\wedge$
	× ×	

It is impossible for the premises to be true and the conclusion false, so there is no counter-example, and the argument is valid.
Proof that $\forall y\,\exists x\,(Ryy \rightarrow Rxy)$ is a tautology:

1.	$\neg\forall y\,\exists x\,[Ryy \rightarrow Rxy]$ ✓	Root
2.	$\exists y\,\neg\exists x\,[Ryy \rightarrow Rxy]$ ✓a	1 $\neg\forall$
3.	$\neg\exists x\,[Raa \rightarrow Rxa]$ ✓	2 $\forall a$
4.	$\forall x\,[\neg(Raa \rightarrow Rxa)]$ ✓a	3 $\neg\exists$
5.	$\neg(Raa \rightarrow Raa))$ ✓	4 $\exists a$
6.	Raa	5 $\neg \rightarrow$
7.	$\neg Raa$	5 $\neg \rightarrow$
	×	

Proof that $\forall x\,\forall y\,\forall z\,[(Rxy \wedge Ryz) \rightarrow Rxz], \forall x\,[\neg Rxx] \therefore \forall x\,\forall y\,[Rxy \rightarrow \neg Ryx]$ is valid:

1.	$\forall x\,\forall y\,\forall z\,[(Rxy \wedge Ryz) \rightarrow Rxz]\ \backslash a$	Premise
2.	$\forall x\,[\neg Rxx]\ \backslash a$	Premise
3.	$\neg\forall x\,\forall y\,[Rxy \rightarrow \neg Ryx]\ \checkmark$	Neg Conc
4.	$\exists x\,\neg\forall y\,[Rxy \rightarrow \neg Ryx]\ \checkmark a$	3 $\neg\forall$
5.	$\neg\forall y\,[Ray \rightarrow \neg Rya]$	4 $\exists$a
6.	$\exists y\,[\neg(Ray \rightarrow \neg Rya)]\ \checkmark b$	5 $\neg\forall$
7.	$\neg(Rab \rightarrow \neg Rba)$	6 $\exists$b
8.	Rab	7 $\neg\rightarrow$
9.	$\neg\neg Rba$	7 $\neg\rightarrow$
10.	Rba	9 $\neg\neg$
11.	$\forall y\,\forall z\,[(Ray \wedge Ryz) \rightarrow Raz]\ \backslash b$	1 $\forall a$
12.	$\forall z\,[(Rab \wedge Rbz) \rightarrow Raz]\ \backslash a$	11 $\forall b$
13.	$(Rab \wedge Rba) \rightarrow Raa\ \checkmark$	12 $\forall a$
14.	$\neg Raa$	2 $\forall a$
15.	$\neg(Rab \wedge Rba)\ \checkmark$ Raa $\times$	13 $\rightarrow$
16.	$\neg Rab$ $\times$ $\neg Rba$ $\times$	15 $\neg\wedge$

Mathematics Corner

The above argument is a proof from mathematics. It says that any relation that is transitive and anti-reflexive (e.g., the greater-than '$>$' and strict subset '$\subset$' relations) must also be anti-symmetric.

The following argument is invalid:

$$\begin{array}{l} \forall x\,[\neg(\exists y\,[Syx] \wedge \neg Ax)] \\ \forall x\,[\exists y\,[Sxy] \to \neg Bx] \\ \therefore\ \forall x\,[Sxx \to (Ax \wedge Bx)] \end{array}$$

1.	$\forall x\,[\neg(\exists y\,[Syx] \wedge \neg Ax)]\ \backslash a$				Premise
2.	$\forall x\,[\exists y\,[Sxy] \to \neg Bx]\ \backslash a$				Premise
3.	$\neg\forall x\,[Sxx \to (Ax \wedge Bx)]$ ✓				Neg Conc
4.	$\exists x\,[\neg(Sxx \to (Ax \wedge Bx))]$ ✓a				$3\ \neg\forall$
5.	$\neg(Saa \to (Aa \wedge Ba))$ ✓				$4\ \exists a$
6.	Saa				$5\ \neg \to$
7.	$\neg(Aa \wedge Ba)$				$5\ \neg \to$
8.	$\neg(\exists y\,[Sya] \wedge \neg Aa)$ ✓				$1\ \forall a$
9.	$\neg\exists y\,[Sya]$ ✓	$\neg\neg Aa$			$8\ \neg\wedge$
10.	$\forall y\,[\neg Sya]\ \backslash a$	|			$9\ \neg\exists$
11.	$\neg Saa$	|			$10\ \forall a$
12.	$\times$	Aa			$9\ \neg\neg$
13.		$\neg Aa$	$\neg Ba$		$7\ \neg\wedge$
14.		$\times$	$\exists y\,[Say] \to \neg Ba$ ✓a		$2\ \forall a$
15.			$\neg\exists y\,[Say]$ ✓	$\neg Ba$	$14 \to$
16.			$\forall y\,[\neg Say]\ \backslash a$	$\uparrow$	$15\ \neg\exists$
17.			$\neg Saa$		$16\ \forall a$
			$\times$		

Fortunately the tree has produced a one-item counter-example. This will lead to a short counter-example and proof:

Domain = $\{a\}$

Sxy	a
a	1

	A	B
a	1	0

x	y	$\forall x\ [\neg\,(\exists y\ [Syx]\wedge\neg\,Ax)]$
a	a	1 1 1 1 0 0 1

x	y	$\forall x\ [\exists y\ [Sxy]\to\neg\,Bx]$
a	a	1 1 1 1 1 0

x	y	$\neg\forall x\ [Sxx\to(Ax\wedge Bx)]$
a	a	1 0 1 0 1 0 0

$\forall x\,[\exists y\,[Rxy] \to Fx], \forall x\,[Fx \to \neg Rxx] \therefore \forall x \forall y\,[Rxx \vee \neg Rxy]$ is invalid:

1.	$\forall x\,[\exists y\,[Rxy] \to Fx] \setminus a, b$			Premise
2.	$\forall x\,[Fx \to \neg Rxx] \setminus a$			Premise
3.	$\neg\forall x \forall y\,[Rxx \vee \neg Rxy]$ ✓			Neg Conc
4.	$\exists x\,[\neg(\forall y\,[Rxx \vee \neg Rxy])]$ ✓ a			3 $\neg\forall$
5.	$\neg(\forall y\,[Raa \vee \neg Ray])$ ✓			4 $\exists a$
6.	$\exists y\,[\neg(Raa \vee \neg Ray)]$ ✓ b			5 $\neg\forall$
7.	$\neg(Raa \vee \neg Rab)$ ✓			6 $\exists b$
8.	$\neg Raa$			7 $\neg\vee$
9.	$\neg\neg Rab$			7 $\neg\vee$
10.	Rab			9 $\neg\neg$
11.	$\exists y\,[Ray] \to Fa$ ✓			1 $\forall a$
12.	$\neg\exists y\,[Ray]$ ✓	Fa		11 $\to$
13.	$\forall y\,[\neg Ray] \setminus b$	|		12 $\neg\exists$
14.	$\neg Rab$	|		13 $\forall b$
15.	×	$Fa \to \neg Raa$ ✓		2 $\forall a$
16.		$\neg Fa$	$\neg Raa$	15 $\to$
17.		×	$\exists y\,[Rby] \to Fa$ ✓	1 $\forall b$
18.			$\neg\exists y\,[Rby]$ ✓ Fb	17 $\to$
19.			$\forall y\,[\neg Rby] \setminus a, b$ ⋮	18 $\neg\exists$
20.			$\neg Rba$	19 $\forall a$
21.			$\neg Rbb$	19 $\forall b$
			↑	

While the branch under Fb might continue, it can't affect our open branch. Our counter-example and proof is:

Domain = $\{a, b\}$

Rxy	a	b
a	0	1
b	0	0

	F
a	1
b	?

x	y	$\forall x$	$[\exists y$	$[Rxy]$	$\to$	$Fx]$	$\forall x$	$[Fx$	$\to$	$\neg$	$Rxx]$	$\neg$	$\forall x$	$\forall y$	$[Rxx$	$\vee$	$\neg$	$Rxy]$
a	a	1	1	0	1	1	1	1	1	1	0	1	0	0	0	1	1	0
a	b			1											0	0	0	1
b	a		0	0	1	?		?	1	1	0			1	0	1	1	0
b	b			0											0	1	1	0

Chapter 23
Tree Strategies

23.1 Order of Decomposition

You can always decompose the complex formulas of a truth tree in any order you want. The resulting trees will all eventually give the same (or equivalent) results. But some trees are shorter, or easier to find good strategies to complete, or easier to complete without making mistakes. Here's the advice we gave for TFL trees:

1. Non-branching rules before branching.
2. Close branches quickly.
3. Simple formulas before complex.
4. Biconditionals last.

All this still holds. However, we also have some quantifier-specific advice.

1. Negated Quantifiers are simple non-branching rules; do them early.
2. Existential Quantifiers after other non-branching rules.
3. Universal Quantifiers only when you have a clear plan for them.
4. Otherwise, delay Universal Quantifiers if there are many names.

This advice isn't perfect, but it will generally serve you well.

Existential quantifiers generate new information and new possibilities. Every existential quantifier will add a new name. Negated quantifiers will either become existential quantifiers, to be decomposed immediately, or universal quantifiers, to be planned for and used wisely (or often).

Because universal quantifiers can generate so many new formulas, you should either use them only when you have a clear plan, or when you are out of plans; at which point, use all the names, and see if anything useful appears!

Let's test if $\forall x\, Rax, \forall x\, \forall y\, (Rxy \to Ryx) \therefore \forall y\, Rya$ is valid.

$$\begin{array}{lcl}
1. & \forall x\,[Rax] & \text{Premise} \\
2. & \forall x\,\forall y\,[Rxy \to Ryx] & \text{Premise} \\
3. & \neg\forall y\,[Rya] & \text{Neg Conc}
\end{array}$$

If we ignored (3) and decomposed our universals first, we'd get

$$\begin{array}{lccl}
1. & \multicolumn{2}{c}{\forall x\,[Rax] \setminus a} & \text{Premise} \\
2. & \multicolumn{2}{c}{\forall x\,\forall y\,[Rxy \to Ryx] \setminus a} & \text{Premise} \\
3. & \multicolumn{2}{c}{\neg\forall y\,[Rya] \checkmark} & \text{Neg Conc} \\
4. & \multicolumn{2}{c}{Raa} & 1\ \forall a \\
5. & \multicolumn{2}{c}{\forall y\,[Ray \to Rya] \setminus a} & 2\ \forall a \\
6. & \multicolumn{2}{c}{Raa \to Raa} & 5\ \forall a \\
 & \swarrow & \searrow & \\
7. & \neg Raa & Raa & 6 \to \\
8. & \times & \exists y\,[\neg Rya] \checkmark b & 3\ \neg\forall \\
9. & & \neg Rba & 8\ \exists b
\end{array}$$

The tree hasn't closed yet, and the new name b on line (8) means we need to rethink our use of lines (1), (2) and (5). If we instead delayed our universals until we had a plan, we might get:

$$\begin{array}{lccl}
1. & \multicolumn{2}{c}{\forall x\,[Rax] \setminus b} & \text{Premise} \\
2. & \multicolumn{2}{c}{\forall x\,\forall y\,[Rxy \to Ryx] \setminus a} & \text{Premise} \\
3. & \multicolumn{2}{c}{\neg\forall y\,[Rya] \checkmark} & \text{Neg Conc} \\
4. & \multicolumn{2}{c}{\exists y\,[\neg Rya] \checkmark b} & 3\ \neg\forall \\
5. & \multicolumn{2}{c}{\neg Rba} & 4\ \exists b \\
6. & \multicolumn{2}{c}{Rab} & 1\ \forall b \\
7. & \multicolumn{2}{c}{\forall y\,[Ray \to Rya] \setminus b} & 2\ \forall a \\
8. & \multicolumn{2}{c}{Rab \to Rba} & 7\ \forall b \\
 & \swarrow & \searrow & \\
9. & \neg Rab & Rba & 8 \to \\
 & \times & \times &
\end{array}$$

With line (5) giving us $\neg Rba$, we could see that line (2), which switches the order of the names, could cause a contradiction with $(Rab \to Rba)$ if we also had Rab. And line (1) could give us Rab. So we had a plan for which names to substitute into lines (1) and (2) – and with a little luck, our plan worked.

