A Modern Formal Logic Primer

Paul Teller

Prentice Hall, 1989

Available online at:
http://tellerprimer.ucdavis.edu/

(PDF partial edition with an added Appendix, 2014)
A Modern
Formal Logic
Primer

Previously published by
Pearson Education, Inc.

Preface to Volumes I and II
A Guide to the Primer

This text is a primer in the best sense of the word: A book which presents the basic elements of a subject. In other respects, I have sought to write a different kind of text, breaking with what I regard as an unfortunate tradition in teaching formal logic. From truth tables through completeness, I seek to explain, as opposed to merely presenting my subject matter. Most logic texts (indeed, most texts) put their readers to sleep with a formal, dry style. I have aimed for a livelier lecture style, which treats students as human beings and not as knowledge receptacles. In a text, as in the classroom, students need to be encouraged and to hear their difficulties acknowledged. They need variation in pace. They need shifts in focus among “I,” “we,” and “you,” just as most of us speak in the classroom. From time to time students simply need to rest their brains.

One fault of logic textbooks especially bothers me: Some authors feel so concerned to teach rigor that they end up beating their students over the head with it. I have not sacrificed rigor. But I have sought to cultivate it rather than rubbing it in.

Now to the contents of the Primer. Volume I presents sentence logic. Volume II, Part I lays out predicate logic, including identity, functions, and definite descriptions; Part II introduces metatheory, including mathematical induction, soundness, and completeness. The text includes completely independent presentations of Fitch-style natural deduction and
the tree method as developed by Richard Jeffrey. I have presented the material with a great deal of modularity.

I have presented the text in two volumes to maximize flexibility of use in a variety of courses. Many introductory courses cover a mix of informal and formal logic. Too often I have heard instructors express dissatisfaction with what they find available for the formal portion of such a course. Volume I provides a new option. Using it in tandem with any of the many available inexpensive informal texts, instructors can combine the best of both subjects. Volume I will present a serious-minded introduction to formal logic, which at the same time should prove accessible and encouraging to those students who will never again take another logic course. The relatively small numbers who continue to a second course, devoted exclusively to formal logic, need only purchase Volume II to build on the foundation already laid.

The Primer incorporates a number of unusual features. Chapters 1, 3, and 4 emphasize the concept of a truth function. Though the idea is simple once you get it, many students need several passes. The optional section 3–4, on disjunctive normal form and the Schiffer stroke, serves the didactic function of providing yet more drill on truth functionality.

Following Richard Jeffrey, I have thoroughly presented `&', `v', and `¬' before treating `→' and `≡'. `&', `v', and `¬' are much less controversial correlates of their English counterparts than is `→'. Using `&', `v' and `¬' as a vehicle for introducing the idea of a truth function, I can deal honestly with the difficulties of giving a truth functional formulation of conditionals. In turn, this honest examination provides further drill with the concept of a truth function.

Sentences in English and logic often do not correspond very accurately. Consequently, I speak of transcription, not translation between logic and English. I treat sentence logic transcription quite briefly in chapter 1 of Volume I and further in the short, optional chapter 2. Predicate logic transcription gets a minimal introduction in chapter 1 of Volume II and then comes in for a thorough workout in chapter 4, also optional. There I deal with the subject matter of domains and the traditional square of opposition by using the much more general method of restricted quantifier subscripts and their elimination. This technique provides an all-purpose tool for untangling complicated transcription problems. Chapter 4 of Volume II also examines quantificational ambiguity in English, which most logic texts strangely ignore.

Training in metatheory begins in Volume I, chapter 1. But the training is largely implicit: I use elementary ideas, such as metavariables, and then call attention to them as use makes their point apparent. After thorough preparation throughout the text, chapter 10 of Volume II brings together the fundamental ideas of metatheory.

Standard treatments of sentence logic present sentence logic semantics, in the form of truth tables, before sentence logic derivation rules. Only in this way do students find the rules clearly intelligible, as opposed to poorly understood cookbook recipes. Often texts do not follow this heuristic for predicate logic, or they do so only half-heartedly. Presumably, authors fear that the concept of an interpretation is too difficult. However, one can transparently define interpretations if one makes the simplifying assumption of including a name for each object in an interpretation's domain, in effect adopting a substitutional interpretation of the quantifiers. I further smooth the way by stressing the analogy of form and function between interpretations and truth value assignments in sentence logic.

This approach is ample for fixing basic ideas of semantics and for making predicate logic rules intelligible. After introducing predicate logic syntax in Volume II, chapter 1, and semantics in chapters 2 and 3, tree rules are almost trivial to teach; and derivation rules, because they can be better motivated, come more easily. I have clearly noted the limitation in my definition of an interpretation, and I have set students thinking, in an exercise, why one may well not want to settle for a substitutional interpretation. Finally, with the ground prepared by the limited but intuitive definitions of chapters 2 and 3 of Volume II, students have a relatively easy time with the full characterization of an interpretation in chapter 15.

No one has an easy time learning—or teaching—natural deduction quantifier rules. I have worked hard to motivate them in the context of informal argument. I have made some minor modifications in detail of formulation, modifications which I believe make the rules a little easier to grasp and understand. For existential elimination, I employ the superficially restrictive requirement that the instantiating name be restricted to the sub-derivation. I explain how this restriction works to impose the more complex and traditional restrictions, and I set this up in the presentation so that instructors can use the more traditional restrictions if they prefer.

For the proof of completeness of the natural deduction system I have fashioned my own semantic tableau proof. I believe that on its own it is at least as accessible as the Henkin and other more familiar proofs. In addition, if you do tree completeness first, you can explain the natural deduction completeness proof literally in a few minutes.

I have been especially careful not to dive into unexplained proofs of soundness and completeness. Instructors will find, in separate sections, informal and intuitive explanations of the sentence logic proofs, unencumbered with formal details, giving an understanding of how the proofs work. These sections require only the first short section of the induction chapter. Instructors teaching metatheory at a more elementary level may
want to conclude with some of these sections. Those ready for the tonic of rigor will find much to satisfy them in the succeeding sections.

In some chapters I have worked as hard on the exercises as on the text. I have graded the skill problems, beginning with easy comprehension checkers, through skill builders, to some problems which will test real skill mastery. I think few will not find enough problems.

Exercises should exercise understanding as well as skills. Any decent mathematics text puts problems to this task, as well as uses them to present auxiliary material. Too few logic texts fall in this tradition. I hope that students and instructors will enjoy my efforts in some of the exercises to introduce auxiliary material, to lay foundations for succeeding material, to engage creative understanding, and to join in the activity of conceptual exploration.

For teaching plans the key word is "modularity." Those using just Volume I in an informal/formal course may teach chapters 1, 2 (optional), 3, and 4 to introduce sentence logic. Then, as taste and time permit, you may do natural deduction (chapters 5, 6, and 7) or trees (chapters 8 and 9), or both, in either order.

Volumes I and II together provide great flexibility in a first symbolic logic course. Given your introduction of sentence logic with chapters 1, 3, and 4 of Volume I and grounding of predicate logic with chapters 1, 2, and 3 of Volume II you can do almost anything you want. I have made treatment of derivations and trees completely independent. You can run through the one from sentence to predicate logic, and then go back and do the other. Or you can treat both natural deduction and trees for sentence logic before continuing to predicate logic. You can spend up to two weeks on transcription in chapter 2 of Volume I and chapter 4 of Volume II, or you can rely on the minimal discussion of transcription in the first chapters of Volumes I and II and omit chapter 2 of Volume I and chapter 4 of Volume II altogether.

If you do both trees and natural deduction, the order is up to you. Trees further familiarize students with semantics, which helps in explaining natural deduction rules. On the other hand, I have found that after teaching natural deduction I can introduce trees almost trivially and still get their benefit for doing semantics and metatheory.

Your only limitation is time. Teaching at an urban commuter university, in one quarter I cover natural deduction (Volume I, chapters 1, 2, 3, 4, 5, 6, 7; Volume II, chapters 1, 2, 3, 5, and perhaps 6), or trees and sentence logic natural deduction (Volume I, chapters 1, 2, 3, 4, 8, 9; Volume II, chapters 1, 2, 3, 5, 8; Volume I, chapters 5, 6, and 7). A semester should suffice for all of Volume I and Volume II through chapter 8, and perhaps 9. Again, you may want to follow the chapter sequencing, or you may want to do natural deduction first, all the way through predicate logic, or trees first.

If you do just natural deduction or just trees you have more time for identity, functions, definite descriptions, and metatheory. Chapter 10 of Volume II, basic metatheoretical concepts, can provide a very satisfying conclusion to a first course. A two quarter sequence may suffice for all of the metatheory chapters, especially if you do not do both natural deduction and trees thoroughly. To this end the metatheory chapters cover soundness and completeness for both natural deduction and trees independently. Or, you may choose to end with the sections presenting the informal explanations of induction and the soundness and completeness proofs. The text will provide a leisurely full year course or a faster paced full year course if you supplement it a bit at the end of the year.

I want to mention several features of my usage. I use single quotes to form names of expressions. I depart from logically correct use of quotation marks in one respect. In stating generalizations about arguments I need a formulation which makes explicit use of metavariables for premise and conclusion. But before chapter 10 of Volume II, where I make the metalanguage/object language distinction explicit, I do not want to introduce a special argument forming operator because I want to be sure that students do not mistake such an operator for a new symbol in the object language. Consequently I use the English word 'therefore'. I found, however, that the resulting expressions were not well enough set off from their context. For clarity I have used double quotes when, for example, I discuss what one means by saying that an argument, "X. Therefore Y." is valid.

Throughout I have worked to avoid sexist usage. This proves difficult with anaphoric reference to quantified variables, where English grammar calls for constructions such as 'If someone is from Chicago he likes big cities.' and 'Anyone who loves Eve loves himself.' My solution is to embrace grammatical reform and use a plural pronoun: 'If someone is from Chicago they like big cities.' and 'Anyone who loves Eve loves themselves.' I know. It grates. But the offense to grammar is less than the offense to social attitudes. As this reform takes hold it will sound right to all of us.

I thank the many friends and family who have actively supported this project, and who have born with me patiently when the toil has made me hard to live with. I do not regard the project as finished. Far from it. I hope that you—teachers and students—will write me. Let me know where I am still unclear. Give me your suggestions for further clarification, for alternative ways to explain, and for a richer slate of problems. Hearing your advice on how to make this a better text will be the best sign that I have part way succeeded.

Paul Teller
Contents

A Modern Formal Logic Primer

Volume I

Sentence Logic

PREFACE TO VOLUMES I AND II: A Guide to the Primer ix

1. BASIC IDEAS AND TOOLS 1
   1–1. Logic As the Science of Argument 1
   1–2. Sentences and Connectives 4
   1–3. Truth Tables and the Meaning of ‘¬’, ‘&’, and ‘∨’ 7
   1–4. Truth Functions 9
   1–5. Compounding Compound Sentences 11
   1–6. Rules of Formation and Rules of Valuation 16

2. TRANSCRIPTION BETWEEN ENGLISH AND SENTENCE LOGIC 21
   2–1. Transcription vs. Translation 21
   2–2. Grouping in Transcription 22
   2–3. Adequacy of Transcriptions 24
1–1. LOGIC AS THE SCIENCE OF ARGUMENT

Adam is happy, or so I tell you. If you don’t believe me, I try to convince you with an argument: Adam just got an ‘A’ on his logic exam. Anyone who gets an ‘A’ on an exam is happy. So Adam is happy. A logician would represent such an argument in this way:

1  Premises  a) Adam just got an ‘A’ on his logic exam.
              b) Anyone who gets an ‘A’ on an exam is happy.
      Conclusion  c) Adam is happy.

We ordinarily think of an argument as an attempt to convince someone of a conclusion by offering what a logician calls premises, that is, reasons for believing the conclusion. But in order to study arguments very generally, we will characterize them by saying:

An Argument is a collection of declarative sentences one of which is called the conclusion and the rest of which are called the premises.

An argument may have just one premise, or it may have many. By declarative sentences, I mean those, such as ‘Adam is happy,’ or
Grass is green.', which we use to make statements. Declarative sentences contrast with questions, commands, and exclamations, such as 'Is Adam happy?', 'Cheer up, Adam!' and 'Boy, is Adam happy!' Throughout this text I will deal only with declarative sentences; though if you continue your study of logic you will encounter such interesting topics as the logic of questions and the logic of commands.

For an argument to have any interest, not just any premises and conclusion will do. In any argument worth its name, we must have some connection or relation between the premises and conclusion, which you can think of intuitively in this way:

Ordinarily, the premises of an argument are supposed to support, or give us reasons, for believing the conclusion.

A good way of thinking about logic, when you are beginning to learn, is to say that logic is the study of this reason-giving connection. I like to say, more generally, that logic is the science of arguments. Logic sets out the important properties of arguments, especially the ways in which arguments can be good or bad. Along the way, logicians also study many things that are not themselves arguments or properties of arguments. These are things which we need to understand in order to understand arguments clearly or things which the study of arguments suggests as related interesting questions.

In order to see our subject matter more clearly, we need to distinguish between inductive and deductive arguments. Argument (1) is an example of a deductive argument. Compare (1) with the following:

1. a) Adam has smiled a lot today.
   b) Adam has not frowned at all today.
   c) Adam has said many nice things to people today, and no unfriendly things.
   d) Adam is happy today.

   The difference between arguments (1) and (2) is this: In (1), without fail, if the premises are true, the conclusion will also be true. I mean this in the following sense: It is not possible for the premises to be true and the conclusion false. Of course, the premises may well be false. (1, for one, would suspect premise (b) of argument (1).) But in any possible situation in which the premises are true, the conclusion will also be true.

   In argument (2) the premises relate to the conclusion in a different way. If you believe the second argument's premises, you should take yourself to have at least some fairly good reasons for believing that the conclusion is true also. But, of course, the premises of (2) could be true and the conclusion nonetheless false. For example, the premises do not rule out the possibility that Adam is merely pretending to be happy.

Logicians mark this distinction with the following terminology:

Valid Deductive Argument: An argument in which, without fail, if the premises are true, the conclusion will also be true.

Good Inductive Argument: An argument in which the premises provide good reasons for believing the conclusion. In an inductive argument, the premises make the conclusion likely, but the conclusion might be false even if the premises are true.

What do we mean by calling an argument 'deductive' or 'inductive', without the qualifiers 'valid' or 'good'? Don't let anyone tell you that these terms have rigorous definitions. Rather,

We tend to call an argument 'Deductive' when we claim, or suggest, or just hope that it is deductively valid. And we tend to call an argument 'Inductive' when we want to acknowledge that it is not deductively valid but want its premises to aspire to making the conclusion likely.

In everyday life we don't use deductively valid arguments too often. Outside of certain technical studies, we intend most of our arguments as inductively good. In simple cases you understand inductive arguments clearly enough. But they can be a bear to evaluate. Even in the simple case of argument (2), if someone suggests that Adam is just faking happiness, your confidence in the argument may waver. How do you decide whether or not he is faking? The problem can become very difficult. In fact, there exists a great deal of practical wisdom about how to evaluate inductive arguments, but no one has been able to formulate an exact theory which tells us exactly when an inductive argument is really good.

In this respect, logicians understand deduction much better. Even in an introductory formal logic course, you can learn the rules which establish the deductive validity of a very wide and interesting class of arguments. And you can understand very precisely what this validity consists in and why the rules establish validity. To my mind, these facts provide the best reason for studying deductive logic: It is an interesting theory of a subject matter about which you can, in a few months, learn a great deal. Thus you will have the experience of finding out what it is like to understand a subject matter by learning a technical theory about that subject matter.

Studying formal logic also has other, more practical, attractions. Much of what you learn in this book will have direct application in mathematics, computer science, and philosophy. More generally, studying deductive logic can be an aid in clear thinking. The point is that, in order to make the nature of deductive validity very precise, we must learn a way of mak-
ing certain aspects of the content of sentences very precise. For this rea-
son, learning deductive logic can pay big dividends in improving your
clarity generally in arguing, speaking, writing, and thinking.

EXERCISES

1-1. Explain in your own words what an argument is. Give an ex-
ample of your own of an inductive argument and of a deductive
argument. Explain why your example of an inductive argument is
an inductive argument and why your example of a deductive argu-
ment is a deductive argument.

1-2. SENTENCES AND CONNECTIVES

I have said that arguments are composed of declarative sentences. Some
logicians prefer to say that arguments are composed of the things we say
with sentences, that is, statements or propositions. Sentences can be prob-
lematic in logic because sentences are often ambiguous. Consider this sen-
tence:

(3) I took my brother's picture yesterday.

I could use this sentence to mean that yesterday I made a photograph of
my brother. Or I could use the sentence to mean that I stole a picture
that belonged to my brother. Actually, this sentence can be used to say a
rather amazingly large number of different things.

Ambiguous sentences can make a problem for logic because they can be
ture in one way of understanding them and false in another. Because
logic has to do with the truth and falsity of premises and conclusions in
arguments, if it is not clear whether the component sentences are true or
false, we can get into some awful messes. Because logic won't be con-
cerned with all the details of the structure of a sentence. Consider,
for example, the sentence 'Adam loves Eve.' In sentence logic we won't
be concerned with the fact that this sentence has a subject and a predicate,
that it uses two proper names, and so on.

Indeed, the only fact about this sentence which is relevant to sentence
logic is whether it happens to be true or false. So let's ignore all the struc-
ture of the sentence and symbolize it in the simplest way possible, say, by
using the letter 'A'. (I put quotes around letters and sentences when I
talk about them as opposed to using them. If this use of quotes seems strange,
don't worry about it—you will easily get used to it.) In other words, for
the moment, we will let the letter 'A' stand for the sentence 'Adam loves
Eve.' When we do another example we will be free to use 'A' to stand for
a different English sentence. But as long as we are dealing with the same
example, we will use 'A' to stand for the same sentence.

Similarly, we can let other capital letters stand for other sentences. Here
is a transcription guide that we might use:

Transcription Guide

A: Adam loves Eve.
B: Adam is blond.
C: Eve is clever.

'A' is standing for 'Adam loves Eve.', 'B' is standing for 'Adam is blond.'
and 'C' is standing for 'Eve is clever.' In general, we will use capital letters
to stand for any sentences we want to consider where we have no interest
in the internal structure of the sentence. We call capital letters used in
this way Atomic Sentences, or Sentence Letters. The word ‘atomic’ is supposed to remind you that, from the point of view of sentence logic, these are the smallest pieces we need to consider. We will always take a sentence letter (and in general any of our sentences) to be true or false (but not both true and false!) and not to change from true to false or from false to true in the middle of a discussion.

Starting with atomic sentences, sentence logic builds up more complicated sentences, or Compound Sentences. For example, we might want to say that Adam does not love Eve. We say this with the Negation of ‘A’, also called the Denial of ‘A’. We could write this as ‘not A’. Instead of ‘not’, though, we will just use the negation sign, ‘~’. That is, the negation of ‘A’ will be written as ‘~A’, and will mean ‘not A’, that is, that ‘A’ is not true. The negation sign is an example of a Connective, that is, a symbol we use to build longer sentences from shorter parts.

We can also use the atomic sentences in our transcription guide to build up a compound sentence which says that Adam loves Eve and Adam is blond. We say this with the Conjunction of the sentence ‘A’ and the sentence ‘B’, which we write as ‘A&B’. ‘A’ and ‘B’ are called Conjuncts or Components of ‘A&B’, and the connective ‘&’ is called the Sign of Conjunction.

Finally, we can build a compound sentence from the sentence ‘A’ and the sentence ‘B’ which means that either Adam loves Eve or Adam is blond. We say this with the Disjunction of the sentence ‘A’ and the sentence ‘B’, which we write as ‘A v B’. ‘A’ and ‘B’ are called Disjuncts or Components of ‘A v B’, and the connective ‘v’ is called the Sign of Disjunction.

You might wonder why logicians use a ‘v’ to mean ‘or’. There is an interesting historical reason for this which is connected with saying more exactly what ‘or’ is supposed to mean. When I say, ‘Adam loves Eve or Adam is blond’, I might actually mean two quite different things. I might mean that Adam loves Eve, or Adam is blond, but not both. Or I might mean that Adam loves Eve, or Adam is blond, or possibly both.

If you don’t believe that English sentences with ‘or’ in them can be understood in these two very different ways, consider the following examples. If a parent says to a greedy child, ‘You can have some candy or you can have some cookies’, the parent clearly means some of one, some of the other, but not both. When the same parent says to an adult dinner guest, ‘We have plenty, would you like some more meat or some more potatoes?’ clearly he or she means to be offering some of either or both.

Again, we have a problem with ambiguity. We had better make up our minds how we are going to understand ‘or’, or we will get into trouble. In principle, we could make either choice, but traditionally logicians have always opted for the second, in which ‘or’ is understood to mean that the first sentence is true, or the second sentence is true, or possibly both sentences are true. This is called the Inclusive Sense of ‘or’. Latin, unlike English, was not ambiguous in this respect. In Latin, the word ‘vel’ very specifically meant the first or the second or possibly both. This is why logicians symbolize ‘or’ with ‘v’. It is short for the Latin ‘vel’, which means inclusive or. So when we write the disjunction ‘AvB’, we understand this to mean that ‘A’ is true, ‘B’ is true, or both are true.

To summarize this section:

Sentence logic symbolizes its shortest unambiguous sentences with Atomic Sentences, also called Sentence Letters, which are written with capital letters: ‘A’, ‘B’, ‘C’ and so on. We can use Connectives to build Compound Sentences out of shorter sentences. In this section we have met the connectives ‘~’ (the Negation Sign), ‘&’ (the Sign of Conjunction), and ‘v’ (the Sign of Disjunction).

EXERCISES

1–2. Transcribe the following sentences into sentence logic, using ‘G’ to transcribe ‘Pudding is good.’ and ‘F’ to transcribe ‘Pudding is fattening.’

   a) Pudding is good and pudding is fattening.
   b) Pudding is both good and fattening.
   c) Pudding is either good or not fattening.
   d) Pudding is not good and not fattening.

You may well have a problem with the following transcriptions, because to do some of them right you need to know something I haven’t told you yet. But please take a try before continuing. Trying for a few minutes will help you to understand the discussion of the problem and its solution in the next section. And perhaps you will figure out a way of solving the problem yourself!

   e) Pudding is neither good nor fattening.
   f) Pudding is both not good and not fattening.
   g) Pudding is neither not good or not fattening.
   h) Pudding is either not good or not fattening.

1–3. TRUTH TABLES AND THE MEANING OF ‘~’, ‘&’, AND ‘v’

We have said that ‘~A’ means not A, ‘A&B’ means A and B, and ‘AvB’ means A or B in the inclusive sense. This should give you a pretty good idea of what the connectives ‘~’, ‘&’, and ‘v’ mean. But logicians need to
be as exact as possible. So we need to specify how we should understand the connectives even more exactly. Moreover, the method which we will use to do this will prove very useful for all sorts of other things.

To get the idea, we start with the very easy case of the negation sign, \( \neg \). The sentence \( A \) is either true or it is false. If \( A \) is true, then \( \neg A \) is false. If \( A \) is false, then \( \neg A \) is true. And that is everything you need to know about the meaning of \( \neg \). We can say this more concisely with a table, called a **Truth Table**:

<table>
<thead>
<tr>
<th>A</th>
<th>( \neg A )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

The column under \( A \) lists all the possible cases involving the truth and falsity of \( A \). We do this by describing the cases in terms of what we call **Truth Values**. The case in which \( A \) is true is described by saying that \( A \) has the truth value \( t \). The case in which \( A \) is false is described by saying that \( A \) has the truth value \( f \). Because \( A \) can only be true or false, we have only these two cases. We explain how to understand \( \neg \) by saying what the truth value of \( \neg A \) is in each case. In case 1, \( \neg A \) has the truth value \( f \); that is, it is false. In case 2, \( \neg A \) has the truth value \( t \); that is, it is true. Although what we have done seems trivial in this simple case, you will see very soon that truth tables are extremely useful.

Let us see how to use truth tables to explain \( \& \). A conjunction has two atomic sentences, so we have four cases to consider:

<table>
<thead>
<tr>
<th>A</th>
<th>( \neg A )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

When \( A \) is true, \( B \) can be true or false. When \( A \) is false, again \( B \) can be true or false. The above truth table gives all possible combinations of truth values which \( A \) and \( B \) can have together.

We now specify how \( \& \) should be understood by specifying the truth value for each case for the compound \( A \& B \):

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>( A &amp; B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

In other words, \( A \& B \) is true when the conjuncts \( A \) and \( B \) are both true. \( A \& B \) is false in all other cases, that is, when one or both of the conjuncts are false.

A word about the order in which I have listed the cases. If you are curious, you might try to guess the recipe I used to order the cases. (If you try, also look at the more complicated example in section 1–5.) But I won't pause to explain, because all that is important about the order is that we don't leave any cases out and all of us list them in the same order, so that we can easily compare answers. So just list the cases as I do.

We follow the same method in specifying how to understand \( \vee \). The disjunction \( A \vee B \) is true when either or both of the disjunctions \( A \) and \( B \) are true. \( A \vee B \) is false only when \( A \) and \( B \) are both false:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>( A \vee B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

We have defined the connectives \( \neg \), \( \& \), and \( \vee \) using truth tables for the special case of sentence letters \( A \) and \( B \). But obviously nothing will change if we use some other pair of sentences, such as \( 'H' \) and \( 'D' \).

This section has focused on the truth table definitions of \( \neg \), \( \& \), and \( \vee \). But along the way I have introduced two auxiliary notions about which you need to be very clear. First, by a **Truth Value Assignment of Truth Values to Sentence Letters**, I mean, roughly, a line of a truth table, and a **Truth Table** is a list of all the possible truth values assignments for the sentence letters in a sentence:

A **Truth Assignment** to a collection of atomic sentence letters is a specification, for each of the sentence letters, whether the letter is (for this assignment) to be taken as true or as false. The word **Case** will also be used for 'assignment of truth values'.

A **Truth Table for a Sentence** is a specification of all possible truth values assignments to the sentence letters which occur in the sentence, and a specification of the truth value of the sentence for each of these assignments.

1–4. **TRUTH FUNCTIONS**

I want to point out one more thing about the way we have defined the connectives \( \neg \), \( \& \), and \( \vee \). Let us start with \( \neg \). What do you have to know in order to determine whether \( \neg A \) is true or false? You don't have to know what sentence \( A \) actually stands for. You don't have to know whether \( A \) is supposed to mean that Adam loves Eve, or that pudding is
fattening, or anything like that. To know whether \( \neg A \) is true or false, all you have to know is whether \( A \) itself is true or false. This is because if you know the truth value of \( A \), you can get the truth value of \( \neg A \) by just looking it up in the truth table definition of \( \neg A \).

The same thing goes for \( \& \) and \( \lor \). To know whether \( A \& B \) is true or false, you don’t have to know exactly what sentences \( A \) and \( B \) are supposed to be. All you need to know is the truth value of \( A \) and the truth value of \( B \). This is because, with these truth values, you can look up the truth value of \( A \& B \) with the truth table definition of \( \& \). Likewise, with truth values for \( A \) and for \( B \), you can look up the truth value for \( A \lor B \).

Logicians have a special word for these simple facts about \( \neg \), \( \& \), and \( \lor \). We say that these connectives are Truth Functionals. In other words (to use \( \& \) as an example), the truth value of the compound sentence \( A \& B \) is a function of the truth values of the components \( A \) and \( B \). In other words, if you put in truth values for \( A \) and for \( B \) as input, the truth table definition of \( \& \) gives you, as an output, the truth value for the compound \( A \& B \). In this way \( A \& B \) is a function in the same kind of way that \( x + y \) is a numerical function. If you put in specific numbers for \( x \) and \( y \), say, 5 and 7, you get a definite value for \( x + y \), namely, 12.

\( A \& B \) is just like that, except instead of number values 1, 2, 3, . . . which can be assigned to \( x \) and to \( y \), we have just two truth values, \( t \) and \( f \), which can be assigned to \( A \) and to \( B \). And instead of addition, we have some other way of combining the assigned values, a way which we gave in the truth table definition of \( \& \). Suppose, for example, that I give you the truth values \( t \) for \( A \) and \( f \) for \( B \). What, then, is the resulting truth value for \( A \& B \)? Referring to the truth table definition of \( A \& B \), you can read off the truth value \( f \) for \( A \& B \). The truth tables for \( \neg \) and \( \lor \) give other ways of combining truth values of components to get truth values for the compound. That is, \( \neg \) and \( \lor \) are different truth functions.

Let’s pull together these ideas about truth functions:

A Truth Function is a rule which, when you give it input truth values, gives you a definite output truth value. A Truth Functional Connective is a connective defined by a truth function. A Truth Functional Compound is a compound sentence formed with truth functional connectives.

**EXERCISES**

1-3. Try to explain what it would be for a declarative compound sentence in English not to be truth functional. Give an example of a declarative compound sentence in English that is not truth functional. (There are lots of them! You may find this exercise hard. Please try it, but don’t get alarmed if you have trouble.)

1-5. COMPOUNDING COMPOUND SENTENCES

We have seen how to apply the connectives \( \neg \), \( \& \), and \( \lor \) to atomic sentences such as \( A \) and \( B \) to get compound sentences such as \( \neg A \), \( A \& B \), and \( A \lor B \). But could we now do this over again? That is, could we apply the connectives not just to atomic sentences \( A \), \( B \), \( C \), etc., but also to the compound sentences \( \neg A \), \( A \& B \), and \( A \lor B \)? Yes, of course. For example, we can form the conjunction of \( \neg A \) with \( B \), giving us \( \neg A \& B \). Using our current transcription guide, this transcribes into ‘Adam does not love Eve and Adam is blond.’

As another example, we could start with the conjunction \( A \& B \) and take this sentence’s negation. But now we have a problem. (This is the problem you encountered in trying to work exercise 1-2, e-i.) If we try to write the negation of \( A \& B \) by putting a \( \neg \) in front of \( A \& B \), we get the sentence we had before. But the two sentences should not be the same! This might be a little confusing at first. Here is the problem: We are considering two ways of building up a complex sentence from shorter parts, resulting in two different complex sentences. In the first way, we take the negation of \( A \), that is, \( \neg A \), and conjoin this with \( B \). In the second way, we first conjoin \( A \) and \( B \) and then negate the whole. In English, the sentence ‘It is not the case both that Adam loves Eve and Adam is blond.’ is very different from the sentence ‘Adam does not love Eve and Adam is blond.’ (Can you prove this by giving circumstances in which one of these compound sentences is true and the other one is false?)

In order to solve this problem, we need some device in logic which does the work that ‘both’ does in English. (If you are not sure you yet understand what the problem is, read the solution I am about to give and then reread the last paragraph.) What we need to do is to make clear the order in which the connectives are applied. It makes a difference whether we first make a negation and then form a conjunction, or whether we first form the conjunction and then make a negation. We will indicate the order of operations by using parentheses, much as one does in algebra. Whenever we form a compound sentence we will surround it by parentheses. Then you will know that the connective inside the parentheses applies before the one outside the parentheses. Thus, when we form the negation of \( A \), we write the final result as \( \neg (A) \). We now take \( \neg (A) \) and conjoin it with \( B \), surrounding the final result with parentheses:

\( (\neg (A)) & B \)

This says, take the sentence ‘\( \neg A \)’ and conjoin it with ‘\( B \)’. To indicate that the final result is a complete sentence (in case we will use it in some still larger compound), we surround the final result in parentheses also. Note
how I have used a second style for the second pair of parentheses—square brackets—to make things easier to read.

Contrast (4) with

(5) \[\neg(A\&B)\]

which means that one is to conjoin 'A' with 'B' and then take the negation of the whole.

In the same kind of way we can compound disjunctions with conjunctions and conjunctions with disjunctions. For example, consider

(6) \[\{(A\&B)vC\}\]

(7) \[\{(A&(BvC))\}\]

Sentence (6) says that we are first to form the conjunctions of 'A' with 'B' and then form the disjunction with 'C'. (7), on the other hand, says that we are first to form the disjunction of 'B' with 'C' and then conjoin the whole with 'A'. These are very different sentences. Transcribed into English, they are 'Adam both loves Eve and is blond, or Eve is clever,' and 'Adam loves Eve, and either Adam is blond or Eve is clever.'