23.2 Truth Tree Complexities

Strategically Selecting Names

In our last example, we had a plan for substituting our universal quantifiers by trying to create contradictions. This sort of strategy is important, particularly when you realise how long your tree could get if you just listed all possible substitutions. Lets try a more complex argument:

$$\forall x\,\exists y\,Rxy, \forall x\,\forall y\,Rxy \to Ryx, \forall x\,\forall y\,\forall z\,(Rxy \land Ryz) \to Rxz \therefore \forall x\,Rxx$$

We'll start by decomposing all the existentials as quickly as we can:

1.	$\forall x\,\exists y\,[Rxy]\ \backslash a$	Premise
2.	$\forall x\,\forall y\,[Rxy \to Ryx]$	Premise
3.	$\forall x\,\forall y\,\forall z\,[(Rxy \land Ryz) \to Rxz]$	Premise
4.	$\neg\forall x\,[Rxx]$ ✓	Neg Conc
5.	$\exists x\,[\neg Rxx]$ ✓a	4 $\neg\forall$
6.	$\neg Raa$	6 $\exists a$
7.	$\exists y\,[Ray]$ ✓b	1 $\forall a$
8.	Rab	7 $\exists b$

Now we ask how we can get a contradiction with these formulas. $\neg Raa$ is our only negative formula, so we need to get Raa from lines (2) or (3). If Raa was the consequent of line (2), the full formula would be $Raa \to Raa$, which doesn't seem useful? If Raa was the consequent of line (3), the full formula would be $(Rab \land Rba) \to Raa$. So we'd need $Rab \land Rba$. And we already have Rab! How can we get Rba? Well, line (2) can give us $Rab \to Rba$, and we already have Rab. We have a plan!!

9.		$\forall y\,[Ray \to Rya]\ \backslash b$			2 $\forall a$
10.		$Rab \to Rba$			9 $\forall b$
11.	$\neg Rab$		Rba		10 $\to$
12.	$\times$		$\forall y\,\forall z\,[(Ray \land Ryz) \to Raz]$ ✓b		3 $\forall a$
13.			$\forall z\,[(Rab \land Rbz) \to Raz]$ ✓a		12 $\forall b$
14.			$(Rab \land Rba) \to Raa$		13 $\forall a$
15.			$\neg(Rab \land Rba)$ ✓	Raa	14 $\to$
				$\times$	
16.		$\neg Rab$	$\neg Rba$		15 $\neg\land$
		$\times$	$\times$		

Infinite Trees

Every decomposition of a TFL tree takes us closer to a completed tree, because we check of formulas as decomposed, and replace them with either one or two simpler formulas. TFL trees are guaranteed to be completed in a finite number of steps — they will either close, or every formula will be decomposed into an atomic or negated atomic formula.

However, the universal quantifier rule can be applied to the same formula once for every name in the tree, and so if we keep generating new names, it is possible for trees to continue indefinitely without closing.

Consider a tree with two formulas in its root: $Raa, \forall x\, \exists y\, [Rxy]$. We will have to decompose the universal, using the existing name 'a'. And that will give us $\exists y\, [Ray]$, which in turn will give us a new name 'b', which we'll use to decompose the universal quantifier...

1.	Raa	Root
2.	$\forall x\, \exists y\, [Rxy] \setminus a, b, c$	Root
3.	$\exists y\, [Ray] \checkmark b$	1 $\forall a$
4.	Rab	2 $\exists b$
5.	$\exists y\, [Rby] \checkmark c$	1 $\forall b$
6.	Rbc	4 $\exists c$
7.	$\exists y\, [Rcy] \checkmark d$	1 $\forall c$
8.	Rcd	5 $\exists d$
9.	$\vdots$	

As the pattern makes clear, this tree will never close, but it will also never be complete. It will just keep decomposing instances of an existential with new names, then decomposing those instances of the universal, which in turn require a new existential, and so on.

The tree will never close, and its open branch is describing a model where all the formulas are true, but it is an infinite model. There's nothing illogical about that, but we certainly can't use truth tables to check formulas in that model. The good news is that in PL whenever there is an infinite model that makes a finite list of formulas true, there is also a finite model that does the same. The bad news is that we can't tell you how to find this finite model.

Logic Corner

Congratulations, you've run into a limit of logic. PL is *semi-decidable*. We can always tell if a set of formulas is inconsistent, but we can't always tell if a set of formulas is consistent – any algorithm will sometimes go into an infinite loop rather than stopping with an answer.

So, what should you do if your tree starts looping? First, check your tree decompositions, in case you've made a mistake. Then, once you are sure your tree is looping, stop extending the tree, draw a table with all the information you have *before the loop starts*, and manually, creatively, add more information to the table until you can make all the formulas true. We can't give you precise instructions on how to do this, because there is no algorithm that will always work.

However, usually (particularly with exercises in this course) you can produce a counter-example which is a model with one or two names. For example, with the tree above, a one-name model where $Raa = 1$ makes both Raa and $\forall x \, \exists y \, [Rxy]$ true. So would any two-name model with $Raa = 1$ and $Rba = 1$, or $Raa = 1$ and $Rbb = 1$. You can then check your counter-example using the techniques we discussed in Part VI.

Here is one counter-example, taken from the first 4 lines of the tree:

Domain $= \{a, b\}$

Rxy	a	b
a	1	1
b	1	?

x	y	Raa	$\forall x$	$\exists y$	$[Rxy]$
a	a	1	1	1	1
a	b				1
b	a			1	1
b	b				?

Logic Corner

In theory, to determine whether a set of formulas is consistent, or an argument valid, you don't need to produce a finite model. If a tree continues infinitely, there will always be an open branch. But in this course you must always produce a finite model.

An Old New Name

There is one case where you can use a *new* name when decomposing a universal quantifier. Suppose that you have no existential quantifiers, and no names already mentioned. Then there are no old names, and you can't decompose any of your universal quantifiers. In this one case, you can (after double-checking), use a new name to decompose a universal quantifier. Why? Because we can assume that the model we are reasoning about is not empty. Empty models aren't interesting. Consider the seemingly valid argument that 'everything is made out of atoms, so something is made of atoms'. If there's nothing at all, then the premise could be true and the conclusion false. But truth trees assume there is always at least one object:

$$\forall x\,[Ax] \mathrel{.\dot{}.} \exists x\,[Ax]$$

1.	$\forall x\,[Ax]$	Premise
2.	$\neg\exists x\,[Ax]\ \backslash b$	Neg Conc
3.	$\forall x\,[\neg Ax]\ \checkmark b$	$2\ \neg\exists$
4.	$\neg Ab$	$3\ \forall b$
5.	Ab	$1\ \forall b$
	$\times$	

Is $\exists x\,\forall y\,[Ryy \to Rxy]$ a tautology?

1.	$\neg\exists x\,\forall y\,[Ryy \to Rxy]\ \checkmark$	Root
2.	$\forall x\,[\neg\forall y\,[Ryy \to Rxy]]\ \backslash a$	$1\ \neg\exists$
3.	$\neg\forall y\,[Ryy \to Ray]\ \checkmark$	$2\ \forall a$
4.	$\exists y\,[\neg Ryy \to Ray]\ \checkmark b$	$3\ \neg\forall$
5.	$\neg(Rbb \to Rab)\ \checkmark$	$4\ \exists b$
6.	Rbb	$5\ \neg \to$
7.	$\neg Rab$	$5\ \neg \to$
	$\uparrow$	

Is this valid: 'All Unicorns have horns, so some Unicorns have horns'?

$$\forall x\,[Ux \to Hx] \mathrel{.\dot{}.} \neg\exists y\, Uy \wedge Hy$$

1.	$\forall x\,[Ux \to Hx]\ \backslash a$				Premise
2.	$\neg\exists y\,[Uy \wedge Hy]\ \checkmark$				Neg Conc
3.	$\forall y\,[\neg(Uy \wedge Hy)]\ \backslash a$				$2\ \neg\exists$
4.	$Ua \to Ha\ \checkmark$				$1\ \forall a$
5.	$\neg(Ua \wedge Ha)\ \checkmark$				$3\ \forall a$
6.	$\neg Ua$		Ha		$4 \to$
7.	$\neg Ua$	$\neg Ha$	$\neg Ua$	$\neg Ha$	$5\ \neg\wedge$
	$\uparrow$			$\times$	

A counter-example is: Domain = $\{a\}$.

	Hx	Ux
a	?	0

You can use new letters for your universals in other situations, but they will only make your tree longer, and never help it close or generate a counter-example. So don't. Just don't.

23.3 Common Student Errors

Using Old Names

The most frequent errors for trees in PL involve ignoring the restrictions on names when taking instances. Particular formulas (existentials and negated universals) require a *new* name — if the name appears anywhere in the branch at the time the existential is taken (including below it), it is not suitable for substitution via these rules.

For example, this tree contains a mistake in a quantifier rule. See if you can spot it:

1.	$\forall x\, \exists y\, [\neg Rxy]\ \backslash a$	
2.	Rab	
3.	$\exists y\, [\neg Ray]\ \checkmark b$	1 $\forall$
4.	$\neg Rab$	3 $\exists$
	$\times$	

Line (3) here is fine; but at line (4), this tree uses the name '*b*' to decompose the existential on line (3), even though '*b*' had already appeared in line (2). The existential rule requires a new name. The correct version of this tree would remain open. (It will extend infinitely.)

Main Connectives

You can only ever decompose the main connective of a formula. If you have a formula like $\forall x\, \exists y\, (\neg Ryx) \wedge (Fa \to Fb)$ you can't decompose the universal quantifier. This formula is a conjunction, so the only decomposition rule possible is that for conjunction.

It's often tempting to decompose a quantifier that's not the main connective. First, you can't. The rules don't work that way. Second, there are several ways that this can cause trouble. For instance, an existential quantifier within the scope of either a negation or the antecedent of a conditional will become a universal quantifier by the time the formula is decomposed. Second, if a quantifier is inside the scope of another quantifier, decomposing the inner quantifier first is effectively moving the inner quantifier outside the other quantifier. This quantifier scope shift is particularly problematic for an existential quantifier inside a universal quantifier.

It may help you to remember this by remembering our earlier example that 'Every day someone is struck by lightning' is true, but 'Someone is struck by lightning every day' is false, as different people are struck each day.

Decomposing Universals Only Once

It's very easy to forget that a universal quantifier can be decomposed multiple times in a tree. And often a single use will close some or all branches. But if there's an open branch, you'll need to identify every name used in that branch, and make sure that every universal quantifier in that branch is decomposed for every name. This is important for populating the counter-example, and also because occasionally it will close a seemingly open branch.

Negating Formulas in the Root

When testing consistency, we want to check if all the formulas can be true together. So no formula is negated. When testing validity, we want to check if all the premises can be true and the conclusion false (a counter-example). So we negate the conclusion only. When testing if a formula is a logical truth, we want to check if it can be false, so we negate the formula. It is very easy to tangle these up - particularly to forget to negate the conclusion of an argument. Then the information in the root of your tree is rotten. And a tree with a rotten root won't turn out well.

Reading off Counter-examples incorrectly

When you read off a counter-example from an open branch, make sure that you only trace up the tree towards the root. Do not use any atomic formulas that are on other branches, even if they are also open. If a name doesn't appear on your chosen open branch, its not needed in the counter-example. All the predicates in the root are needed, however.

Once you have your list of names from the open branch, and predicates from the root, draw up your tables – one for all the one-place predicates, and one for each two-place predicate. Then fill in the values for each atom in your open branch. Any cell in the tables that isn't filled with a 1 or 0 is then filled with a ?.

When you complete your partial predicate truth table, you should be able to fill in all the cells even with the missing truth values. If you need a truth value to calculate a row, check if you didn't decompose all names in one of the universal quantified formulas, as that's the most likely way to skip generating a needed atomic formula.

Practice Exercises

$\star$ **Exercise A:** Use a tree to test whether the following formulas are tautologies. If they are not tautologies, describe a model on which they are false.