We show more clearly that (6) and (7) are different sentences by writing out truth tables for them. We now have three atomic sentences, 'A', 'B', and 'C'. Each can be true or false, whatever the others are, so that we now have eight possible cases. For each case we work out the truth value of a larger compound from the truth value of the parts, using the truth value of the intermediate compound when figuring the truth value of a compound of a compound:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>g</th>
<th>h</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>2</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>3</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>5</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>6</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>7</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>8</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

Let's go over how we got this truth table. Columns a, b, and c simply give all possible truth value assignments to the three sentence letters 'A', 'B', and 'C'. As before, in principle, the order of the cases does not matter. But to make it easy to compare answers, you should always list the eight possible cases for three letters in the order I have just used. Then, for each case, we need to calculate the truth value of the compounds in columns d through h from the truth values given in columns a, b, and c.

Let us see how this works for case 5, for example. We first need to determine the truth value to put in column d, for '(A&B)' from the truth values given for this case. In case 5 'A' is false and 'B' is true. From the truth table definition of '&', we know that a conjunction (here, 'A&B') is false when the first conjunct (here, 'A') is false and the second conjunct (here, 'B') is true. So we write an 'f' for case 5 in column d. Column e is the disjunction of 'B' with 'C'. In case 5 'B' is true and 'C' is true. When we disjoin something true with something true, we get a true sentence. So we write the letter 't', standing for the truth value t, in column e for case 5.

Moving on to column g, we are looking at the disjunction of '(A&B)' with 'C'. We have already calculated the truth value of '(A&B)' for case 5—that was column d—and the truth value of 'C' for case 5 is given in column c. Reading off columns c and d, we see that '(A&B)' is false and 'C' is true in case 5. The sentence of column g, '[(A&B)vC]', is the disjunction of these two components and we know that the disjunction of something false with something true is, again, true. So we put a 't' in column g for case 5. Following the same procedure for column h, we see that for case 5 we have a disjunction of something false with something true, which gives the truth value f. So we write 'f' for case 5 in column h.

Go through all eight cases and check that you understand how to determine the truth values for columns d through h on the basis of what you are given in columns a, b, and c.

Now, back to the point that got us started on this example. I wanted to prove that the sentences '[(A&B)vC]' and '[A&(BvC)]' are importantly different. Note that in cases 5 and 7 they have different truth values. That is, there are two assignments of truth values to the components for which one of these sentences is true and the other is false. So we had better not confuse these two sentences. You see, we really do need the parentheses to distinguish between them.

Actually, we don't need all the parentheses I have been using. We can make two conventions which will eliminate the need for some of the parentheses without any danger of confusing different sentences. First, we can eliminate the outermost parentheses, as long as we put them back in if we decide to use a sentence as a component in a larger sentence. For example, we can write 'A&B' instead of '(A&B)' as long as we put the parentheses back around 'A&B' before taking the negation of the whole to form '~(A&B)'. Second, we can agree to understand '~' always to apply to the shortest full sentence which follows it. This eliminates the need to surround a negated sentence with parentheses before using it in a larger sentence. For example, we will write '~A&B' instead of '(~A)&B'. We know that '~A&B' means '(~A)&B' and not '~(A&B)' because the '~' in
'~A&B' applies to the shortest full sentence which follows it, which is 'A' and not 'A&B'.

This section still needs to clarify one more aspect of dealing with compound sentences. Suppose that, before you saw the last truth table, I had handed you the sentence '(A&B)vC' and asked you to figure out its truth value in each line of a truth table. How would you know what parts to look at? Here's the way to think about this problem. For some line of a truth table (think of line 5, for example), you want to know the truth value of '(A&B)vC'. You could do this if you knew the truth values of 'A&B' and of 'C'. With their truth values you could apply the truth table definition of 'v' to get the truth value of '(A&B)vC'. This is because '(A&B)vC' just is the disjunction of 'A&B' with 'C'. Thus you know that '(A&B)vC' is true if at least one of its disjuncts, that is, either 'A&B' or 'C', is true; and '(A&B)vC' is false only if both its disjuncts, 'A&B' and 'C', are false.

And how are you supposed to know the truth values of 'A&B' and of 'C'? Since you are figuring out truth values of sentences in the line of a truth table, all you need do to figure out the truth value of 'C' on that line is to look it up under the 'C' column. Thus, if we are working line 5, we look under the 'C' column for line 5 and read that in this case 'C' has the truth value t. Figuring out the truth value of 'A&B' for this line is almost as easy. 'A&B' is, by the truth table definition of conjunction, true just in case both conjuncts (here, 'A' and 'B') are true. In line 5 'A' is false and 'B' is true. So for this line, 'A&B' is false. Now that we finally have the truth values for the parts of '(A&B)vC', that is, for 'A&B' and for 'C', we can plug these truth values into the truth table definition for 'v' and get the truth value t for '(A&B)vC'.

Now suppose that you have to do the same thing for a more complicated sentence, say

\[ \neg[(Av-C)&[Bv(-A&C)]] \]

Don't panic. The principle is the same as for the last, simpler example. You can determine the truth value of the whole if you know the truth value of the parts. And you can determine the truth value of the parts if you can determine the truth value of their parts. You continue this way until you get down to atomic sentence letters. The truth value of the atomic sentence letters will be given to you by the line of the truth table. With them you can start working your way back up.

You can get a better grip on this process with the idea of the Main Connective of a sentence. Look at sentence (8) and ask yourself, "What is the last step I must take in building this sentence up from its parts?" In the case of (8) the last step consists in taking the sentence '[Av¬C]&[Bv(¬A&C)]' and applying '¬' to it. Thus (8) is a negation, '¬'.

is the main connective of (8), and '[Av¬C]&[Bv(¬A&C)]' is the component used in forming (8).

What, in turn, is the main connective of '[Av¬C]&[Bv(¬A&C)]'? Again, what is the last step you must take in building this sentence up from its parts? In this case you must take 'Av¬C' and conjoin it with 'Bv(¬A&C)'. Thus this sentence is a conjunction, '&', is its main connective, and its components are the two conjuncts 'Av¬C' and 'Bv(¬A&C)'. In like manner, 'Bv(¬A&C)' is a disjunction, with 'v' its main connective, and its components are the disjuncts 'B' and '¬A&C'. To summarize,

The Main Connective in a compound sentence is the connective which was used last in building up the sentence from its component or components.

Now, when you need to evaluate the truth value of a complex sentence, given truth values for the atomic sentence letters, you know how to proceed. Analyze the complete sentence into its components by identifying main connectives. Write out the components, in order of increasing complexity, so that you can see plainly how the larger sentences are built up from the parts.

In the case of (8), we would lay out the parts like this:

\[ A, B, C, \neg A, \neg C, Av¬C, \neg A&C, Bv(¬A&C), [Av¬C]&[Bv(¬A&C)] \]

You will be given the truth values of the atomic sentence letters, either by me in the problem which I set for you or simply by the line of the truth table which you are working. Starting with the truth values of the atomic sentence letters, apply the truth table definitions of the connectives to evaluate the truth values of the successively larger parts.

<table>
<thead>
<tr>
<th>EXERCISES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1–4. For each of the following sentences, state whether its main connective is '¬', '&amp;', or 'v' and list each sentence's components. Then do the same for the components you have listed until you get down to atomic sentence letters. So you can see how you should present your answers, I have done the first problem for you.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Main Sentence</th>
<th>Connective</th>
<th>Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A&amp;¬B)</td>
<td>¬</td>
<td>Av¬B</td>
</tr>
<tr>
<td>Av¬B</td>
<td>v</td>
<td>A, ¬B</td>
</tr>
<tr>
<td>¬B</td>
<td>¬</td>
<td>B</td>
</tr>
</tbody>
</table>
always invent new ones, for example, by using subscripts, as in $\neg(A \lor B)$.

If we run out of letters, we can only apply if somehow we have been given a truth value assignment to the atomic sentence letters. That is, if we have been given truth values for the ultimate constituent atomic sentence letters, then, using the rules of valuation, we can always calculate the truth value of a compound sentence, no matter how complex. Once again, this is what we mean when we say that the connectives are truth functional.

How does one determine the truth value of atomic sentences? That’s not a job for logicians. If we really want to know, we will have to find out the actual truth values of the atomic constituents once they are given to us. And when we do truth tables, we don’t have to worry about the actual truth values of the atomic sentence letters. In truth tables, like those in the following exercises, we consider all possible combinations of truth values which the sentence letters could have.

The truth table definitions of the connectives give a graphic summary of these rules of valuation. I’m going to restate those truth table definitions here because, if truth be told, I didn’t state them quite right. I gave them only for sentence letters, ‘A’ and ‘B’. I did this because, at that point in the exposition, you had not yet heard about long compound sentences, and I didn’t want to muddy the waters by introducing too many new things at once. But now that you are used to the idea of compound sentences, I can state the truth table definitions of the connectives with complete generality.

Suppose that $X$ and $Y$ are any two sentences. They might be atomic sentence letters, or they might themselves be very complex compound sentences. Then we specify that:

<table>
<thead>
<tr>
<th>( X )</th>
<th>( \neg X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

Note that these rules apply to any compound sentence. However, they only apply if somehow we have been given a truth value assignment to the atomic sentence letters. That is, if we have been given truth values for the ultimate constituent atomic sentence letters, then, using the rules of valuation, we can always calculate the truth value of a compound sentence, no matter how complex. Once again, this is what we mean when we say that the connectives are truth functional.

As I explained earlier, we agree to cheat on these strict rules in two ways (and in these two ways only!). We omit the outermost parentheses, and we omit parentheses around a negated sentence even when it is not the outermost sentence, because we agree to understand ‘\~’ always to apply to the shortest full sentence which follows it.

I should also clarify something about formation rule i). In principle, sentence logic can use as many atomic sentences as you like. It is not limited to the 26 letters of the alphabet. If we run out of letters, we can always invent new ones, for example, by using subscripts, as in ‘$A_1$’ and ‘$C_{37}$’. In practice, of course, we will never need to do this.
The difference between my earlier, restricted truth table definitions and these new general definitions might seem a bit nitpicky. But the difference is important. You probably understood the intended generality of my first statement of the truth table definitions. However, a computer, for example, would have been totally confused. Logicians strive, among other things, to give very exact statements of everything. They enjoy exactness for its own sake. But exactness has practical value too, for example, when one needs to write a program that a computer can understand.

This section has also illustrated another thing worth pointing out. When I talked about sentences generally, that is, when I wanted to say something about any sentences, \(X\) and \(Y\), I used boldface capital letters from the end of the alphabet. I'm going to be doing this throughout the text. But rather than dwell on the point now, you will probably best learn how this usage works by reading on and seeing it illustrated in practice.

**EXERCISES**

1-5. Which of the following expressions are sentences of sentence logic and which are not?

- a) \(A \& \sim B\)
- b) \(A \sim \& B\)
- c) \(Gv(\sim B \& \sim H)\)
- d) \(A \& (G \sim (DvH))\)
- e) \((A \& B)v(G \& D)\)
- f) \((A \& B) \& CvD\)

1-6. Construct a complete truth table for each of the following sentences. The first one is done for you:

<table>
<thead>
<tr>
<th>(X)</th>
<th>(Y)</th>
<th>(X &amp; Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(A)</th>
<th>(B)</th>
<th>(\sim B)</th>
<th>(\sim BvA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

a) \(\sim BvA\)

b) \(\sim (BvA)\)

c) \(((QvT) \& \sim (Qv \sim T))\)

d) \((D \& \sim G)v(G \& D)\)

e) \(Av(\sim BvC)\)

f) \(Kv(\sim P \& (\sim PvM))\)

g) \(((Dv \sim B) \& (Dv \sim B)) \& (DvB)\)

h) \(L \& (Mv(\sim N \& (\sim MvL)))\)

1-7. Philosopher's problem: Why do I use quotation marks around sentences, writing things like

'\(B\)'

and

'\(\sim (CvA)\)'

but no quotation marks about boldface capital letters, writing

\(X, Y, XvY,\) etc.

when I want to talk about sentences generally?

**CHAPTER SUMMARY EXERCISE**

The following list gives you the important terms which have been introduced in this chapter. Make sure you understand all of them by writing a short explanation of each. Please refer back to the text to make sure, in each case, that you have correctly explained the term. Keep your explanation of these terms in your notebook for reference and review later on in the course.

- a) Argument
- b) Valid Deductive Argument
c) Good Inductive Argument
d) Deductive Argument
e) Inductive Argument
f) Atomic Sentence (also called 'Sentence Letter' or 'Atomic Sentence Letter')
g) Compound Sentence
h) Connective
i) Component
j) ~ (called the 'Negation Sign' or 'Sign of Denial')
k) Negation
l) & (called the 'Sign of Conjunction')
m) Conjunction
n) Conjunct
o) v (called the 'Sign of Disjunction')
p) Disjunction
q) Disjunct
r) Inclusive Or
s) Exclusive Or
t) Truth Value
u) Truth Table
v) Truth Table Definition
w) Assignment of Truth Values
x) Case
y) Truth Function
z) Truth Functional Connective
aa) Truth Functional Compound
bb) Main Connective
2-1. TRANSCRIPTION VS. TRANSLATION

As we saw in chapter 1, for many English sentences we can find corresponding sentences of sentence logic. For example, if 'A' stands for the sentence 'Adam loves Eve.' and 'B' for the sentence 'Adam is blond.' then 'Bv¬A' corresponds to 'Either Adam is blond or he does not love Eve.'

Many logicians use the word 'translation' to describe the relation between a sentence of English and a corresponding sentence of logic. I think that 'translation' is the wrong word to use. If a first sentence translates a second, the two sentences are supposed to have exactly the same meaning. But the correspondence between English and logic is often looser than having the same meaning, as the next examples show.

Consider the sentence 'Adam loves Eve, but he left her.' This English sentence is a compound of two shorter sentences, 'Adam loves Eve.', which we will transcribe with the sentence letter 'A', and 'He left her.' (that is, Adam left Eve), which we will transcribe with the sentence letter 'L'. These two sentences have been connected in English with the word 'but'. So we can get a partial transcription into logic by writing 'A but L'. We are still not finished, however, because 'but' is a word of English, not logic. What in logic corresponds to 'but'?

If I assert the sentence 'Adam loves Eve but he left her.', what am I
telling you? Well, first of all, I assert that Adam loves Eve. In asserting the original sentence, I am also telling you that Adam left Eve. In other words, so far, I seem to be saying: 'Adam loves Eve and he left her.'

What's the difference between 'Adam loves Eve but he left her.' and Adam loves Eve and he left her.'? That is, between 'A but L' and 'A&L'? Not much. In English, we tend to use the word 'but' when we want to assert two things (a conjunction), but the first thing asserted may well lead one to expect the opposite of the second thing asserted. 'But' functions much as do the words '. . . and, contrary to what I just said would lead you to expect. . .': 'Adam loves Eve, and, contrary to what I just said would lead you to expect, he left her.'

Logic has no way of expressing the idea of 'contrary to what the first conjunct would lead you to expect.' So we simply transcribe 'but' as '&'. In sentence logic we can't improve upon 'A&L' as a transcription of 'Adam loves Eve but he left her.' Several other English words function very much like 'but', and should likewise get transcribed as '&': 'however', 'nevertheless', 'although', and 'despite (the fact that)'.

Perhaps you are starting to see why I want to talk about transcribing, instead of translating, English into logic. 'A&L' isn't a very good translation of 'Adam loves Eve but he left her.' If it were a good translation, we would have to say that 'and' means the same thing as 'but'. However, '(&)' is the closest we have to 'but' in logic, so that's what we use.

A Transcription of an English sentence into sentence logic is a sentence of sentence logic which expresses, as closely as possible, what the English sentence expresses.

Logicians sometimes use the words 'paraphrasing' or 'symbolizing' for what I am calling 'transcribing' English sentences in logic.

2-2. GROUPING IN TRANSCRIPTION

Here is another problem which comes up in transcribing English into logic. Consider the sentence.

(1) Eve is clever and Eve is dark-eyed or Adam is blond.

How do we transcribe this? Should we understand (1) as

(1a) (Eve is clever and Eve is dark-eyed) or Adam is blond.

Or should we understand it as

(1b) Eve is clever and (Eve is dark-eyed or Adam is blond).

As we know from section 1-5, the grouping makes a difference. The problem here is that (1) is bad English. In English we should also indicate the grouping, which we can easily do with a comma. Thus (1a) corresponds to

(1c) Eve is clever and Eve is dark-eyed, or Adam is blond.

and (1b) corresponds to

(1d) Eve is clever, and Eve is dark-eyed or Adam is blond.

Using the following transcription guide

B: Adam is blond.
C: Eve is clever.
D: Eve is dark-eyed.

we get as transcriptions of (1a) and (1c):

(1e) (C&D)vB

And as transcriptions of (1b) and (1d):

(1f) C&(DvB)

Notice that by using parentheses in (1a) and (1b) I have used a mixture of English and sentence logic as an aid to figuring out what seems to be going on. Such mixtures often help in transcription. If you don't see a correct transcription right away, transcribe part, or features of, the English sentence. Then go to work on the parts which you did not transcribe in your first pass at the problem.

The expression 'Either . . . or . . . ' functions in English to indicate grouping in some respects as do parentheses in logic. Anything that goes where you see the '. . . ' acts as if it had parentheses around it, even if it is quite complex. (Often something which goes where you see the '. . . ' also acts like it had parentheses around it, but this English device does not always work.) Thus we could write (1a) and (1c) as

(1g) Either Eve is clever and Eve is dark-eyed, or Adam is blond.

'Both . . . and . . . ' serves much as does 'Either . . . or . . . ', although the complexities of English grammar don't let you say things such as 'Both Eve is clever and Eve is dark-eyed, or Adam is blond.' To speak grammatical English, one has to say

(1h) Eve is both clever and dark-eyed, or Adam is blond.

which we clearly transcribe as (1e).
Notice that in (1h) we have done some collapsing of English sentence units. When transcribing into logic, you should rewrite 'Eve is clever and dark-eyed.' as a conjunction of two atomic sentences, that is, as 'Eve is clever and Eve is dark-eyed.' or finally as 'C&D'. And, to consider a new example, you should rewrite 'Eve is clever or dark-eyed.' as a disjunction of two atomic sentences, that is, as 'Eve is clever or Eve is dark-eyed.', or finally as 'CvD'.

2-3. Adequacy of Transcriptions

It's your turn to figure out an example. Before reading on, try transcribing

(2) Adam is neither ugly nor dumb.

What did you get? 'Neither' suggests a negation, and 'nor' suggests a disjunction. But (2) is tricky. If we use 'U' for 'Adam is ugly.' and 'D' for 'Adam is dumb.' then (2) is not true. Suppose you proposed (2) as a transcription of (2). Transcribe (2) back into English, and the English sentence false, or the transcription false and the English sentence true? If so, reject the proposed transcription. If there is no such case, the transcription is as good as it can get. Of course, in applying this test you will have to do the best you can to determine whether or not, for a case described in terms of truth values assigned to sentence letters, your English sentence is true. The structure of English is complicated, so there are no simple rules for determining the truth value of arbitrary English sentences. Nonetheless, this test can often help you to decide whether a proposed transcription is adequate.

We summarize the test by saying:

Second Transcription Test: Given a sentence of sentence logic as a proposed transcription of an English sentence, try to imagine a case, described in terms of an assignment of truth values to sentence letters (a case) which makes the proposed transcription true and the English sentence false, or the transcription false and the English sentence true? If so, reject the proposed transcription. If there is no such case, the transcription is as good as it can get. Of course, in applying this test you will have to do the best you can to determine whether or not, for a case described in terms of truth values assigned to sentence letters, your English sentence is true. The structure of English is complicated, so there are no simple rules for determining the truth value of arbitrary English sentences. Nonetheless, this test can often help you to decide whether a proposed transcription is adequate.

We will say that two such sentences are logically equivalent, a notion which I won't dwell on now because it provides the subject of the next chapter. But even this quick description of logical equivalence will help
you pull together the ideas of the last few paragraphs. At least so far as sentence logic goes, two sentences say the same thing if and only if they are logically equivalent. With this way of understanding "saying the same thing," our two tests for adequacy of transcription ultimately do the same work. For if "saying the same thing" just means "being true in exactly the same cases," two sentences say the same thing (our first test for an adequate transcription) if and only if they are true in the same cases (our second test for an adequate transcription).

Chapter 3 will clarify your understanding of logical equivalence. For the moment, however, you will be served by an intuitive understanding of a summary of this section:

If two sentence logic sentences are logically equivalent to each other, they provide equally good transcriptions of a given English sentence.

**EXERCISES**

2-1. Consider the sentence

(2*) Adam is not both ugly and dumb.

Carry out a study of its transcription into sentence logic which is similar to the study of (2). In particular, show that this sentence has two logically equivalent, and so equally accurate, transcriptions, both of which need carefully to be distinguished from a somewhat similar, but inadequate, transcription. If you have trouble with this exercise, spend a minute guessing at a transcription of (2*). Write down your guess and then reread the discussion of the transcription of (2).

2-2. Using this transcription guide, transcribe the following sentences into sentences of sentence logic.

A: Adam loves Eve.
B: Adam is blond.
C: Eve is clever.
D: Eve is dark-eyed.
E: Eve loves Adam.

a) Eve is clever or Eve is dark-eyed.
b) Eve is clever or dark-eyed.
c) Eve is clever and dark-eyed.
d) Eve is clever but not dark-eyed.
e) Eve either is not clever or she is not dark-eyed.
f) Eve is either not clever or not dark-eyed.

g) Eve is dark-eyed and Adam loves her.
h) Either Adam is blond and loves Eve, or he is not blond and Eve loves him.
i) Eve is both not dark-eyed and either clever or in love with Adam.
j) Eve is dark-eyed, but Adam does not love her.
k) Adam is either blond or in love with Eve; nevertheless, Eve does not love him.
l) Although either Eve is dark-eyed or Adam is blond, Adam does not love Eve.
m) Despite Eve's being clever and not loving Adam, Adam does love Eve.
n) Adam loves Eve even though she is not dark-eyed.
o) Adam not only loves Eve, Eve also loves Adam.
p) Even though Eve is either clever or not dark-eyed, either Adam is blond or in love with Eve.
q) Eve is both in love with Adam and not dark-eyed, despite Adam's being either blond or not in love with Eve.
r) Adam does not love Eve. Also, Adam is blond, and Eve is either clever or in love with Adam.
s) Adam is either in love with Eve or not.
t) Adam is either in love with Eve or not. However, although she is clever, Eve is either dark-eyed or in love with Adam.
u) Either Adam is blond, or it is both the case that Eve loves Adam and is either dark-eyed or clever.
v) Either it is the case that both Adam is blond or not in love with Eve and Eve is dark-eyed or in love with Adam, or it is the case that both Adam does love Eve or is not blond and Eve is clever but not dark-eyed.

2-3. Using the same transcription guide as in exercise 2-2, transcribe the following into English:

a) Bv~B
b) A&~B
c) ~(AvC)
d) Bv(D&~C)
e) (Ev~C)&(~BvA)
f) [(AvE)&(~Cv(C&~D))]

g) ![If(¬BvA)&Dv(¬E&vB))&C (This is almost impossible to transcribe into English, but do the best you can. I'm giving this problem not to give you a bad time but to illustrate how logic has certain capacities to state things exactly, no matter how complex they are, while English, in practice, breaks down.)

2-4. Make up your own transcription guide and transcribe the following sentences into sentence logic. Your transcriptions should be as detailed as possible. For example, transcribe 'Roses are red and violets are blue,' not with one sentence letter but with two sentence
letters conjoined, like this: 'R&B' (R: Roses are red, B: Violets are blue).

a) Roses are red or Teller will eat his hat.
b) Monty Python is funny but Robert Redford is not.
c) Chicago is not bigger than New York even though New York is not the largest city.
d) Either I will finish this logic course or I will die trying.
e) W. C. Fields was not both handsome and smart.
f) Uncle Scrooge was neither generous nor understanding.
g) Although Minnesota Fats tried to diet, he was very overweight.
h) Peter likes pickles and ice cream, but he does not like to eat them together.
i) Roses are red and violets are blue. Transcribing this jingle is not hard to do.
j) Columbus sailed the ocean blue in 1491 or 1492, but in any case he discovered neither the South nor the North Pole.
k) Either Luke will catch up with Darth Vader and put an end to him or Darth Vader will get away and cause more trouble. But eventually the Empire will be destroyed.

CHAPTER SUMMARY EXERCISE

Give brief explanations of the following terms introduced in this chapter. Again, please refer to the text to make sure you have the ideas right.

a) Transcription
b) Adequate Transcription

Also, give a brief description of how English marks the grouping of sentences, that is, describe how English accomplishes the work done in logic by parentheses.
3-1. LOGICAL EQUIVALENCE

I introduced logic as the science of arguments. But before turning to arguments, we need to extend and practice our understanding of logic's basic tools as I introduced them in chapter 1. For starters, let's look at the truth table for 'A', '~A', and the negation of the negation of 'A', namely, '~~A'.

<table>
<thead>
<tr>
<th>A</th>
<th>~A</th>
<th>~~A</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td></td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

This truth table exhibits the special situation which I mentioned at the end of the last chapter: The truth value of '~~A' is always the same as that of 'A'. Logicians say that 'A' and '~~A' are Logically Equivalent.

As we will see in a moment, much more complicated sentences can be logically equivalent to each other. To get clear on what this means, let us review some of the things that truth tables do for us. Suppose we are looking at a compound sentence, perhaps a very complicated one which uses many sentence letters. When we write out the truth table for such a
sentence, we write out all the possible cases, that is, all the possible assignments of truth values to sentence letters in all possible combinations. In each one of these possible cases our original sentence has the truth value $t$ or the truth value $f$.

Now suppose that we look at a second sentence which uses the same sentence letters, or perhaps only some of the sentence letters that the first sentence uses, and no new sentence letters. We say that the two sentences are logically equivalent if in each possible case, that is, for each line of the truth table, they have the same truth value.

Two sentences of sentence logic are *Logically Equivalent* if and only if in each possible case (for each assignment of truth values to sentence letters) the two sentences have the same truth value.

What we said about the double negation of 'A' naturally holds quite generally:

The Law of Double Negation (DN): For any sentence $X$, $X$ and $\neg\neg X$ are logically equivalent.

Here are two more laws of logical equivalence:

*De Morgan's Laws* (DM): For any sentences $X$ and $Y$, $\neg(X\&Y)$ is logically equivalent to $\neg X \lor \neg Y$. And $\neg(X\lor Y)$ is logically equivalent to $\neg X \land \neg Y$.

Thus 'Adam is not both ugly and dumb.' is logically equivalent to 'Either Adam is not ugly or Adam is not dumb.' And 'Adam is not either ugly or dumb.' is logically equivalent to 'Adam is not ugly and Adam is not dumb.' You should check these laws with truth tables. But I also want to show you a second, informal way of checking them which allows you to "see" the laws. This method uses something called *Venn Diagrams*.

A Venn diagram begins with a box. You are to think of each point inside the box as a possible case in which a sentence might be true or false. That is, think of each point as an assignment of truth values to sentence letters, or as a line of a truth table. Next we draw a circle in the box and label it with the letter 'X', which is supposed to stand for some arbitrary sentence, atomic or compound. The idea is that each point inside the circle represents a possible case in which $X$ is true, and each point outside the circle represents a possible case in which $X$ is false.

Look at Figure 3-1. What area represents the sentence $\neg X$? The area outside the circle, because these are the possible cases in which $X$ is false.

Now let's consider how Venn diagrams work for compound sentences built up from two components, $X$ and $Y$. Depending on what the sentences $X$ and $Y$ happen to be, both of them might be true, neither might be true, or either one but not the other might be true. Not to omit any of these eventualities, we must draw the circles representing $X$ and $Y$ as overlapping, as in Figure 3-2 and 3-3.

The conjunction $X\&Y$ is true in just those cases represented by points that lie inside both the $X$ and $Y$ circles, that is, the shaded area in Figure 3-2. The disjunction $X \lor Y$ is true in just those cases represented by points that lie inside either the $X$ or the $Y$ circle (or both), that is, the shaded area in Figure 3-3.

Now we can use Venn diagrams to check De Morgan's laws. Consider first a negated conjunction. Look for the area, shown in Figure 3-4, which represents $\neg(X \land Y)$. This is just the area outside the shaded lens in Figure 3-2.

Let us compare this with the area which represents $\neg X \lor \neg Y$. We draw overlapping $X$ and $Y$ circles. Then we take the area outside the first circle (which represents $\neg X$; see Figure 3-5), and we take the area outside the second (which represents $\neg Y$; see Figure 3-6). Finally, we put these two areas together to get the area representing the disjunction $\neg X \lor \neg Y$, as represented in Figure 3-7.

Notice that the shaded area of Figure 3-7, representing $\neg X \lor \neg Y$, is the same as that of Figure 3-4, representing $\neg(X \land Y)$. The fact that the same
32 Logical Equivalence, Logical Truths, and Contradictions

3-2. Substitution of Logical Equivalents and Some More Laws

The shaded area represents both \( \sim X \lor \sim Y \) and \( \sim (X \land Y) \) means that the two sentences are true in exactly the same cases and false in the same cases. In other words, they always have the same truth value. And that is just what we mean by two sentences being logically equivalent.

Now try to prove the other of De Morgan's laws for yourself using Venn diagrams.

Here are two more laws of logical equivalence:

The Distributive Laws: For any three sentences, \( X \), \( Y \), and \( Z \), \( X \land (Y \lor Z) \) is logically equivalent to \( (X \land Y) \lor (X \land Z) \). And \( X \lor (Y \land Z) \) is logically equivalent to \( (X \lor Y) \land (X \lor Z) \).

For example, 'Adam is both bold and either clever or lucky.' comes to the same thing as 'Adam is either both bold and clever or both bold and lucky.' You should prove these two laws for yourself using Venn diagrams. To do so, you will need a diagram with three circles, one each representing \( X \), \( Y \), and \( Z \). Again, to make sure that you omit no possible combination of truth values, you must draw these so that they all overlap, as in Figure 3-8.

Fill in the areas to represent \( Y \lor Z \), and then indicate the area which represents the conjunction of this with \( X \). In a separate diagram, first fill in the areas representing \( X \land Y \) and \( X \land Z \), and then find the area corresponding to the disjunction of these. If the areas agree, you will have demonstrated logical equivalence. Do the second of the distributive laws similarly. Also, if you feel you need more practice with truth tables, prove these laws using truth tables.

**EXERCISES**

3-1. Prove the second of De Morgan's laws and the two distributive laws using Venn diagrams. Do this in the same way that I proved the first of De Morgan's laws in the text, by drawing a Venn diagram for each proof, labeling the circles in the diagram, and explaining in a few sentences how the alternate ways of getting the final area give the same result. Use more than one diagram if you find that helpful in explaining your proof.

3-2. SUBSTITUTION OF LOGICAL EQUIVALENTS AND SOME MORE LAWS

We can't do much with our laws of logical equivalence without using a very simple fact, which our next example illustrates. Consider

\[ (1) \quad \sim \sim A \lor B. \]

'\( \sim \sim A \)' is logically equivalent to 'A'. This makes us think that (1) is logically equivalent to

\[ (2) \quad A \lor B. \]

This is right. But it is important to understand why this is right. A compound sentence is made up of component sentences, which in turn may be made up of further component sentences. How do subsentences (components, or components of components, or the like) affect the truth value of the original sentence? Only through their truth values. The only way that a subsentence has any effect on the truth values of a larger sentence is through the subsentence's truth value. (This, again, is just what we mean by saying that compound sentences are truth functions.) But if only the truth values matter, then substituting another sentence which always has the same truth value as the first can't make any difference.
I'll say it again in different words: Suppose that X is a subsentence of some larger sentence. Suppose that Y is logically equivalent to X, which means that Y and X always have the same truth value. X affects the truth value of the larger sentence only through its (i.e., X's) truth value. So, if we substitute Y for X, there will be no change in the larger sentence's truth value.

But this last fact is just what we need to show our general point about logical equivalence. The larger sentence will have the same truth value before and after the substitution; that is, the two versions of the larger sentence will be logically equivalent:

The Law of Substitution of Logical Equivalents (SLE): Suppose that X and Y are logically equivalent, and suppose that X occurs as a subsentence of some larger sentence Z. Let Z* be the new sentence obtained by substituting Y for X in Z. Then Z is logically equivalent to Z*.

Let's apply these laws to an example. Starting with the sentence

\[ \neg((\neg A \land B) \land (\neg A \land B)) \]

we can apply one of De Morgan's laws. This sentence is the negation of a conjunction, with the conjuncts \( \neg A \land B \) and \( \neg A \land B \). De Morgan's law tells us that this first line is logically equivalent to the disjunction of the negation of the two original conjuncts:

\[ \neg(\neg A \land B) \lor \neg(\neg A \land B) \]  
DM

(The 'DM' on the right means that this line was obtained from the previous line by applying one of De Morgan's laws.)