1. $\forall x \forall y [Gxy \to \exists z [Gxz]]$
2. $\forall x [Fx] \lor \forall x [Fx \to Gx]$
3. $\forall x [Fx \to (\neg Fx \to \forall y [Gy]]$
4. $\exists x [Jx] \leftrightarrow \neg \forall x [\neg Jx]$
5. $\exists x [Fx \lor \neg Fx]$
6. $\forall x [Fx \lor Gx] \to (\forall y [Fy] \lor \exists x [Gx])$

$\star$ **Exercise B:** Use a tree to test whether the following arguments are valid. If they are not, give a model as a counterexample.

1. $Fa, Ga \therefore \forall x [Fx \to Gx]$
2. $Fa, Ga \therefore \exists x [Fx \land Gx]$
3. $\forall x \exists y [Lxy] \therefore \exists x \forall y [Lxy]$
4. $\exists x [Fx \land Gx], Fb \leftrightarrow Fa, Fc \to Fa \therefore Fa$
5. $\forall x \exists y [Gyx] \therefore \forall x \exists y [Gxy \lor Gyx]$

Exercise C: Translate each argument into PL, then use a tree to evaluate for validity. If they are invalid give a model as a counterexample.

1. Every logic student is studying. Patrick is not studying. Therefore, Patrick is not a logic student.
2. Kane Williamson is a New Zealand male athlete. Therefore, some athletes are New Zealand.
3. The All Blacks are going to win the game. Every team who wins the game will be celebrated. Therefore, the All Blacks will be celebrated.
4. The All Blacks are going to win the game. Therefore, the Springboks are not going to win the game.
5. All cats make Deborah sneeze, unless they are hairless. Some hairless cats are cuddly. Therefore, some cuddly things make Deborah sneeze.

Chapter 24

Truth Trees With Identity

We introduced Identity in Chapter 18, and Identity tables in Chapter 20. It should come as no surprise that we can also add Identity to Truth Trees.

The rules for Identity are a little different from the other Truth Tree rules.

The first rule is that we can always introduce the logical truth $(\alpha = \alpha)$ into any branch, for any name α.

$$\boxed{\alpha = \alpha \qquad \texttt{id}}$$

The only use for this rule is to close branches that already contain $(\alpha \neq \alpha)$, which is shorthand for $\neg(\alpha = \alpha)$.

The second rule is more complicated. If one formula in the current branch is an identity statement $\alpha = \beta$, then we can substitute some or all instances of α with β, or β with α, in a copy of another formula in the current branch.

$$\boxed{\begin{array}{cl} \alpha = \beta & \\ A & \\ A[\alpha/\beta] & \qquad \texttt{sub} \end{array}}$$

This `sub`rule uses 2 formulas (the identity, and the formula being copied), so both need to be listed. But we aren't decomposing these formulas, so they aren't ticked (✓) – they can be used as many times as needed. We represent this rule by $[\alpha/\beta]$, where we replace α with β in the formula.

An example will help to make these rules clearer. Let's test this argument: $(a = b), (a = a) \to Fa \therefore Fb$.

1.	$a = b$ ✓		Premise
2.	$(a = a) \to Fa$ ✓		Premise
3.	$\neg Fb$		Neg Conc
4.	$\neg(a = a)$	Fa	$2 \to$
5.	$(a = a)$	$\vert$	`id`
6.	$\times$	Fb	$1, 4$ $[a/b]$
		$\times$	

Adding $(a = a)$ on line #5 lets us close the branch. Otherwise we'd need a special closing rule for $\neg(a = a)$. On line #6 we copy the formula Fa on line #4 and replace an a with a b, because line #1 says that a and b are identical.

Practice Exercises

⋆ **Exercise A:** Let A be this formula:

$$a = b \to (Rca \lor Pb)$$

Write all formulas that can be obtained from the following substitutional instances:

1. $A[a/b]$
2. $A[b/a]$
3. $A[a/c]$
4. $A[c/a]$
5. $A[b/c]$
6. $A[c/b]$

24.1 Logical Truths of Identity

In Chapter 20 we learned several rules to help us to complete Identity tables. The first rule was 'Reflexivity', which says that everything is identical with itself: $\forall x\,[x = x]$. The second rule was 'Symmetry', which says that if a is identical to b, then b is identical to a: $\forall x\,\forall y\,[x = y \to y = x]$. The third rule was 'Transitivity', which says that if anything is identical to two other things, they are identical with each other: $\forall x\,\forall y\,\forall z\,[(x = y \land y = z) \to x = z]$. Let's check these rules are logical truths, using Truth Trees.

1.	$\neg\forall x\,[x = x]$ ✓	Root
2.	$\exists x\,[x \neq x]$ ✓a	1 $\neg\forall$
3.	$a \neq a$	2 $\exists$ a
4.	$a = a$	`id`
	$\times$	

The last line is one in which we use the rule of identity to close the branch.

1.	$\neg\forall x\,\forall y\,[x = y \rightarrow y = x]$ ✓	Root
2.	$\exists x\,[\neg\forall y\,[x = y \rightarrow y = x]]$ ✓a	1 $\neg\forall$
3.	$\neg\forall y\,[a = y \rightarrow y = a]$ ✓	2 $\exists$ a
4.	$\exists y\,[\neg(a = y \rightarrow y = a)]$ ✓b	3 $\neg\forall$
5.	$\neg(a = b \rightarrow b = a)$ ✓	4 $\exists$ b
6.	$a = b$	5 $\neg \rightarrow$
7.	$\neg(b = a)$	5 $\neg \rightarrow$
8.	$a = a$	`id`
9.	$b = a$	$8, 6\ [a/b]$
	$\times$	

1.	$\neg\forall x\,\forall y\,\forall z\,[(x = y \wedge y = z) \rightarrow x = z]$ ✓	Root
2.	$\exists x\,[\neg\forall y\,\forall z\,[(x = y \wedge y = z) \rightarrow x = z]]$ ✓a	1 $\neg\forall$
3.	$\neg\forall y\,\forall z\,[(a = y \wedge y = z) \rightarrow a = z]$ ✓	2 $\exists$ a
4.	$\exists y\,[\neg\forall z\,[(a = y \wedge y = z) \rightarrow a = z]]$ ✓b	3 $\neg\forall$
5.	$\neg\forall z\,[(a = b \wedge b = z) \rightarrow a = z]$ ✓	4 $\exists$ b
6.	$\exists z\,[\neg((a = b \wedge b = z) \rightarrow a = z)]$ ✓c	5 $\neg\forall$
7.	$\neg((a = b \wedge b = c) \rightarrow a = c)$ ✓	6 $\exists$ c
8.	$a = b \wedge b = c$	7 $\neg \rightarrow$
9.	$\neg(a = c)$	7 $\neg \rightarrow$
10.	$a = b$	8 $\wedge$
11.	$b = c$	8 $\wedge$
12.	$a = c$	$10, 11\ [b/a]$
	$\times$	

We also learned how to complete predicate tables from information in identity tables, and identity tables from predicate tables. If two names are identical, then they have exactly the same truth values for each predicate: $\forall x\,\forall y\,[(x = y) \rightarrow (Px \leftrightarrow Py)]$. And if two names had different values for at least one predicate, then they were not identical: $\forall x\,\forall y\,[\neg(Px \leftrightarrow Py) \rightarrow (x \neq y)]$. These formulas are equivalent, so we'll only test the first one.

1.	$\neg\forall x\,\forall y\,[(x=y)\rightarrow(Px\leftrightarrow Py)]\ \checkmark$		Root
2.	$\exists x\,[\neg\forall y\,[(x=y)\rightarrow(Px\leftrightarrow Py)]]\ \checkmark a$		1 ¬∀
3.	$\neg\forall y\,[(a=y)\rightarrow(Pa\leftrightarrow Py)]\ \checkmark$		2 ∃ a
4.	$\exists y\,[\neg((a=y)\rightarrow(Pa\leftrightarrow Py))]\ \checkmark b$		3 ¬∀
5.	$\neg((a=b)\rightarrow(Pa\leftrightarrow Pb))\ \checkmark$		4 ∃ b
6.	$a=b$		5 →
7.	$\neg(Pa\leftrightarrow Pb)\ \checkmark$		5 →
8.	Pa	$\neg Pa$	7 ¬ ↔
9.	$\neg Pb$	Pb	7 ¬ ↔
10.	Pb	$\neg Pb$	$6,8\ [a/b]$
	$\times$	$\times$	

One final logical truth about Identity: $\forall x\,[\forall y\,[(x=y)\rightarrow Fy]\leftrightarrow Fx]$.

1.	$\neg\forall x\,[\forall y\,[(x=y)\rightarrow Fy]\leftrightarrow Fx]\ \checkmark$			Root
2.	$\exists x\,[\neg(\forall y\,[(x=y)\rightarrow Fy]\leftrightarrow Fx)]\ \checkmark a$			1 ¬∀
3.	$\neg(\forall y\,[(a=y)\rightarrow Fy]\leftrightarrow Fa)\ \checkmark$			2 ∃ a
4.	$\forall y\,[(a=y)\rightarrow Fy]\ \backslash a$		$\neg\forall y\,[(a=y)\rightarrow Fy]\ \checkmark$	3 ¬ ↔ a
5.	$\neg Fa$		Fa	3 ¬ ↔ a
6.	$(a=a)\rightarrow Fa\ \checkmark$			4 ∀ a
7.	$\neg(a=a)$	Fa		6 →
8.	$a=a$	$\times$		`id`
9.	$\times$		$\exists y\,[\neg((a=y)\rightarrow Fy)]\ \checkmark b$	4 ¬∀
10.			$\neg((a=b)\rightarrow Fb)\ \checkmark$	4 ∃ b
11.			$a=b$	10 ¬ →
12.			$\neg Fb$	10 ¬ →
13.			$\neg Fa$	$11,12\ [b/a]$
			$\times$	

24.2 Arguments with Identity

Checking arguments that use Identity for validity is the same as any other argument. Consider $Rab, a=c\ \therefore\ Rbc$:

1.	Rab	Premise
2.	$a = c$	Premise
3.	$\neg Rbc$	Neg Conc
4.	Rcb	$1, 2\ [a/c]$
5.	$\neg Rba$	$2, 3\ [c/a]$
	$\uparrow$	

R	a	b	c
a	?	1	?
b	0	?	0
c	?	1	?

=	a	b	c
a	1	0	1
b	0	1	0
c	1	0	1

Counter-examples are read off an open branch of the tree, just like any other counter-example. Once you've read off the counter-example, you'll need to do a little more work to complete the identity and other predicate tables, exactly as we've shown you in Chapter 20.