Did you have trouble understanding that one of De Morgan's laws applies to the sentence? If so, try using the idea of the main connective introduced in chapter 1. Ask yourself: "In building this sentence up from its parts, what is the last thing I do?" You apply the negation sign to \( (\neg A \land B) \land (\neg A \land B) \). So you know the original sentence is a negation. Next, ask yourself, what is the last thing I do in building \( (\neg A \land B) \land (\neg A \land B) \) up from its parts? Conjoin \( \neg A \land B \) with \( \neg A \land B \). So \( (\neg A \land B) \land (\neg A \land B) \) is a conjunction. The original sentence, then, is the negation of a conjunction, that is, a sentence of the form \( \neg(X \land Y) \), where, in our example, X is the sentence \( \neg A \land B \) and Y is the sentence \( \neg A \land B \). Applying De Morgan's law to \( \neg(X \land Y) \) gives \( \neg X \lor \neg Y \); in other words, in our example, \( \neg(\neg A \land B) \lor \neg(\neg A \land B) \).

Next, we can apply De Morgan's law to each of the components, \( \neg(\neg A \land B) \) and \( \neg(\neg A \land B) \), and then use the law of substitution of logical equivalents to substitute the results back into the full sentence. Doing this, we get

\[ \neg((\neg A \land B) \land (\neg A \land B)) \]

(As before, 'DM' on the right means that we have used one of De Morgan's laws. 'SLE' means that we have also used the law of substitution of logical equivalents in getting the last line from the previous one.)

Now we can apply the law of double negation (abbreviated 'DN') to \( \neg X \) and to \( \neg Y \) and once more substitute the results into the larger sentence. This gives

\[ (A \land B) \lor (A \land B) \]
DN, SLE

We have only one more step to do. If you look carefully, you will see that the distributive law (abbreviated 'D') applies to the last line. So the last line is logically equivalent to

\[ A \land B \]
D

This might not be clear at first. As I stated the distributive law, you might think it applies only to show that the very last line is logically equivalent to the next to last line. But if X is logically equivalent to Y, then Y is logically equivalent to X! Logical equivalence is a matter of always having the same truth value, so if two sentences are logically equivalent, it does not matter which one gets stated first. Often students only think to apply a law of logical equivalents in the order in which it happens to be stated. But the order makes no difference—the relation of logical equivalence is symmetric, as logicians say.

Let's put all the pieces of this problem together. In the following summary, each sentence is logically equivalent to the previous sentence and the annotations on the right tell you what law or laws give you a line from the previous one.

\[ \neg((\neg A \land B) \land (\neg A \land B)) \]
DM

(As before, 'DM' on the right means that we have used one of De Morgan's laws.)

\[ (A \land B) \lor (A \land B) \]
DN, SLE

Actually, all I have really proved is that each of the above sentences is logically equivalent to the next. I really want to show that the first is logically equivalent to the last. Do you see why that must be so? Because being logically equivalent just means having the same truth value in all possible cases, we trivially have

The Law of Transitivity of Logical Equivalence (TLE): For any sentences X, Y, and Z, if X is logically equivalent to Y and Y is logically equivalent to Z, then X is logically equivalent to Z.
Repeated use of this law allows us to conclude that the first sentence in our list is logically equivalent to the last. Many of you may find this point obvious. From now on, transitivity of logical equivalence will go without saying, and you do not need explicitly to mention it in proving logical equivalences.

Here are some more easy, but very important, laws:

The **Commutative Law** (CM): For any sentences \( X \) and \( Y \), \( X \& Y \) is logically equivalent to \( Y \& X \). And \( X \lor Y \) is logically equivalent to \( Y \lor X \).

In other words, order in conjunctions and disjunctions does not make a difference. Note that the commutative law allows us to apply the distributive law from right to left as well as from left to right. For example, \( (A \& B) \lor C \) is logically equivalent to \( (A \lor C) \& (B \lor C) \). You should write out a proof of this fact using the commutative law and the distributive law as I stated it originally.

Next, the **Associative Law** tells us that \( A \&(B \& C) \) is logically equivalent to \( (A \& B) \& C \). To check this, try using a Venn diagram, which in this case gives a particularly quick and clear verification. Or simply note that both of these sentences are true only when \( A \), \( B \), and \( C \) are all true, and are false when one or more of the sentence letters are false. This fact shows that in this special case we can safely get away with dropping the parentheses and simply writing \( A \& B \& C \), by which we will mean either of the logically equivalent \( A \&(B \& C) \) or \( (A \& B) \& C \). Better yet, we will extend the way we understand the connective \( \& \). We will say that \( \& \) can appear between any number of conjuncts. The resulting conjunction is true just in case all of the conjuncts are true, and the conjunction is false in all other cases.

The same sort of generalization goes for disjunction. \( A \lor (B \lor C) \) is logically equivalent to \( (A \lor B) \lor C \). Both of these are true just in case one or more of \( A \), \( B \), and \( C \) are true and false only if all three of \( A \), \( B \), and \( C \) are false. (Again, a Venn diagram provides a particularly swift check.) We extend our definition of \( \lor \) so that it can appear between any number of disjuncts as we like. The resulting disjunction is true just in case at least one of the disjuncts is true and the disjunction is false only if all the disjuncts are false.

The **Associative Law** (A): For any sentences \( X \), \( Y \), and \( Z \), \( X \&(Y \& Z) \), \( (X \& Y) \& Z \), and \( X \& Y \& Z \) are logically equivalent to each other. And \( X \lor (Y \lor Z) \), \( (X \lor Y) \lor Z \), and \( X \lor Y \lor Z \) are logically equivalent to each other. Similarly, conjunctions with four or more components may be arbitrarily grouped and similarly for disjunctions with four or more disjuncts.

Here is yet another easy law. Clearly, \( X \& X \) is logically equivalent to \( X \). Likewise, \( X \lor X \) is logically equivalent to \( X \).

The **Law of Redundancy** (RD): For any sentence \( X \), \( X \& X \) is logically equivalent to \( X \). Similarly, \( X \lor X \) is logically equivalent to \( X \).

Let us apply this law in a little example. Again, each line is logically equivalent to the next (RD stands for the law of redundancy):

\[
\neg(A \& B) \& \neg(A \lor B) \quad \text{DM, SLE}
\]
\[
\neg(A \lor B) \& \neg(A \lor B) \quad \text{DM, SLE}
\]
\[
\neg(A \& B) \quad \text{RD}
\]

Before asking you to practice these laws, let me give you a more extended example which illustrates all the laws I have introduced so far:

\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \\
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{DM, SLE}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{CM, SLE}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{A}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{CM, SLE}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{DN, SLE}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{CM, SLE}
\]
\[
\neg(A \& B) \lor (C \lor (B \& \neg A)) \quad \text{RD, SLE}
\]

**EXERCISES**

3-2. Prove the following logical equivalences. Write out your proofs as I did in the text specifying which laws you use in getting a line from the previous line. You can use the abbreviations for the laws found in the text. Until you feel comfortable with the easy laws, please include all steps. But when they begin to seem painfully obvious, you may combine the following laws with other steps and omit mentioning that you have used them: double negation, the associative law, the commutative law, and the law of substitution of logical equivalents. You must explicitly specify any other law you use.

a) \( 'B \lor A' \) is logically equivalent to \( \neg(\neg A \& \neg B)' \).

b) \( '(A \& B) \lor C' \) is logically equivalent to \( '(A \lor C) \& (B \lor C)' \). (Show all steps in this problem.)

c) \( 'A \lor (\neg C \& B)' \) is logically equivalent to \( '(A \& C) \lor (A \& B)' \).

d) \( '\neg(\neg A \& B) \lor (C \& B)' \) is logically equivalent to \( '(\neg A \& C) \lor (\neg B \& C)' \).

e) \( '(A \& B) \lor (A \& C)' \) is logically equivalent to \( 'A \&(B \& C)' \).

f) \( '(A \& B) \lor (A \& C)' \) is logically equivalent to \( '(A \& B) \lor (A \& C)' \).

g) \( '(A \& B) \lor (A \& C)' \) is logically equivalent to \( '(A \& B) \lor (A \& C)' \).

h) \( 'A \& (\neg A \& B)' \) is logically equivalent to \( 'A \lor (\neg A \& C)' \).
i) ‘¬A&B&C’ is logically equivalent to ‘C&[(¬A¬B)v(B¬(¬CvA))].’

3–3. Give a formal statement of De Morgan’s laws in application to negations of conjunctions and disjunctions with three components. Model your formal statement on the formal statement in the text. It should begin as follows:

De Morgan’s Laws:  For any sentences X, Y, and Z . . .

3–3. LOGICAL TRUTHS AND CONTRADICTIONS

Let us look at another interesting example:

<table>
<thead>
<tr>
<th>A</th>
<th>¬A</th>
<th>Av¬A</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

‘Av¬A’ is true no matter what. Such a sentence is called a Logical Truth.

A sentence of sentence logic is a Logical Truth just in case it is true in all possible cases, that is, just in case it is true for all assignments of truth values to sentence letters.

Many authors use the word Tautology for a logical truth of sentence logic. I prefer to use the same expression, ‘logical truth’, for this idea as it applies in sentence logic and as it applies to predicate logic, which we will study in volume II.

Clearly there will also be sentences which are false no matter what, such as

<table>
<thead>
<tr>
<th>A</th>
<th>¬A</th>
<th>A&amp;¬A</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

Such a sentence is called a Contradiction.

A sentence of sentence logic is a Contradiction just in case it is false in all possible cases, that is, just in case it is false for all assignments of truth values to sentence letters.

Later on in the course, logical truths and contradictions will concern us quite a bit. They are interesting here because they provide several further laws of logical equivalence:

The Law of Logically True Conjunct (LTC): If X is any sentence and Y is any logical truth, then X&Y is logically equivalent to X.

The Law of Contradictory Disjunct (CD): If X is any sentence and Y is any contradiction, then XvY is logically equivalent to X.

You should be able to show that these laws are true. Furthermore, you should satisfy yourself that a conjunction is always a contradiction if one of its conjuncts is a contradiction and that a disjunction is always a logical truth if one of its disjuncts is a logical truth.

EXERCISES

3–4. Explain why a disjunction is always a logical truth if one of its disjuncts is a logical truth. Explain why a conjunction is always a contradiction if one of its conjuncts is a contradiction.

3–5. Further simplify the sentence ‘A&(Bv¬B)’, which was the last line of the first example in section 3–2.

3–6. Prove the following logical equivalences, following the same instructions as in exercise 3–2:

a) ‘A&(¬AvB)’ is logically equivalent to ‘A&B’.
b) ‘AvB’ is logically equivalent to ‘Av(A&¬B)’.
c) ‘A’ is logically equivalent to ‘(A&B)v(A&¬B)’. (This equivalence is called the Law of Expansion. You may find it useful in some of the other problems.)
d) ‘A’ is logically equivalent to ‘(AvB)&(Av¬B)’.
e) ‘A&(Av¬A&C)’ is logically equivalent to ‘A&B’.
f) ‘CvB’ is logically equivalent to ‘(C&A)v(B&A)v(C&¬A)v(B&¬A)’.
g) ‘CvB’ is logically equivalent to ‘(CvA)&(Bv¬D)&(¬AvC)&(DvB)’.
h) ‘A&(¬AvB)’ is logically equivalent to ‘(¬AvB)&(¬BvA)’.
i) ‘Av¬BvC’ is logically equivalent to ‘(¬AvB)vAvC’.

3–7. For each of the following sentences, determine whether it is a logical truth, a contradiction, or neither. (Logicians say that a sentence which is neither a logical truth nor a contradiction is Contingent, that is, a sentence which is true in some cases and false in others.) Simplify the sentence you are examining, using the laws of logical equivalence, to show that the sentence is logically equivalent to a sentence you already know to be a logical truth, a contradiction, or neither.

a) (B&A)v(B&¬A)
b) B&[(¬AvA)&¬B)]
Now that we understand logical equivalence, we can use it to put any sentence into a form which shows very clearly what the sentence says. As usual, we will start by looking at an example. Start with the truth table for Av-B:

<table>
<thead>
<tr>
<th>Case</th>
<th>A</th>
<th>B</th>
<th>Av-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>2</td>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>3</td>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

The truth table tells us that 'Av-B' is true in cases 1, 2, and 4. We can easily say what case 1 says using a sentence of sentence logic. Case 1 just says that 'A' and 'B' are both true, which we can say with 'A&B'. In the same way, case 2 says that 'A' is true and 'B' is false, which we say in sentence logic with 'A&-B'. Finally, '-A&-B' says that case 4 holds, that is, that 'A' is false and 'B' is false. Of course, none of these things says what 'Av-B' says, which was that either case 1 is true or case 2 is true or case 4 is true. But, clearly, we can say this in sentence logic by using the disjunction of the three sentences, each one of which describes one of the cases covered by 'Av-B'. That is

'Av-B' is logically equivalent to '(A&B)v(A&-B)v(-A&-B)'.

'(A&B)v(A&-B)v(-A&-B)' is said to be in Disjunctive Normal Form, and it says that either 'A' and 'B' are both true or 'A' is true and 'B' is false or 'A' is false and 'B' is false. This disjunction is logically equivalent to 'Av-B' because the disjunction says just what 'Av-B' says, as shown by its truth table.

Here is a slightly different way of putting the same point. The truth table shows us the possible cases in which the sentence under study will be true. We can always write a logically equivalent sentence in disjunctive normal form by literally writing out the information contained in the truth table. For each case in which the original sentence comes out true, write the conjunction of sentence letters and negated sentence letters which describe that case. Then take the disjunction of these conjunctions. This sentence will be true in exactly those cases described by the disjunctions (in our example, in the cases described by 'A&B', by 'A&-B', and by '-A&-B'). But the original sentence is true in just these same cases—that is what its truth table tells us. So the sentence in disjunctive normal form is true in exactly the same cases as the original, which is to say that the two are logically equivalent.

I want to give you a formal summary description of disjunctive normal form. But first we must deal with three troublesome special cases. I will neutralize these troublesome cases with what may at first strike you as a very odd trick.

Consider the atomic sentence 'A'. As I have so far defined disjunctions and conjunctions, 'A' is neither a disjunction nor a conjunction. But because 'A' is logically equivalent to 'AvA', it will do no harm if we extend the meaning of 'disjunction' and say that, in a trivial way, 'A' will also count as a disjunction—that is, as a degenerate disjunction which has only one disjunct.

We can play the same trick with conjunctions. 'A' is logically equivalent to 'A&A'. So we do no harm if we extend the meaning of 'conjunction' and say that, in a trivial way, 'A' also counts as a conjunction—the degenerate conjunction which has only one conjunct.

Finally, we will want to say the same thing, not just about atomic sentence letters, but about any sentence, X, atomic or compound. Whatever the form of X, we can always view X as a degenerate disjunction with just one disjunct or as a degenerate conjunction with just one conjunct.

What is the point of this apparently silly maneuver? I am working toward giving a definition of disjunctive normal form which will work for any sentence. The idea of disjunctive normal form is that it involves a disjunction of conjunctions. But what should we say, for example, is the disjunctive normal form of the sentence 'A&B'? If we allow degenerate disjunctions with one disjunct, we can say that 'A&B' is already in disjunctive normal form—think of it as '(A&B)v(A&B)'. Again, what should we say is the disjunctive normal form of 'A'? Let's count 'A' as a degenerate conjunction with just one conjunct (think of 'A&A') and let's count this conjunction as a degenerate disjunction, as in the last example. So we can say that 'A' is already in disjunctive normal form and still think of disjunctive normal form as a disjunction of conjunctions.

We still have to discuss one more special case. What should we say is
the disjunctive normal form of a contradiction, such as 'A&~A'? We will allow repetitions of sentence letters with and without negation signs, so that, again, 'A&~A' will itself already count as being in disjunctive normal form.

Now we can say very simply:

A sentence is in **Disjunctive Normal Form** if it is a disjunction, the disjuncts of which are themselves conjunctions of sentence letters and negated sentence letters. In this characterization we allow as a special case that a disjunction may have only one disjunct and a conjunction may have only one conjunct.

For any sentence, X, of sentence logic, the disjunctive normal form of X is given by a sentence Y if Y is in disjunctive normal form and is logically equivalent to X. Except for contradictions, the disjunctive normal form of a sentence is the sentence’s truth table expressed in sentence logic.

The fact that every sentence of sentence logic is logically equivalent to a sentence in disjunctive normal form helps to show something interesting about the connectives. All our sentences are put together using '&', 'v', and '~'. But are these connectives all we really need? Could we say new things if we added new connectives? The answer is no, if we limit ourselves to sentences which can be given in terms of a truth table. Because we can write any truth table in disjunctive normal form, using only '&', 'v' and '~', anything which we can express using a truth table we can express using just these three connectives. In other words, '&', 'v', and '~' are enough if we limit ourselves to a logic all the sentences of which are truth functions of atomic sentence letters. We say that '&', 'v', and '~' are, together, **Expressively Complete**. For given the truth table of any sentence which we might want to write, we can always write it with a sentence in disjunctive normal form.

Even more interestingly, '&', 'v' and '~' are more than we need. Using De Morgan’s laws and double negation, we can always get rid of a conjunction in favor of a disjunction and some negation signs. And we can always get rid of a disjunction in favor of a conjunction and some negation signs. (Do you see how to do this?) Thus any sentence which can be represented by a truth table can be expressed using just '&', '~' and 'v'. And any such sentence can be expressed using just 'v' and '~'. So '&', 'v' and '~' are expressively complete, and 'v' and '~' are also expressively complete.

We have just seen that anything that can be represented with truth tables can be expressed with a sentence using just two connectives. Could we make do with just one connective? Clearly, we can’t make do with just ‘&’, with just ‘v’, or with just ‘~’. (Can you see why?) But perhaps we could introduce a new connective which can do everything all by itself. Consider the new connective ‘|’, called the **Sheffer Stroke**, defined by

| X | Y | X|Y |
|---|---|---|
| t | t | f |
| t | f | f |
| f | t | f |
| f | f | t |

Work out the truth table and you will see that X|X is logically equivalent to ~X. Similarly, you can prove that (X|Y)|(X|Y) is logically equivalent to XvY. With this new fact, we can prove that ‘|’ is expressively complete. We can express any truth function in disjunctive normal form. Using De Morgan’s law and the law of double negation, we can get rid of the ‘&’s in the disjunctive normal form. So we can express any truth function using just ‘v’ and ‘~’. But now for each negation we can substitute a logically equivalent expression which uses just ‘|’. And for each disjunction we can also substitute a logically equivalent expression which uses just ‘|’. The final result uses ‘|’ as its only connective. Altogether, the sentence in disjunctive normal form has been transformed into a logically equivalent sentence using just ‘|’. And because any truth function can be put in disjunctive normal form, we see that any truth function, that is, any sentence which could be given a truth table definition, can be expressed using just ‘|’.

The important idea here is that of expressiveness completeness:

A connective, or set of connectives, is **Expressively Complete** for truth functions if and only if every truth function can be represented using just the connective or connectives.

Actually, the really important idea is that of a truth function. Understanding expressiveness completeness for truth functions will help to make sure you have the idea of a truth function clearly in mind.

**EXERCISES**

3-8: Put the following sentences in disjunctive normal form. You can do this most straightforwardly by writing out truth tables for the sentences and then reading off the disjunctive normal form from the truth tables. Be sure you know how to work the problems this way. But you might have more fun trying to put a sentence in disjunctive normal form by following this procedure: First, apply De Morgan’s laws to drive all negations inward until negation signs apply only to sentence letters. Then use other laws to get the sentence in the final disjunctive normal form.
3-9. Suppose you are given a sentence in which 'v' occurs. Explain in general how you can write a logically equivalent sentence in which 'v' does not occur at all. Similarly, explain how a sentence in which '&' occurs can be replaced by a logically equivalent sentence in which '&' does not occur. (Hint: You will need to appeal to De Morgan's laws.)

3-10. Define a new connective, '∗', as representing the following truth function:

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X∗Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

Show that '∗' is expressively complete.

3-11. Show that '&' is not expressively complete. That is, give a truth function and show that this truth function cannot be expressed by using '&' as the only connective. Similarly, show that 'v' is not expressively complete and show that '¬' is not expressively complete. (You may find this problem hard, but please take a few minutes to try to work it.)

CHAPTER SUMMARY EXERCISE

Once again, you will find below the important terms which I have introduced in this chapter. Make sure you understand all of them by writing out a short explanation of each. You should refer to the text to make sure that you have correctly explained each term. Please keep your explanations of these terms in your notebook for reference and review.

a) Logical Equivalence
b) Venn Diagram
c) Law of Double Negation
d) De Morgan's Laws
e) Distributive Laws
f) Law of Substitution of Logical Equivalents
g) Law of Transitivity of Logical Equivalence

h) Commutative Laws
i) Associative Law
j) Law of Redundancy
k) Logical Truth
l) Contradiction
m) Law of Logically True Conjunct
n) Law of Contradictory Disjunct

If you have read section 3-4, also explain

o) Disjunctive Normal Form
p) Expressively Complete
q) Sheffer Stroke
Validity and Conditionals

4-1. VALIDITY

Consider the following argument:

\[
\begin{align*}
A \vee B & \quad \text{Adam loves Eve or Adam loves Bertha.} \\
\neg A & \quad \text{Adam does not love Eve.} \\
B & \quad \text{Adam loves Bertha.}
\end{align*}
\]

If you know, first of all, that either 'A' or 'B' is true, and in addition you know that 'A' itself is false; then clearly, 'B' has to be true. So from 'A \vee B' and '\neg A' we can conclude 'B'. We say that this argument is Valid, by which we mean that, without fail, if the premises are true, then the conclusion is going to turn out to be true also. We interpret this to mean that in each possible case (in each of the cases 1 through 4), if the premises are true in that case, then the conclusion is true in that case. In other words, in all cases in which the premises are true, the conclusion is also true. In yet other words:

"To say that an argument (expressed with sentences of sentence logic) is Valid is to say that any assignment of truth values to sentence letters which makes all of the premises true also makes the conclusion true."

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>\neg A</th>
<th>A \vee B</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
</tbody>
</table>

We know that cases 1 through 4 constitute all the ways in which any of the sentences in the argument may turn out to be true or false. This enables us to explain very exactly what we mean by saying that, without fail, if the premises are true, then the conclusion is going to turn out to be true also. We interpret this to mean that in each possible case (in each of the cases 1 through 4), if the premises are true in that case, then the conclusion is true in that case. In other words, in all cases in which the premises are true, the conclusion is also true. In yet other words:

"To say that an argument (expressed with sentences of sentence logic) is Valid is to say that any assignment of truth values to sentence letters which makes all of the premises true also makes the conclusion true."

4-2. INVALIDITY AND COUNTEREXAMPLES

Let's look at an example of an Invalid argument (an argument which is not valid):

\[
\begin{align*}
A \vee B & \\
A & \\
\frac{}{B}
\end{align*}
\]

I have set up a truth table which shows the argument to be invalid. First I use a '*' to mark each case in which the premises are all true. In one of these cases (the second) the conclusion is false. This is what can't happen in a valid argument. So the argument is invalid. I will use the term Counterexample for a case which in this way shows an argument to be invalid. A counterexample to an argument is a case in which the premises are true and the conclusion is false.

In fact, we can use this idea of a counterexample to reword the defini-
tion of validity. To say that an argument is valid is to say that any assignment of truth values to sentence letters which makes all of the premises true also makes the conclusion true. We reword this by saying: An argument is valid just in case there is no possible case, no assignment of truth values to sentence letters, in which all of the premises are true and the conclusion is false. To be valid is to rule out any such possibility. We can break up this way of explaining validity into two parts:

A Counterexample to a sentence logic argument is an assignment of truth values to sentence letters which makes all of the premises true and the conclusion false.

An argument is Valid just in case there are no counterexamples to it.

Now let us reexpress all of this using sentences of sentence logic and the idea of logical truth. Let us think of an argument in which X is the conjunction of all the premises and Y is the conclusion. X and Y might be very complicated sentences. The argument looks like this:

\[ \neg(X \land \neg Y) \]

I will express an argument such as this with the words “X. Therefore Y”.

A counterexample to such an argument is a case in which X is true and Y is false, that is, a case in which \(X \land \neg Y\) is true. So to say that there are no possible cases in which there is a counterexample is to say that in all possible cases \(X \land \neg Y\) is false, or, in all possible cases \(\neg(X \land \neg Y)\) is true. But to say this is just to say that \(\neg(X \land \neg Y)\) is a logical truth. The grand conclusion is that

The argument “X. Therefore Y” is valid just in case the sentence \(\neg(X \land \neg Y)\) is a logical truth.

4-3. SOUNDNESS

Logic is largely about validity. So to understand clearly what much of the rest of this book is about, you must clearly distinguish validity from some other things.

If I give you an argument by asserting to you something of the form “X. Therefore Y”, I am doing two different things. First, I am asserting the premise or premises, X. Second, I am asserting to you that from these premises the conclusion, Y follows.

To see clearly that two different things are going on here, consider that there are two ways in which I could be mistaken. It could turn out that I am wrong about the claimed truth of the premises, X. Or I could be wrong about the ‘therefore’. That is, I could be wrong that the conclusion, Y, validly follows from the premises, X. To claim that X is true is one thing. It is quite another thing to make a claim corresponding to the ‘therefore’, that the argument is valid, that is, that there is no possible case in which the premises are true and the conclusion is false.

Some further, traditional terminology helps to emphasize this distinction. If I assert that the argument, “X. Therefore Y”, is valid, I assert something about the relation between the premises and the conclusion, that in all lines of the truth table in which the premises all turn out true, the conclusion turns out true also. In asserting validity, I do not assert that the premises are in fact true. But of course, I can make this further assertion. To do so is to assert that the argument is not only valid, but Sound:

An argument is Sound just in case, in addition to being valid, all its premises are true.

Logic has no special word for the case of a valid argument with false premises.

To emphasize the fact that an argument can be valid but not sound, here is an example:

Teller is ten feet tall or Teller has never taught logic. AvB
Teller is not ten feet tall. \(-A\)
Teller has never taught logic. \(B\)

Viewed as atomic sentences, ‘Teller is ten feet tall.’ and ‘Teller has never taught logic.’ can be assigned truth values in any combination, so that the truth table for the sentences of this argument looks exactly like the truth table of section 4-1. The argument is perfectly valid. Any assignment of truth values to the atomic sentences in which the premises both come out true (only case 3) is an assignment in which the conclusion comes out true also. But there is something else wrong with the argument of the present example. In the real world, case 3 does not in fact apply. The argument’s first premise is, in fact, false. The argument is valid, but not sound.

EXERCISES

4-1. Give examples, using sentences in English, of arguments of each of the following kind. Use examples in which it is easy to tell whether the premises and the conclusion are in fact (in real life) true or false.
Validity and Conditionals

4-2. Use truth tables to determine which of the following arguments are valid. Use the following procedure, showing all your work: First write out a truth table for all the sentences in the argument. Then use a * to mark all the lines of the truth table in which all of the argument's premises are true. Next look to see whether the conclusion is true in the *ed lines. If you find any *ed lines in which the conclusion is false, mark these lines with the word 'counterexample'. You know that the argument is valid if and only if there are no counterexamples, that is, if and only if all the cases in which all the premises are true are cases in which the conclusion is also true. Write under the truth table whether the argument is valid or invalid (i.e., not valid).

<table>
<thead>
<tr>
<th>a) ( \neg (A &amp; B) )</th>
<th>b) ( \neg A )</th>
<th>c) ( A )</th>
<th>d) ( \neg B )</th>
<th>e) ( B )</th>
<th>If ( A ) then ( B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \neg A )</td>
<td>( A )</td>
<td>( \neg B )</td>
<td>( A )</td>
<td>( \neg B )</td>
<td>( B )</td>
</tr>
</tbody>
</table>

4-3. Show that \( X \) is logically equivalent to \( Y \) if and only if the arguments “\( X \) therefore \( Y \)” and “\( Y \) therefore \( X \)” are both valid.

4-4. THE CONDITIONAL

In section 4-2 we saw that the argument, “\( X \) Therefore \( Y \)”, is intimately related to the truth function \( \neg (X \& \neg Y) \). This truth function is so important that we are going to introduce a new connective to represent it. We will define \( X \Rightarrow Y \) to be the truth function which is logically equivalent to \( \neg (X \& \neg Y) \). You should learn its truth table definition:

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>( X \Rightarrow Y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

Again, the connection between \( X \Rightarrow Y \) and the argument “\( X \) Therefore \( Y \)” is that \( X \Rightarrow Y \) is a logical truth just in case the argument “\( X \) Therefore \( Y \)” is valid.

Logicians traditionally read a sentence such as ‘\( A \Rightarrow B \)’ with the words ‘If \( A \), then \( B \)’, and the practice is to transcribe ‘If . . . then . . .’ sentences of English by using ‘\( \Rightarrow \)’. So (to use a new example) we would transcribe ‘If the cat is on the mat, then the cat is asleep,’ as ‘\( A \Rightarrow B \)’.

In many ways, this transcription proves to be problematic. To see why, let us forget ‘\( \Rightarrow \)’ for a moment and set out afresh to define a truth functional connective which will serve as a transcription of the English ‘If . . . then . . .’:

\[
\begin{array}{c|c|c}
\text{case} & A & \text{If } A \text{ then } B \\
\hline
1 & t & t \\
2 & t & f \\
3 & f & t \\
4 & f & f \\
\end{array}
\]

That is, by choosing t or f for each of the boxes under ‘If \( A \) then \( B \)’ in the truth table, we want to write down a truth function which says as closely as possible what ‘If \( A \) then \( B \)’ says in English.

The only really clear-cut case is case 2, the case in which the cat is on the mat but is not asleep. In this circumstance, the sentence ‘If the cat is on the mat, then the cat is asleep.’ is most assuredly false. So we have to put f for case 2 in the column under ‘If \( A \) then \( B \)’. If the cat is both on the mat and is asleep, that is, if we have case 1, we may plausibly take the conditional sentence to be true. So let us put t for case 1 under ‘If \( A \) then \( B \)’. But what about cases 3 and 4, the two cases in which \( A \) is false? If the cat is not on the mat, what determines whether or not the conditional, ‘If the cat is on the mat, then the cat is asleep.’, is true or false?

Anything we put for cases 3 and 4 is going to give us problems. Suppose we put t for case 3. This is to commit ourselves to the following: When the cat is not on the mat and the cat is asleep somewhere else, then the conditional, ‘If the cat is on the mat, then the cat is asleep.’, is true. But suppose we have sprinkled the mat with catnip, which always makes the cat very lively. Then, if we are going to assign the conditional a truth value at all, it rather seems that it should count as false. On the other hand, if we put f for case 3, we will get into trouble if the mat has a cosy place by the fire which always puts the cat to sleep. For then, if we assign a truth value at all, we will want to say that the conditional is true. Similar examples show that neither t nor f will always work for case 4.

Our problem has a very simple source: ‘If . . . then . . .’ in English can be used to say various things, many of which are not truth functional.
Whether or not an ‘If . . . then . . .’ sentence of English is true or false in these nontruth functional uses depends on more than just the truth values of the sentences which you put in the blanks. The truth of ‘If you are five feet five inches tall, then you will not be a good basketball player.’ depends on more than the truth or falsity of ‘You are five feet five inches tall.’ and ‘You will not be a good basketball player.’ It depends on the fact that there is some factual, nonlogical connection between the truth and falsity of these two component sentences.

In many cases, the truth or falsity of an English ‘If . . . then . . .’ sentence depends on a nonlogical connection between the truth and falsity of the sentences which one puts in the blanks. The connection is often causal, temporal, or both. Consider the claim that ‘If you stub your toe, then it will hurt.’ Not only does assertion of this sentence claim that there is some causal connection between stubbing your toe and its hurting, this assertion also claims that the pain will come after the stubbing. However, sentence logic is insensitive to such connections. Sentence logic is a theory only of truth functions, of connectives which are defined entirely in terms of the truth and falsity of the component sentences. So no connective defined in sentence logic can give us a good transcription of the English ‘If . . . then . . .’ in all its uses.