Is this argument valid?

$$\exists x\,[Fx], \exists x\,[Gx], \forall x\,[\neg(Fx \wedge Gx)] \therefore \exists x\,\exists y\,[x \neq y]$$

1.	$\exists x\,[Fx]$		Premise
2.	$\exists x\,[Gx]$		Premise
3.	$\forall x\,[\neg(Fx \wedge Gx)]$		Premise
4.	$\neg\exists x\,\exists y\,[x \neq y]$		Neg Conc
5.	Fa		1 ∃ a
6.	Gb		2 ∃ b
7.	$\forall x\,\neg\exists y\,[x \neq y]$		4 ¬∃
8.	$\neg\exists y\,[a \neq y]$		7 ∀ a
9.	$\forall y\,[\neg(a \neq y)]$		8 ¬∃
10.	$\neg\neg(a = b)$		9 ∀ a
11.	$a = b$		10 ¬¬ a
12.	$\neg(Fa \wedge Ga)$		2 ∀ a
13.	$\neg Fa$	$\neg Ga$	12 ∨
14.	$\times$	Ga	$5, 11\ [b/a]$
		$\times$	

Is this argument valid?

$$Rab \to Sa, a = c, Rcb \therefore Sa$$

1.	$Rab \to Sa$		Premise
2.	$a = c$		Premise
3.	Rcb		Premise
4.	$\neg Sa$		Neg Conc
5.	$\neg Rab$	Sa	$1 \to$
6.	$\neg Rcb$	$\times$	$3, 5$ $[a/c]$
	$\times$		

Is this argument valid?

$$\forall x \forall y\, [(Fx \wedge Gy) \to x \neq y] \;\therefore\; \neg\exists x\, [Fx \wedge Gx]$$

1.	$\forall x \forall y\, [(Fx \wedge Gy) \to x \neq y] \setminus a$		Premise
2.	$\neg\neg\exists x\, [Fx \wedge Gx]$		Neg Conc
3.	$\exists x\, [Fx \wedge Gx] \checkmark a$		$2 \neg\neg$
4.	$Fa \wedge Ga$		3 ∃ a
5.	$\forall y\, [(Fa \wedge Gy) \to a \neq y] \setminus a$		1 ∀ a
6.	$(Fa \wedge Ga) \to a \neq a \checkmark$		5 ∀ a
7.	$\neg(Fa \wedge Ga)$	$\neg(a = a)$	$6 \to$
8.	$\times$	$a = a$	`id`
		$\times$	

Is this argument valid?

$$\forall x\, [Fx \to x = a], \forall x\, [Fx \vee Gx], \exists x\, [\neg Gx] \;\therefore\; Fa$$

1.	$\forall x\, [Fx \to x = a] \setminus b$			Premise
2.	$\forall x\, [Fx \vee Gx] \setminus b$			Premise
3.	$\exists x\, [\neg Gx] \checkmark b$			Premise
4.	$\neg Fa$			Neg Conc
5.	$\neg Gb$			3 ∃ b
6.	$Fb \vee Gb \checkmark$			2 ∀ b
7.	Fb		Gb	$6 \vee$
8.	$Fb \to b = a \checkmark$		$\times$	1 ∀ b
9.	$\neg Fb$	$b = a$		$8 \to$
10.	$\times$	Fa		$7, 9$ $[b/a]$
		$\times$		

Identity can be used for counting. Here's an argument involving counting:

> Suppose there are at most two things, and Angela and Belinda are both frogs, and Angela is not Belinda. Then obviously, everything is a frog.

And here's a symbolisation of this argument, using initials as an implicit symbolisation key:

$$\exists x\, \exists y\, [\forall z\, [z = x \vee z = y]], Fa, Fb, a \neq b \;\therefore\; \forall x\, [Fx]$$

1.	$\exists x\, \exists y\, [\forall z\, [z = x \vee z = y]]$ ✓d				Premise
2.	Fa				Premise
3.	Fb				Premise
4.	$(a \neq b)$				Premise
5.	$\neg\forall x\, [Fx]$				Neg Conc
6.	$\exists x\, [\neg Fx]$ ✓c				5 ¬∀
7.	$\neg Fc$				6 ∃ c
8.	$\exists y\, [\forall z\, [z = d \vee z = y]]$ ✓e				1 ∃ d
9.	$\forall z\, [z = d \vee z = e]$ ✓				8 ∃ e
10.	$a = d \vee a = e$ ✓				9 ∀ a
11.	$b = d \vee b = e$ ✓				9 ∀ b
12.	$c = d \vee c = e$ ✓				9 ∀ c
13.	$a = d$		$a = e$		10 ∨
14.	$b = d$	$b = e$	$b = e$	$b = d$	11 ∨
15.	$a = b$	|	$a = b$	|	$13, 14\ [d/b]$; $13, 14\ [e/b]$
16.	×	Fd	×	Fd	$2, 13\ [a/d]$; $2, 13\ [b/d]$
17.		Fe		Fe	$3, 14\ [b/e]$; $3, 14\ [a/e]$
18.	$c = d$	$c = e$	$c = d$	$c = e$	12 ∨
19.	Fc	Fc	Fc	Fc	$14, 18\ [d/c]$; $14, 18\ [e/c]$
	×	×	×	×	

You can see the slightly different substitutions in the parallel branches on lines #15-#19. Identity trees with many parallel branches – like this one – can get messy. When the different substitutions become too messy to keep track of, we can abuse notation, and just write:

19.	Fc	Fc	Fc	Fc	$14, 18[/]$

Practice Exercises

⋆ **Exercise A:** Use a tree to show that the following arguments are valid.

1. $\forall x(x = a \lor x = b)$, $\neg(Fa \land Ga)$, $Gb \to Hb \therefore \forall x((Fx \land \neg Hx) \to \neg Gx)$
2. $\exists xFx$, $\forall x\forall y((Fx \land Fy) \to x = y) \therefore \exists x\forall y(Fy \leftrightarrow (x = y))$
3. $\forall x\forall y(Fx \to x = y)$, $\forall x(Fx \lor Gx) \therefore (\forall xFx \lor \forall xGx)$
4. $\exists x\forall y(Fy \leftrightarrow (x = y)) \therefore \exists xFx$
5. $\exists x\forall y(Fy \leftrightarrow (x = y)) \therefore \forall x\forall y((Fx \land Fy) \to (x = y))$
6. $\exists x\forall y(Fy \leftrightarrow (x = y))$, $\forall x(Fx \lor Gx) \therefore \forall x\forall y((Gx \lor Gy) \lor x = y)$

Exercise B: Use a tree to test whether the following arguments are valid.

1. $(\forall x\forall y\forall z((Rxy \land Rxz) \to y = z) \lor \forall x\forall y\forall z((Rxy \land Rzy) \to x = z))$, $\forall xRxx \therefore \forall x\forall y(Rxy \leftrightarrow (x = y))$
2. $\forall x\neg Rxx$, $\forall x\forall y((Rxy \land Ryx) \to x = y)$, $\forall x\forall y\forall z((Rxy \land Ryz) \to Rxz)$ $\therefore \forall x\forall y\forall z(\neg Rxy \lor \neg Ryz \lor \neg Rzx)$
3. $\exists x\forall y(Fy \leftrightarrow (x = y))$, $\exists x\forall y(Gy \leftrightarrow (x = y))$, $\neg\exists x(Fx \land Gx)$ $\therefore \exists x\exists y(x \neq y \land \forall z((Fz \lor Gz) \leftrightarrow (z = x \lor z = y)))$
4. $\forall x\forall y((Rxy \land Ryx) \to x = y)$, $\forall x(Fx \to \exists y(Rxy \land \neg Fy))$ $\therefore \forall x\exists y\neg(Ryx \land Fx)$
5. $\forall x\exists y\forall z(Rxz \leftrightarrow (z = y)$, $\forall x\forall y\forall z((Rxy \land Ryz) \to Rxz)$ $\therefore \forall x\exists y(Rxy \land Ryx)$

Part VIII

Appendix

Chapter A
Solutions to selected exercises

Many of the exercises may be answered correctly in different ways. Where that is the case, the solution here represents one possible correct answer.

Chapter 1 Ex A:

1. I should take my sunglasses.
2. It must have been sunny.
3. You took the cookie from the cookie-jar.
4. Colonel Mustard did it in the kitchen with the lead pipe.

Chapter 2 Ex B:

1. Yes, valid arguments can have a mix of true and false premises.
   e.g., You are a lizard (false). If you are a lizard, you are a reptile (true). Therefore, you are a reptile.
2. Yes, a valid argument can have only false premises.
   e.g., You are a lizard (false). If you are a lizard, you are a mammal (false). Therefore, you are a mammal.
3. Yes, a valid argument can have false premises and conclusion.
   e.g., You are a lizard (false). If you are a lizard, you are a fish (false). Therefore, you are a fish.
4. Yes, an invalid argument can be made valid by adding a new premise.
   e.g., You are alive. Therefore you breathe. This is invalid. But adding 'Everything that is alive breathes' makes the argument valid.
5. No, you can't make a valid argument invalid by adding a new premise. Adding new premises just makes it harder to make all the premises true. Recall that a valid argument has a true conclusion whenever its premises are true, so adding premises can only make it easier to be valid.

Chapter 3 Ex B: Are these logical truths, falsehoods, or contingent?

1. contingent
2. contingent
3. logical truth
4. logical falsehood
5. contingent

Chapter 3 Ex D: Which pairs of statements are logically equivalent?

1. Not equivalent
2. Not equivalent
3. Not equivalent
4. Equivalent
5. Equivalent

Chapter 3 Ex F:

1. Consistent.
2. Consistent.
3. Inconsistent.
4. Consistent.
5. Inconsistent.
6. Inconsistent.
7. Inconsistent.

Chapter 3 Ex G:

1. Possible. 20=30. Therefore 2 = 3.
2. Impossible. Invalid arguments have counter-examples. Counter-examples make the conclusion false. Logical truths can never be false.
3. Impossible. Valid arguments have a true conclusion if the premises are true, and they are true always, so the conclusion is true always.
4. Impossible. Contingent statements are those that are neither logical truths nor falsehoods.
5. Possible. Any two logical truths are equivalent.
6. Impossible. The contingent statement will be false in a situation where the logical truth is true, so they are not equivalent.
7. Possible. Any two logical falsehoods are equivalent, and inconsistent.
8. Impossible. A falsehood is never true, so all the statements can't be true.
9. Possible. As long as the other statements are mutually inconsistent, adding a logical truth won't make the other statements all true.

Chapter 4 Ex A:

1. $(R \wedge W)$
2. $(R \wedge \neg S$
3. $\neg(R \wedge S)$
4. $\neg(R \vee S)$
5. $(R \rightarrow (W \wedge G))$
6. $(G \wedge \neg W)$
7. $(R \rightarrow (W \wedge \neg S))$
8. $((R \rightarrow W) \wedge \neg S)$

Chapter 4 Ex C:
The letters in your symbolisation keys may differ from ours.

1. $(\neg Z \rightarrow \neg C), (Z \rightarrow T), (T \rightarrow \neg C) \therefore \neg C.$
2. $(R \rightarrow S), (R \rightarrow E) \therefore (\neg E \rightarrow (\neg R \wedge \neg S)).$
3. $(R \rightarrow \neg D), (P \rightarrow M), ((P \rightarrow M) \rightarrow (\neg D \rightarrow \neg N)) \therefore (N \rightarrow \neg R).$

Chapter 6 Ex A:

1. $\neg M$
2. $(M \vee \neg M)$
3. $(G \vee C)$
4. $\neg(C \vee G)$
5. $C \rightarrow \neg(G \vee M)$
6. $\neg M \rightarrow (C \vee G)$

Chapter 6 Ex C:

1. $(A \wedge B)$
2. $(C \rightarrow E)$
3. $(C \vee A)$
4. $(B \wedge \neg F)$
5. $(\neg A \wedge \neg B)$
6. $(A \wedge B \wedge \neg(E \vee F))$
7. $(F \rightarrow D)$
8. $((\neg A \rightarrow \neg B) \wedge (A \rightarrow B))$
9. $(E \leftrightarrow \neg F)$
10. $((B \wedge D) \rightarrow F)$
11. $\neg(B \wedge D)$

Chapter 6 Ex E:

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

Chapter 6 Ex G:
The letters in your symbolisation keys may differ from ours.

1. $(A \rightarrow E), (\neg D \rightarrow A) \therefore (\neg E \rightarrow D)$.
2. $(R \vee S), (R \rightarrow D), (S \rightarrow C) \therefore (D \vee C)$.
3. $(R \rightarrow (C \wedge \neg N)), (\neg R \rightarrow (N \wedge \neg C)) \therefore ((N \vee C) \wedge \neg(N \wedge C))$.
4. $(\neg M \rightarrow D), (\neg D \rightarrow M), \neg(M \wedge D) \therefore ((M \vee D) \wedge \neg(M \wedge D))$.

Chapter 8 Ex C:

1. $\neg(S \leftrightarrow (P \rightarrow S))$

P	S	$\neg$	$(S$	$\leftrightarrow$	$(P$	$\rightarrow$	$S))$
1	1	0	1	1	1	1	1
1	0	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	0	1	0	0	0	1	0

2. $\neg((X \wedge Y) \vee (X \vee Y))$

X	Y	$\neg$	$((X$	$\wedge$	$Y)$	$\vee$	$(X$	$\vee$	$Y))$
1	1	0	1	1	1	1	1	1	1
1	0	0	1	0	0	1	1	1	0
0	1	0	0	0	1	1	0	1	1
0	0	1	0	0	0	0	0	0	0

3. $((\neg P \vee \neg M) \leftrightarrow M)$

P	M	$((\neg$	P	$\vee$	$\neg$	$M)$	$\leftrightarrow$	$M)$
1	1	0	1	0	0	1	0	1
1	0	0	1	1	1	0	0	0
0	1	1	0	1	0	1	1	1
0	0	1	0	1	1	0	0	0

4. $((A \rightarrow B) \leftrightarrow (\neg B \leftrightarrow \neg A))$

A	B	$((A$	$\rightarrow$	$B)$	$\leftrightarrow$	$(\neg$	B	$\leftrightarrow$	$\neg$	$A))$
1	1	1	1	1	1	0	1	1	0	1
1	0	1	0	0	1	1	0	0	0	1
0	1	0	1	1	0	0	1	0	1	0
0	0	0	1	0	1	1	0	1	1	0

5. $\neg\neg(\neg A \land \neg B)$

A	B	$\neg$	$\neg$	$(\neg$	A	$\land$	$\neg$	$B)$
1	1	0	1	0	1	0	0	1
1	0	0	1	0	1	0	1	0
0	1	0	1	1	0	0	0	1
0	0	1	0	1	0	1	1	0

6. $(((D \land R) \to I) \to \neg(D \lor R))$

D	I	R	$((D$	$\land$	$R)$	$\to$	$I)$	$\to$	$\neg$	$(D$	$\lor$	$R))$
1	1	1	1	1	1	1	1	0	0	1	1	1
1	1	0	1	0	0	1	1	0	0	1	1	0
1	0	1	1	1	1	0	0	1	0	1	1	1
1	0	0	1	0	0	1	0	0	0	1	1	0
0	1	1	0	0	1	1	1	0	0	0	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0
0	0	1	0	0	1	1	0	0	0	0	1	1
0	0	0	0	0	0	1	0	1	1	0	0	0

7. $((C \leftrightarrow (D \lor E)) \land \neg C)$

C	D	E	$((C$	$\leftrightarrow$	$(D$	$\lor$	$E))$	$\land$	$\neg$	$C)$
1	1	1	1	1	1	1	1	0	0	1
1	1	0	1	1	1	1	0	0	0	1
1	0	1	1	1	0	1	1	0	0	1
1	0	0	1	0	0	0	0	0	0	1
0	1	1	0	0	1	1	1	0	1	0
0	1	0	0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1	0	1	0
0	0	0	0	1	0	0	0	1	1	0

8. $(\neg(G \land (B \land H)) \leftrightarrow (G \lor \neg(B \lor H)))$

B	G	H	$(\neg$	$(G$	$\land$	$(B$	$\land$	$H))$	$\leftrightarrow$	$(G$	$\lor$	$\neg$	$(B$	$\lor$	$H))$
1	1	1	0	1	1	1	1	1	0	1	1	0	1	1	1
1	1	0	1	1	0	1	0	0	1	1	1	0	1	1	0
1	0	1	1	0	0	1	1	1	0	0	0	0	1	1	1
1	0	0	1	0	0	1	1	0	0	0	0	0	1	1	0
0	1	1	1	1	0	0	0	1	1	1	1	0	0	1	1
0	1	0	1	1	0	0	0	0	1	1	1	1	0	0	0
0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0

Chapter 9 Ex B:

1. $A \to A$, $\neg A \to \neg A$, $A \wedge A$, $A \vee A$ are mutually consistent.

A	$(A$	$\to$	$A)$	$(\neg$	A	$\to$	$\neg$	$A)$	$(A$	$\wedge$	$A)$	$(A$	$\vee$	$A)$
1	1	1	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	0	0	0	0	0	0

2. $A \vee B$, $A \to C$, $B \to C$ are mutually consistent.

A	B	C	$(A$	$\vee$	$B)$	$(A$	$\to$	$C)$	$(B$	$\to$	$C)$
1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	0	0	1	0	0
1	0	1	1	1	0	1	1	1	0	1	1
1	0	0	1	1	0	1	0	0	0	1	0
0	1	1	0	1	1	0	1	1	1	1	1
0	1	0	0	1	1	0	1	0	1	0	0
0	0	1	0	0	0	0	1	1	0	1	1
0	0	0	0	0	0	0	1	0	0	1	0

3. $B \wedge (C \vee A)$, $A \to B$, $\neg(B \vee C)$ are mutually *in*consistent.

A	B	C	$(B$	$\wedge$	$(C$	$\vee$	$A))$	$(A$	$\to$	$B)$	$(\neg$	$(B$	$\vee$	$C))$
1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	0	1	1	0	1	1	1	1	1	0	1	1	0
1	0	1	0	0	1	1	1	1	0	0	0	0	1	1
1	0	0	0	0	0	1	1	1	0	0	1	0	0	0
0	1	1	1	1	1	1	0	0	1	1	0	1	1	1
0	1	0	1	0	0	0	0	0	1	1	0	1	1	0
0	0	1	0	0	1	1	0	0	1	0	0	0	1	1
0	0	0	0	0	0	0	0	0	1	0	1	1	0	0

4. $A \leftrightarrow (B \vee C)$, $C \to \neg A$, $A \to \neg B$ are mutually consistent.

A	B	C	$(A$	$\leftrightarrow$	$(B$	$\vee$	$C))$	$(C$	$\to$	$\neg$	$A)$	$(A$	$\to$	$\neg$	$B)$
1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1
1	1	0	1	1	1	1	0	0	1	0	1	1	0	0	1
1	0	1	1	1	0	1	1	1	0	0	1	1	1	1	0
1	0	0	1	0	0	0	0	0	1	0	1	1	1	1	0
0	1	1	0	0	1	1	1	1	1	1	0	0	1	0	1
0	1	0	0	0	1	1	0	0	1	1	0	0	1	0	1
0	0	1	0	0	0	1	1	1	1	1	0	0	1	1	0
0	0	0	0	1	0	0	0	0	1	1	0	0	1	1	0

Chapter 9 Ex C:

1. $A \to A .\therefore. A$ is invalid.

A	$(A$	$\to$	$A)$	A
1	1	1	1	1
0	0	1	0	0

2. $A \to (A \wedge \neg A) .\therefore. \neg A$ is valid.

A	$(A$	$\to$	$(A$	$\wedge$	$\neg$	$A)$	$\neg$	A
1	1	0	1	0	0	1	0	1
0	0	1	0	0	1	0	1	0

3. $A \vee (B \to A) .\therefore. \neg A \to \neg B$ is valid.

A	B	$(A$	$\vee$	$(B$	$\to$	$A)$	$(\neg$	A	$\to$	$\neg$	$B)$
1	1	1	1	1	1	1	0	1	1	0	1
1	0	1	1	0	1	1	0	1	1	1	0
0	1	0	0	1	0	0	1	0	0	0	1
0	0	0	1	0	1	0	1	0	1	1	0

4. $A \vee B, B \vee C, \neg A .\therefore. B \wedge C$ is invalid.

A	B	C	$(A$	$\vee$	$B)$	$(B$	$\vee$	$C)$	$\neg$	A	$(B$	$\wedge$	$C)$
1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	0	1	1	1	1	1	0	0	1	1	0	0
1	0	1	1	1	0	0	1	1	0	1	0	0	1
1	0	0	1	1	0	0	0	0	0	1	0	0	0
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	0	0	1	1	1	1	0	1	0	1	0	0
0	0	1	0	0	0	0	1	1	1	0	0	0	1
0	0	0	0	0	0	0	0	0	1	0	0	0	0

5. $(B \wedge A) \to C, (C \wedge A) \to B .\therefore. (C \wedge B) \to A$ is invalid.

A	B	C	$((B$	$\wedge$	$A)$	$\to$	$C)$	$((C$	$\wedge$	$A)$	$\to$	$B)$	$((C$	$\wedge$	$B)$	$\to$	$A)$
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	0	0	0	0	1	1	1	0	0	1	1	1
1	0	1	0	0	1	1	1	1	1	1	0	0	1	0	0	1	1
1	0	0	0	0	1	1	0	0	0	1	1	0	0	0	0	1	1
0	1	1	1	0	0	1	1	1	0	0	1	1	1	1	1	0	0
0	1	0	1	0	0	1	0	0	0	0	1	1	0	0	1	1	0
0	0	1	0	0	0	1	1	1	0	0	1	0	1	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0

Chapter 9 Ex K:

1. $\mathcal{A}$ and $\mathcal{B}$ have the same truth value on every line of a complete truth table, so $\mathcal{A} \leftrightarrow \mathcal{B}$ is true on every line. It is a logical truth.
2. $(\mathcal{A} \wedge \mathcal{B}) \rightarrow \mathcal{C}$ is false on some line of a complete truth table. On that line, $\mathcal{A}$ and $\mathcal{B}$ are true and $\mathcal{C}$ is false. So the argument is invalid.
3. Since $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ are mutually inconsistent, there is no line on which all three are true, so their conjunction will be a logical falsehood.
4. Since $\mathcal{A}$ is false on every line of a complete truth table, there is no line on which $\mathcal{A}$ and $\mathcal{B}$ are true and $\mathcal{C}$ is false. So the argument is valid.
5. Since $\mathcal{C}$ is true on every line of a complete truth table, there is no line on which $\mathcal{A}$ and $\mathcal{B}$ are true and $\mathcal{C}$ is false. So the argument is valid.
6. $(\mathcal{A} \vee \mathcal{B})$ is equivalent to $\mathcal{A}$. So $(\mathcal{A} \vee \mathcal{B})$ is a logical truth iff $\mathcal{A}$ is; a contradiction iff $\mathcal{A}$ is; and contingent iff $\mathcal{A}$ is.
7. $\mathcal{A}$ and $\mathcal{B}$ have different truth values on at least one line of a complete truth table, and $(\mathcal{A} \vee \mathcal{B})$ will be true on that line. On other lines, it might be true or false. So $(\mathcal{A} \vee \mathcal{B})$ is either a logical truth or it is contingent; it is *not* a contradiction.

Chapter 10 Ex A:

1. A, $\neg A$ are not equivalent.

A	A	$\neg$	A
1	1	0	1

2. A, $A \vee A$ are equivalent.
3. $A \rightarrow A$, $A \leftrightarrow A$ are equivalent.
4. $A \vee \neg B$, $A \rightarrow B$ are not equivalent.

A	B	A	$\vee$	$\neg$	B	A	$\rightarrow$	B
1	0	1	1	1	0	1	0	0

5. $A \wedge \neg A$, $\neg B \leftrightarrow B$ are equivalent.
6. $\neg(A \wedge B)$, $\neg A \vee \neg B$ are equivalent.
7. $\neg(A \rightarrow B)$, $\neg A \rightarrow \neg B$ are not equivalent.

A	B	$\neg$	$(A$	$\rightarrow$	$B)$	$\neg$	A	$\rightarrow$	$\neg$	B
0	0	0	0	1	0	1	0	1	1	0
1	1	0	1	1	1	0	1	1	0	1

8. $(A \rightarrow B)$, $(\neg B \rightarrow \neg A)$ are equivalent.

Chapter 10 Ex D:

1. $A \vee (A \rightarrow (A \leftrightarrow A)) \therefore A$ is invalid.

A	$(A$	$\vee$	$(A$	$\rightarrow$	$(A$	$\leftrightarrow$	$A))$	A
0	0	1	0	1	0	1	0	0

2. $A \leftrightarrow \neg(B \leftrightarrow A) \therefore A$ is invalid.

A	B	$(A$	$\leftrightarrow$	$\neg$	$(B$	$\leftrightarrow$	$A))$	A
0	0	0	1	0	0	1	0	0

3. $A \rightarrow B, B \therefore A$ is invalid.

A	B	$(A$	$\rightarrow$	$B)$	B	A
0	1	0	1	1	1	0

4. $A \vee B, B \vee C, \neg B \therefore A \wedge C$ is valid.
5. $A \leftrightarrow B, B \leftrightarrow C \therefore A \leftrightarrow C$ is valid.

Chapter 11 Ex A:

1. The root of a truth tree is the list of formulas at the beginning of the tree. Trees test whether the root is consistent.
2. You should tick (✓) a formula when you've decomposed it on every open branch below that formula. This is simply to remind yourself that it has been decomposed, and so avoid decomposing it again.
3. A truth tree branch represents a potential valuation that may make the root formulas all true.
4. You should create branches when the formula being decomposed has more than one valuation that would make it true.
5. You should close a branch when it contains any formula and its negation, as we know that branch represents an impossible valuation.
6. An open branch is complete when all the formulas above it either have a check mark, or are atomic. Once you have found a complete open branch, you can stop growing the tree.
7. The collection of atomic formulas in an open branch describe a valuation under consideration. A *complete* open branch represents a valuation where all the formulas in the root are true.