What should we do? Thus far, one choice for cases 3 and 4 seems as good (or as bad) as another. But the connection between the words ‘. . . therefore . . .’ and ‘If . . . then . . .’ suggests how we should make up our minds. When we use ‘If . . . then . . .’ to express some causal, temporal, or other nonlogical connection between things in the world, the project of accurately transcribing into sentence logic is hopeless. But when we use ‘If . . . then . . .’ to express what we mean by ‘. . . therefore . . .’, our course should be clear. To assert ‘X. Therefore Y’, is to advance the argument with X as premise(s) and Y as conclusion. And to advance the argument, ‘X. Therefore Y’, is (in addition to asserting X) to assert that the present case is not a counterexample; that is, it is to assert that the sentence ∼(X&∼Y) is true. In particular, if the argument, ‘X. Therefore Y’, is valid, there are no counterexamples, which, as we saw, comes to the same thing as ∼(X&∼Y) being a logical truth.

Putting these facts together, we see that when “IF X THEN Y” conveys what the ‘therefore’ in “X. Therefore Y” conveys, we can transcribe the “IF X THEN Y” as ∼(X&∼Y), for which we have introduced the new symbol X⇒Y. In short, when ‘If . . . then . . .’ can be accurately transcribed into sentence logic at all, we need to choose t for both cases 3 and 4 to give us the truth table for X⇒Y defined as ∼(X&∼Y).

Logicians recognize that ‘⇒’ is not a very faithful transcription of ‘If . . . then . . .’. when ‘If . . . then . . .’ expresses any sort of nonlogical connection. But just since ‘⇒’ agrees with ‘If . . . then . . .’ in the clear case 2 and the fairly clear case 1, ‘⇒’ is going to be at least as good a transcription as any alternative. And the connection with arguments at least makes ‘⇒’ the right choice for cases 3 and 4 when there is a right choice, that is, when ‘If . . . then . . .’ means ‘. . . therefore . . .’.

We have labored over the introduction of the sentence logic connective ‘⇒’. Some logic texts just give you its truth table definition and are done with it. But logicians use the ‘⇒’ so widely to transcribe the English ‘If . . . then . . .’ that you should appreciate as clearly as possible the (truth functional) ways in which ‘⇒’ does and the (nontruth functional) ways in which ‘⇒’ does not correspond to ‘If . . . then . . .’.

In these respects, the English ‘and’ and ‘or’ seem very different. ‘And’ and ‘or’ seem only to have truth functional aspects, so that they seem to correspond very closely to the truth functionally defined ‘&’ and ‘v’. Now that you have been through some consciousness raising about how English can differ from logic in having nontruth functional aspects, it is time to set the record straight about the ‘and’ and ‘or’ of English.

Surely, when I assert, ‘Adam exchanged vows with Eve, and they became man and wife.’ I do more than assert the truth of the two sentences ‘Adam exchanged vows with Eve’ and ‘They became man and wife.’ I assert that there is a connection, that they enter into the state of matrimony as a result of exchanging vows. Similarly, if I yell at you, ‘I agree with you or I’ll knock your block off!’ I do more than assert that either ‘You will agree with me’ or ‘I will knock your block off’ is true. I assert that nonagreement will produce a blow to your head. In these examples ‘and’ and ‘or’ convey some causal, intentional, or conventional association which goes above and beyond the truth functional combination of the truth values of the component sentences. ‘And’ can likewise clearly express a temporal relation which goes beyond the truth values of the components. When I say, ‘Adam put on his seat belt and started the car.’ I assert not only that ‘Adam put on his seat belt.’ and ‘He started the car.’ are both true. I also assert that the first happened before the second.

Although ‘and’, ‘or’, and ‘If . . . then . . .’ all have their nontruth functional aspects, in this respect ‘If . . . then . . .’ is the most striking. ‘⇒’ is much weaker than ‘if . . . then . . .’. inasmuch as ‘⇒’ leaves out all of the nontruth functional causal, temporal, and other connections often conveyed when we use ‘If . . . then . . .’. Students sometimes wonder: If ‘⇒’ (and ‘&’ and ‘v’) are so much weaker than their English counterparts, why should we bother with them? The answer is that although truth functional sentence logic will only serve to say a small fraction of what we can say in English, what we can say with sentence logic we can say with profound clarity. In particular, this clarity serves as the basis for the beautifully clear exposition of the nature of deductive argument.

When the language of logic was discovered, its clarity so dazzled philosophers and logicians that many hoped it would ultimately replace English, at least as an all-encompassing exact language of science. Historically, it
took decades to realize that the clarity comes at the price of important expressive power.

But back to ‘3’.

Here are some things you are going to need to know about the connective ‘3’:

A sentence of the form X > Y is called a Condition. X is called its Antecedent and Y is called its Consequent.

Look at the truth table definition of X > Y and you will see that, unlike conjunctions and disjunctions, conditions are not symmetric. That is, X > Y is not logically equivalent to Y > X. So we need names to distinguish between the components. This is why we call the first component the antecedent and the second component the consequent (not the conclusion—a conclusion is a sentence in an argument).

Probably you will most easily remember the truth table definition of the conditional if you focus on the one case in which it is false, the one case in which the conditional always agrees with English. Just remember that a conditional is false if the antecedent is true and the consequent is false, and true in all other cases. Another useful way for thinking about the definition is to remember that if the antecedent of a conditional is false, then the whole conditional is true whatever the truth value of the consequent. And if the consequent is true, then again the conditional is true, whatever the truth value of the antecedent.

Finally, you should keep in mind some logical equivalences:

The Law of the Conditional (C): X > Y is logically equivalent to ~ (X & ~ Y) and (by De Morgan’s law) to ~ X V Y.

The Law of Contraposition (CP): X > Y is logically equivalent to ~ Y > ~ X.

4–5. THE BICONDITIONAL

We introduce one more connective into sentence logic. Often we will want to study cases which involve a conjunction of the form (X > Y) & (Y > X). This truth function of X and Y occurs so often in logic that we give it its own name, the Biconditional, which we write as X = Y. Working out the truth table of (X > Y) & (Y > X) we get as our definition of the biconditional:

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X = Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

Because a biconditional has a symmetric definition, we don’t have different names for its components. We just call them ‘components’. You will remember this definition most easily by remembering that a biconditional is true if both components have the same truth value (both true or both false), and it is false if the two components have different truth values (one true, the other false). We read the biconditional X = Y with the words ‘X if and only if Y’. With the biconditional, we get into much less trouble with transcriptions between English and sentence logic than we did with the conditional.

Given the way we define ‘=’, we have the logical equivalence:

The Law of the Biconditional (B): X = Y is logically equivalent to (X > Y) & (Y > X).

Remember that the conditional, X > Y, is a logical truth just in case the corresponding argument, “X. Therefore Y”, is valid. Likewise, there is something interesting we can say about the biconditional, X = Y, being a logical truth:

X = Y is a logical truth if and only if X and Y are logically equivalent.

Can you see why this is true? Suppose X = Y is a logical truth. This means that in every possible case (for every assignment of truth values to sentence letters) X = Y is true. But X = Y is true only when its two components have the same truth value. So in every possible case, X and Y have the same truth value, which is just what we mean by saying that they are logically equivalent. On the other hand, suppose that X and Y are logically equivalent. This just means that in every possible case they have the same truth value. But when X and Y have the same truth value, X = Y is true. So in every possible case X = Y is true, which is just what is meant by saying that X = Y is a logical truth.

EXERCISES

4–4. In section 1–6 I gave rules of formation and valuation for sentence logic. Now that we have extended sentence logic to include the connectives ‘&’ and ‘=’, these rules also need to be extended. Write the full rules of formation and valuation for sentence logic, where sentence logic may now use all of the connectives ‘=’, ‘&’, ‘V’, ‘>’, and ‘=’. In your rules, also provide for three and more place conjunctions and disjunctions as described in section 3–2 in the discussion of the associative law.
4-5. Follow the same instructions as for exercise 4-2.

\[\begin{array}{cccc}
a) & A \to B & b) & A \to \neg B \\
& B & & B \\
a & \neg A & & \neg A \\
b) & (A \vee B) \to (A \land C) & f) & (A \vee B) = (A \land \neg C) \\
& C \land A & & \neg B \land C \\
& \neg C & & A \land C \\
c) & A \equiv B \\
d) & A \equiv \neg B \\
e) & (A \to B) \equiv (A \land \neg B) \\
f) & \neg (A \vee B) \equiv \neg (A \land \neg B) \\
g) & (A \equiv A) \to (B \equiv \neg B) \\
h) & \neg (B \equiv A) \to (C \equiv B) \\
i) & (A \equiv (B \equiv C)) \equiv ((A \equiv B) \equiv (A \equiv C)) \\
\end{array}\]

4-6. For each of the following sentences, establish whether it is a logical truth, a contradiction, or neither. Use the laws of logical equivalence in chapter 3 and sections 4-3 and 4-4, and use the fact that a biconditional is a logical truth if and only if its components are logically equivalent.

\[\begin{array}{c}
a) & (A \equiv B) = (\neg B \equiv \neg A) \\
b) & (A \equiv \neg A) \equiv (B \equiv B) \\
c) & A \equiv A \\
d) & A \equiv B \\
e) & (A \equiv B) \equiv (A \equiv \neg B) \\
f) & \neg (A \lor B) \equiv \neg (A \land \neg B) \\
g) & (A \equiv A) \equiv (B \equiv \neg B) \\
h) & \neg (B \equiv A) \equiv (C \equiv B) \\
i) & (A \equiv (B \equiv C)) \equiv ((A \equiv B) \equiv (A \equiv C)) \\
\end{array}\]

4-7. Discuss how you would transcribe 'unless' into sentence logic. Experiment with some examples, trying out the use of 'v', '\lor', and '\neg'. Bear in mind that one connective might work well for one example, another connective for another example. As you work, pay attention to whether or not the compound English sentences you choose as examples are truth functional. Report the results of your research by giving the following:

a) Give an example of a compound English sentence using 'unless' which seems to be nontruth functional, explaining why it is not truth functional.
b) Give an example of a compound English sentence using 'unless' which seems to be truth functional, explaining why it is truth functional.
c) Give one example of each English sentences using 'unless' which can be fairly well transcribed into sentence logic using 'v', '\lor', '\neg', giving the transcriptions into sentence logic.

4-8. Transcribe the following sentences into sentence logic, using the given transcription guide:

\[\begin{array}{ll}
A: & \text{Adam loves Eve.} \\
B: & \text{Adam is blond.} \\
C: & \text{Eve is clever.} \\
D: & \text{Eve has dark eyes.} \\
E: & \text{Eve loves Adam.} \\
\end{array}\]

\[\begin{array}{l}
a) & \text{If Eve has dark eyes, then Adam does not love her.} \\
b) & \text{Adam loves Eve if she has dark eyes.} \\
c) & \text{If Adam loves Eve, Eve does not love Adam.} \\
d) & \text{Eve loves Adam only if he is not blond.} \\
e) & \text{Adam loves Eve if and only if she has dark eyes.} \\
f) & \text{Eve loves Adam provided he is blond.} \\
g) & \text{Provided she is clever, Adam loves Eve.} \\
h) & \text{Adam does not love Eve unless he is blond.} \\
i) & \text{Unless Eve is clever, she does not love Adam.} \\
j) & \text{If Adam is blond, then he loves Eve only if she has dark eyes.} \\
k) & \text{If Adam is not blond, then he loves Eve whenever or not she has dark eyes.} \\
l) & \text{Adam is blond and in love with Eve if and only if she is clever.} \\
m) & \text{Only if Adam is blond is Eve both clever and in love with Adam.} \\
\end{array}\]

4-9. Consider the following four different kinds of nontruth functional connectives that can occur in English:

\[\begin{array}{l}
a) \text{Connectives indicating connections (causal, intentional, or conventional)} \\
b) \text{Modalities (what must, can, or is likely to happen)} \\
c) \text{So-called "propositional attitudes," having to do with what people know, believe, think, hope, want, and the like} \\
d) \text{Temporal connectives, having to do with what happens earlier, later, or at the same time as something else.} \\
\end{array}\]

Give as many English connectives as you can in each category. Keep in mind that some connectives will go in more than one category. ('Since' is such a connective. What two categories does it go into?) To get you started, here are some of these connectives: 'because', 'after', 'more likely than', 'Adam knows that', 'Eve hopes that'.

[The text continues with additional content, but it is not necessary to transcribe it further for this task.]
CHAPTER SUMMARY EXERCISES

Give brief explanations for each of the following. As usual, check your explanations against the text to make sure you get them right, and keep them in your notebook for reference and review.

a) Valid
b) Invalid
c) Counterexample
d) Sound
e) Conditional
f) Biconditional
g) Law of the Conditional
h) Law of Contraposition
i) Law of the Biconditional
5–1. THE IDEA OF NATURAL DEDUCTION

In chapter 4 you learned that saying an argument is valid means that any case which makes all of the argument's premises true also makes its conclusion true. And you learned how to test for validity by using truth tables, by exhaustively checking all the relevant cases, that is, all the lines of the truth table. But truth tables are horribly awkward. It would be nice to have a way to check validity which looked more like the forms of argument we know from everyday life.

Natural deduction does just that. When we speak informally, we use many kinds of valid arguments. (I'll give some examples in a moment.) Natural deduction makes these familiar forms of argument exact. It also organizes them in a system of valid arguments in which we can represent absolutely any valid argument.

Let's look at some simple and, I hope, familiar forms of argument. Suppose I know (say, because I know Adam's character) that if Adam loves Eve, then he will ask Eve to marry him. I then find out from Adam's best friend that Adam does indeed love Eve. Being a bright fellow, I immediately conclude that a proposal is in the offing. In so doing I have used the form of argument traditionally called 'modus ponens', but which I am going to call Conditional Elimination.
Conditional Elimination

\[
\begin{align*}
X \rightarrow Y \\
X & \quad \rightarrow E \\
Y
\end{align*}
\]

Logicians call such an argument form a Rule of Inference. If, in the course of an argument, you are given as premises (or you have already concluded) a sentence of the form \(X \rightarrow Y\) and the sentence \(X\), you may draw as a conclusion the sentence \(Y\). This is because, as you can check with a truth table, in any case in which sentences of the form \(X \rightarrow Y\) and \(X\) are both true, the sentence \(Y\) will be true also. You may notice that I have stated these facts, not for some particular sentences 'A\(\rightarrow\)B', 'A', and 'B', but for sentence forms expressed with boldfaced 'X' and 'Y'. This is to emphasize the fact that this form of argument is valid no matter what specific sentences might occur in the place of 'X' and 'Y'.

Here is another example of a very simple and common argument form, or rule of inference:

Disjunction Elimination

\[
\begin{align*}
X \lor Y \\
\neg X & \quad \lor E \\
Y
\end{align*}
\]

If I know that either Eve will marry Adam or she will marry no one, and I then somehow establish that she will not marry Adam (perhaps Adam has promised himself to another), I can conclude that Eve will marry no one. (Sorry, even in a logic text not all love stories end happily!) Once again, as a truth table will show, this form of argument is valid no matter what sentences occur in the place of 'X' and in the place of 'Y'.

Though you may never have stopped explicitly to formulate such rules of argument, all of us use rules like these. When we argue we also do more complicated things. We often give longer chains of argument which start from some premises and then repeatedly use rules in a series of steps. We apply a rule to premises to get an intermediate conclusion. And then, having established the intermediate conclusion, we can use it (often together with some of the other original premises) to draw further conclusions.

Let's look at an example to illustrate how this process works. Suppose you are given the sentences 'A\(\rightarrow\)B', 'B\(\lor\)C', and 'A' as premises. You are asked to show that from these premises the conclusion 'C' follows. How can you do this?

It's not too hard. From the premises 'A\(\rightarrow\)B' and 'A', the rule of conditional elimination immediately allows you to infer 'B':

\[
\begin{align*}
A \rightarrow B \\
A & \quad \rightarrow E \\
B
\end{align*}
\]

But now you have 'B' available in addition to the original premise 'B\(\lor\)C'. From these two sentences, the rule of conditional elimination allows you to infer the desired conclusion 'C':

\[
\begin{align*}
B \lor C \\
B & \quad \lor E \\
C
\end{align*}
\]

I hope this example is easy to follow. But if I tried to write out an example with seven steps in this format, things would get impossibly confusing. We need a streamlined way of writing chains of argument.

The basic idea is very simple. We begin by writing all our premises and then drawing a line to separate them from the conclusions which follow. But now we allow ourselves to write any number of conclusions below the line, as long as the conclusions follow from the premises. With some further details, which I'll explain in a minute, the last example looks like this:

<table>
<thead>
<tr>
<th>Line</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A(\rightarrow)B, P</td>
</tr>
<tr>
<td>2</td>
<td>B(\lor)C, P</td>
</tr>
<tr>
<td>3</td>
<td>A, P</td>
</tr>
<tr>
<td>4</td>
<td>B, 1, 3, \rightarrow E</td>
</tr>
<tr>
<td>5</td>
<td>C, 2, 4, \rightarrow E</td>
</tr>
</tbody>
</table>

Lines 1 through 5 constitute a Derivation of conclusions 4 and 5 from premises 1, 2, and 3. In thinking about such a derivation, you should keep most clearly in mind the idea that the conclusions are supposed to follow from the premises, in the following sense: Any assignment of truth values to sentence letters which makes the premises all true will also make all of the conclusions true.

In a derivation, every sentence below the horizontal line follows from the premises above the line. But sentences below the line may follow directly or indirectly. A sentence follows directly from the premises if a rule of inference applies directly to premises to allow you to draw the sentence as a conclusion. This is the way I obtained line 4. A sentence follows indirectly from the premises if a rule of inference applies to some conclu-
The Idea of Natural Deduction

62  Natural Deduction for Sentence Logic

The derivation looks like this:

1  A ⊃ ¬B  P
2  BvC  P
3  A  P
4  ¬B  1, 3, Df
5  C  2, 4, Df
6  CvD  5, Dv

The sentence of line 4 (I'll just say "line 4" for short) is licensed by applying conditional elimination to lines 1 and 3. Line 5 is licensed by applying disjunction elimination to lines 2 and 4. Finally, I license line 6 by applying disjunction introduction to line 5.

EXERCISES

5–1. For each of the following arguments, provide a derivation which shows the argument to be valid. That is, for each argument construct a derivation which uses as premises the argument's premises and which has as final conclusion the conclusion of the argument. Be sure to number and annotate each step as I have done with the examples in the text. That is, for each conclusion, list the rule which licenses drawing the conclusion and the line numbers of the sentences to which the rule applies.

a) ¬P ⊃ ¬Q
b) ¬C ⊃ ¬D
c) Fv ⊃ G
d) A ⊃ B
e) Lv ⊃ M
f) C
  C ⊃ (NvA)
g) (Kv ⊃ D) ⊃ F
h) D
  (DvB) ⊃ ¬G
  (¬GvH) ⊃ (GvQ)
i) (MvT) ⊃ (AvJ)
j) ¬A
  BvM
  ¬A ⊃ ¬B
  jvD

For the moment don’t worry too much about the vertical line on the left. It’s called a Scope Line. Roughly speaking, the scope line shows what hangs together as one extended chain of argument. You will see why scope lines are useful when we introduce a new idea in the next section.

You should be sure you understand why it is legitimate to draw conclusions indirectly from premises, by appealing to previous conclusions. Again, what we want to guarantee is that any case (i.e., any assignment of truth values to sentence letters) which makes the premises true will also make each of the conclusions true. We design the rules of inference so that whenever they apply to sentences and these sentences happen to be true, then the conclusion licensed by the rule will be true also. For short, we say that the rules are Truth Preserving.

Suppose we have a case in which all of the premises are true. We apply a rule to some of the premises, and because the rule is truth preserving, the conclusion it licenses will, in our present case, also be true. (Line 4 in the last example illustrates this.) But if we again apply a rule, this time to our first conclusion (and possibly some premise), we are again applying a rule to sentences which are, in the present case, all true. So the further conclusion licensed by the rule will be true too. (As an illustration, look at line 5 in the last example.) In this way, we see that if we start with a case in which all the premises are true and use only truth preserving rules, all the sentences which follow in this manner will be true also.

To practice, let’s try another example. We’ll need a new rule:

\[
\begin{align*}
\text{Disjunction Introduction} \\
X & \quad \text{v} \\
\hline
X \lor Y
\end{align*}
\]

which says that if \( X \) is true, then so is \( X \lor Y \). If you recall the truth table definition of \( \lor \), you will see that disjunction introduction is a correct, truth preserving rule of inference. The truth of even one of the disjuncts in a disjunction is enough to make the whole disjunction true. So if \( X \) is true, then so is \( X \lor Y \), whatever the truth value of \( Y \).

Let’s apply this new rule, together with our two previous rules, to show that from the premises ‘\( A \supset \sim B \)’, ‘\( B \lor C \)’, and ‘\( A \)’, we can draw the conclusion ‘\( C \lor D \)’. But first close the book and see if you can do it for yourself.
5–2. SUBDERIVATIONS

Many of you have probably been thinking: So far, we have an “introduction” and an “elimination” rule for disjunction and just an “elimination” rule for the conditional. I bet that by the time we’re done we will have exactly one introduction and one elimination rule for each connective. That’s exactly right. Our next job is to present the introduction rule for the conditional, which involves a new idea.

How can we license a conclusion of the form X ⊃ Y? Although we could do this in many ways, we want to stick closely to argument forms from everyday life. And most commonly we establish a conclusion of the form X ⊃ Y by presenting an argument with X as the premise and Y as the conclusion. For example, I might be trying to convince you that if Adam loves Eve, then Adam will marry Eve. I could do this by starting from the assumption that Adam loves Eve and arguing, on that assumption, that matrimony will ensue. Altogether, I will not have shown that Adam and Eve will get married, because in my argument I used the unargued assumption that Adam loves Eve. But I will have shown that if Adam loves Eve, then Adam will marry Eve.

Let’s fill out this example a bit. Suppose that you are willing to grant, as premises, that if Adam loves Eve, Adam will propose (A ⊃ B), and that if Adam proposes, marriage will ensue (B ⊃ C). But neither you nor I have any idea whether or not Adam does love Eve (whether ‘A’ is true). For the sake of argument, let’s add to our premises the temporary assumption, ‘A’, which says that Adam loves Eve, and see what follows. Assuming ‘A’, that Adam loves Eve, we can conclude ‘B’ which says that Adam will propose (by conditional elimination, since we have as a premise ‘A ⊃ B’, that if Adam loves Eve, he will propose). And from the conclusion ‘B’, that Adam will propose, we can further conclude ‘C’, that marriage will ensue (again by conditional elimination, this time appealing to the premise ‘B ⊃ C’, that proposal will be followed by marriage). So, on the temporary assumption ‘A’, that Adam loves Eve, we can conclude ‘C’, that marriage will ensue. But the assumption was only temporary. We are not at all sure that it is true, and we just wanted to see what would follow from it. So we need to discharge the temporary assumption, that is, restate what we can conclude from our permanent premises without making the temporary assumption. What is this? Simply ‘A ⊃ C’, that if Adam loves Eve, marriage will ensue.

Presenting this example in English takes a lot of words, but the idea is in fact quite simple. Again, we badly need a streamlined means of representing what is going on. In outline, we have shown that we can establish a conditional of the form X ⊃ Y not on the basis of some premises (or not from premises alone), but on the strength of an argument. We need to write down the argument we used, and, after the whole argument, write down the sentence which the argument establishes. We do it like this:

\[
\begin{array}{|c|}
\hline
3 & A \\
4 & A \supset B \\
5 & B \supset C \\
6 & B & 3, 4, \supset E \\
7 & C & 5, 6, \supset E \\
8 & A \supset C & 3–7, \text{Conditional Introduction (\text{CI})} \\
\hline
\end{array}
\]

For right now, don’t worry about where lines 4 and 5 came from. Focus on the idea that lines 3 through 7 constitute an entire argument, which we call a Subderivation, and the conclusion on line 8 follows from the fact that we have validly derived ‘C’ from ‘A’. A subderivation is always an integral part of a larger, or Outer Derivation. Now you can see why I have been using the vertical scope lines. We must keep outer derivations and subderivations separated. A continuous line to the left of a series of sentences indicates to you what pieces hang together as a derivation. A derivation may have premises, conclusions, and subderivations, which are full-fledged derivations in their own right.

A subderivation can provide the justification for a new line in the outer derivation. For the other rules we have learned, a new line was justified by applying a rule to one or two prior lines. Our new rule, conditional introduction (\text{CI}), justifies a new line, in our example, by appealing to a whole subderivation, 3–7 in our example. When a rule applies to two prior lines, we list the line numbers separated by commas—in the example line 6 is licensed by applying \supset E to lines 3 and 4. But when we justify a new line (8 in our example) by applying a rule (here, \text{CI}) to a whole subderivation, we cite the whole subderivation by writing down its inclusive lines numbers (3–7 in our example).

Now, where did lines 4 and 5 come from in the example, and why did I start numbering lines with 3? I am trying to represent the informal example about Adam and Eve, which started with the real premises that if Adam loves Eve, Adam will propose (A ⊃ B), and that if Adam proposes, they will marry (B ⊃ C). These are premises in the original, outer derivation, and I am free to use them anywhere in the following argument, including in any subderivation which forms part of the main argument. Thus the whole derivation looks like this:

\[
\begin{array}{|c|}
\hline
1 & A \supset B \\
2 & B \supset C \\
3 & A & \text{Assumption (A)} \\
4 & A \supset B & 1, \text{Reiteration (R)} \\
5 & B \supset C & 2, \text{Reiteration (R)} \\
6 & B & 3, 4, \supset E \\
7 & C & 5, 6, \supset E \\
8 & A \supset C & 3–7, \text{Conditional Introduction (\text{CI})} \\
\hline
\end{array}
\]
I am licensed to enter lines 4 and 5 in the subderivation by the rule:

**Reiteration:** If a sentence occurs, either as a premise or as a conclusion in a derivation, that sentence may be copied (reiterated) in any of that derivation's lower subderivations, or lower down in the same derivation.

In the present example, ‘\( A \supset B \)’ and ‘\( B \supset C \)’ are assumed as premises of the whole argument, which means that everything that is supposed to follow is shown to be true only on the assumption that these original premises are true. Thus we are free to assume the truth of the original premises anywhere in our total argument. Furthermore, if we have already shown that something follows from our original premises, this conclusion will be true whenever the original premises are true. Thus, in any following subderivation, we are free to use any conclusions already drawn.

At last I can give you the full statement of what got us started on this long example: the rule of **Conditional Introduction.** We have been looking only at a very special example. The same line of thought applies whatever the details of the subderivation. In the following schematic presentation, what you see in the box is what you must have in order to apply the rule of conditional introduction. You are licensed to apply the rule when you see something which has the form of what is in the box. What you see in the circle is the conclusion which the rule licenses you to draw.

**Conditional Introduction**

<table>
<thead>
<tr>
<th>( X ) Assumption (A)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>( X \supset Y ) Conditional Introduction (( \supset ))</td>
</tr>
</tbody>
</table>

In words: If you have, as part of an outer derivation, a subderivation with assumption \( X \) and final conclusion \( Y \), then \( X \supset Y \) may be entered below the subderivation as a further conclusion of the outer derivation. The subderivation may use any previous premise or conclusion of the outer derivation, entering these with the reiteration rule.

You will have noticed that the initial sentences being assumed in an outer, or main, derivation get called “premises,” while the initially assumed sentence in a subderivation gets called an “assumption.” This is because the point of introducing premises and assumptions is slightly different. While we are arguing, we appeal to premises and assumptions in exactly the same way. But premises always stay there. The final conclusion of the outer derivation is guaranteed to be true only in those cases in which the premises are true. But an assumption introduced in a subderivation gets discharged.

This is just a new word for what we have been illustrating. The point of the subderivation, beginning with assumption \( X \) and ending with final conclusion \( Y \), is to establish \( X \supset Y \) as part of the outer derivation. Once the conclusion, \( X \supset Y \), has been established and the subderivation has been ended, we say that the assumption, \( X \), has been discharged. In other words, the scope line which marks the subderivation signals that we may use the subderivation’s special assumption only within that subderivation. Once we have ended the subderivation (indicated with the small stroke at the bottom of the vertical line), we are not, in the outer derivation, subject to the restriction that \( X \) is assumed to be true. If the premises of the original derivation are true, \( X \supset Y \) will be true whether \( X \) is true or not.

It’s very important that you understand why this last statement is correct, for understanding this amounts to understanding why the rule for conditional introduction works. Before reading on, see if you can state for yourself why, if the premises of the original derivation are true, and there is a subderivation from \( X \) as assumption to \( Y \) as conclusion, \( X \supset Y \) will be true whether or not \( X \) is true.

The key is the truth table definition of \( X \supset Y \). If \( X \) is false, \( X \supset Y \) is, by definition, true, whatever the truth value of \( Y \). So we only have to worry about cases in which \( X \) is true. If \( X \) is true, then for \( X \supset Y \) to be true, we need \( Y \) to be true also. But this is just what the subderivation shows: that for cases in which \( X \) is true, \( Y \) is also true. Of course, if the subderivation used premises from the outer derivation or used conclusions that followed from those premises, the subderivation only shows that in all cases in which \( X \) and the original premises are true, \( Y \) will also be true. But then we have shown that \( X \supset Y \) is true, not in absolutely all cases, but in at least those cases in which the original premises are true. But that’s just right, since we are entering \( X \supset Y \) as a conclusion of the outer derivation, subject to the truth of the original premises.

**EXERCISES**

5–2. Again, for each of the following arguments, provide a derivation which shows the argument to be valid. Be sure to number and annotate each step to show its justification. All of these exercises will
require you to use conditional introduction and possibly other of the rules you have already learned. You may find the use of conditional introduction difficult until you get accustomed to it. If so, don’t be alarmed, we’re going to work on it a lot. For these problems you will find the following strategy very helpful: If the final conclusion which you are trying to derive (the “target conclusion”) is a conditional, set up a subderivation which has as its assumption the antecedent of the target conclusion. That is, start your outer derivation by listing the initial premises. Then start a subderivation with the target conclusion’s antecedent as its assumption. Then reiterate your original premises in the subderivation and use them, together with the subderivation’s assumptions, to derive the consequent of the target conclusion. If you succeed in doing this, the rule of conditional introduction licenses drawing the target conclusion as your final conclusion of the outer derivation.

\[ \begin{align*}
\text{a) } & \quad B \implies C \\
\text{b) } & \quad \neg B \quad \neg C \\
\text{c) } & \quad \neg \neg B \\
\text{d) } & \quad \neg B \quad \neg C \\
\text{e) } & \quad (B \lor C) \quad \neg C \\
\text{f) } & \quad A \lor B \\
\text{g) } & \quad A \lor B \\
\text{h) } & \quad (A \lor B) \lor C \\
\text{i) } & \quad (A \lor B) \lor C \\
\text{j) } & \quad (A \lor B) \lor C \\
\text{k) } & \quad (A \lor B) \lor C \\
\text{l) } & \quad (A \lor B) \lor C \\
\text{m) } & \quad (A \lor B) \lor C \\
\text{n) } & \quad (A \lor B) \lor C \\
\text{o) } & \quad (A \lor B) \lor C \\
\text{p) } & \quad (A \lor B) \lor C \\
\text{q) } & \quad (A \lor B) \lor C \\
\end{align*} \]

5–3. THE COMPLETE RULES OF INFERENCE

We now have in place all the basic ideas of natural deduction. We need only to complete the rules. So that you will have them all in one place for easy reference, I will simply state them all in abbreviated form and then comment on the new ones. Also, I will now state all of the rules using the same format. For each rule I will show a schematic derivation with one part in a box and another part in a circle. In the box you will find, depending on the rule, either one or two sentence forms or a subderivation form. In the circle you will find a sentence form. To apply a given rule in an actual derivation, you proceed as follows: You look to see whether the derivation has something with the same form as what’s in the box. If so, the rule licenses you to write down, as a new conclusion, a sentence with the form of what’s in the circle.

\[ \begin{align*}
\text{Conjunction Introduction} & \quad \text{Conjunction Elimination} \\
\text{Disjunction Introduction} & \quad \text{Disjunction Elimination} \\
\text{Conditional Introduction} & \quad \text{Conditional Elimination} \\
\text{Biconditional Introduction} & \quad \text{Biconditional Elimination} \\
\end{align*} \]
In interpreting these schematic statements of the rules, you must remember the following: When a rule applies to two sentences, as in the case of conjunction introduction, the two sentences can occur in either order, and they may be separated by other sentences. The sentences to which a rule applies may be premises, an assumption, or prior conclusions, always of the same derivation, that is, lying along the same scope line. Also, the sentence which a rule licenses you to draw may be written anywhere below the licensing sentences or derivation, but as part of the same derivation, again, along the same scope line.

Conjunction introduction and elimination are so simple we rarely bother to mention them when we argue informally. But to be rigorous and complete, our system must state and use them explicitly. Conjunction introduction states that when two sentences, X and Y, appear in a derivation, in either order and whether or not separated by other sentences, we may conclude their conjunction, X&Y, anywhere below the two conjuncts. Conjunction introduction just tells us that if a conjunction of the form X&Y appears on a derivation, we may write either conjunct (or both, on different lines) anywhere lower down on the derivation. We have already discussed the rules for disjunction and the conditional. Here we need only add that in the elimination rules, the sentences to which the rules apply may occur in either order and may be separated by other sentences. For example, when applying disjunction elimination, the rule applies to sentences of the form X∨Y and ~X, in whatever order those sentences occur and whether or not other sentences appear between them.

Biconditional introduction and elimination really just express the fact that a biconditional of the form X=Y is logically equivalent to the conjunction of sentences of the form X⇒Y and Y⇒X. If the two conditionals appear on a derivation, whatever the order, and whether or not separated by other sentences, we may write the biconditional lower down as a conclusion. Conversely, if a biconditional of the form X=Y appears, one may write lower down, as a conclusion, X⇒Y, Y⇒X, or both (on separate lines).

Note that negation elimination licenses dropping a double negation, and is justified by the fact that X is always logically equivalent to ~~X.

Negation introduction requires some comment. Once again, natural deduction seeks to capture and make precise conventional forms of informal argument. This time we express what traditionally goes under the name of "reductio ad absurdum," or "reduction to the absurd." Here the idea is that if we begin with an assumption from which we can deduce a contradiction, the original assumption must be false. Natural deduction employs this strategy as follows: Begin a subderivation with an assumption, X. If one succeeds in deriving both a sentence of the form Y and its negation, ~Y, write the sentence of the form ~X as a conclusion of the outer derivation anywhere below the subderivation.

As with the other rules, you should be sure you understand why this rule works. Suppose in a subderivation we have drawn the conclusions Y and ~Y from the assumption X. This is (by the rules for conjunction) equivalent to deriving the contradiction Y&~Y from X. Now, X must be either true or false. If it is true, and we have drawn from it the conclusion that Y&~Y, we have a valid argument from a true premise to a false conclusion. But that can't happen—our rules for derivations won't let that happen. So X must have been false, in which case ~X must be true and can be entered as a conclusion in the outer derivation. Finally, if the subderivation has used premises or conclusions of the outer derivation, we can reason in exactly the same way, but subject to the restriction that we consider only cases in which the original premises were true.

In annotating negation introduction, keep in mind the same consideration which applied in annotating conditional introduction. The new line is justified by appeal, not to any one or two lines, but to a whole argument, represented by a subderivation. Consequently, the justification for the new line appeals to the whole subderivation. Indicate this fact by writing down the inclusive line numbers of the subderivation (the first and last of its line numbers separated by a dash).

In applying these rules, be sure to keep the following in mind: To apply the rules for conditional and negation introduction, you must always have a completed subderivation of the form shown. It's the presence of the subderivation of the right form which licenses the introduction of a conditional or a negated sentence. To apply any of the other rules, you must have the input sentence or sentences (the sentence or sentences in the box in the rule's schematic statement) to be licensed to write the output sentence of the rule (the sentence in the circle in the schematic pre-
sensation). But an input sentence can itself be either a prior conclusion in the derivation or an original premise or assumption.

Incidentally, you might have been puzzled by the rule for negation introduction. The rule for negation elimination has the form "\(~X\). Therefore X". Why not, you might wonder, use the rule "X. Therefore \(~X\)" for negation introduction? That's a good question. The rule "X. Therefore \(~X\)" is a correct rule in the sense that it is truth preserving. It will never get you a false sentence out of true ones. But the rule is not strong enough. For example, given the other rules, if you restrict yourself to the rule "X. Therefore \(~X\)" for negation introduction, you will never be able to construct a derivation that shows the argument

\[
\frac{\neg A}{\neg (A \land B)}
\]

to be valid. We want our system of natural deduction not only to be Sound, which means that every derivation represents a valid argument. We also want it to be Complete, which means that every valid argument is represented by a derivation. If we use the rule "X. Therefore \(~X\)" for negation introduction, our system of natural deduction will not be complete. The rules will not be strong enough to provide a correct derivation for every valid argument.

**EXERCISES**

5–3. Below you find some correct derivations without the annotations which tell you, for each line, which rule was used in writing the line and to which previous line or lines the rule appeals. Copy the derivations and add the annotations. That is, for each line, write the line number of the previous line or lines and the rule which, applying to those previous lines, justifies the line you are annotating.

a) 1 \[ B \land (B \lor A) \] p 2 \[ B \] 3 \[ B \lor A \] 4 \[ \neg A \] 5 \[ \neg C \equiv (A \lor B) \] 6 \[ A \equiv B \] 7 \[ B \lor C \] 8 \[ A \lor C \] 9 \[ A \] 10 \[ A \equiv B \]

b) 1 \[ \neg C \equiv (A \lor B) \] p 2 \[ A \] 3 \[ (A \lor B) \lor C \] 4 \[ A \lor C \] 5 \[ B \lor C \] 6 \[ C \]

c) 1 \[ A \lor B \] p 2 \[ B \lor C \] 3 \[ A \]

5–4. For each of the following arguments, provide a derivation which shows the argument to be valid. Follow the same directions as you did for exercises 5–1 and 5–2.

a) \[ (C \land \neg H) \vdash \neg H \]
b) \[ (A \land B) \vdash \neg A \]
c) \[ (A \land B) \vdash \neg A \]
d) \[ (A \land B) \vdash \neg A \]
e) \[ (A \land B) \vdash \neg A \]
f) \[ (A \land B) \vdash \neg A \]
g) [other arguments provided but not transcribed here]
In chapter 3 we defined triple conjunctions and disjunctions, that is, sentences of the form $X & Y & Z$ and $X \lor Y \lor Z$. Write introduction and elimination rules for such triple conjunctions and disjunctions.

5–6. Suppose we have a valid argument and an assignment of truth values to sentence letters which makes one or more of the premises false. What, then, can we say about the truth value of the conclusions which follow validly from the premises? Do they have to be false? Can they be true? Prove what you say by providing illustrations of your answers.

### CHAPTER SUMMARY EXERCISES

Give brief explanations for each of the following, referring back to the text to make sure your explanations are correct and saving your answers in your notebook for reference and review.

- a) Derivation
- b) Subderivation
- c) Outer Derivation
- d) Scope Line
- e) Premise
- f) Assumption
- g) Rule of Inference
- h) License (to draw a conclusion)
- i) Truth Preserving Rule
- j) Discharging an Assumption
- k) Conjunction Introduction
- l) Conjunction Elimination
- m) Disjunction Introduction
- n) Disjunction Elimination
- o) Conditional Introduction
- p) Conditional Elimination
- q) Biconditional Introduction
- r) Biconditional Elimination
- s) Negation Introduction
- t) Negation Elimination
- u) Reiteration
6-1. CONSTRUCTING CORRECT DERIVATIONS

Knowing the rules for constructing derivations is one thing. Being able to apply the rules successfully is another. There are no simple mechanical guidelines to tell you which rule to apply next, so constructing derivations is a matter of skill and ingenuity. Long derivations can be extremely difficult. (It's not hard to come up with problems which will stump your instructor!) At first, most students feel they don't even know how to get started. But with a bit of practice and experience, you will begin to develop some intuitive skill in knowing how to organize a derivation. To get you started, here are some examples and practical strategies.

Usually you will be setting a problem in the following form: You will be given some premises and a conclusion. You will be told to prove that the conclusion follows validly from the premises by constructing a derivation which begins with the given premises and which terminates with the given conclusion. So you already know how your derivation will begin and end.

Your job is to fill in the intermediate steps so that each line follows from previous lines by one of the rules. In filling in, you should look at both the beginning and the end of the derivation.
Let's illustrate this kind of thinking with a simple example. Suppose you are asked to derive ‘B&C’ from the premises ‘A&B’, ‘A>C’, and ‘A’. Right off, you know that the derivation will take the form

\[
\begin{array}{c|c|c|c|c|c}
1 & A&B & P \\
2 & A>C & P \\
3 & A & P \\
\hline
4 & & & & B&C \\
\end{array}
\]

where you still have to figure out what replaces the question marks.

First, look at the conclusion. It is a conjunction, which can most straightforwardly be introduced with the rule for &I. (From now on, I'm going to use the shorthand names of the rules.) What do you need to apply that rule? You need ‘B’ and you need ‘C’. So if you can derive ‘B’ and ‘C’, you can apply &I to get ‘B&C’. Can you derive ‘B’ and ‘C’? Look at the premises. Can you get ‘B’ out of them? Yes, by applying &E to lines 1 and 3. Similarly, you can derive ‘C’ from lines 2 and 3. Altogether, the derivation will look like this:

\[
\begin{array}{c|c|c|c|c|c|c|c|c}
1 & A&B & P \\
2 & A>C & P \\
3 & A & P \\
\hline
4 & B & 1, 3, &E \\
5 & C & 2, 3, &E \\
6 & B&C & 4, 5, &I \\
\end{array}
\]

Let's try a slightly harder example. Suppose you are asked to derive ‘C>B’ from the premises ‘AvB’ and ‘C>B’. Your target conclusion is a conditional. Well, what rule allows you to conclude a conditional? >I. So you will try to set things up so that you can apply >I. This will involve starting a subderivation with ‘C’ as its assumption, in which you will try to derive ‘A’. In outline, the derivation you are hoping to construct can be expected to look like this:

\[
\begin{array}{c|c|c|c|c|c|c|c|c}
1 & AvB & P \\
2 & C>B & P \\
\hline
3 & C & A \\
\hline
4 & ? & ? \\
5 & ? & A \\
6 & C>B & A \\
\end{array}
\]

(Your derivation won't have to look like this. In every case there is more than one correct derivation of a conclusion which follows from a given set of premises. But in this case, this is the obvious thing to try, and it provides the simplest correct derivation.)

To complete the derivation, you must fill in the steps in the subderivation to show that (given the premises of the outer derivation) ‘A’ follows from ‘C’.

How will you do this? Let's study what you have available to use. In the subderivation you are allowed to use the subderivation's assumption and also any previous premise or conclusion in the outer derivation. Notice that from ‘C’ and the premise ‘C>B’ you can get ‘~B’ by &E. Is that going to do any good? Yes, for you can then apply &E to ‘~B’ and the premise ‘AvB’ to get the desired ‘A’. All this is going to take place in the subderivation, so you will have to reiterate the premises. The completed derivation looks like this:

\[
\begin{array}{c|c|c|c|c|c|c|c|c}
1 & AvB & P \\
2 & C>B & P \\
\hline
3 & C & A \\
\hline
4 & C>B & 2, R \\
5 & ~B & 3, 4, &E \\
6 & AvB & 1, R \\
7 & A & 5, 6, &vE \\
8 & C>B & 3-7, &I \\
\end{array}
\]

If you are still feeling a little lost and bewildered, reread the text from the beginning of this section.
When you have understood the examples given so far, you are ready for something new. Let’s try to derive ‘A→¬B’ from ‘B→¬A’. As in the second example, our first effort to derive a conditional should be by using ∨I. So we want a subderivation with ‘A’ as assumption and ‘¬B’ as final conclusion:

```
1  B→¬A  P
2  A   A
    ?
    ?
  ¬B
```

A→¬B  ∨I

But how can we get ‘¬B’ from the assumption of ‘A’, using the premise of the outer derivation?

‘¬B’ is the negation of the sentence ‘B’. Unless there is some really obvious simple alternative, one naturally tries to use ¬I. ¬I works by starting a subderivation with the sentence to be negated as assumption and then deriving some sentence and its negation. In the present situation this involves something that might not have occurred to you, namely, creating a subderivation of a subderivation. But that’s fine. All the rules for working on a derivation apply to subderivations also, including the creation of subderivations. The only difference between a subderivation and a derivation is that a subderivation ends when we discharge its assumption, returning to its outer derivation; and that in a subderivation we may reiterate prior premises or conclusions from an outer derivation (or from any outer-outer derivation, as you will see in a moment). This is because in a subderivation we are working under the assumption that all outer assumptions and premises are true.

Will this strategy work? Before writing anything down, let me illustrate the informal thinking you should go through to see whether a strategy promises to be successful. Look back at the outline we have already written of how we hope the derivation will look. We are proposing to start a sub-sub-derivation with the new assumption ‘B’. That sub-sub-derivation can use the original premise ‘B→¬A’, which, together with the assumption ‘B’, will give ‘¬A’ by ∨E. But the sub-sub-derivation is also within its outer derivation beginning with the assumption of ‘A’. So ‘A’ is also being assumed in the sub-sub-derivation, which we express by reiterating ‘A’ in the sub-sub-derivation. The sub-sub-derivation now has both ‘A’ and ‘¬A’, which constitutes the contradiction we needed:

```
1  B→¬A  P
2  A   A
    ?
    ?
  ¬B
```

A→¬B  ∨I

```
3  B     1, R
4  ¬A    3, 4, ∨E
5   A     2, R
6  ¬B    3–6, ¬I
7  A→¬B  2–7, ∨I
```

How are you doing? If you have had trouble following, rest for a moment, review to be sure you have gotten everything up to this point, and then we’ll try something one step more involved.

Let’s try deriving ‘A≡¬B’ from ‘AvB’ and ‘¬(A&B)’. The conclusion is a biconditional, and one derives a biconditional most easily by using ≡I. Think of a biconditional as the conjunction of two conditionals, the two conditionals we need to derive the biconditional using ≡I. So you should aim to develop a derivation along these lines:

```
1  AvB  P
2  ¬(A&B)  P
    ?
    ?
  ¬B
```

A→¬B  ∨I

```
3  B     1, R
4  ¬A    3, 4, ∨E
5   A     2, R
6  ¬B    3–6, ¬I
7  A→¬B  2–7, ∨I
```

We have reduced the complicated problem of deducing ‘A≡¬B’ to the simpler problems of deducing ‘¬B→A’ and ‘A→¬B’.

In constructing derivations, you should learn to think in this kind of
pattern. Try to resolve the problem of deriving the final conclusion (your target conclusion) by breaking it down into simpler problems of deriving simpler sentences (your new target conclusions). You may actually need to resolve your simpler problems into still more simple problems. You continue working backward toward the middle until you can see how to derive your simple sentences from the premises. At this point you start working from the premises forward and fill everything in.

How, in this example, can we derive our simplified new target conclusions? They are both conditionals, and as we saw in the second example, the straightforward way to derive conditionals uses \( \rightarrow I \). This involves starting one subderivation for each of the conditionals to be derived:

1. \( \text{AvB} \)
2. \( \neg(\text{A&B}) \)

\[
\begin{align*}
&\quad \neg B \\
&\quad A \\
&\quad \newline \\
&\quad \neg B \lor A \\
&\quad \newline \\
&\quad \neg B
\end{align*}
\]

We have now resolved our task into the problem of filling in the two subderivations.

Can you see how to complete the subderivations by working with the premises of the outer derivation? The first subderivation is easy: `\( \neg B \)` and `\( \text{AvB} \)` give `\( A \)` by \( \lor E \). The second subderivation presents more of a challenge. But we can complete it by using the same tactics illustrated in the previous example. We've assumed `\( A \)` and want to get `\( \neg B \)`. To get `\( \neg B \)`, we can try `\( \neg I \)` (unless a really simple alternative suggests itself). `\( \neg I \)` will require us to start a sub-sub-derivation with `\( B \)` as assumption. In this sub-sub-derivation we can reiterate anything which is above in an outer derivation of the derivation on which we are working. So we can reiterate `\( A \)`,
No rule applies immediately to the premises to give ‘C’. Because ‘C’ is atomic, no introduction rule for a connective will give ‘C’. What on earth can you do?

Sometimes when you are stuck, you can succeed by arranging to use ~I in what I am going to call the reductio ad absurdum strategy. This strategy proceeds by assuming the negation of what you want and then from this assumption (and prior premises and conclusions) deriving a contradiction. As you will see in the example, you will then be able to apply ~I to derive the double negation of what you want, followed by ~E to get rid of the double negation. In outline, the reductio ad absurdum strategy, applied to this problem, will look like this:

1 | A&B  
2 | ~C~B  
3 | X  
   | ~X
   | ~C
   | ~I
   | C

Will this strategy work in this case? If you assume ‘~C’, you will be able to use that assumption with the premise ‘~C~B’ to get ‘~B’. But ‘~B’ will contradict the ‘B’ in the premise ‘A&B’, and you can dig ‘B’ out of ‘A&B’ with &E. In sum, from ‘~C’ and the premises you will be able to derive both ‘B’ and ‘~B’, ~I then allows you to conclude ‘~~C’ (the negation of the assumption which led to the contradiction). ~E finally gives ‘C’:

1 | A&B  
2 | ~C~B  
3 | ~C
   | A
4 | ~C~B
5 | ~B  
6 | A&B
7 | B
8 | ~C
9 | C

The first time you see an example like this it may seem tricky. But you will soon get the hang of it.

You do need to be a little cautious in grasping at the reductio strategy when you are stuck. Often, when students have no idea what to do, they assume the opposite of what they want to conclude and then start blindly applying rules. This almost never works. To use the reductio strategy successfully, you need to have a more specific plan. Ask yourself: “Can I, by assuming the opposite of what I want to derive, get a contradiction (a sentence and its negation) out of the assumption?” If you can see how to do this, you are all set, and you can start writing down your derivation. If you think you see a way which might work, it may be worth starting to write to clarify your ideas. But if you have no idea of how you are going to get a contradiction out of your assumption, go slow. Spend a little time brainstorming about how to get a contradiction. Then, if you find you are getting nowhere, you may well need to try an entirely different approach to the problem.

I should also comment on the connection between what I have called the reductio ad absurdum strategy and the rule for ~I. They really come to pretty much the same thing. If you need to derive a sentence of the form ~X, consider assuming X, trying to derive a contradiction, and applying ~I to get ~X. To derive a sentence of the form X, assume ~X, and derive ~~X by ~I. Then eliminate the double negation with ~E.

**EXERCISES**

6–1. For each of the following arguments, provide a derivation, complete with annotations, which shows the argument to be valid. If you find you are having difficulty with these problems, go over the examples earlier in this chapter and then try again.

a) Kv~I   b) ~C~A   c) ~D~K   d) ~F~G
   ~B~A   B   K   D~E
   ~(~K~l)
   B
   C
   D&H

f) AvB   e) A~B
   B&C
   ~A&B
   ~CvD
   ~DDA
6–2. RECOGNIZING THE MAIN CONNECTIVE

Suppose you are asked to provide a derivation which shows the following argument to be valid:

\[
\begin{align*}
(1) & \quad (\neg B \land C = (\neg B)) \\
(2) & \quad C
\end{align*}
\]

The premise is a mess! How do you determine which rule applies to it? After your labor with some of the exercises in the last chapter, you probably can see that the key lies in recognizing the main connective. Even if you got all of those exercises right, the issue is so important that it’s worth going over from the beginning.

Let’s present the issue more generally. When I stated the rules of inference, I expressed them in general terms, using boldface capital letters, ‘X’ and ‘Y’. For example, the rule for &E is:

\[
\begin{align*}
(3) & \quad \begin{align*}
& \quad X & \quad \text{and} & \quad \begin{align*}
& \quad Y
\end{align*}
\end{align*}
\]

The idea is that whenever one encounters a sentence of the form X&Y in a derivation, one is licensed to write either the sentence X or the sentence Y (or both on separate lines) further down. Focus on the fact that this is so whatever sentences X and Y might be. This is the point of using boldface capital letters in the presentation. X and Y don’t stand for any particular sentences. Rather, the idea is that if you write any sentence you want for X and any sentence you want for Y, you will have a correct instance of the rule for &E. This is what I mean by saying that I have expressed the rule “in general terms” and by talking about a sentence “of the form X&Y”.

How will these facts help you to deal with sentence (1)? Here’s the technique you should use if a sentence such as (1) confuses you. Ask yourself: “How do I build this sentence up from its parts?” You will be particularly interested in the very last step in putting (1) together from its parts. In this last step you take the sentence:

\[
(4) \quad \neg B \quad \text{which you can think of as } X
\]

and the sentence

\[
(5) \quad C = (\neg B) \quad \text{which you can think of as } Y
\]

The sentence

\[
(6) \quad C = (\neg B)
\]

ask yourself “What is the last thing I do to build this sentence up from its parts?” You take the sentence

\[
(7) \quad C \quad \text{which you can think of as } X
\]

and you put them on either side of a biconditional, ‘=’, giving

\[
(8) \quad \neg B \quad \text{which you can think of as } Y
\]

Consequently, if you find sentence (6), you can apply the rule of inference for =E:

\[
\begin{align*}
(9) & \quad C = (\neg B) \quad \text{which thus has the form } X = Y
\end{align*}
\]

Thus =E applies to sentence (6), licensing us to follow (6) on a derivation either with the sentence ‘C = (\neg B)’ or the sentence ‘(\neg B) = C’, or both on separate lines.

In a moment we will apply what we have just done to provide a deri-
Natural Deduction for Sentence Logic

The last thing you do in putting this one together from its parts is to put '(A>B)' and 'C' on either side of a conditional, '→'. So the sentence has the form X→Y, with 'A→B' as X and 'C' as Y. If we have 'A→B' as well as '(A→B)→C' in a derivation, we can apply →E to the two to derive 'C'.

Perhaps you can now see how we can write a derivation which shows how (2) follows from (1). But we will need to see how to treat one more compound sentence. This time, try to figure it out for yourself. What is the form of the sentence '(A→B)→C'? The key idea you need to untangle a sentence such as (1) is that of a main connective:

The Main Connective in a sentence is the connective which was used last in building up the sentence from its component or components.

(A negated sentence, such as '~(A∨B)', has just one component, 'A∨B' in this case. All other connectives use two components in forming a sentence.) Once you see the main connective, you will immediately spot the component or components to which it has been applied (so to speak, the X and the Y), and then you can easily determine which rules of inference apply to the sentence in question.

Let's practice with a few examples:

<table>
<thead>
<tr>
<th>SENTENCE</th>
<th>MAIN CONNECTIVE</th>
<th>COMPONENT OR COMPONENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A∨B)→(B&amp;D)</td>
<td>∨</td>
<td>AvB and ~(B&amp;D)</td>
</tr>
<tr>
<td>[(A→B)∨(A=D)]</td>
<td>v</td>
<td>(A→B)∨D and ~A=D</td>
</tr>
<tr>
<td>~[(A∧B)∨(B¬A)]</td>
<td>~</td>
<td>(A∧B)∨(B¬A)</td>
</tr>
</tbody>
</table>

The second and third examples illustrate another fact to which you must pay attention. In the second example, the main connective is a '∨'. But which occurrence of '∨'? Notice that the sentence uses two '∨'s, and not both occurrences count as the main connective! Clearly, it is the second occurrence, the one used to put together the components '(A→B)∨D' and '~A=D', to which we must pay attention. Strictly speaking, it is an occurrence of a connective which counts as the main connective. It is the occurrence last used in putting the sentence together from its parts. In the third example, '~' occurs three times! Which occurrence counts as the main connective? The very first.

In the following exercises you will practice picking out the main connective of a sentence and determining which rule of inference applies. But let's pause first to say more generally how this works:

The elimination rule for '&' applies to a sentence only when an '&' occurs as the sentence's main connective. The same thing goes for '∨', '→', and '='. The components used with the main connective are the components to which the elimination rule makes reference.

The elimination rule for '~' applies only to a doubly negated sentence, '~~X'; that is, only when '~' is the sentence's main connective, and the '~' is applied to a component, ~X, which itself has a '~' as its main connective.

The introduction rule for '&' licenses you to write as a conclusion a sentence, the main connective of which is '&'. The same thing goes for '∨', '→', '=', and '~'.

**EXERCISES**

6–2. Give derivations which establish the validity of the following arguments:

- a) (AvB)∧[(AvB)→C]  b) A
  - A)
  - (AvB)=[(A∧K)∧(B∧K)]
  - C
  - (AvB)=[(A∧K)∧(B∧K)]
  - K
  - (B∧A)
6-3. DERIVATIONS: OVERVIEW, DEFINITIONS, AND POINTS TO WATCH OUT FOR

This chapter and chapter 5 have described, explained, and illustrated derivations. Let's pull these thoughts together with some explicit definitions and further terminology.

A Rule of Inference tells when you are allowed, or Licensed, to draw a conclusion from one or more sentences or from a whole argument (as represented by a subderivation).

A Derivation is a list of which each member is either a sentence or another derivation. If a first derivation has a second derivation as one of the first derivation's parts, the second derivation is called a Subderivation of the first and the first is called the Outer Derivation of the second. Each sentence in a derivation is a premise or assumption, or a reiteration of a previous sentence from the same derivation or an outer derivation, or a sentence which follows by one of the rules of inference from previous sentences or subderivations of the derivation.

In practice, we always list the premises or assumptions of a derivation at its beginning, and use a horizontal line to separate them from the further sentences which follow as conclusions. What's the difference between premises and assumptions? From a formal point of view, there is no difference at all, in the sense that the rules of inference treat premises and assumptions in exactly the same way. In practice, when an unargued sentence is assumed at the beginning of the outermost deduction, we call it a premise. When an unargued sentence is assumed at the beginning of a subderivation, we call it an assumption. The point is that we always terminate subderivations before the end of a main derivation, and when we terminate a subderivation, in some sense we have gotten rid of, or Discharged, the subderivation's assumptions.

To make these ideas clearer and more precise, we have to think through what the vertical lines, or Scope Lines, are doing for us?

A Scope Line tells us what sentences and subderivations hang together as a single derivation. Given a vertical scope line, the derivation it marks begins where the line begins and ends where the line ends. The derivation marked by a scope line includes all and only the sentences and subderivations immediately to the right of the scope line.

To help sort out these definitions, here is a schematic example:

```
Q
R
S

\[\begin{array}{c}
| \text{T} \\
| \text{U} \\
\end{array}\]

\[\begin{array}{c}
| \text{V} \\
| \text{W} \\
\end{array}\]

\[\begin{array}{c}
| \text{X} \\
\end{array}\]

\[\begin{array}{c}
| \text{Y} \\
\end{array}\]

\[\begin{array}{c}
| \text{Z} \\
\end{array}\]
```

Notice that at the bottom of each scope line I have written a number to help us in talking about the different component derivations. Consider first the main derivation, derivation 1, marked with the leftmost scope line numbered '1' at its bottom. Derivation 1 includes premises Q and R and has a first conclusion S, other conclusions not explicitly shown, indicated by . . . , and the final conclusion Z. Derivation 1 also includes two subderivations, derivations 2 and 3. Derivation 2 has assumption T, various conclusions not explicitly indicated (again signaled by . . . ), and final conclusion U. Derivation 3 starts with assumption V, has final conclusion Y, and includes a subderivation of its own, derivation 4.
This organization serves the purpose of keeping track of what follows from what. In the outermost derivation 1, all the conclusions of the derivation (S . . . Z) follow from the derivation's premises, Q and R. This means that every assignment of truth values to sentence letters which makes the premises Q and R true will make all the conclusions of derivation 1 true. But the conclusions of a subderivation hold only under the subderivation's additional assumption. For example, the conclusion U of subderivation 2 is subject to the assumption T as well as the premises Q and R. This means that we are only guaranteed that any assignment of truth values to sentence letters which makes Q, R, and T all true will make U true also. In other words, when we start a subderivation, we add an additional assumption which is assumed in effect just in the subderivation. Any premises or assumptions from outer derivations also apply in the subderivation, since they and their consequences can be reiterated into the subderivation.

You should particularly notice that when a subderivation has ended, its special assumption is no longer assumed. It is not assumed in any conclusions drawn as part of the outer derivation, nor is it assumed as part of a new subderivation which starts with a different assumption. Thus the truth of T is not being assumed anywhere in derivation 1, 3, or 4. This is what we mean by saying that the assumption of a subderivation has been discharged when the subderivation is terminated.

These facts are encoded in the reiteration rule which we can now spell out more clearly than before. The reiteration rule spells out the fact that a subderivation assumes the truth, not only of its own assumption, but of the prior assumptions, premises, and conclusions of any outer derivation. Thus, in subderivation 2, reiteration permits us to write, as part of 2, Q, R, S, or any other conclusion of 1 which appears before 2 starts. This is because inside 2, we assume that the premises of outer derivation 1 are true. And because whenever the premises are true, conclusions which follow from them are true, we may also use the truth of any such conclusions which have already followed from these premises.

But we cannot reiterate a sentence of 2 in, for example, 1. This is because when we end subderivation 2 we have discharged its premise. That is, we are no longer arguing under the assumption that the assumption of 2 is true. So, for example, it would be a mistake to reiterate U as part of 1. U has been proved only subject to the additional assumption T. In 1, T is not being assumed. In the same way, we cannot reiterate U as part of 3 or 4. When we get to 3, subderivation 2 has been ended. Its special assumption, T, has been discharged, which is to say that we no longer are arguing under the assumption of T.

Students very commonly make the mistake of copying a conclusion of a subderivation, such as U, as a conclusion of an outer derivation—in our schematic example, listing U as a conclusion in derivation 1 as well as in subderivation 2. I'll call this mistake the mistake of hopping scope lines.

Don't hop scope lines!

We can, however, reiterate Q, R, S, or any prior conclusion in 1 within sub-sub-derivation 4. Why? Because 4 is operating under its special assumption, W, as well as all the assumptions and premises of all derivations which are outer to 4. Inside 4 we are operating under all the assumptions which are operative in 3, which include not only the assumption of 3 but all the premises of the derivation of which 3 is a part, namely, 1. All this can be expressed formally with the reiteration rule, as follows: To get a premise or prior conclusion of 1 into 4, first reiterate the sentence in question as part of 3. Now that sentence, officially part of 3, can be reiterated again in 4. But we can dispense with the intermediate step.

Incidentally, once you clearly understand the reiteration rule, you may find it very tiresome always to have to explicitly copy the reiterated sentences you need in subderivations. Why, you may wonder, should you not be allowed, when you apply other rules, simply to appeal to prior sentences in outer derivations, that is, to the sentences which the reiteration rule allows you to reiterate? If you fully understand the reiteration rule, you will do no harm in thus streamlining your derivations. I will not use this abbreviation, because I want to be sure that all of my readers understand as clearly as possible how reiteration works. You also should not abbreviate your derivations in this way unless your instructor gives you explicit permission to do so.

Scope lines also indicate the sentences to which we can apply a rule in deriving a conclusion in a derivation or subderivation. Let us first focus on rules which apply only to sentences, that is, rules such as vE or E, which have nothing to do with subderivations. The crucial feature of such a rule is that, if the sentences to which we apply it are true, the conclusion will be true also. Suppose, now, we apply such a rule to the premises Q and R of derivation 1. Then, if the premises are true, so will the rule's conclusion, so that we can write any such conclusion as part of derivation 1. In further application of such rules in reaching conclusions of derivation 1, we may appeal to 1's prior conclusions as well as its premises, since if the premises are true, so will the prior conclusions. In this way we are still guaranteed that if the premises are true, so will the new conclusion.

But we cannot apply such a rule to assumptions or conclusions of a subderivation in drawing conclusions to be made part of derivation 1. For example, we can't apply a rule to sentences S and U in drawing a conclusion which will be entered as a part of derivation 1. Why not? Because we want all the conclusions of 1 to be guaranteed to be true if 1's premises are true. But assumptions or conclusions of a subderivation, say, 2, are only sure to be true if 1's premises and 2's special assumption are true.

In sum, when applying a rule of inference which provides a conclusion
when applied to sentences ("input sentences"), the input sentences must already appear before the rule is applied, and all input sentences as well as the conclusion must appear in the same derivation. Violating this instruction constitutes a second form of the mistake of hopping scope lines.

What about $\supset I$ and $\neg I$, which don't have sentences as input? Both these rules have the form: If a subderivation of such and such a form appears in a derivation, you may conclude so and so. It is important to appreciate that these two rules do not appeal to the sentences which appear in the subderivation. They appeal to the subderivation as a whole. They appeal not to any particular sentences, but to the fact that from one sentence we have derived certain other sentences. That is why when we annotate these rules we cite the whole subderivation to which the rule applies, by indicating the inclusive line numbers of the subderivation.