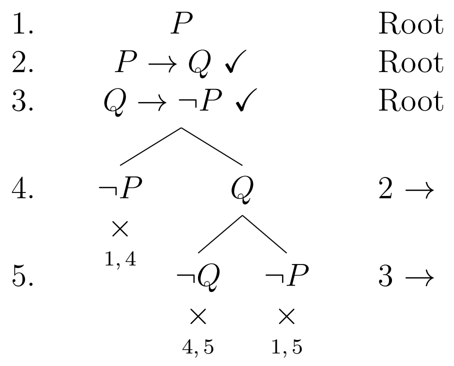
Chapter 12 Ex A: To be updated shortly.
Chapter 13 Ex A:

1. (a) $\{P, P \rightarrow Q, Q \rightarrow \neg P\}$; (b) false.
2. (a) $\neg((P \rightarrow Q) \leftrightarrow (Q \rightarrow P))$; (b) true.
3. (a) $\{P \wedge Q, \neg R \rightarrow \neg Q, \neg(P \wedge R)\}$; (b) true.
4. (a) $\{A \vee B, B \rightarrow C, A \leftrightarrow C, \neg C\}$; (b) true.

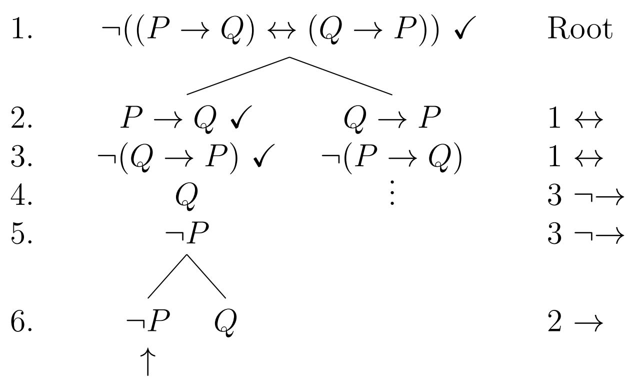
5. (a) $A \leftrightarrow \neg A$; (b) true.
6. (a) $\{P, P \to Q, \neg Q, \neg A\}$; (b) true.
7. (a) $\{P \to Q, \neg P \vee \neg Q, Q \to P\}$; (b) false.

Chapter 13 Ex B:

1. $P, P \to Q, Q \to \neg P$ are NOT mutually consistent

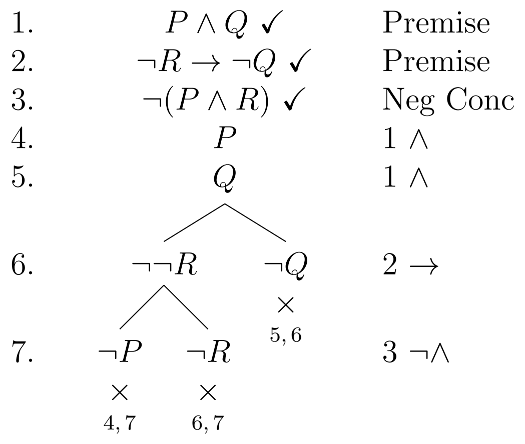


2. $(P \to Q) \leftrightarrow (Q \to P)$ is NOT a tautology.

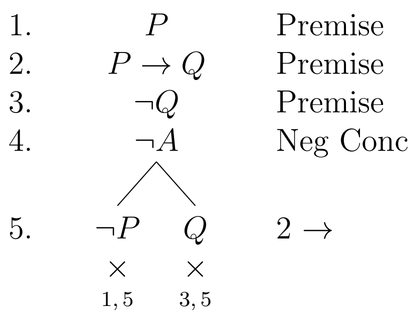

1. $\neg((P \to Q) \leftrightarrow (Q \to P))$ ✓ Root
2. $P \to Q$ ✓ $Q \to P$ $1 \leftrightarrow$
3. $\neg(Q \to P)$ ✓ $\neg(P \to Q)$ $1 \leftrightarrow$
4. Q $\vdots$ $3 \neg\to$
5. $\neg P$ $3 \neg\to$
6. $\neg P$ Q $2 \to$
$\uparrow$

P	Q	$(P$	$\to$	$Q)$	$\leftrightarrow$	$(Q$	$\to$	$P)$
0	1	0	1	1	0	1	0	0

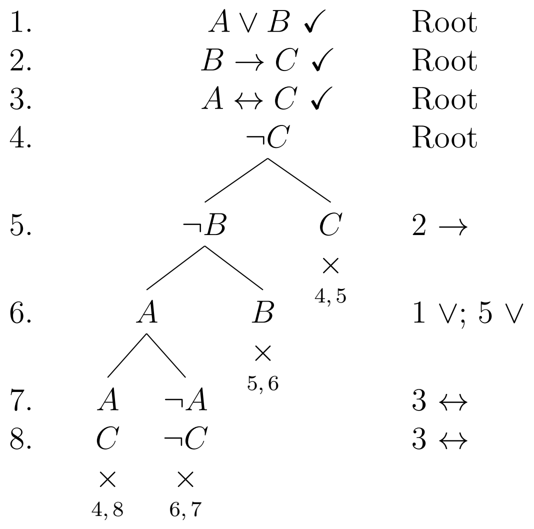
3. $P \wedge Q$, $\neg R \to \neg Q$ $\therefore$ $P \wedge R$ is valid.


1. $P \wedge Q$ ✓ Premise
2. $\neg R \to \neg Q$ ✓ Premise
3. $\neg(P \wedge R)$ ✓ Neg Conc
4. P $1 \wedge$
5. Q $1 \wedge$
6. $\neg\neg R$ $\neg Q$ $2 \to$
$\times$ $5, 6$
7. $\neg P$ $\neg R$ $3 \neg\wedge$
$\times$ $4, 7$ $\times$ $6, 7$

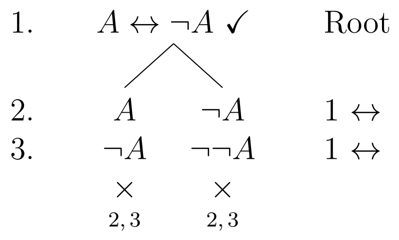
4. Every valuation making P, $P \to Q$, and $\neg Q$ true also makes A true.



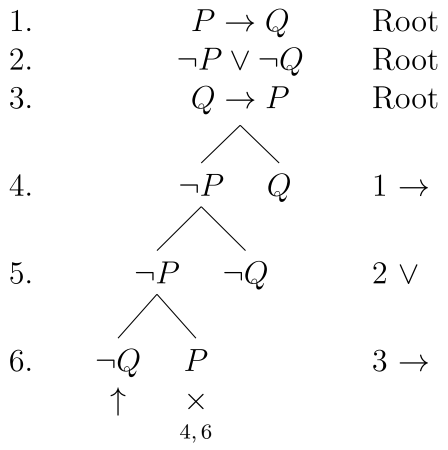
5. $A \vee B$, $B \rightarrow C$, and $A \leftrightarrow C$ cannot all be true but C false.


1. $A \vee B$ ✓ Root
2. $B \rightarrow C$ ✓ Root
3. $A \leftrightarrow C$ ✓ Root
4. $\neg C$ Root
5. $\neg B$ C $2 \rightarrow$
× 4, 5
6. A B $1 \vee$; $5 \vee$
× 5, 6
7. A $\neg A$ $3 \leftrightarrow$
8. C $\neg C$ $3 \leftrightarrow$
× 4, 8
× 6, 7

6. $A \leftrightarrow \neg A$ is a contradiction.

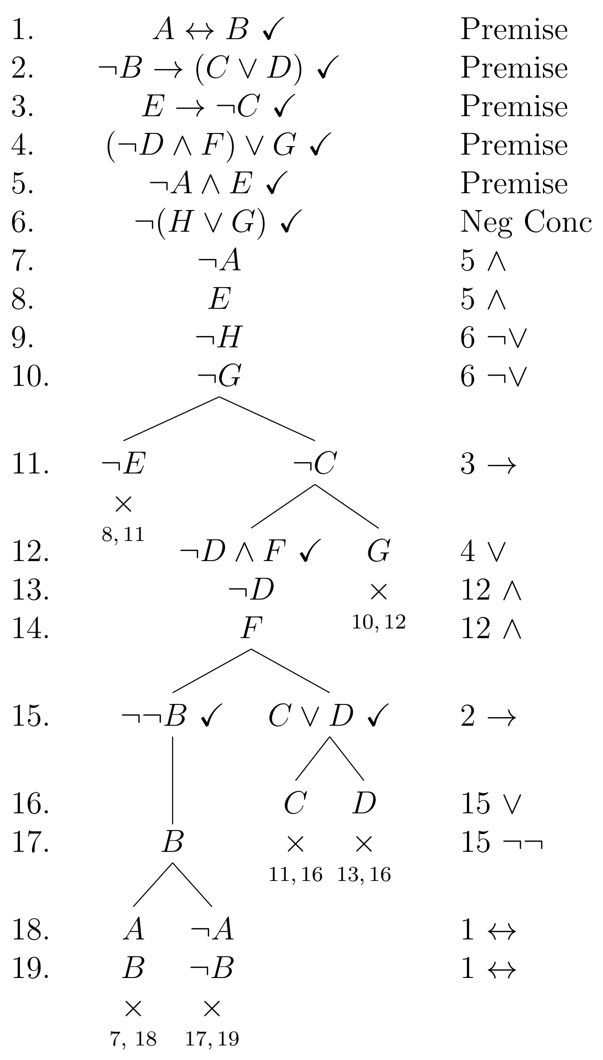

1. $A \leftrightarrow \neg A$ ✓ Root
2. A $\neg A$ $1 \leftrightarrow$
3. $\neg A$ $\neg\neg A$ $1 \leftrightarrow$
× 2, 3
× 2, 3

7. $P \rightarrow Q$, $\neg P \vee \neg Q$, and $Q \rightarrow P$ are mutually consistent.


1. $P \rightarrow Q$ Root
2. $\neg P \vee \neg Q$ Root
3. $Q \rightarrow P$ Root
4. $\neg P$ Q $1 \rightarrow$
5. $\neg P$ $\neg Q$ $2 \vee$
6. $\neg Q$ P $3 \rightarrow$
↑
× 4, 6

P	Q	$(P$	$\rightarrow$	$Q)$	$(\neg$	P	$\vee$	$\neg$	$Q)$	$(Q$	$\rightarrow$	$P)$
0	0	0	1	0	1	0	1	1	0	0	1	0

Chapter 13 Ex C:


1. $A \leftrightarrow B$ ✓ Premise
2. $\neg B \to (C \vee D)$ ✓ Premise
3. $E \to \neg C$ ✓ Premise
4. $(\neg D \wedge F) \vee G$ ✓ Premise
5. $\neg A \wedge E$ ✓ Premise
6. $\neg(H \vee G)$ ✓ Neg Conc
7. $\neg A$ 5 $\wedge$
8. E 5 $\wedge$
9. $\neg H$ 6 $\neg\vee$
10. $\neg G$ 6 $\neg\vee$
11. $\neg E$ | $\neg C$ 3 $\to$
× 8, 11
12. $\neg D \wedge F$ ✓ | G 4 $\vee$
13. $\neg D$ | × 12 $\wedge$
14. F | 10, 12 12 $\wedge$
15. $\neg\neg B$ ✓ | $C \vee D$ ✓ 2 $\to$
16. C | D 15 $\vee$
17. B | × × 15 $\neg\neg$
11, 16 13, 16
18. A | $\neg A$ 1 $\leftrightarrow$
19. B | $\neg B$ 1 $\leftrightarrow$
× ×
7, 18 17, 19

This argument is valid.