Consider $\supset I$. Suppose that from $T$ we have derived $U$, perhaps using the premises and prior conclusions of our outer derivation. Given this fact, any assignment of truth values to sentence letters which makes the outer derivation's premises true will also make the conditional $T \supset U$ true. (I explained why in the last chapter.) Thus, given a subderivation like 2 from $T$ to $U$, we can write the conclusion $T \supset U$ as part of the outer derivation 1. If 1's premises are true, $T \supset U$ will surely be true also.

The key point to remember here is that when $\supset I$ and $\neg I$ apply to a subderivation, the conclusion licensed appears in the same derivation in which the input subderivation appeared as a part. Subderivation 2 licenses the conclusion $T \supset Y$ as a conclusion of 1, by $\supset I$; and $\supset I$, similarly applied to derivation 4, licenses concluding $W \supset X$ as part of 3, but not as part of 1.

By this time you may be feeling buried under a pile of details and mistakes to watch out for. Natural deduction may not yet seem all that natural. But, as you practice, you will find that the bits come to hang together in a very natural way. With experience, all these details will become second nature so that you can focus on the challenging task of devising a good way of squeezing a given conclusion out of the premises you are allowed to use.

EXERCISES

6–3. For each of the following arguments, provide a derivation which shows the argument to be valid. If you get stuck on one problem, try another. If you get good and stuck, read over the examples in this chapter, and then try again.

a) $R$  b) $\neg(A\&B)\backslash C$  c) $\neg(Hv\neg D)$  d) $\neg C\&\neg D$

$\frac{\neg C\&\neg D}{R(D)\&(RvK)}$  $\frac{\neg C\&\neg D}{\neg C\&\neg D}$  $\frac{\neg C\&\neg D}{\neg C\&\neg D}$  $\frac{\neg C\&\neg D}{\neg C\&\neg D}$

6–4. Write a rule of inference for the Sheffer stroke, defined in section 3–5.

CHAPTER SUMMARY EXERCISES

This chapter has focused on improving your understanding of material introduced in chapter 5, so there are only a few new ideas. Complete short explanations in your notebook for the following terms. But also go back to your explanations for the terms in the chapter summary exercises for chapter 5 and see if you can now explain any of these terms more accurately and clearly.

a) Reductio Ad Absurdum Strategy
b) Main Connective
c) Discharging an Assumption
d) Hopping Scope Lines
PART I: PREDICATE LOGIC

1. PREDICATE LOGIC: Syntax
   1–1. We Need More Logical Form  1
   1–2. Quantifiers and Variables  5
   1–3. The Sentences of Predicate Logic  9

2. PREDICATE LOGIC: Semantics and Validity
   2–1. Interpretations  13
   2–2. Truth in an Interpretation  18
   2–3. Validity in Predicate Logic  24
### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td><strong>MORE ABOUT QUANTIFIERS</strong></td>
<td></td>
</tr>
<tr>
<td>3–1.</td>
<td>Some Examples of Multiple Quantification</td>
<td>28</td>
</tr>
<tr>
<td>3–2.</td>
<td>Quantifier Scope, Bound Variables, and Free Variables</td>
<td>29</td>
</tr>
<tr>
<td>3–3.</td>
<td>Correct Definitions of Substitution Instance and Truth in an Interpretation</td>
<td>33</td>
</tr>
<tr>
<td>3–4.</td>
<td>Some Logical Equivalences</td>
<td>36</td>
</tr>
<tr>
<td>4.</td>
<td><strong>TRANSCRIPTION</strong></td>
<td>40</td>
</tr>
<tr>
<td>4–1.</td>
<td>Restricted Quantifiers</td>
<td>40</td>
</tr>
<tr>
<td>4–2.</td>
<td>Transcribing from English into Logic</td>
<td>45</td>
</tr>
<tr>
<td>4–3.</td>
<td>Transcription Strategies</td>
<td>54</td>
</tr>
<tr>
<td>5.</td>
<td><strong>NATURAL DEDUCTION FOR PREDICATE LOGIC:</strong> Fundamentals</td>
<td>62</td>
</tr>
<tr>
<td>5–1.</td>
<td>Review and Overview</td>
<td>62</td>
</tr>
<tr>
<td>5–2.</td>
<td>The Universal Elimination Rule</td>
<td>63</td>
</tr>
<tr>
<td>5–3.</td>
<td>The Existential Introduction Rule</td>
<td>65</td>
</tr>
<tr>
<td>5–4.</td>
<td>The Existential Elimination and Universal Introduction Rules: Background in Informal Argument</td>
<td>70</td>
</tr>
<tr>
<td>5–5.</td>
<td>The Universal Introduction Rule</td>
<td>73</td>
</tr>
<tr>
<td>5–6.</td>
<td>The Existential Elimination Rule</td>
<td>83</td>
</tr>
<tr>
<td>6.</td>
<td><strong>MORE ON NATURAL DEDUCTION FOR PREDICATE LOGIC</strong></td>
<td>91</td>
</tr>
<tr>
<td>6–1.</td>
<td>Multiple Quantification and Harder Problems</td>
<td>91</td>
</tr>
<tr>
<td>6–2.</td>
<td>Some Derived Rules</td>
<td>96</td>
</tr>
<tr>
<td>6–3.</td>
<td>Logical Truth, Contradictions, Inconsistency, and Logical Equivalence</td>
<td>99</td>
</tr>
<tr>
<td>7.</td>
<td><strong>TRUTH TREES FOR PREDICATE LOGIC:</strong> Fundamentals</td>
<td>106</td>
</tr>
<tr>
<td>7–1.</td>
<td>The Rule for Universal Quantification</td>
<td>106</td>
</tr>
<tr>
<td>7–2.</td>
<td>The Rule for Existential Quantification</td>
<td>111</td>
</tr>
<tr>
<td>7–3.</td>
<td>Applying the Rules</td>
<td>114</td>
</tr>
<tr>
<td>7–4.</td>
<td>Negated Quantified Sentences</td>
<td>118</td>
</tr>
<tr>
<td>8.</td>
<td><strong>MORE ON TRUTH TREES FOR PREDICATE LOGIC</strong></td>
<td>123</td>
</tr>
<tr>
<td>8–1.</td>
<td>Contradictions, Logical Truth, Logical Equivalence, and Consistency</td>
<td>123</td>
</tr>
<tr>
<td>8–2.</td>
<td>Truth Trees with Multiple Quantifiers</td>
<td>128</td>
</tr>
<tr>
<td>8–3.</td>
<td>Three Shortcuts</td>
<td>131</td>
</tr>
<tr>
<td>8–4.</td>
<td>Infinite Trees</td>
<td>133</td>
</tr>
<tr>
<td>9.</td>
<td><strong>IDENTITY, FUNCTIONS, AND DEFINITE DESCRIPTIONS</strong></td>
<td>138</td>
</tr>
<tr>
<td>9–1.</td>
<td>Identity</td>
<td>138</td>
</tr>
<tr>
<td>9–2.</td>
<td>Inference Rules for Identity</td>
<td>141</td>
</tr>
<tr>
<td>9–3.</td>
<td>Functions</td>
<td>147</td>
</tr>
<tr>
<td>9–4.</td>
<td>Definite Descriptions</td>
<td>153</td>
</tr>
</tbody>
</table>

**PART II: METATHEORY**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.</td>
<td><strong>METATHEORY:</strong> The Basic Concepts</td>
<td>157</td>
</tr>
<tr>
<td>10–1.</td>
<td>Object Language and Metalanguage</td>
<td>157</td>
</tr>
<tr>
<td>10–2.</td>
<td>Syntax and Semantics</td>
<td>161</td>
</tr>
<tr>
<td>10–3.</td>
<td>Soundness and Completeness</td>
<td>162</td>
</tr>
<tr>
<td>10–4.</td>
<td>Some Further Notation and Definitions</td>
<td>165</td>
</tr>
</tbody>
</table>
11. MATHEMATICAL INDUCTION 169
   11-1. Informal Introduction 169
   11-2. The Principle of Weak Induction 170
   11-3. Strong Induction 172

12. SOUNDNESS AND COMPLETENESS FOR SENTENCE LOGIC TREES 176
   12-1. Preliminaries 176
   12-2. Soundness and Completeness of the Tree Method: Informal Statement 177
   12-3. Soundness and Completeness for Sentence Logic Trees: Formal Details 181

13. SOUNDNESS AND COMPLETENESS FOR SENTENCE LOGIC DERIVATIONS 191
   13-1. Soundness for Derivations: Informal Introduction 191
   13-2. Soundness for Derivations: Formal Details 193
   13-4. Completeness for Derivations: Formal Details 203

14. KOENIG'S LEMMA, COMPACTNESS, AND GENERALIZATION TO INFINITE SETS OF PREMISES 212
   14-1. Koenig's Lemma 212
   14-2. Compactness and Infinite Sets of Premises 214

15. INTERPRETATIONS, SOUNDNESS, AND COMPLETENESS FOR PREDICATE LOGIC 220
   15-1. Interpretations 220
   15-2. Soundness and Completeness for Trees 230
In Volume I you gained a firm foundation in sentence logic. But there must be more to logic, as you can see from the next examples. Consider the following two English arguments and their transcriptions into sentence logic:

1. Everyone loves Adam. 
2. Eve loves Adam. 

In sentence logic, we can only transcribe the sentences in these arguments as atomic sentence letters. But represented with sentence letters, both natural deduction and truth trees tell us that these arguments are invalid. No derivation will allow us to derive 'B' from 'A' or 'C' from 'B'. A&¬B is a counterexample to the first argument, and B&¬C is a counterexample to the second. An argument is valid only if it has no counterexamples.

Something has gone terribly wrong. Clearly, if everyone loves Adam, then so does Eve. If the premise is true, without fail the conclusion will be true also. In the same way, if Eve loves Adam, then someone loves Adam. Once again, there is no way in which the premise could be true.
and the conclusion false. But to say that if the premises are true, then without fail the conclusion will be true also is just what we intend when we say that an argument is valid. Since sentence logic describes these arguments as invalid, it looks like something has to be wrong with sentence logic.

Sentence logic is fine as far as it goes. The trouble is that it does not go far enough. These two arguments owe their validity to the internal logical structure of the sentences appearing in the arguments, and sentence logic does not describe this internal logical structure. To deal with this shortcoming, we must extend sentence logic in a way which will display the needed logical structure and show how to use this structure in testing arguments for validity. We will keep the sentence logic we have learned in Volume I. But we will extend it to what logicians call Predicate Logic (also sometimes called Quantificational Logic).

Predicate logic deals with sentences which say something about someone or something. Consider the sentence ‘Adam is blond.’ This sentence attributes the property of being blond to the person named ‘Adam’. The sentence does this by applying the predicate (the word) ‘blond’ to the name ‘Adam’. A sentence of predicate logic does the same thing but in a simplified way.

We will put capital letters to a new use. Let us use the capital letter ‘B’, not now as a sentence letter, but to transcribe the English word ‘blond’. And let us use ‘a’ to transcribe the name ‘Adam’. For ‘Adam is blond.’, predicate logic simply writes ‘Ba’, which you should understand as the predicate ‘B’ being applied to the name ‘a’. This, in turn, you should understand as stating that the person named by ‘a’ (namely, Adam) has the property indicated by ‘B’ (namely, the property of being blond).

Of course, on a different occasion, we could use ‘B’ to transcribe a different English predicate, such as ‘bachelor’, ‘short’, or ‘funny’. And we could use ‘a’ as a name for different people or things. It is only important to stick to the same transcription use throughout one problem or example.

Predicate logic can also express relations which hold between things or people. Let’s consider the simple statement that Eve loves Adam. This tells us that there is something holding true of Eve and Adam together, namely, that the first loves the second. To express this in predicate logic we will again use our name for Adam, ‘a’. We will use a name for Eve, say, the letter ‘e’. And we will need a capital letter to stand for the relation of loving, say, the letter ‘L’. Predicate logic writes the sentence ‘Eve loves Adam’ as ‘Lea’. This is to be read as saying that the relation indicated by ‘L’ holds between the two things named by the lowercase letters ‘e’ and ‘a’. Once again, in a different example or problem, ‘L’, ‘a’, and ‘e’ could be used for different relations, people, or things.

You might be a little surprised by the order in which the letters occur in ‘Lea’. But don’t let that bother you. It’s just the convention most often used in logic: To write a sentence which says that a relation holds between two things, first write the letter which indicates the relation and then write the names of the things between which the relation is supposed to hold. Some logicians write ‘Lea’ as ‘L(e,a)’, but we will not use this notation.

Note, also, the order in which the names ‘e’ and ‘a’ appear in ‘Lea’. ‘Lea’ is a different sentence from ‘Lae’. ‘Lae’ says that Eve loves Adam. ‘Lae’ says that Adam loves Eve. One of these sentences might be true while the other one is false! Think of ‘L’ as expressing the relation, which holds just in case the first thing named loves the second thing named.

Here is a nasty piece of terminology which I have to give you because it is traditional and you will run into it if you continue your study of logic. Logicians use the word Argument for a letter which occurs after a predicate or a relation symbol. The letter ‘a’ in ‘Ba’ is the argument of the predicate ‘B’. The letters ‘e’ and ‘a’ in ‘Lea’ are the arguments of the relation symbol ‘L’. This use of the word ‘argument’ has nothing to do with the use in which we talk about an argument from premises to a conclusion.

At this point you might be perplexed by the following question. I have now used capital letters for three different things. I have used them to indicate atomic sentences. I have used them as predicates. And I have used them as relation symbols. Suppose you encounter a capital letter in a sentence of predicate logic. How are you supposed to know whether it is an atomic sentence letter, a predicate, or a relation symbol?

Easy. If the capital letter is followed by two lowercase letters, as in ‘Lea’, you know the capital letter is a relation symbol. If the capital letter is followed by one lowercase letter, as in ‘Ba’, you know the capital letter is a predicate. And if the capital letter is followed by no lowercase letters at all, as in ‘A’, you know it is an atomic sentence letter.

There is an advantage to listing the arguments of a relation symbol after the relation symbol, as in ‘Lea’. We can see that there is something important in common between relation symbols and predicates. To attribute a relation as holding between two things is to say that something is true about the two things taken together and in the order specified. To attribute a property as holding of one thing is to say that something is true about that one thing. In the one case we attribute something to one thing, and in the other we attribute something to two things.

We can call attention to this similarity between predicates and relations in a way which also makes our terminology a bit smoother. We can indicate the connection by calling a relation symbol a Two Place Predicate, that is, a symbol which is very like an ordinary predicate except that it has two argument places instead of one. In fact, we may sometimes want to talk
about three place predicates (equally well called 'three place relation symbols'). For example, to transcribe 'Eve is between Adam and Cid', I introduce 'c' as a name for Cid and the three place predicate 'K' to indicate the three place relation of being between. My transcription is 'Keac', which you can think of as saying that the three place relation of being between holds among Eve, Adam, and Cid, with the first being between the second and the third.

This is why our new logic is called 'predicate logic': It involves predicates of one place, two places, three places, or indeed, any number of places. As I mentioned, logicians also refer to these symbols as one place, two place, or many place relation symbols. But logicians never call the resulting system of logic 'relation logic'. I have no idea why not.

Our familiar sentence logic built up all sentences from atomic sentence letters. Predicate logic likewise builds up compound sentences from atomic sentences. But we have expanded our list of what counts as an atomic sentence. In addition to atomic sentence letters, we will include sentences such as 'Ba' and 'Lea'. Indeed, any one place predicate followed by one name, any two place predicate followed by two names, and so on, will now also count as an atomic sentence. We can use our expanded stock of atomic sentences to build up compound sentences with the help of the connectives, just as before.

In summarizing this section, we say

EXERCISES

In the following exercises, use this transcription guide:

- a: Adam
- e: Eve
- c: Cid
- Bx: x is blond
- Cx: x is a cat
- Lxy: x loves y
- Txy: x is taller than y

1-1. Transcribe the following predicate logic sentences into English:

a) Tce
b) Lce
c) ¬Tcc
d) Bc
e) Tce ⊃ Lce
f) Lce v Lcc
g) ¬(Lce & Lca)
h) Bc = (Lce v Lcc)

1-2. Transcribe the following English sentences into sentences of predicate logic:

a) Cid is a cat.
b) Cid is taller than Adam.
c) Either Cid is a cat or he is taller than Adam.
d) If Cid is taller than Eve then he loves her.
e) Cid loves Eve if he is taller than she is.
f) Eve loves both Adam and Cid.
g) Eve loves either Adam or Cid.
h) Either Adam loves Eve or Eve loves Adam, but both love Cid.
i) Only if Cid is a cat does Eve love him.
j) Eve is taller than but does not love Cid.

1-2. QUANTIFIERS AND VARIABLES

We still have not done enough to deal with arguments (1) and (2). The sentences in these arguments not only attribute properties and relations to things, but they involve a certain kind of generality. We need to be able to express this generality, and we must be careful to do it in a way which will make the relevant logical form quite clear. This involves a way of writing general sentences which seems very awkward from the point of view of English. But you will see how smoothly everything works when we begin proving the validity of arguments.

English has two ways of expressing general statements. We can say 'Everyone loves Adam.' (Throughout, 'everybody' would do as well as 'everyone'.) This formulation puts the general word 'everyone' where ordinarily we might put a name, such as 'Eve'. Predicate logic does not work this way. The second way of expressing general statements in English uses expressions such as 'Everyone is such that they love Adam.' or 'Everything is such that it loves Adam.' Predicate logic uses a formulation of this kind.
Read the symbol \((\forall x)\) as 'Every x is such that'. Then we transcribe 'Everyone loves Adam.' as \((\forall x)Lxa\). In words, we read this as "Every x is such that x loves Adam." \((\forall x)\) is called a Universal Quantifier. In other logic books you may see it written as \((x)\).

We are going to need not only a way of saying that everyone loves Adam but also a way of saying that someone loves Adam. Again, English does this most smoothly by putting the general word 'someone' where we might have placed a name like 'Eve'. And again logic does not imitate this style. Instead, it imitates English expressions such as 'Someone is such that he or she loves Adam.,' or 'Some person is such that he or she loves Adam.', or 'Something is such that it loves Adam.' Read the symbol \((\exists x)\) as 'Some x is such that.' Then we transcribe 'Someone loves Adam.' as \((\exists x)Lxa\). \((\exists x)\) is called an Existential Quantifier.

In one respect, \((\exists x)\) corresponds imperfectly to English expressions which use words such as 'some', 'there is a', and 'there are'. For example, we say 'Some cat has caught a mouse' and 'There is a cat which has caught a mouse' when we think that there is exactly one such cat. We say 'Some cats have caught a mouse' or 'There are cats which have caught a mouse' when we think that there are more than one. Predicate logic has only the one expression, \((\exists x)\), which does not distinguish between 'exactly one' and 'more than one'. \((\exists x)\) means that there is one or more x such that.

(In chapter 9 we will learn about an extension of our logic which will enable us to make this distinction not made by \((\exists x)\):)

In English, we also make a distinction by using words such as 'Everyone' and 'everybody' as opposed to words like 'everything'. That is, English uses one word to talk about all people and another word to talk about all things which are not people. The universal quantifier, \((\forall x)\), does not mark this distinction. If we make no qualification, \((\forall x)\), means all people and things. The same comments apply to the existential quantifier. English contrasts 'someone' and 'somebody' with 'something'. But in logic, if we make no qualification, \((\exists x)\) means something, which can be a person or a thing. All this is very inconvenient when we want to transcribe sentences such as 'Someone loves Adam.' and 'Everybody loves Eve.' into predicate logic.

Many logicians try to deal with this difficulty by putting restrictions on the things to which the 'x' in \((\forall x)\) and \((\exists x)\) can refer. For example, in dealing with a problem which deals only with people, they say at the outset: For this problem 'x' will refer only to people. This practice is called establishing a Universe of Discourse or Restricting the Domain of Discourse. I am not going to fill in the details of this common logical practice because it really does not solve our present problem. If we resolved to talk only about people, how would we say something such as 'Everybody likes something'? In chapter 4 I will show you how to get the effect of restricting the domain of discourse in a more general way which will also allow us to talk at the same time about people, things, places, or whatever we like.

But until chapter 4 we will make do with the intuitive idea of restricting 'x' to refer only to people when we are transcribing sentences using expressions such as 'anybody', 'no one', and 'someone'. In other words, we will, for the time being indulge in the not quite correct practice of transcribing \((\forall x)\) as 'everyone', 'anybody', etc., and \((\exists x)\) as 'someone', 'somebody', or the like, when this is the intuitively right way to proceed, instead of the strictly correct 'everything', 'something', and similar expressions.

The letter 'x' in \((\forall x)\) and \((\exists x)\) is called a Variable. Variables will do an amazing amount of work for us, work very similar to that done by English pronouns, such as 'he', 'she', and 'it'. For example, watch the work 'it' does for me when I say the following: "I felt something in the closed bag. It felt cold. I pulled it out." This little discourse involves existential quantification. The discourse begins by talking about something without saying just which thing this something is. But then the discourse goes on to make several comments about this thing. The important point is that all the comments are about the same thing. This is the work that 'it' does for us. It enables us to cross-reference, making clear that we are always referring to the same thing, even though we have not been told exactly what that thing is.

A variable in logic functions in exactly the same way. For example, once we introduce the variable 'x' with the existential quantifier, \((\exists x)\) we can use 'x' repeatedly to refer to the same (unknown) thing. So I can say, 'Someone is blond and he or she loves Eve' with the sentence \((\exists x)(Bx & Lxe)\). Note the use of parentheses here. They make clear that the quantifier \((\exists x)\) applies to all of the sentence 'Bx & Lxe'. Like negation, a quantifier applies to the shortest full sentence which follows it, where the shortest full following sentence may be marked with parentheses. And the 'x' in the quantifier applies to, or is linked to, all the occurrences of 'x' in this shortest full following sentence. We say that

A quantifier Governs the shortest full sentence which follows it and Binds the variables in the sentence it governs. The latter means that the variable in the quantifier applies to all occurrences of the same variable in the shortest full following sentence.

Unlike English pronouns, variables in logic do not make cross-references between sentences.

These notions actually involve some complications in sentences which use two quantifiers, complications which we will study in chapter 3. But this rough characterization will suffice until then.

Let us look at an example with the universal quantifier, \((\forall x)\). Consider the English sentences 'Anyone blond loves Eve.', 'All blonds love Eve.',
'Any blond loves Eve.' and 'All who are blond love Eve.' All these sentences say the same thing, at least so far as logic is concerned. We can express what they say more painstakingly by saying, 'Any people are such that if they are blond then they love Eve.' This formulation guides us in transcribing into logic. Let us first transcribe a part of this sentence, the conditional, which talks about some unnamed people referred to with the pronoun 'they': 'If they are blond then they love Eve.' Using the variable 'x' for the English pronoun 'they', this comes out as 'Bx ⊃ Lxe'. Now all we have to do is to say that this is true whoever "they" may be. This gives us '(∀x)(Bx ⊃ Lxe)'. Note that I have enclosed 'Bx ⊃ Lxe' in parentheses before prefixing the quantifier. This is to make clear that the quantifier applies to the whole sentence.

I have been using 'x' as a variable which appears in quantifiers and in sentences governed by quantifiers. Obviously, I would just as well have used some other letter, such as 'y' or 'z'. In fact, later on, we will need to use more than one variable at the same time with more than one quantifier. So we will take '(∀x)', '(∀y)', and '(∀z)' all to be universal quantifiers, as well as any other variable prefixed with '∀' and surrounded by parentheses if we should need still more universal quantifiers. In the same way, '(∃x)', '(∃y)', and '(∃z)' will all function as existential quantifiers, as will any similar symbol obtained by substituting some other variable for 'x', 'y', or 'z'.

To make all this work smoothly, we should clearly distinguish the letters which will serve as variables from other letters. Henceforth, I will reserve lowercase 'w', 'x', 'y', and 'z' to use as variables. I will use lowercase 'a' through 'r' as names. If one ever wanted more variables or names, one could add to these lists indefinitely by using subscripts. Thus 'a₁' and 'd₁₁' are both names, and 'x₁' and 'z₂₄' are both variables. But in practice we will never need that many variables or names.

What happened to 's', 't', 'u', and 'v'? I am going to reserve these letters about names and variables. The point is this: As I have mentioned, when I want to talk generally in English about sentences in sentence logic, I use boldface capital 'X', 'Y', and 'Z'. For example, when I stated the & rule I wrote, "For any sentences X and Y...". The idea is that what I wrote is true no matter what sentence you might write in for 'X' and 'Y'. I will need to do the same thing when I state the new rules for quantifiers. I will need to say something which will be true no matter what names you might use and no matter what variables you might use. I will do this by using boldface 's' and 't' when I talk about names and boldface 'u' and 'v' when I talk about variables.

To summarize our conventions for notation:

We will use lowercase letter 'a' through 'r' as names, and 'w', 'x', 'y' and 'z' as variables. We will use boldface 's' and 't' to talk generally about names and boldface 'u' and 'v' to talk generally about variables.

1-3. The Sentences of Predicate Logic

We now have all the pieces for saying exactly which expressions are going to count as sentences of predicate logic. First, all the sentences of sentence logic count as sentences of predicate logic. Second, we expand our stock of atomic sentences. I have already said that we will include among the atomic sentences predicates followed by the right number of names (one name for one place predicates, two names for two place predicates, and so on). We will do the same thing with variables and with variables mixed with names. So 'Bx' will count as an atomic sentence, as will 'Lxx', 'Lxy', and 'Lxz'. In general, any predicate followed by the right number of names and/or variables will count as an atomic sentence.

We get all the rest of the sentences of predicate logic by using connectives to build longer sentences from shorter sentences, starting from atomic sentences. We use all the connectives of sentence logic. And we add to these '(∀x)', '(∃x)', '(∀y)', and other quantifiers, all of which count as new connectives. We use a quantifier to build a longer sentence from a shorter one in exactly the same way that we use the negation sign to build up sentences. Just put the quantifier in front of any expression which is already itself a sentence. We always understand the quantifier to apply to the shortest full sentence which follows the quantifier, as indicated by parentheses. Thus, if we start with 'Lxa', '(∀x)Lxa' counts as a sentence. We could have correctly written '(∀x)(Lxa)', though the parentheses around 'Lxa' are not needed in this case. To give another example, we can start with the atomic sentences 'Bx' and 'Lxe'. We build a compound by joining these with the conditional, '⊃', giving 'Bx ⊃ Lxe'. Finally, we apply '(∀x)' to this compound sentence. We want to be clear that '(∀x)' applies to the whole of 'Bx ⊃ Lxe', so we have to put parentheses around it before prefixing '(∀x)'. This gives '(∀x)(Bx ⊃ Lxe)'.

Here is a formal definition of sentences of predicate logic:

All sentence letters and predicates followed by the appropriate number of names and/or variables are sentences of predicate logic. (These are the atomic sentences.) If X is any sentence of predicate logic and u is any variable, then '(∀u)X' (a universally quantified sentence) and '(∃u)X' (an existentially quantified sentence) are both sentences of predicate logic. If X and Y are both sentences of predicate logic, then any expression formed from X and Y using the connectives of sentence logic are sentences of predicate logic. Finally, only these expressions are sentences of predicate logic.

Logicians often use the words Well Formed Formula (Abbreviated wff) for any expression which this definition classifies as a predicate logic sentence.

You may have noticed something a little strange about the definition. It tells us that an expression such as '(∀x)Ba' is a predicate logic sentence. If 'A' is a sentence letter, even '(∀x)A' is going to count as a sentence! But how should we understand '(∀x)Ba' and '(∀x)A'? Since the variable 'x' of
the quantifier does not occur in the rest of the sentence, it is not clear what these sentences are supposed to mean.

To have a satisfying definition of predicate logic sentence, one might want to rule out expressions such as '(\(\forall x\))Ba' and '(\(\forall x\))A'. But it will turn out that keeping these as official predicate logic sentences will do no harm, and ruling them out in the definition makes the definition messier. It is just not worth the effort to rule them out. In the next chapter we will give a more exact characterization of how to understand the quantifiers, and this characterization will tell us that "vacuous quantifiers," as in '(\(\forall x\))Ba' and '(\(\forall x\))A', have no effect at all. These sentences can be understood as the sentences 'Ba' and 'A', exactly as if the quantifiers were not there.

The definition also counts sentences such as 'By', 'Lze', and 'Bx & Lxe' as sentences, where 'x' and 'z' are variables not governed by a quantifier. Such sentences are called Open Sentences. Open sentences can be a problem in logic in the same way that English sentences are a problem when they contain "open" pronouns. You fail to communicate if you say, 'He has a funny nose,' without saying or otherwise indicating who "he" is.

Many logicians prefer not to count open sentences as real sentences at all. Where I use the expression 'open sentence', often logicians talk about 'open formulas' or 'propositional functions'. If you go on in your study of logic, you will quickly get used to these alternative expressions, but in an introductory course I prefer to keep the terminology as simple as possible.

Have you been wondering what the word 'syntax' means in the title of this chapter? The Syntax of a language is the set of rules which tell you what counts as a sentence of the language. You now know what constitutes a sentence of predicate logic, and you have a rough and ready idea of how to understand such a sentence. Our next job will be to make the interpretation of these sentences precise. We call this giving the Semantics for predicate logic, which will be the subject of the next chapter. But, first, you should practice what you have learned about the syntax of predicate logic to make sure that your understanding is secure.

EXERCISES

1–3. Which of the following expressions are sentences of predicate logic?

- a) Ca
- b) Tab
- c) aTb
- d) Ca ⊃ Tab
- e) (\(\exists x\))¬Cx

f) (\(\forall x\))(Cx ⊃ Tax)
g) (\(\forall x\))Cx & Tax(\(\forall x\))
h) (\(\forall x\))(Txa v Tax)
i) (\(\exists x\))Txa v (\(\exists x\))Txc

In the following exercises, use this transcription guide:

- a: Adam
- e: Eve
- c: Cid
- Bx: x is blond
- Cx: x is a cat
- Lxy: x loves y
- Txy: x is taller than y

Before you begin, I should point out something about transcribing between logic and pronouns in English. I used the analogy to English pronouns to help explain the idea of a variable. But that does not mean that you should always transcribe variables as pronouns or that you should always transcribe pronouns as variables. For example, you should transcribe 'If Eve is a cat, then she loves herself.' with the predicate logic sentence 'Ce ⊃ Lee'. Notice that 'she' and 'herself' are both transcribed as 'e'. That is because in this case we have been told who she and herself are. We know that they are Eve, and so we use the name for Eve, namely, 'e' to transcribe these pronouns. How should we describe 'Ca ⊃ ¬Ba'? We could transcribe this as 'If Adam is a cat then Adam is not blond.' But a nicer transcription is simply 'If Adam is a cat then he is not blond.'

Now do your best with the following transcriptions.

1–4. Transcribe the following predicate logic sentences into English:

- a) ¬Laa
- b) Laa ⊃ ¬Taa
- c) ¬(Bc v Lee)
- d) Ca = (Ba v Lae)
- e) (\(\exists x\))Txc
- f) (\(\forall x\))(Lax & (\(\forall x\))Lcx)
- g) (\(\forall x\))(Lax & Lcx)
- h) (\(\exists x\))(Txa v (\(\exists x\))Txc
- i) (\(\exists x\))(Txa v Txc)
- j) (\(\forall x\))(Cx ⊃ Lxe)
- k) (\(\exists x\))(Cx & ¬Lcx)
- l) ¬(\(\forall x\))(Cx ⊃ Lcx)
- m) (\(\forall x\))(Cx ⊃ (Lcx v Lcx))
- n) (\(\exists x\))(Cx & (Bx & Txc))
1–5. Transcribe the following English sentences into sentences of predicate logic:

   a) Everyone loves Eve.
   b) Everyone is loved by either Cid or Adam.
   c) Either everyone is loved by Adam or everyone is loved by Cid.
   d) Someone is taller than both Adam and Cid.
   e) Someone is taller than Adam and someone is taller than Cid.
   f) Eve loves all cats.
   g) All cats love Eve.
   h) Eve loves some cats.
   i) Eve loves no cats.
   j) Anyone who loves Eve is not a cat.
   k) No one who loves Eve is a cat.
   l) Somebody who loves Adam loves Cid.
   m) No one loves both Adam and Cid.