Chapter 15.2 Ex C:

1. The scope of $\exists x$ is Fx and the scope of $\exists y$ is Gy.
2. The scope of $\exists x$ is Fx and the scope of $\exists y$ is Fy.
3. The scope of $\exists x$ is $(Fx \wedge \exists y Gy)$ and the scope of $\exists y$ is Gy.
4. The scope of $\exists x$ is $(Fx \to \exists y((Gy \vee Hy) \wedge \neg\forall z(Fx \vee Gz)))$, the scope of $\exists y$ is $((Gy \vee Hy) \wedge \neg\forall z(Fx \vee Gz))$ and the scope of $\forall z$ is $(Fx \vee Gz)$.
5. The scope of $\exists x$ is $(Ga \wedge Hx))$.

Chapter 16 Ex E:

1. $Za \wedge Zb \wedge Zc$
2. $Rb \wedge \neg Ab$
3. $Lcb \to Mb$
4. $(Ab \wedge Ac) \to (Lab \wedge Lac)$
5. $\exists x(Rx \wedge Zx)$
6. $\forall x(Ax \to Rx)$
7. $\forall x\big[Zx \to (Mx \vee Ax)\big]$
8. $\exists x(Rx \wedge \neg Ax)$
9. $\exists x(Rx \wedge Lcx)$
10. $\forall x\big[(Mx \wedge Zx) \to Lbx\big]$
11. $\forall x\big[(Mx \wedge Lax) \to Lxa\big]$
12. $\exists x Rx \to Ra$
13. $\forall x(Ax \to Rx)$
14. $\forall x\big[(Mx \wedge Lcx) \to Lax\big]$
15. $\exists x(Mx \wedge Lxb \wedge \neg Lbx)$

Chapter 17 Ex G:

1. $\forall x[Cxp \to Dx]$
2. $Cjp \wedge Fj$
3. $\exists x[Cxp \wedge Fx]$
4. $\neg\exists x[Sxj]$
5. $\forall x\big[(Cxp \wedge Mx) \to Dx\big]$
6. $\neg\exists x[Cxp \wedge Mx]$
7. $\exists x[Cjx \wedge Sxe \wedge Fj]$
8. $Spe \wedge Mp$
9. $\forall x\big[(Sxp \wedge Mx) \to \neg\exists y Cyx\big]$
10. $\exists x[Sxj \wedge \exists y[Cyx] \wedge Fj]$
11. $\forall x\big[Dx \to \exists y(Sxy \wedge My \wedge Dy)\big]$
12. $\forall x\big[(Fx \wedge Dx) \to \exists y[Cxy \wedge Dy]\big]$

Chapter 18 Ex D:

1. $\forall x[Cx \to (Bx \wedge Sx)]$.
2. $\neg\exists x[Wx \wedge Sx]$
3. $\exists x\exists y[Cx \wedge Cy \wedge x \neq y]$
4. $\exists x\exists y[Jx \wedge Ox \wedge Jy \wedge Oy \wedge x \neq y]$
5. $\forall x\forall y\forall z\big[(Jx \wedge Ox \wedge Jy \wedge Oy \wedge Jz \wedge Oz) \to (x = y \vee x = z \vee y = z)\big]$
6. $\exists x\exists y\big[Jx \wedge Bx \wedge Jy \wedge By \wedge x \neq y \wedge \forall z[(Jz \wedge Bz) \to (x = z \vee y = z)]\big]$
7. $\exists x_1\exists x_2\exists x_3\exists x_4\big[Dx_1 \wedge Dx_2 \wedge Dx_3 \wedge Dx_4 \wedge x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge x_3 \neq x_4 \wedge \neg\exists y[Dy \wedge y \neq x_1 \wedge y \neq x_2 \wedge y \neq x_3 \wedge y \neq x_4]\big]$

8. $\exists x\big[Dx \land Cx \land \forall y[(Dy \land Cy) \to x = y] \land (Bx \land Sx)\big]$
9. $\forall x\big[(Ox \land Jx) \to Wx\big] \land \exists x\big[Mx \land \forall y[My \to x = y] \land Wx\big]$
10. $\exists x\big[Dx \land Cx \land \forall y[(Dy \land Cy) \to x = y] \land (Wx \land Sx)\big] \to$
    $\exists x \forall y[(Wx \land Sx) \leftrightarrow x = y]$
11. wide scope: $\neg\exists x\big[Mx \land \forall y(My \to x = y) \land Jx\big]$
    narrow scope: $\exists x\big[Mx \land \forall y(My \to x = y) \land \neg Jx\big]$
12. wide scope: $\neg\exists x \exists z\big[Dx \land Cx \land Mz \land \forall y[(Dy \land Cy) \to x = y] \land$
    $\forall y[(My \to z = y) \land x = z]\big]$
    narrow scope: $\exists x \exists z\big[Dx \land Cx \land Mz \land \forall y[(Dy \land Cy) \to x = y] \land$
    $\forall y[(My \to z = y) \land x \neq z]\big]$

Chapter 21 Ex A:

1. True
2. False
3. True
4. False
5. False
6. False
7. True
8. True
9. False
10. True
11. True
12. False
13. True
14. False
15. True

Chapter 23 Ex A:

1. $\forall x \forall y(Gxy \to \exists z Gxz)$

1.	$\neg\forall x\,\forall y\,Gxy \to \exists z\,[Gxz]$ ✓	Root
2.	$\exists x\,\neg\forall y\,Gxy \to \exists z\,[Gxz]$ ✓a	1 $\neg\forall$
3.	$\neg\forall y\,Gay \to \exists z\,[Gaz]$ ✓	2 $\exists$ a
4.	$\exists y\,\neg(Gay \to \exists z\,[Gaz])$ ✓b	3 $\neg\forall$
5.	$\neg(Gab \to \exists z\,[Gaz])$	4 $\exists$ b
6.	Gab	5 $\neg\to$
7.	$\neg\exists z\,[Gaz]$ ✓	5 $\neg\to$
8.	$\forall z\,[\neg Gaz]\ \backslash b$	7 $\neg\exists$
9.	$\neg Gab$	8 $\forall$ b
	$\times$	

2. $\forall xFx \lor \forall x(Fx \to Gx)$

1.	$\neg(\forall x\,[Fx] \lor \forall x\,[Fx \to Gx])\ \checkmark$	Root
2.	$\neg\forall x\,[Fx]\ \checkmark$	1 $\neg\lor$
3.	$\neg\forall x\,[Fx \to Gx]\ \checkmark$	1 $\neg\lor$
4.	$\exists x\,[\neg Fx]\ \checkmark a$	2 $\neg\forall$
5.	$\neg Fa\ \checkmark$	4 $\exists$ a
6.	$\exists x\,[\neg(Fx \to Gx)]\ \checkmark b$	3 $\neg\forall$
7.	$\neg(Fb \to Gb)\ \checkmark$	3 $\exists$ b
8.	Fb	7 $\neg \to$
9.	$\neg Gb$	7 $\neg \to$
	$\uparrow$	

	F	G
a	0	?
b	1	0

3. $\forall x(Fx \to (\neg Fx \to \forall yGy))$

1.	$\neg\forall x\,[Fx \to (\neg Fx \to \forall y\,[Gy])]\ \checkmark$	Root
2.	$\exists x\,[\neg(Fx \to (\neg Fx \to \forall y\,[Gy]))]\ \checkmark$	1 $\neg\forall$
3.	$\neg(Fa \to (\neg Fa \to \forall y\,[Gy]))\ \checkmark$	2 $\exists$ a
4.	Fa	3 $\neg \to$
5.	$\neg(\neg Fa \to \forall y\,[Gy])$	3 $\neg \to$
6.	$\neg Fa$	5 $\neg \to$
7.	$\neg\forall y\,[Gy]$	5 $\neg \to$
	$\times$	

4. $\exists xJx \leftrightarrow \neg\forall x\neg Jx$

1.	$\neg(\exists x\,[Jx] \leftrightarrow \neg\forall x\,[\neg Jx])\ \checkmark$		Root
2.	$\exists x\,[Jx]\ \checkmark a$	$\neg\exists x\,[Jx]$	1 $\neg \leftrightarrow$
3.	$\neg\neg\forall x\,[\neg Jx]\ \checkmark$	$\neg\forall x\,[\neg Jx]$	1 $\neg \leftrightarrow$
4.	$\forall x\,[\neg Jx]\ \checkmark$		3 $\neg\neg$
5.	Ja		2 $\exists$ a
6.	$\neg Ja$		4 $\forall$ a
7.	$\times$	$\forall x\,[\neg Jx]\ \backslash b$	2 $\neg\exists$
8.		$\exists x\,[\neg\neg Jx]\ \checkmark b$	3 $\neg\forall$
9.		$\neg\neg Jb$	8 $\exists$ b
10.		$\neg Jb$	7 $\forall$ b
		$\times$	

5. $\exists x(Fx \vee \neg Fx)$

1.	$\neg\exists x\,[Fx \vee \neg Fx]$ ✓	Root
2.	$\forall x\,[\neg(Fx \vee \neg Fx)]\ \backslash a$	1 $\neg\exists$
3.	$\neg(Fa \vee \neg Fa)$ ✓	2 $\forall$ a
4.	$\neg Fa$	3 $\neg\vee$
5.	$\neg\neg Fa$	3 $\neg\vee$
	$\times$	

6. $\forall x(Fx \vee Gx) \rightarrow (\forall yFy \vee \exists xGx)$

1.	$\neg(\forall x\,[Fx \vee Gx] \rightarrow (\forall y\,[Fy] \vee \exists x\,[Gx]))$ ✓	Root
2.	$\forall x\,[Fx \vee Gx]\ \backslash a$	1 $\neg \rightarrow$
3.	$\neg(\forall y\,[Fy] \vee \exists x\,[Gx])$ ✓	1 $\neg \rightarrow$
4.	$\neg\forall y\,[Fy]$ ✓	3 $\neg\vee$
5.	$\neg\exists x\,[Gx]$ ✓	3 $\neg\vee$
6.	$\exists y\,[\neg Fy]$ ✓a	4 $\neg\exists$
7.	$\neg Fa$	6 $\neg\exists$
8.	$\forall x\,[\neg Gx]\ \backslash a$	5 $\neg\exists$
9.	$Fa \vee Ga$	2 $\forall$ a
10.	$Fa \quad Ga$	9 $\vee$
11.	$\times \quad \neg Ga$	8 $\forall$ a
	$\times$	

Chapter 23 Ex B:

1. Fa, Ga $\therefore$ $\forall x(Fx \rightarrow Gx)$

1.	Fa	Premise
2.	Ga	Premise
3.	$\neg\forall x\,[Fx \rightarrow Gx]$ ✓	Neg Conc
4.	$\exists x\,[\neg(Fx \rightarrow Gx)]$ ✓b	3 $\neg\forall$
5.	$\neg(Fb \rightarrow Gb)$	4 $\exists$ b
6.	Fb	5 $\neg \rightarrow$
7.	$\neg Gb$	5 $\neg \rightarrow$
	$\uparrow$	

	F	G
a	1	1
b	1	0

2. Fa, Ga $\therefore$ $\exists x(Fx \wedge Gx)$

1.	Fa		Premise
2.	Ga		Premise
3.	$\neg\exists x\,[Fx \wedge Gx]$ ✓		Neg Conc
4.	$\forall x\,[\neg(Fx \wedge Gx)] \setminus a$		3 $\neg\exists$
5.	$\neg(Fa \wedge Ga)$ ✓		4 $\forall$ a
6.	$\neg Fa$	$\neg Ga$	5 $\neg\wedge$
	$\times$	$\times$	

3. $\forall x \exists y Lxy$ $\therefore$ $\exists x \forall y Lxy$

1.	$\forall x\,\exists y\,[Lxy] \setminus a$	Premise
2.	$\neg\exists x\,\forall y\,[Lxy]$ ✓	Neg Conc
3.	$\forall x\,\neg\forall y\,[Lxy] \setminus a$	2 $\neg\exists$
4.	$\exists y\,[Lay]$ ✓b	1 $\forall$ a
5.	Lab	4 $\exists$ b
6.	$\neg\forall y\,[Lay]$ ✓	3 $\forall$ a
7.	$\exists y\,[\neg Lay]$ ✓c	6 $\neg\forall$
8.	$\neg Lac$	7 $\exists$ c
	$\uparrow$	

L	a	b	c
a	?	1	0
b	1	?	?
c	1	?	?