**CHAPTER SUMMARY EXERCISES**

Provide short explanations for each of the following. Check against the text to make sure that your explanations are correct, and keep your explanations in your notebook for reference and review.

   a) Predicate Logic
   b) Name
   c) Predicate
   d) One Place Predicate
   e) Two Place Predicate
   f) Relation
   g) Variable
   h) Universal Quantifier
   i) Existential Quantifier
   j) Universe, or Domain of Discourse
   k) Govern
   l) Bind
   m) Open Sentence
   n) Sentence of Predicate Logic
   o) Well Formed Formula (wff)
   p) Syntax
   q) Semantics
2–1. INTERPRETATIONS

Recall that we used truth tables to give very precise definitions of the meaning of '&, ', 'v', '¬', '→', and '≡'. We would like to do the same for the meaning of quantifiers. But, as you will see very soon, truth tables won't do the job. We need something more complicated.

When we were doing sentence logic, our atomic sentences were just sentence letters. By specifying truth values for all the sentence letters with which we started, we already fixed the truth values of any sentence which we could build up from these smallest pieces. Now that we are doing predicate logic, things are not so easy. Suppose we are thinking about all the sentences which we could build up using the one place predicate 'B', the two place predicate 'L', the name 'a', and the name 'e'. We can form six atomic sentences from these ingredients: 'Ba', 'Be', 'Laa', 'Lae', 'Lea', and 'Lee'. The truth table formed with these six atomic sentences would have 64 lines. Neither you nor I are going to write out a 64-line truth table, so let's consider just one quite typical line from the truth table:

\[
\begin{array}{cccccc}
Ba & Be & Laa & Lae & Lea & Lee \\
\top & \top & \bot & \top & \top & \bot \\
\end{array}
\]

Figure 2-1
Even such an elementary case in predicate logic begins to get quite complicated, so I have introduced a pictorial device to help in thinking about such cases (see figure 2-1). I have drawn a box with two dots inside, one labeled ‘a’ and the other labeled ‘e’. This box is very different from a Venn diagram. This box is supposed to picture just one way the whole world might be. In this very simple picture of the world, there are just two things, Adam and Eve. The line of the truth table on the left gives you a completed description of what is true and what is false about Adam and Eve in this very simple world: Adam is blond, Eve is not blond, Adam does not love himself, Adam loves Eve, Eve does not love Adam, and Eve does love herself.

You can also think of the box and the description on the left as a very short novel. The box gives you the list of characters, and the truth table line on the left tells you what happens in this novel. Of course, the novel is not true. But if the novel were true, if it described the whole world, we would have a simple world with just Adam and Eve having the properties and relations described on the left.

Now, in writing this novel, I only specified the truth value for atomic sentences formed from the one and two place predicates and from the two names. What about the truth value of more complicated sentences? We can use our old rules for figuring out the truth value of compounds formed from these atomic sentences using '&', 'v', '~', '>', and '='. For example, in this novel 'Ba & Lac' is true because both the components are true.

What about the truth value of '(3x)Bx'? Intuitively, '(3x)Bx' should be true in the novel because in the novel there is someone, namely Adam, who is blond. As another example, consider '(3x)Lxa'. In this novel '(3x)Lxa' is false because Eve does not love Adam and Adam does not love Eve. And in this novel there isn't anyone (or anything) else. So no one loves Adam. In other words, in this novel it is false that there is someone who loves Adam.

Let's move on and consider the sentence '(Vx)Lxe'. In our novel this sentence is true, because Adam loves Eve, and Eve loves herself, and that's all the people there are in this novel. If this novel were true, it would be true that everyone loves Eve. Finally, '(Vx)Bx' is false in the novel, for in this novel Eve is not blond. So in this novel it is false that everyone is blond.

Remember what we had set out to do: We wanted to give a precise account of the meaning of the quantifiers very like the precise account which truth table definitions gave to '&' and the other sentence logic connectives. In sentence logic we did this by giving precise rules which told us when a compound sentence is true, given the truth value of the compound's components.

We now have really done the same thing for '(Vx)' and '(3x)' in one special case. For a line of a truth table (a "novel") that gives a truth value for all atomic sentences using 'B', 'L', 'a', and 'e', we can say whether a universally quantified or an existentially quantified sentence is true or false. For example, the universally quantified sentence '(Vx)Lxe' is true just in case 'Lxe' is true for all values of 'x' in the novel. At the moment we are considering a novel in which the only existing things are Adam and Eve. In such a novel '(Vx)Lxe' is true if both 'Lxe' is true when we take 'x' to refer to Adam and 'Lxe' is also true when we take 'x' to refer to Eve. Similarly, '(3x)Bx' is true in such a novel just in case 'Bx' is true for some value of 'x' in the novel. As long as we continue to restrict attention to a novel with only Adam and Eve as characters, '(3x)Bx' is true in the novel if either 'Bx' is true when we take 'x' to refer to Adam or 'Bx' is true if we take 'x' to refer to Eve.

If the example seems a bit complicated, try to focus on this thought: All we are really doing is following the intuitive meaning of "all x" and "some x" in application to our little example. If you got lost in the previous paragraph, go back over it with this thought in mind.

Now comes a new twist, which might not seem very significant, but which will make predicate logic more interesting (and much more complicated) than sentence logic. In sentence logic we always had truth tables with a finite number of lines. Starting with a fixed stock of atomic sentence letters, we could always, at least in principle, write out all possible cases to consider, all possible assignments of truth values to sentence letters. The list might be too long to write out in practice, but we could at least understand everything in terms of such a finite list of cases.

Can we do the same thing when we build up sentences with predicates and names? If, for example, we start with just 'B', 'L', 'a', and 'e', we can form six atomic sentences. We can write out a 64-line truth table which will give us the truth value for any compound built up from these six atomic sentences, for any assignment of truth values to the atomic sentences. But the fact that we are using quantifiers means that we must also consider further possibilities.

Consider the sentence '(Vx)Bx'. We know this is false in the one case we used as an example (in which 'Ba' is true and 'Be' is false). You will immediately think of three alternative cases (three alternative "novels") which must be added to our list of relevant possible cases: the case in which Eve is blond and Adam is not, the case in which Adam and Eve are both blond, and the case in which both are not blond. But there are still more cases which we must include in our list of all possible cases! I can generate more cases by writing new novels with more characters. Suppose I write a new novel with Adam, Eve, and Cid. I now have eight possible ways of distributing hair color (blond or not blond) among my characters, which can be combined with 512 different possible combinations of who does or does not love whom! And, of course, this is just the beginning of an unending list of novels describing possible cases in which '(Vx)Bx' will have a truth value. I can always expand my list of novels by adding new
characters. I can even describe novels with infinitely many characters, although I would not be able to write such a novel down.

How are we going to manage all this? In sentence logic we always had, for a given list of atomic sentences, a finite list of possible cases, the finite number of lines of the corresponding truth table. Now we have infinitely many possible cases. We can’t list them all, but we can still say what any one of these possible cases looks like. Logicians call a possible case for a sentence of predicate logic an Interpretation of the sentence. The example with which we started this chapter is an example of an interpretation, so actually you have already seen and understood an example of an interpretation. We need only say more generally what interpretations are.

We give an interpretation, first, by specifying a collection of objects which the interpretation will be about, called the Domain of the interpretation. A domain always has at least one object. Then we give names to the objects in the domain, to help us in talking about them. Next, we must say which predicates will be involved. Finally, we must go through the predicates and objects and say which predicates are true of which objects. If we are concerned with a one place predicate, the interpretation specifies a list of objects of which the object is true. If the predicate is a two place predicate, then the interpretation specifies a list of pairs of objects between which the two place relation is supposed to hold, that is, pairs of objects of which the two place relation is true. Of course, order is important. The pair a-followed-by-b counts as a different pair from the pair b-followed-by-a. Also, we must consider objects paired with themselves. For example, we must specify whether Adam loves himself or does not love himself. The interpretation deals similarly with three and more place predicates.

In practice, we often specify the domain of an interpretation simply by giving the interpretation’s names for those objects. I should mention that in a fully developed predicate logic, logicians consider interpretations which have unnamed objects. In more advanced work, interpretations of this kind become very important. But domains with unnamed objects would make it more difficult to introduce basic ideas and would gain us nothing for the work we will do in part 1 of this volume. So we won’t consider interpretations with unnamed objects until part II.

The following gives a summary and formal definition of an interpretation:

An Interpretation consists of

a) A collection of objects, called the interpretation’s Domain. The domain always has at least one object.
b) A name for each object in the domain. An object may have just one name or more than one name. (In part II we will expand the definition to allow domains with unnamed objects.)
c) A list of predicates.
d) A specification of the objects of which each predicate is true and the objects of which each predicate is false—that is, which one place predicates apply to which individual objects, which two place predicates apply to which pairs of objects, and so on. In this way every atomic sentence formed from predicates and names gets a truth value.
e) An interpretation may also include atomic sentence letters. The interpretation specifies a truth value for any included atomic sentence letter.

By an Interpretation of a Sentence, we mean an interpretation which is sure to have enough information to determine whether or not the sentence is true or false in the interpretation:

An Interpretation of a Sentence is an interpretation which includes all the names and predicates which occur in the sentence and includes truth values for any atomic sentence letters which occur in the sentence.

For example, the interpretation of figure 2–1 is an interpretation of ‘Ba’ and of ‘(\(\forall x\))Lxx’. In this interpretation ‘Ba’ is true and ‘(\(\forall x\))Lxx’ is false. Note that for each of these sentences, the interpretation contains more information than is needed to determine whether the sentence is true or false. This same interpretation is not an interpretation of ‘Bc’ or of ‘(\(\exists x\))Txe’. This is because the interpretation does not include the name ‘c’ or the two place predicate ‘T’, and so can’t tell us whether sentences which use these terms are true or false.

EXERCISES

2–1. I am going to ask you to give an interpretation for some sentences. You should use the following format. Suppose you are describing an interpretation with a domain of three objects named ‘a’, ‘b’, and ‘c’. Specify the domain in this way: D = \{a,b,c\}. That is, specify the domain by giving a list of the names of the objects in the domain. Then specify what is true about the objects in the domain by using a sentence of predicate logic. Simply conjoin all the atomic and negated atomic sentences which say which predicates are true of which objects and which are false. Here is an example. The following is an interpretation of the sentence ‘Tb & Kbd’.

D = {b,d}; Tb & Td & Kbb & Kbd & Kdb & Kdd.

In this interpretation all objects have property T and everything stands in the relation K to itself and to everything else. Here is another interpretation of the same sentence:

D = {b,d}; \neg Tb & Td & Kdb & \neg Kbd & \neg Kdb & Kdd.
Sometimes students have trouble understanding what I want in this exercise. They ask, How am I supposed to decide which interpretation to write down? You can write down any interpretation you want as long as it is an interpretation of the sentence I give you. In every case you have infinitely many interpretations to choose from because you can always get more interpretations by throwing in more objects and then saying what is true for the new objects. Choose any you like. Just make sure you are writing down an interpretation of the sentence I give you.

a) Lab        d) $(\forall x)(Fx=Rxb)$
b) Lab $\supset$ Ta  e) Ga & $(\exists x)(Lxb v Rax)$
c) Lab $\neg$ Lba     f) $(Kx & (\forall x)Rax) \supset (\exists x)(Mx v Rcx)$

2-2. TRUTH IN AN INTERPRETATION

Just like a line of a truth table, an interpretation tells us whether each atomic sentence formed from predicates and names is true or false. What about compound sentences? If the main connective of a compound sentence does not involve a quantifier, we simply use the old rules for the connectives of sentence logic. We have only one more piece of work to complete: We must make more exact our informal description of the conditions under which a quantified sentence is true or is false in an interpretation. We have only one more piece of work to complete: We must make more exact our informal description of the conditions under which a quantified sentence is true or is false in an interpretation.

Intuitively, a universally quantified sentence is going to be true in an interpretation if it is true in the interpretation for everything to which the variable could refer in the interpretation. (Logicians say, "For every value of the universally quantified variable.") An existentially quantified sentence will be true in an interpretation if it is true for something to which the variable could refer in the interpretation (that is, "for some value of the existentially quantified variable.") What we still need to do is to make precise what it is for a quantified sentence to be true for a value of a variable. Let's illustrate with the same example we have been using, the interpretation given in figure 2-1.

Consider the sentence $(\forall x)Bx$. In the interpretation we are considering, there are exactly two objects, a, and e. $(\forall x)Bx$ will be true in the interpretation just in case, roughly speaking, it is true both for the case of $x$ referring to a and the case of $x$ referring to e. But when $x$ refers to a, we have the sentence 'Ba'. And when $x$ refers to e, we have the sentence 'Be'. Thus $(\forall x)Bx$ is true in this interpretation just in case both 'Ba' and 'Be' are true. We call 'Ba' the Substitution Instance of $(\forall x)Bx$ formed by substituting 'a' for 'x'. Likewise, we call 'Be' the substitution instance of $(\forall x)Bx$ formed by substituting 'e' for 'x'. Our strategy is to explain the meaning of universal quantification by defining this notion of substitution instance and then specifying that a universally quantified sentence is true in an interpretation just in case it is true for all substitution instances in the interpretation:

(Incomplete Definition) For any universally quantified sentence $(\forall u)(\ldots u \ldots)$, the Substitution Instance of the sentence with the name $s$ substituted for the variable $u$ is $(\ldots s \ldots)$, the sentence formed by dropping the initial universal quantifier and writing $s$ wherever $u$ had occurred.

A word of warning: This definition is not yet quite right. It works only as long as we don't have multiple quantification, that is, as long as we don't have sentences which stack one quantifier on top of other quantifiers. But until chapter 3 we are going to keep things simple and consider only simple sentences which do not have one quantifier applying to a sentence with another quantifier inside. When we have the basic concepts we will come back and give a definition which is completely general.

Now we can easily use this definition of substitution instance to characterize truth of a universally quantified sentence in an interpretation:

(Incomplete Definition) A universally quantified sentence is true in an interpretation just in case all of the sentence's substitution instances, formed with names in the interpretation, are true in the interpretation.

Another word of warning: As with the definition of substitution instance, this definition is not quite right. Again, chapter 3 will straighten out the details.

To practice, let's see whether $(\forall x)(Bx \supset Lxe)$ is true in the interpretation of figure 2-1. First we form the substitution instances with the names of the interpretation, 'a', and 'e'. We get the first substitution instance by dropping the quantifier and writing in 'a' everywhere we see 'x'. This gives

$Ba \supset Lae$.

Note that because 'Ba' and 'Lae' are both true in the interpretation, this first substitution instance is true in the interpretation. Next we form the second substitution instance by dropping the quantifier and writing in 'e' wherever we see 'x':

$Be \supset Lec$.

Because 'Be' is false and 'Lee' is true in the interpretation, the conditional 'Be $\supset$ Lee' is true in the interpretation. We see that all the substitution
instances of \((\forall x)(Bx \supset Lxe)\) are true in the interpretation. So this universally quantified sentence is true in the interpretation.

To illustrate further our condition for truth of a universally quantified sentence, consider the sentence \((\forall x)(Bx \supset Lxa)\). This has the substitution instance \(Ba \supset Laa\). In this interpretation \(Ba\) is true and \(Laa\) is false, so \(Ba \supset Laa\) is false in the interpretation. Because \((\forall x)(Bx \supset Lxa)\) has a false substitution instance in the interpretation, it is false in the interpretation.

You may have noticed the following fact about the truth of a universally quantified sentence and the truth of its substitution instances. By definition \((\forall x)(Bx \supset Lxe)\) is true in the interpretation just in case all of its instances are true in the interpretation. But its instances are all true just in case the conjunction of the instances is true. That is, \((\forall x)(Bx \supset Lxe)\) is true in the interpretation just in case the conjunction

\[(Ba \supset Lae) \& (Be \supset Lee)\]
is true in the interpretation. If you think about it, you will see that this will hold in general. In the interpretation we have been discussing (or any interpretation with two objects named \('a\' and \('e\)'), any universally quantified sentence, \((\forall x)\ldots x \ldots\); will be true just in case the conjunction of its substitution instance, \((\ldots a \ldots) \& (\ldots e \ldots)\), is true in the interpretation.

It's looking like we can make conjunctions do the same work that the universal quantifier does. A universally quantified sentence is true in an interpretation just in case the conjunction of all its substitution instances is true in the interpretation. Why, then, do we need the universal quantifier at all?

To answer this question, ask yourself what happens when we shift to a new interpretation with fewer or more things in its domain. In the new interpretation, what conjunction will have the same truth value as a given universally quantified sentence? If the new interpretation has a larger domain, our conjunction will have more conjuncts. If the new interpretation has a smaller domain, our conjunction will have fewer conjuncts. In other words, when we are looking for a conjunction of instances to give us the truth value of a universally quantified sentence, the conjunction will change from interpretation to interpretation. You can see in this way that the universal quantifier really does add something new. It acts rather like a variable conjunction. It has the effect of forming a long conjunction, with one conjunct for each of the objects in an interpretation's domain. If an interpretation's domain has infinitely many objects, a universally quantified sentence has the effect of an infinitely long conjunction!

What about existentially quantified sentences? All the work is really done. We repeat everything we said for universal quantification, replacing the word 'all' with 'some':

(Incomplete Definition) For any existentially quantified sentence \((\exists x)(\ldots u \ldots)\), the Substitution Instance of the sentence, with the name \(u\) substituted for the variable \(u\) is \((\ldots s \ldots)\), the sentence formed by dropping the initial existential quantifier and writing \(s\) wherever \(u\) had occurred.

(Incomplete Definition) An existentially quantified sentence is true in an interpretation just in case some (i.e., one or more) of the sentence's substitution instances, formed with names in the interpretation, are true in the interpretation.

As with the parallel definitions for universally quantified sentences, these definitions will have to be refined when we get to chapter 3.

To illustrate, let's see whether the sentence \((\exists x)(Bx \& Lxe)\) is true in the interpretation of figure 2–1. We will need the sentence's substitution instances. We drop the quantifier and write in 'a' wherever we see 'x', giving 'Ba & Lae', the instance with 'a' substituted for 'x'. In the same way, we form the instance with 'e' substituted for 'x', namely, 'Be & Lee'. '(3x)(Bx & Lxe)' is true in the interpretation just in case one or more of its substitution instances are true in the interpretation. Because 'Ba' and 'Lae' are true in the interpretation, the first instance, 'Ba & Lae', is true, and so \((\exists x)(Bx \& Lxe)\) is true.

Have you noticed that, just as we have a connection between universal quantification and conjunction, we have the same connection between existential quantification and disjunction: \((\exists x)(Bx \& Lxe)\) is true in our interpretation just in case one or more of its instances are true. But one or more of its instances are true just in case their disjunction

\[(Ba \& Lae) v (Be & Lee)\]
is true. In a longer or shorter interpretation we will have the same thing with a longer or shorter disjunction. Ask yourself, when is an existentially quantified sentence true in an interpretation? It is true just in case the disjunction of all its substitution instances in that interpretation is true in the interpretation. Just as the universal quantifier acted like a variable conjunction sign, the existential quantifier acts like a variable disjunction sign. In an interpretation with an infinite domain, an existentially quantified sentence even has the effect of an infinite disjunction.

I hope that by now you have a pretty good idea of how to determine whether a quantified sentence is true or false in an interpretation. In understanding this you also come to understand everything there is to know about the meaning of the quantifiers. Remember that we explained the meaning of the sentence logic connectives 'v', '&', 'v', '!', and '=' by giving their truth table definitions. For example, explaining how to determine whether or not a conjunction is true in a line of a truth table tells you everything there is to know about the meaning of '&'. In the same way, our characterization of truth of a quantified sentence in an interpre-
This point about the meaning of the quantifiers illustrates a more general fact. By a "case" in sentence logic we mean a line of a truth table, that is, an assignment of truth values to sentence letters. The interpretations of predicate logic generalize this idea of a case. Keep in mind that interpretations do the same kind of work in sentence logic that assignments of truth values to sentence letters do in sentence logic, and you will easily extend what you already know to understand validity, logical truth, contradictions, and other concepts in predicate logic.

By now you have also seen how to determine the truth value which an interpretation gives to any sentence, not just to quantified sentences. An interpretation itself tells you which atomic sentences are true and which are false. You can then use the rules of valuation for sentence logic connectives together with our two new rules for the truth of universally and existentially quantified sentences to determine the truth of any compound sentence in terms of the truth of shorter sentences. Multiple quantification still calls for some refinements, but in outline you have the basic ideas.

**EXERCISES**

2-2. Consider the interpretation

D = {a,b}; \neg Ba & Bb & Laa & \neg Lab & Lba & \neg Lbb.

For each of the following sentences, give all of the sentence's substitution instances in this interpretation, and for each substitution instance say whether the instance is true or false in the interpretation. For example, for the sentence '(\forall x)Bx', your answer should look like this:

<table>
<thead>
<tr>
<th>GIVEN SENTENCE</th>
<th>SUBSTITUTION INSTANCES</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\forall x)Bx</td>
<td>Ba, false in the interpretation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bb, true</td>
<td></td>
</tr>
<tr>
<td>a) (\exists x)Bx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b) (\exists x)\neg Lxa</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c) (\forall x)Lxa</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d) (\exists x)Lbx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>e) (\forall x)(Bx v Lax)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>f) (\exists x)(Lxa &amp; Lbx)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g) (\forall x)(Bx \supset Lbx)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>h) (\exists x)(Lbx &amp; Bb) v Bx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>i) (\forall x)(Bx \supset (Lxx \supset Lxa))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>j) (\forall x)[(Bx v Lax) \supset (Lxb v \neg Bx)]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k) (\exists x)[(Lax &amp; Lxa) = (Bx v Lxb)]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2-3. For each of the sentences in exercise 2-2, say whether the sentence is true or false in the interpretation of exercise 2-2.

2-4. For each of the following sentences, determine whether the sentence is true or false in the interpretation of exercise 2-2. In this exercise, you must carefully determine the main connective of a sentence before applying the rules to determine its truth in an interpretation. Remember that a quantifier is a connective which applies to the shortest full sentence which follows it. Remember that the main connective of a sentence is the last connective that gets used in building the sentence up from its parts. To determine whether a sentence is true in an interpretation, first determine the sentence's main connective. If the connective is '&', 'v', '~', '\supset', or '=' you must first determine the truth value of the components, and then apply the rules for the main connective (a conjunction is true just in case both conjuncts are true, and so on). If the main connective is a quantifier, you have to determine the truth value of the substitution instances and then apply the rule for the quantifier, just as you did in the last exercise.

a) (\exists x)\neg Lxx \supset (\forall x)(Bx v Lbx)

b) (\neg (\exists x)\neg Lxx \supset Bx) & (\forall x)(Bx \supset Lxx)

c) (\neg Bx = (Lax v Lxb))

d) (\neg (\exists x)(Lxx v Bx) \supset (\neg Lab v \neg Ba))

e) (\neg (\forall x)(\neg Lxx v Lxb) \supset (\neg Lab v Lba))

f) (\neg (\exists x)(Lxx v Bx) \supset (\neg Lab v \neg Bx))

g) (\forall x)(\neg Lxx = Bx) \supset (\neg Lab = Lxx))

h) (\forall x)(\neg Lxx v Lxb) v (\exists x)(Lxx v Lxb)

i) (\exists x)(\neg Lab & Lax) & (\exists x)(\neg Lab = Lxx)

j) (\forall x)[(Bx v (Lxx \& \neg Lxb)) \supset (Bx v Lxx)]

2-5. In the past exercises we have given interpretations by explicitly listing objects in the domain and explicitly saying which predicates apply to which things. We can also describe an interpretation in more general terms. For example, consider the interpretation given by

i) Domain: All U.S. citizens over the age of 21.

ii) Names: Each person in the domain is named by 'a' subscripted by his or her social security number.

iii) Predicates: Mx: x is a millionaire.

Hx: x is happy.
Predicate Logic: Semantics and Validity

Predicate Logic: Semantics and Validity

2-3. Validity in Predicate Logic

In sentence logic, we said that an argument is valid if and only if, for all possible cases in which all the premises are true, the conclusion is true also. In predicate logic, the intuitive notion of validity remains the same. We change things only by generalizing the notion of possible case. Where before we meant that all lines in the truth table which make all premises true also make the conclusion true, now we mean that all interpretations where all the premises are true also make the conclusion true:

An argument expressed with sentences in predicate logic is valid if and only if the conclusion is true in every interpretation in which all the premises are true.

You may remember that we got started on predicate logic at the beginning of chapter 1 because we had two arguments which seemed valid but which sentence logic characterized as invalid. To test whether predicate logic is doing the job it is supposed to do, let us see whether predicate logic gives us the right answer for these arguments:

Everyone loves Eve.  \(\forall x Lxe\)
Adam loves Eve.  \(La\)

Suppose we have an interpretation in which \(\forall x Lxe\) is true. Will \(La\) have to be true in this interpretation also? Notice that \(La\) is a substitution instance of \(\forall x Lxe\). A universally quantified sentence is true in an interpretation just in case all its substitution instances are true in the interpretation. So in any interpretation in which \(\forall x Lxe\) is true, the instance \(La\) will be true also. And this is just what we mean by the argument being valid.

Let's examine the other argument:

Adam loves Eve.  \(La\)
Someone loves Eve.  \(\exists x Lxe\)

Suppose we have an interpretation in which \(La\) is true. Does \(\exists x Lxe\) have to be true in this interpretation? Notice that \(La\) is an instance of \(\exists x Lxe\). We know that \(\exists x Lxe\) is true in an interpretation if even one of its instances is true in the interpretation. Thus, if \(La\) is true in an interpretation, \(\exists x Lxe\) will also be true in that interpretation. Once again, the argument is valid.

Along with validity, all our ideas about counterexamples carry over from sentence logic. When we talked about the validity of a sentence logic argument, we first defined it in this way: An argument is valid just in case any line of the truth table which makes all the premises true makes the conclusion true also. Then we reexpressed this by saying: An argument is valid just in case it has no counterexamples; that is, no lines of the truth table make all the premises true and the conclusion false. For predicate logic, all the ideas are the same. The only thing that has changed is that we now talk about interpretations where before we talked about lines of the truth table:
A **Counterexample** to a predicate logic argument is an interpretation in which the premises are all true and the conclusion is false.

A predicate logic argument is **Valid** if and only if it has no counterexamples.

Let's illustrate the idea of counterexamples in examining the validity of

\[ \begin{align*}
&\text{Lae} \\
&\exists x \text{Lxe}
\end{align*} \]

Is there a counterexample to this argument? A counterexample would be an interpretation with 'Lae' true and '(∃x)Lxe' false. But there can be no such interpretation. 'Lae' is an instance of '(∃x)Lxe', and '(∃x)Lxe' is true in an interpretation if even one of its instances is true in the interpretation. Thus, if 'Lae' is true in an interpretation, '(∃x)Lxe' will also be true in that interpretation. In other words, there can be no interpretation in which 'Lae' is true and '(∃x)Lxe' is false, which is to say that the argument has no counterexamples. And that is just another way of saying that the argument is valid.

For comparison, contrast the last case with the argument

\[ \begin{align*}
&\exists x \text{Bx} \\
&\text{Ba}
\end{align*} \]

It's easy to construct a counterexample to this argument. Any case in which someone other than Adam is blond and Adam is not blond will do the trick. So an interpretation with Adam and Eve in the domain and in which Eve is blond and Adam is not blond gives us a counterexample, showing the argument to be invalid.

This chapter has been hard work. But your sweat will be repaid. The concepts of interpretation, substitution instance, and truth in an interpretation provide the essential concepts you need to understand quantification. In particular, once you understand these concepts, you will find proof techniques for predicate logic to be relatively easy.

### EXERCISES

2–6. For each of the following arguments, determine whether the argument is valid or invalid. If it is invalid, show this by giving a counterexample. If it is valid, explain your reasoning which shows it to be valid. Use the kind of informal reasoning which I used in discussing the arguments in this section.

---

You may find it hard to do these problems because I haven't given you any very specific strategies for figuring out whether an argument is valid. But don't give up! If you can't do one argument, try another first. Try to think of some specific, simple interpretation of the sentences in an argument, and ask yourself—"Are the premise and conclusion both true in that interpretation?" Can I change the interpretation so as to make the premise true and the conclusion false? If you succeed in doing that, you will have worked the problem because you will have constructed a counterexample and shown the argument to be invalid. If you can't seem to be able to construct a counterexample, try to understand why you can't. If you can see why you can't and put this into words, you will have succeeded in showing that the argument is valid. Even if you might not succeed in working many of these problems, playing around in this way with interpretations, truth in interpretations, and counterexamples will strengthen your grasp of these concepts and make the next chapter easier.

\[ \begin{align*}
&\begin{array}{c}
\forall x \text{Lxe} \\
\exists x \text{Lxe}
\end{array} \\
&\begin{array}{c}
\text{Lae} \\
\forall x \text{Lxe}
\end{array} \\
&\begin{array}{c}
\exists x \text{Lxe} \\
\forall x \text{Lxe}
\end{array}
\]

\[ \begin{align*}
&\begin{array}{c}
\forall x (\text{Bx} \land \text{Lxe}) \\
\text{Lae}
\end{array} \\
&\begin{array}{c}
\forall x (\text{Bx} \lor \text{Lxe}) \\
\exists x \text{Bx}
\end{array} \\
&\begin{array}{c}
\exists x (\text{Bx} \lor \text{Lxe}) \\
\forall x (\text{Bx} \lor \text{Lxe}) \\
\& \forall x (\neg \text{Bx} \lor \text{Lxa})
\end{array}
\]

---

### CHAPTER SUMMARY EXERCISES

Provide short explanations for each of the following, checking against the text to make sure you understand each term clearly and saving your answers in your notebook for reference and review.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a) Interpretation</td>
<td>b) Interpretation of a Sentence</td>
<td>c) Substitution Instance</td>
</tr>
<tr>
<td>d) Truth in an Interpretation</td>
<td>e) Validity of a Predicate Logic Argument</td>
<td></td>
</tr>
</tbody>
</table>
More about Quantifiers

3-1. SOME EXAMPLES OF MULTIPLE QUANTIFICATION

All of the following are sentences of predicate logic:

1. \((\forall x)(\forall y)L_{xy}\)
2. \((\exists x)(\exists y)L_{xy}\)
3. \((\exists x)(\forall y)L_{xy}\)
4. \((\exists x)(\forall y)L_{yx}\)
5. \((\forall x)(\exists y)L_{xy}\)
6. \((\forall x)(\exists y)L_{yx}\)

Let's suppose that 'L' stands for the relation of loving. What do these sentences mean?

Sentence (1) says that everybody loves everybody (including themselves). (2) says that somebody loves somebody. (The somebody can be oneself or someone else.) Sentences (3) to (6) are a little more tricky. (3) says that there is one person who is such that he or she loves everyone. (There is one person who is such that, for all persons, the first loves the second—think of God as an example.) We get (4) from (3) by reversing the order of the 'x' and 'y' as arguments of 'L'. As a result, (4) says that there is one person who is loved by everyone. Notice what a big difference the order of the 'x' and 'y' makes.

Next, (5) says that everyone loves someone: Every person is such that there is one person such that the first loves the second. In a world in which (5) is true, each person has an object of their affection. Finally we get (6) out of (5) by again reversing the order of 'x' and 'y'. As a result, (6) says that everyone is loved by someone or other. In a world in which (6) is true no one goes unloved. But (6) says something significantly weaker than (3). (3) say that there is one person who loves everyone. (6) says that each person gets loved, but Adam might be loved by one person, Eve by another, and so on.

Can we say still other things by further switching around the order of the quantifiers and arguments in sentences (3) to (6)? For example, switching the order of the quantifiers in (6) gives

\[(\exists y)(\forall x)L_{yx}\]

Strictly speaking, (7) is a new sentence, but it does not say anything new because it is logically equivalent to (3). It is important to see why this is so:

These diagrams will help you to see that (7) and (3) say exactly the same thing. The point is that there is nothing special about the variable 'x' or the variable 'y'. Either one can do the job of the other. What matters is the pattern of quantifiers and variables. These diagrams show that the pattern is the same. All that counts is that the variable marked at position 1 in the existential quantifier is tied to, or, in logicians' terminology, Binds the variable at position 3; and the variable at position 2 in the universal quantifier binds the variable at position 4. Indeed, we could do without the variables altogether and indicate what we want with the third diagram. This diagram gives the pattern of variable binding which (7) and (3) share.