Note that this tree is infinite, as both #1/#4 and #3/#6 will generate new atomic letters every time they are decomposed. The greyed values are not read from the tree, but instead 'guessed' to complete the counter-example.

4. $\exists x(Fx \wedge Gx)$, $Fb \leftrightarrow Fa$, $Fc \rightarrow Fa$ $\therefore$ Fa

1.	$\exists x\,[Fx \wedge Gx]$ ✓			Premise
2.	$Fb \leftrightarrow Fa$			Premise
3.	$Fc \rightarrow Fa$			Premise
4.	$\neg Fa$			Neg Conc
5.	$Fd \wedge Gd$			1 ∃ d
6.	$\neg Fc$		Fa	3 →
			×	
7.	Fb	$\neg Fb$		2 ↔
8.	Fa	$\neg Fa$		2 ↔
9.	×	Fd		5 ∧
10.		Gd		5 ∧
		↑		

	F	G
a	0	?
b	0	?
c	0	?
d	1	1

5. $\forall x \exists y Gyx$ $\therefore$ $\forall x \exists y(Gxy \vee Gyx)$

1.	$\forall x\,[\exists y\, Gyx] \setminus a$	Premise
2.	$\neg\forall x\,[\exists y\, Gxy \vee Gyx]$ ✓	Neg Conc
3.	$\exists x\,[\neg\exists y\, Gxy \vee Gyx]$ ✓a	2 ¬∀
4.	$\neg\exists y\,[Gay \vee Gya]$ ✓	3 ∃ a
5.	$\exists y\, Gya$ ✓b	1 ∀ a
6.	Gba	5 ∃ b
7.	$\forall y\,[\neg(Gay \vee Gya)] \setminus b$	4 ¬∀
8.	$\neg(Gab \vee Gab)$ ✓	7 ∀ b
9.	$\neg Gab$	8 ¬∨
10.	$\neg Gba$	8 ¬∨
	×	

Chapter 24 Ex A:

1. $b = b \to (Rca \lor Pb), a = b \to (Rcb \lor Pb), b = b \to (Rcb \lor Pb)$
2. $a = a \to (Rca \lor Pb), a = b \to (Rca \lor Pa), a = a \to (Rca \lor Pa)$.
3. $c = b \to (Rca \lor Pb), a = b \to (Rcc \lor Pb), c = b \to (Rcc \lor Pb)$.
4. $a = b \to (Raa \lor Pb)$.
5. $a = c \to (Rca \lor Pb), a = b \to (Rca \lor Pc), a = c \to (Rca \lor Pc)$.
6. $a = b \to (Rba \lor Pb)$.

Chapter 24 Ex A:

1. $\forall x(x = a \lor x = b)$, $\neg(Fa \land Ga)$, $Gb \to Hb$ $\therefore$ $\forall x((Fx \land \neg Hx) \to \neg Gx)$

1.	$\forall x\,[x = a \lor x = b]\ \backslash c$	Premise
2.	$\neg(Fa \land Ga)$ ✓	Premise
3.	$Gb \to Hb$ ✓	Premise
4.	$\neg\forall x\,[(Fx \land \neg Hx) \to \neg Gx]$ ✓	Neg Conc
5.	$\exists x\,[\neg((Fx \land \neg Hx) \to \neg Gx)]$ ✓a	4 $\neg\forall$
6.	$\neg((Fc \land \neg Hc) \to \neg Gc)$ ✓	5 $\exists$ c
7.	$Fc \land \neg Hc$ ✓	6 $\neg \to$
8.	$\neg\neg Gc$ ✓	6 $\neg \to$
9.	Gc	8 $\neg\neg$
10.	Fc	7 $\land$
11.	$\neg Hc$	7 $\land$
12.	$c = a \lor c = b$ ✓	1 $\forall$ c
13.	$c = a$ $\qquad$ $c = b$	12 $\lor$
14.	$\neg(Fc \land Gc)$ ✓ $\qquad$ $\vert$	$2, 13\ [a/c]$
15.	$\neg Fc$ $\quad$ $\neg Gc$ $\qquad$ $\vert$	14 $\neg\land$
16.	$\times$ $\quad$ $\times$ $\quad$ $Gc \to Hc$ ✓	$3, 13\ [b/c]$
17.	$\neg Gc$ $\quad$ Hc	16 $\to$
	$\times$ $\quad$ $\times$	

2. $\exists x Fx$, $\forall x \forall y((Fx \wedge Fy) \to x = y)$ $\therefore$ $\exists x \forall y(Fy \leftrightarrow (x = y))$

1.	$\forall x\,[Fx]$ ✓a				Premise
2.	$\forall x\,\forall y\,[(Fx \wedge Fy) \to x = y]$ \\$a$				Premise
3.	$\neg\exists x\,\forall y\,[Fy \leftrightarrow (x = y)]$ ✓				Neg Conc
4.	$\forall x\,[\neg(\forall y\,[Fy \leftrightarrow (x = y)])]$ \\$a$				3 $\neg\exists$
5.	Fa				1 $\exists$ a
6.	$\neg(\forall y\,[Fy \leftrightarrow (a = y)])$ ✓				4 $\forall$ a
7.	$\exists y\,[\neg(Fy \leftrightarrow (a = y))]$ ✓b				6 $\neg\forall$
8.	$\neg(Fb \leftrightarrow (a = b))$ ✓				7 $\exists$ b
9.	Fb			$\neg Fb$	8 $\neg\leftrightarrow$
10.	$\neg(a = b)$			$a = b$	8 $\neg\leftrightarrow$
11.	$\forall y\,[(Fa \wedge Fy) \to a = y]$ \\$b$				2 $\forall$ a
12.	$(Fa \wedge Fb) \to a$ ✓				11 $\forall$ b
13.				$\neg Fa$	$9, 10\ [b/a]$
14.	$\neg(Fa \wedge Fb)$ ✓		$a = b$	$\times$	12 $\to$
			$\times$		
15.	$\neg Fa$	$\neg Fb$			14 $\neg\wedge$
	$\times$	$\times$			

3. $\forall x \forall y(Fx \to x = y)$, $\forall x(Fx \vee Gx)$ $\therefore$ $(\forall x Fx \vee \forall x Gx)$

1.	$\forall x\,\forall y\,[Fx \to x = y]$ \\$b$			Premise
2.	$\forall x\,[Fx \vee Gx]$ \\b			Premise
3.	$\neg(\forall x\,[Fx] \vee \forall x\,[Gx])$ ✓			Neg Conc
4.	$\neg\forall x\,[Fx]$ ✓			3 $\neg\vee$
5.	$\neg\forall x\,[Gx]$ ✓			3 $\neg\vee$
6.	$\exists x\,[\neg Fx]$ ✓a			4 $\neg\forall$
7.	$\neg Fa$			6 $\exists$ a
8.	$\exists x\,[\neg Gx]$ ✓b			5 $\neg\forall$
9.	$\neg Gb$			8 $\exists$ b
10.	$Fb \vee Gb$ ✓			2 $\forall$ b
11.	Fb		Gb	10 $\vee$
12.	$\forall y\,[Fb \to b = y]$ \\$a$		$\times$	1 $\forall$ b
13.	$Fb \to b = a$ ✓			12 $\forall$ a
14.	$\neg Fb$	$b = a$		13 $\to$
15.	$\times$	Fa		$11, 14\ [b/a]$
		$\times$		

4. $\exists x \forall y (Fy \leftrightarrow (x = y)) \therefore \exists x Fx$

1.	$\exists x \, [\forall y \, [Fy \leftrightarrow (x = y)]] \checkmark a$		Premise
2.	$\neg\exists x \, [Fx] \checkmark$		Neg Conc
3.	$\forall x \, [\neg Fx] \setminus a$		2 ¬∃
4.	$\forall y \, [Fy \leftrightarrow (a = y)] \setminus a$		1 ∃ a
5.	$\neg Fa$		3 ∀ a
6.	$Fa \leftrightarrow (a = a)$		4 ∀ a
7.	Fa	$\neg Fa$	6 ↔
8.	$a = a$	$\neg(a = a)$	6 ↔
9.	$\times$	$a = a$	Identity
		$\times$	

5. $\exists x \forall y (Fy \leftrightarrow (x = y)) \therefore \forall x \forall y ((Fx \wedge Fy) \rightarrow (x = y))$

1.	$\exists x \, \forall y \, [Fy \leftrightarrow (x = y)] \checkmark c$			Premise
2.	$\neg\forall x \, \forall y \, [(Fx \wedge Fy) \rightarrow (x = y)] \checkmark$			Neg Conc
3.	$\exists x \, [\neg\forall y \, [(Fx \wedge Fy) \rightarrow (x = y)]] \checkmark a$			2 ¬∀
4.	$\neg\forall y \, [(Fa \wedge Fy) \rightarrow (a = y)] \checkmark$			3 ∃ a
5.	$\exists y \, [\neg((Fa \wedge Fy) \rightarrow (a = y))] \checkmark b$			4 ¬∀
6.	$\neg((Fa \wedge Fb) \rightarrow (a = b)) \checkmark$			5 ∃ b
7.	$Fa \wedge Fb \checkmark$			6 ¬ →
8.	$\neg(a = b) \checkmark$			6 ¬ →
9.	Fa			7 ∧
10.	Fb			7 ∧
11.	$\forall y \, [Fy \leftrightarrow (c = y)] \setminus a, b$			1 ∃ c
12.	$Fa \leftrightarrow (c = a) \checkmark$			11 ∀ a
13.	Fa		$\neg Fa$	12 ↔
14.	$c = a$		$\neg(c = a)$	12 ↔
15.	$Fb \leftrightarrow (c = b) \checkmark$		$\times$	11 ∀ b
16.	Fb	$\neg Fb$		15 ↔
17.	$c = b$	$\neg(c = b)$		15 ↔
18.	$a = b$	$\times$		$14, 17$ $[c/a]$
	$\times$			

6. $\exists x \forall y (Fy \leftrightarrow (x = y)), \forall x (Fx \vee Gx) \therefore \forall x \forall y ((Gx \vee Gy) \vee x = y)$

1.	$\exists x\, [\forall y\, [Fy \leftrightarrow (x = y)]]\ \checkmark c$	Premise
2.	$\forall x\, [Fx \vee Gx]\ \backslash a, b$	Premise
3.	$\neg \forall x\, [\forall y\, [(Gx \vee Gy) \vee x = y]]\ \checkmark$	Neg Conc
4.	$\exists x\, [\neg \forall y\, [(Gx \vee Gy) \vee x = y]]\ \checkmark a$	3 $\neg\forall$
5.	$\neg \forall y\, [(Ga \vee Gy) \vee a = y]\ \checkmark$	4 $\exists$ a
6.	$\exists y\, [\neg((Ga \vee Gy) \vee a = y)]\ \checkmark b$	5 $\neg\forall$
7.	$\neg((Ga \vee Gb) \vee a = b)\ \checkmark$	6 $\exists$ b
8.	$\neg(Ga \vee Gb)\ \checkmark$	7 $\neg\vee$
9.	$\neg(a = b)\ \checkmark$	7 $\neg\vee$
10.	$\neg Ga$	8 $\neg\vee$
11.	$\neg Gb$	8 $\neg\vee$
12.	$\forall y\, [Fy \leftrightarrow (c = y)]\ \backslash a, b$	1 $\exists$ c
13.	$Fa \vee Ga\ \checkmark$	2 $\forall$ a
14.	$Fa \qquad Ga$	13 $\vee$
15.	$Fb \vee Gb\ \checkmark \qquad \times$	2 $\forall$ b
16.	$Gb \qquad Fb$	15 $\vee$
17.	$\times \qquad Fa \leftrightarrow (c = a)\ \checkmark$	12 $\forall$ a
18.	$Fa \qquad \neg Fa$	17 $\leftrightarrow$
19.	$c = a \qquad \neg(c = a)$	17 $\leftrightarrow$
20.	$Fb \leftrightarrow (c = b)\ \checkmark \qquad \times$	12 $\forall$ a
21.	$Fb \qquad \neg Fb$	20 $\leftrightarrow$
22.	$c = b \qquad \neg(c = b)$	20 $\leftrightarrow$
23.	$a = b \qquad \times$	$19, 22\ [c/a]$
	$\times$	