3-2. QUANTIFIER SCOPE, BOUND VARIABLES, AND FREE VARIABLES

In the last example we saw that the variable at 3 is bound by the quantifier at 1 and the variable at 4 is bound by the quantifier at 2. This case contrasts with that of a variable which is not bound by any quantifier, for example
In (8), the occurrence of 'x' at 3 is bound by the quantifier at 2. However, the occurrence of 'x' at 1 is not bound by any quantifier. Logicians say that the occurrence of 'x' at 1 is free. In (9), the occurrence of 'x' at 3 is free because the quantifier at 1 binds only variables in the shortest full sentence which follows it. Logicians call the shortest full sentence following a quantifier the quantifier's scope. In (9), the 'x' at 3 is not in the scope of the quantifier at 1. Consequently, the quantifier does not bind 'x' at 3.

All the important ideas of this section have now been presented. We need these ideas to understand clearly how to apply the methods of derivations and truth trees when quantifiers get stacked on top of each other. All we need do to complete the job is to give the ideas an exact statement and make sure you know how to apply them in more complicated situations.

Everything can be stated in terms of the simple idea of scope. A quantifier is a connective. We use a quantifier to build longer sentences out of shorter ones. In building up sentences, a quantifier works just like the negation sign: It applies to the shortest full sentence which follows it. This shortest full following sentence is the quantifier's scope:

The Scope of a quantifier is the shortest full sentence which follows it. Everything inside this shortest full following sentence is said to be in the scope of the quantifier.

We can now define 'bound' and 'free' in terms of scope:

A variable, \( u \), is Bound just in case it occurs in the scope of a quantifier, \( (\forall u) \) or \( (\exists u) \).

A variable, \( u \), is Free just in case it is not bound; that is, just in case it does not occur in the scope of any quantifier, \( (\forall u) \) or \( (\exists u) \).

Clearly, a variable gets bound only by using a quantifier expressed with the same variable. 'x' can never be bound by quantifiers such as '(Vy)' or '(3z)'.

Occasionally, students ask about the variables that occur within the quantifiers—the 'x' in '(3x)' and in '(\forall x)'. Are they bound? Are they free? The answer to this question is merely a matter of convention on which nothing important turns. I think the most sensible thing to say is that the variable within a quantifier is part of the quantifier symbol and so does not count as either bound or free. Only variables outside a quantifier can be either bound or free. Some logicians prefer to deal with this question by defining the scope of a quantifier to include the quantifier itself as well as the shortest full sentence which follows it. On this convention one would say that a variable within a quantifier always binds itself.

These definitions leave one fine point unclear. What happens if the variable \( u \) is in the scope of two quantifiers that use \( u \)? For example, consider

\[
(10) \quad (\exists x)(\forall x)Lxa \supset Lxb
\]

The occurrence of 'x' at 3 is in the scope of both the 'x' quantifiers. Which quantifier binds 'x' at 3?

To get straight about this, think through how we build (10) up from atomic constituents. We start with the atomic sentences 'Lxa' and 'Lxb'. Because atomic sentences have no quantifiers, 'x' is free in both of these atomic sentences. Next we apply '(\forall x)' to 'Lxa', forming '(\forall x)Lxa', which we use as the antecedent in the conditional

\[
(11) \quad (\forall x)Lxa \supset Lxb
\]

In (11), the occurrence of 'x' at 3 is bound by the quantifier at 2. The occurrence of 'x' at 4 is free in (11).

Finally, we can clearly describe the effect of '(\exists x)' when we apply it to (11). '(\exists x)' binds just the free occurrences of 'x' in (11). The occurrence at 4 is free and so gets bound by the new quantifier. The occurrence at 3 is already bound, so the new quantifier can't touch it. The following diagram describes the overall effect:

\[
(10) \quad (\exists x)(\forall x)Lxa \supset Lxb
\]

First, the occurrence at 3 is bound by the quantifier at 2. Then the occurrence at 4 is bound by the quantifier at 1. The job being done by the 2–3 link is completely independent of the job being done by the 1–4 link.

Let's give a general statement to the facts we have uncovered:

A quantifier \( (\forall u) \) or \( (\exists u) \) binds all and only all free occurrences of \( u \) in its scope. Such a quantifier does not bind an occurrence of \( u \) in its scope which is already bound by some other quantifier in its scope.

We can make any given case even clearer by using different variables where we have occurrences of a variable bound by different quantifiers. So, for example, (10) is equivalent to

\[
(12) \quad (\exists x)(\forall y)Lxa \supset Lxb
\]
In (12), there can be no confusion about which quantifier binds which variable—we keep track of everything by using different variables. Why, then, didn’t we just resolve to use different variables from the beginning and save ourselves a lot of trouble? We could have done that, but then the definition of the sentences of predicate logic would have been much more complicated. Either way, we have work to do. Besides, the formulation I have presented here is the one traditionally used by logicians and so the one you will need to know if you study more logic.

Let’s look at another, slightly more complicated, example to make sure you have put this all together. Draw in the lines which show which quantifier binds which variable in the following:

(13) \( (\exists x)(\exists x)(Bx \lor Lxa) \supset (Bx & Lxb) \)

If you are having trouble, think through how (13) gets built up from its parts. In

(14) \( (\exists x)(Bx \lor Lxa) \supset (Bx & Lxb) \)

the quantifier at 2 applies only to the shortest full sentence which follows it, which ends before the ‘\( \supset \)’. So the occurrences of ‘\( x \)’ at 3 and 4 are both bound by the quantifier at 2. The two occurrences of ‘\( x \)’ at 5 and 6 are not in the scope of a quantifier and are both free. So when we apply the second ‘\( (\exists x) \)’ to all of (14), the new ‘\( (\exists x) \)’ binds only the ‘\( x \)’s which are still free in (14), namely, the ‘\( x \)’s which occur at 5 and 6. In sum, the pattern of binding is

(13) \( (\exists x)(\exists x)(Bx \lor Lxa) \supset (Bx & Lxb) \)

1 2 3 4 5 6

We can make this pattern even clearer by writing the sentence equivalent to (13):

(15) \( (\exists x)(\exists x)(Bx \lor Lza) \supset (Bx & Lxb) \)

In practice, of course, it is much better to use sentences such as (15) and (12) instead of the equivalent (13) and (10), which are more difficult to interpret.

**EXERCISES**

3–1. In the following sentences draw link lines to show which quantifiers bind which variables and say which occurrences of the variables are bound and which are free:

- a) Lxx
- b) (Vy)(Vz)Lzy
- c) (Vz)(Bz \& Lxz)
- d) (3x)[Lxz & (Vz)(Lxy v Lxz)]
- e) (Vx)(Lax & Bz) = (Lxx \& (3x)Bx)
- f) (Vx)(Lxy \& (Bz \& (3x)Lyx)]

3-3. **CORRECT DEFINITIONS OF SUBSTITUTION INSTANCE AND TRUTH IN AN INTERPRETATION**

In chapter 2 I gave an incorrect definition of ‘substitution instance.’ I said that we get the substitution instance of \( (\forall u) \) \( . . . u . . . \) by simply dropping the initial \( (u) \) and writing in \( s \) wherever we find \( u \) in \( . . . u . . . \). This is correct as long as neither a second \( (\forall u) \) nor a \( (\exists u) \) occurs within the scope of the initial \( (\forall u) \), that is, within the sentence \( . . . u . . . \). Since I used only this kind of simple sentence in chapter 2, there we could get away with the simple but incorrect definition. But now we must correct our definition so that it will apply to any sentence. Before reading on, can you see how multiple quantification can make trouble for the simple definition of substitution instance, and can you see how to state the definition correctly?

To correct the definition of substitution instance, all we have to do is to add the qualification that the substituted occurrences of the variable be free:

For any universally quantified sentence \( (\forall u) \) \( . . . u . . . \), the **Substitution Instance** of the sentence, with the name \( s \) substituted for the variable \( u \), is \( . . . s . . . \), the sentence formed by dropping the initial universal quantifier and writing \( s \) for all free occurrences of \( u \) in \( . . . u . . . \).

For any existentially quantified sentence \( (\exists u) \) \( . . . u . . . \), the **Substitution Instance** of the sentence, with the name \( s \) substituted for the variable \( u \), is \( . . . s . . . \), the sentence formed by dropping the initial existential quantifier and writing \( s \) for all free occurrences of \( u \) in \( . . . u . . . \).

For example, look back at (13). Its substitution instance with ‘\( c \)’ substituted for ‘\( x \)’ is

(16) \( (\exists x)(Bx \lor Lxa) \supset (Bc & Lcb) \)

2 3 4 5 6

The occurrences of ‘\( x \)’ at 3 and 4 are not free in the sentence which results from (13) by dropping the initial quantifier. So we don’t substitute
'c' for 'x' at 3 and 4. We substitute 'c' only at the free occurrences, which were at 5 and 6.

Can you see why, when we form substitution instances, we pay attention only to the occurrences which are free after dropping the outermost quantifier? The occurrences at 3 and 4, bound by the '(∃x)' quantifier at 2, have nothing to do with the outermost quantification. When forming substitution instances of a quantified sentence, we are concerned only with the outermost quantifier and the occurrences which it binds.

To help make this clear, once again consider (15), which is equivalent to (13). In (15), we have no temptation to substitute 'c' for 'x' when forming the 'c'-substitution instance for the sentence at a whole. (15) says that there is some x such that so on and so forth about x. In making this true for some specific x, say c, we do not touch the occurrences of 'x'. The internal 'x'-quantified sentence is just part of the so on and so forth which is asserted about x in the quantified form of the sentence, that is, in (15).

Now let's straighten out the definition of truth of a sentence in an interpretation. Can you guess what the problem is with our old definition? I'll give you a clue. Try to determine the truth value of 'Lxe' in the interpretation of figure 2–1. You can't do it! Nothing in our definition of an interpretation gives us a truth value for an atomic sentence with a free variable. An interpretation only gives truth values for atomic sentences which use no variables. You will have just as much trouble trying to determine the truth value of '(∃x)Lxy' in any interpretation. A substitution instance of '(∃x)Lxy' will still have the free variable 'y', and no interpretation will assign such a substitution instance a truth value.

Two technical terms (mentioned in passing in chapter 1) will help us in talking about our new problem:

A sentence which has one or more free variables is called an Open Sentence.

A sentence which is not open (i.e., a sentence with no free variables) is called a Closed Sentence.

In a nutshell, our problem is that our definitions of truth in an interpretation do not specify truth values for open sentences. Some logicians deal with this problem by treating all free variables in an open sentence as if they were universally quantified. Others do what I will do here: We simply say that open sentences have no truth value.

If you think about it, this is really very natural. What, anyway, is the truth value of the English "sentence" 'He is blond,' when nothing has been said or done to give you even a clue as to who 'he' refers to? In such a situation you can't assign any truth value to 'He is blond.' 'He is blond,' functions syntactically as a sentence—it has the form of a sentence. But there is still something very problematic about it. In predicate logic we allow such open sentences to function syntactically as sentences. Doing this is very useful in making clear how longer sentences get built up from shorter ones. But open sentences never get assigned a truth value, and in this way they fail to be full-fledged sentences of predicate logic.

It may seem that I am dealing with the problem of no truth value for open sentences by simply ignoring the problem. In fact, as long as we acknowledge up-front that this is what we are doing, saying that open sentences have no truth value is a completely adequate way to proceed.

We have only one small detail to take care of. As I stated the definitions of truth of quantified sentences in an interpretation, the definitions were said to apply to any quantified sentences. But they apply only to closed sentences. So we must write in this restriction:

A universally quantified closed sentence is true in an interpretation just in case all of the sentence's substitution instances, formed with names in the interpretation, are true in the interpretation.

An existentially quantified closed sentence is true in an interpretation just in case some (i.e., one or more) of the sentence's substitution instances, formed with names in the interpretation, are true in the interpretation.

These two definitions, together with the rules of valuation given in chapters 1 and 4 of volume I for the sentence logic connectives, specify a truth value for any closed sentence in any of our interpretations.

You may remember that in chapter 1 in volume I we agreed that sentences of logic would always be true or false. Sticking by that agreement now means stipulating that only the closed sentences of predicate logic are real sentences. As I mentioned in chapter 1 in this volume, some logicians use the phrase Formulas, or Propositional Functions for predicate logic open sentences, to make the distinction clear. I prefer to stick with the word 'sentence' for both open and closed sentences, both to keep terminology to a minimum and to help us keep in mind how longer (open and closed) sentences get built up from shorter (open and closed) sentences. But you must keep in mind that only the closed sentences are full-fledged sentences with truth values.

EXERCISES

3–2. Write a substitution instance using 'a' for each of the following sentences:

a) (∀y)(∃x)Lxy  
b) (∃z)(∀x)(Bx v Bz)

c) (∃x)(Bx = (∀x)(Lax v Bx))  
d) (∀y)(∃x)(Bx ⊃ By) & (∃x)(By ⊃ Bx)

e) (∀y)(∃x)(Bx v [(∃y)By ⊃ Lyy])
f) \((\forall x)(\exists y)[(Rxy \supset Dy) \supset Ryx]\)
g) \((\forall x)(\forall y)[(Sxy \vee (Kz \supset Lxz)) = (Scx \& Hy)]\)
h) \((\exists x)(\forall z)[(Pxa \supset Kz) \& (\exists y)(Pxy \supset Kc) \& Pxz]\)
i) \((\forall x)(\exists y)[(\exists x)Mzx \supset (\exists x)(Mxy \supset Myz)] \& (\exists x)Mzx\)
j) \((\forall x)[[(\forall x)Rxa \supset Rxb] \vee [(\exists x)(Rcx \& Rxa) \supset Rxx]]\)

3–3. If \(u\) does not occur free in \(X\), the quantifiers \((\forall u)\) and \((\exists u)\) are said to occur \textit{vacuously} in \((\forall u)X\) and \((\exists u)X\). Vacuous quantifiers have no effect. Let’s restrict our attention to the special case in which \(X\) is closed, so that it has a truth value in any of its interpretations. The problem I want to raise is how to apply the definitions for interpreting quantifiers to vacuously occurring quantifiers. Because truth of a quantified sentence is defined in terms of substitution instances of \((\forall u)X\) and \((\exists u)X\), when \(u\) does not occur free in \(X\), we most naturally treat this vacuous case by saying that \(X\) counts as a substitution instance of \((\forall u)(X)\) and \((\exists u)(X)\). (If you look back at my definitions of ‘substitution instance’, you will see that they really say this if by ‘for all free occurrences of \(u\)’ you understand ‘for no occurrences of \(u\)’ when \(u\) does not occur free in \(X\).) With this understanding, show that \((\forall u)X\), \((\exists u)X\), and \(X\) all have the same truth value in any interpretation of \(X\).

3–4. a) As I have defined interpretation, every object in an interpretation has a name. Explain why this chapter’s definitions of truth of existentially and universally quantified sentences would not work as intended if interpretations were allowed to have unnamed objects.
b) Explain why one might want to consider interpretations with unnamed objects.

In part II we will consider interpretations with unnamed objects and revise the definitions of truth of quantified sentences accordingly.

### 3–4. Some Logical Equivalences

The idea of logical equivalence transfers from sentence logic to predicate logic in the obvious way. In sentence logic two sentences are logically equivalent if and only if in all possible cases the sentences have the same truth value, where a possible case is just a line of the truth table for the sentence, that is, an assignment of truth values to sentence letters. All we have to do is to redescribe possible cases as interpretations:

Two closed predicate logic sentences are \textit{Logically Equivalent} if and only if in each of their interpretations the two sentences are either both true or both false.

Notice that I have stated the definition only for closed sentences. Indeed, the definition would not make any sense for open sentences because open sentences don’t have truth values in interpretations. Nonetheless, one can extend the idea of logical equivalence to apply to open sentences. That’s a good thing, because otherwise the law of substitution of logical equivalents would break down in predicate logic. We won’t be making much use of these further ideas in this book, so I won’t hold things up with the details. But you might amuse yourself by trying to extend the definition of logical equivalence to open sentences in a way which will make the law of substitution of logical equivalents work in just the way you would expect.

Let us immediately take note of two equivalences which will prove very useful later on. By way of example, consider the sentence, ‘No one loves Eve’, which we transcribe as ‘\(- (\exists x) Lxe\)’, that is, as ‘\(I\) is not the case that someone loves Eve’. How could this unromantic situation arise? Only if \textit{everyone didn’t} love Eve. In fact, saying ‘\(- (\exists x) Lxe\)’ comes to the same thing as saying ‘\((\forall x)\sim Lxe\)’. If there is not a single person who does love Eve, then it has to be that everyone does not love Eve. And conversely, if positively everyone does not love Eve, then not even one person does love Eve.

There is nothing special about the example I have chosen. If our sentence is of the form ‘\(- (\exists u) . . . \)’, this says that there is not a single \(u\) such that so on and so forth about \(u\). But this comes to the same as saying about each and every \(u\) that so on and so forth is not true about \(u\), that is, that ‘\((\forall u)\sim . . . \)’.

We can easily prove the equivalence of ‘\(- (\exists u) . . . \)’ and ‘\((\forall u)\sim . . . \)’ by appealing to De Morgan’s laws. We have to prove that these two sentences have the same truth value in each and every interpretation. In any one interpretation, ‘\(- (\exists u) . . . \)’ is true just in case the negation of the disjunction of the instances

\[\sim[(. . . a . . .) \vee (. . . b . . .) \vee (. . . c . . .) \vee . . .]\]

is true in the interpretation, where we have included in the disjunction all the instances formed using names which name things in the interpretation. By De Morgan’s laws, this is equivalent to the conjunction of the negation of the instances

\[\sim(. . . a . . .) \& \sim(. . . b . . .) \& \sim(. . . c . . .) \& . . .\]

which is true in the interpretation just in case ‘\((\forall u)\sim . . .\)’ is true in the interpretation. Because this is true in all interpretations, we see that

Rule \(- \exists: \sim (\exists u)(. . . u . . .)\) is logically equivalent to ‘\((\forall u)\sim . . .\)’.

Now consider the sentence ‘Not everyone loves Eve,’ which we transcribe as ‘\(\sim(\forall x)Lxe\)’. If not everyone loves Eve, then there must be some-
one who does not love Eve. And if there is someone who does not love Eve, then not everyone loves Eve. So \( \sim (\forall x) Lxe \) is logically equivalent to \( (\exists x) \sim Lxe \).

Pretty clearly, again there is nothing special about the example. \( \sim (\forall u)(\ldots u \ldots) \) is logically equivalent to \( (\exists u) \sim (\ldots u \ldots) \). If it is not the case that, for all \( u \), so on and so forth about \( u \), then there must be some \( u \) such that not so on and so forth about \( u \). And, conversely, if there is some \( u \) such that not so on and so forth about \( u \), then it is not the case that for all \( u \), so on and so forth about \( u \). In summary

\[ \text{Rule } \sim \forall: \sim (\forall u)(\ldots u \ldots) \text{ is logically equivalent to } (\exists u) \sim (\ldots u \ldots). \]

You can easily give a proof of this rule by imitating the proof of the rule \( \sim \exists \). But I will let you write out the new proof as an exercise.

EXERCISES

3–5. a) Give a proof of the rule of logical equivalence, \( \sim \forall \). Your proof will be very similar to the proof given in the text for the rule \( \sim \exists \).

b) The proof for the rule \( \sim \exists \) is flawed! It assumes that all interpretations have finitely many things in their domain. But not all interpretations are finite in this way. (Exercise 2–5 gives an example of an infinite interpretation.) The problem is that the proof tries to talk about the disjunction of all the substitution instances of a quantified sentence. But if an interpretation is infinite, there are infinitely many substitution instances, and no sentence can be infinitely long. Since I instructed you, in part (a) of this problem, to imitate the proof in the text, probably your proof has the same problem as mine.

Your task is to correct this mistake in the proofs. Give informal arguments for the rules \( \sim \exists \) and \( \sim \forall \) which take account of the fact that some interpretations have infinitely many things in their domain.

3–6. In the text I defined logical equivalence for closed sentences of predicate logic. However, this definition is not broad enough to enable us to state a sensible law of substitution of logical equivalents for predicate logic. Let me explain the problem with an example. The following two sentences are logically equivalent:

1. \( \sim (\forall x)(\forall y) Lxy \)
2. \( (\exists x)(\exists y) \sim Lxy \)

But we cannot prove that (1) and (2) are logically equivalent with the rule \( \sim \forall \) as I have stated it. Here is the difficulty. The rule \( \sim \forall \) tells us that (1) is logically equivalent to

3. \( (\exists x) \sim (\forall y) Lxy \)

What we would like to say is that \( \sim (\forall y) Lxy \) is logically equivalent to \( (\exists y) \sim Lxy \), again by the rule \( \sim \forall \). But the rule \( \sim \forall \) does not license this because I have defined logical equivalence only for closed sentences and \( \sim (\forall y) Lxy \) and \( (\exists y) \sim Lxy \) are open sentences. (Strictly speaking, I should have restricted the \( \sim \forall \) and \( \sim \exists \) rules to closed sentences. I didn’t because I anticipated the results of this exercise.) Since open sentences are never true or false, the idea of logical equivalence for open sentences does not make any sense, at least not on the basis of the definitions I have so far introduced.

Here is your task:

a) Extend the definition of logical equivalence for predicate logic sentences so that it applies to open as well as closed sentences. Do this in such a way that the law of substitution of logical equivalents will be correct when one open sentence is substituted for another when the two open sentences are logically equivalent according to your extended definition.

b) Show that the law of substitution of logical equivalents works when used with open sentences which are logically equivalent according to your extended definition.

CHAPTER SUMMARY EXERCISES

Here are this chapter’s important terms. Check your understanding by writing short explanations for each, saving your results in your notebook for reference and review.

a) Bound Variables
b) Free Variables
c) Scope
d) Closed Sentence
e) Open Sentence
f) Truth of a Sentence in an Interpretation
g) Rule \( \sim \exists \)
h) Rule \( \sim \forall \)
4-1. RESTRICTED QUANTIFIERS

For three chapters now I have been merrily transcribing '(3x)' both as 'something' and 'someone', and I have been transcribing '(Vx)' both as 'everything' and 'everyone.' I justified this by saying that when we talked only about people we would restrict the variables 'x', 'y', etc. to refer only to people, and when we talked about everything, we would let the variables be unrestricted. It is actually very easy to make precise this idea of restricting the universe of discourse. If we want the universe of discourse to be restricted to people, we simply declare that all the objects in our interpretations must be people. If we want a universe of discourse consisting only of cats, we declare that all the objects in our interpretations must be cats. And so on.

As I mentioned, this common practice is not fully satisfactory. What if we want to talk about people and things, as when we assert, 'Everyone likes sweet things.'? Restricted quantifiers will help us out here. They also have the advantage of getting what we need explicitly stated in the predicate logic sentences themselves.

We could proceed by using '(3x)' and '(Vx)' to mean 'something' and 'everything' and introduce new quantifiers for 'someone' and 'everyone'. To see how to do this, let's use the predicate 'P' to stand for 'is a person.' Then we can introduce the new quantifier '(∃x)_P' to stand for some x chosen from among the things that are P, that is, chosen from among people. We call this a restricted quantifier. You should think of a restricted quantifier as saying exactly what an unrestricted quantifier says except that the variable is restricted to the things of which the subscripted predicate is true. With 'P' standing for 'is a person', '(∃x)_P' has the effect of 'someone' or 'somebody'. We can play the same game with the universal quantifier. '(∀x)_P' will mean all x chosen from among the things that are P. With 'P' standing for 'is a person', '(∀x)_P' means, not absolutely everything, but all people, that is, everyone or everybody or anyone or anybody.

This notion of a restricted quantifier can be useful for other things. Suppose we want to transcribe 'somewhere' and 'everywhere' or 'sometimes' and 'always'. Let's use 'N' stand for 'is a place' or 'is a location'. 'Somewhere' means 'at some place' or 'at some location'. So we can transcribe 'somewhere' as '(∃x)_N' and 'everywhere as '(∀x)_N'. For example, to transcribe 'There is water everywhere', I would introduce the predicate 'Wx' to stand for 'there is water at x'. Then '(∀x)_Wx' says that there is water everywhere. Continuing the same strategy, let's use 'Q' to stand for 'is a time'. Then '(∃x)_Q' stands for 'sometime(s)' and '(∀x)_Q' stands for 'always' ('at all times').

In fact, we can also use the same trick when English has no special word corresponding to the restricted quantifier. Suppose I want to say something about all cats, for example, that all cats are furry. Let 'Cx' stand for 'x is a cat' and 'Fx' stand for 'x is furry'. Then '(∀x)_CxFx' says that all things which are cats are furry; that is, all cats are furry. Suppose I want to say that some animals have tails. Using 'Ax' for 'x is an animal' and 'Tx' for 'x is a tail of y', I write '(∃x)_A(∃y)Txy': There is an animal, x, and there is a thing, y, such that y is a tail of x.

As you will see, restricted quantifiers are very useful in figuring out transcriptions, but there is a disadvantage in introducing them as a new kind of quantifier in our system of logic. If we have many different kinds of quantifiers, we will have to specify a new rule for each of them to tell us the conditions under which a sentence formed with the quantifier is true. And when we get to checking the validity of arguments, we will have to have a new rule of inference to deal with each new quantifier. We could state the resulting mass of new rules in a systematic way. But the whole business would still require a lot more work. Fortunately, we don't have to do any of that, for we can get the full effect of restricted quantifiers with the tools we already have.

Let's see how to rewrite subscripted quantifiers. Consider the restricted quantifier '(∃x)_C', which says that there is cat such that, or there are cats such that, or some cats are such that. We say 'some cats are furry' (or 'there is a furry cat' or the like) with '(∃x)_C Fx'. Now what has to be true
for it to be true that some cats are furry, or that there is a furry cat? There has to be one or more things that is both a cat and is furry. If there is not something which is both a cat and is furry, it is false that there is a furry cat. So we can say that some cats are furry by writing '(∃x)(Cx & Fx)'. In short, we can faithfully rewrite '(∃x)Fx' as '(∃x)(Cx & Fx)'.

This strategy will work generally:

Rule for rewriting Subscripted Existential Quantifiers: For any predicate S, any sentence of the form (S_u)(... u ...) is shorthand for (S_u)[Su & (... u ...)].

Here are some examples:

Some cats are blond. (∃x)Bx (∃x)(Cx & Bx)
Eve loves a cat. (∃x)Lxe (∃x)(Cx & Lxe)
Eve loves a furry cat. (∃x)(Fx & Lxe) (∃x)(Cx & (Fx & Lxe))

Clearly, we can proceed in the same way with 'someone' and 'somebody':

Someone loves Eve. (∃x)Lxe (∃x)(Px & Lxe)
Somebody loves Eve or Adam. (∃x)((Lxe v Lxa) (Px)[&Lxe v Lxa])
If somebody loves Eve, then Eve loves somebody. (∃x)(Lxe v Lxa) (∃x)(Px & Lxe) (∃x)(Px & Lxe)
(∃x)Lxe

Notice that in the last example I used the rule for rewriting the subscript on each of two sentences X and Y, inside a compound sentence, X ⊃ Y.

How should we proceed with restricted universal quantifiers? This is a little tricky. Let's work on '(∀x)Fx'—that is, 'All cats are furry'. Under what conditions is this sentence true? To answer the question, imagine that everything in the world is lined up in front of you: All the cats, dogs, people, stones, basketballs, everything. You go down the line and examine the items, one by one, to determine whether all cats are furry. If the first thing in line is a dog, you don't have to determine whether or not it is furry. If the second thing is a basketball, you don't have to worry about it either. But as soon as you come to a cat you must examine it further to find out if it is furry. When you finally come to the end of the line, you will have established that all cats are furry if you have found of each thing that, if it is a cat, then it is furry. In short, to say that all cats are furry is to say '(∀x)(Cx ⊃ Fx)'.

At this point, many students balk. Why, they want to know, should we rewrite a restricted universal quantifier with the '⊂' when we rewrite a restricted existential quantifier with the '&'? Shouldn't '&' work also for restricted universal quantifiers? Well, I'm sorry. It doesn't. That is just not what restricted universal quantifiers mean.

You can prove this for yourself by trying to use '& ' in rewriting the subscripted 'C' in our transcription of 'All cats are furry.' You get

(1) (∀x)(Cx & Fx)

What does (1) say? It says that everything is a furry cat, and in particular that everything is a cat! That's much too strong. All cats could be furry even though there are lots of things which are not cats. Thus 'All cats are furry' could be true even when (1) is false, so that (1) cannot be the right way to rewrite '(∀x)Fx'.

What has gone wrong? The unrestricted universal quantifier applies to everything. So we can use conjunction in expressing the restriction of cats only if we somehow disallow or except the cases of noncats. We can do this by saying that everything is either not a cat or is a cat and is furry:

(2) (∀x)[¬C x v (Cx & Fx)]

(2) does indeed say what 'All cats are furry' says. So (2) should satisfy your feeling that an '&' also comes into the restricted universal quantifier in some way. But you can easily show that (2) is logically equivalent to '(∀x)(Cx ⊃ Fx)'! As the formulation with the '⊂' is more compact, and is also traditional, it is the one we will use.

In general, we rewrite restricted universal quantifiers according to the rule:

Rule for rewriting Subscripted Universal Quantifiers: For any predicate S, any sentence of the form (∀u)(... u ...) is shorthand for (∀u)[Su ⊃ (... u ...)].

Here are some examples to make sure you see how this rule applies:

Eve loves all cats. (∀x)Lxe (∀x)(Cx ⊃ Lxe)
Everybody loves Eve. (∀x)Lxe (∀x)(Px ⊃ Lxe)
Everyone loves either Adam or Eve. (∀x)(Lxa v Lxe) (∀x)(Px ⊃ (Lxa v Lxe))
Not everyone loves both Adam and Eve. (∼(∀x)(Lxa & Lxe)) (∼(∀x)(Px ⊃ (Lxa & Lxe))

In the last example, I used the rewriting rule on a sentence, X, inside a negated sentence of the form ∼X.

If you are still feeling doubtful about using the '⊂' to rewrite restricted universal quantifiers, I have yet another way to show you that this way of rewriting must be right. I am assuming that you agree that our way of rewriting restricted existential quantifiers is right. And I will use a new rule of logical equivalence. This rule tells us that the same equivalences that hold for negated unrestricted universal quantifiers hold for negated restricted universal quantifiers. In particular, saying that not all cats are furry is clearly the same as saying that some cat is not furry. In general
Rule \( \neg \forall \mathbf{u} \): A sentence of the form \( \neg (\forall \mathbf{u}) \) is logically equivalent to \( (\exists \mathbf{u}) \neg \).

You can prove this new rule along the same lines we used in proving the rule \( \neg \).

Now, watch the following chain of logical equivalents:

1. \((\forall \mathbf{u})\)
2. \((\exists \mathbf{u})\)
3. \((\forall \mathbf{u})\) Rule for rewriting \( \exists \mathbf{u} \)
4. \((\forall \mathbf{u})\) DN
5. \((\forall \mathbf{u})\) DN, DN
6. \((\forall \mathbf{u})\) C
7. \((\forall \mathbf{u})\) DN

Since the last line is logically equivalent to the first, it must be a correct way of rewriting the first.

If you are having a hard time following this chain of equivalents, let me explain the strategy. Starting with a restricted universal quantifier, I turn it into a restricted existential quantifier in lines 2 and 3 by using double denial and pushing one of the two negation signs through the restricted quantifier. I then get line 4 by using the rule we have agreed on for rewriting restricted existential quantifiers. Notice that I am applying this rule inside a negated sentence, so that here (and below) I am really using substitution of logical equivalents. In lines 5, 6, and 7 I use rules of logical equivalence to transform a conjunction into a conditional. These steps are pure sentence logic. They involve no quantifiers. Line 8 comes from line 7 by pushing the negation sign back out through what is now an unrestricted existential quantifier, changing it into an unrestricted universal quantifier. Finally, in line 9, I drop the double negation. It's almost like magic!

**EXERCISES**

4-1. Give an argument which shows that the rule \( \neg \exists \mathbf{u} \) is correct.

Similarly, show that the rule \( \neg \exists \mathbf{u} \) is also correct.

4-2. Use the rule \( \exists \mathbf{u} \) to show that, starting from the rule for rewriting subscripted universal quantifiers, you can derive the rule for rewriting subscripted existential quantifiers. Your argument will closely follow the one given in the text for arguing the rule for rewriting subscripted universal quantifiers from the rule for rewriting subscripted existential quantifiers.

4-3. Transcribe the following English sentences into the language of predicate logic. Use this procedure: In a first step, transcribe into a sentence using one or more subscripted quantifiers. Then rewrite the resulting sentence using the rules for rewriting subscripted quantifiers. Show both your first and second steps. Here are two examples of sentences to transcribe and the two sentences to present in presenting the problem:

**Transcription Guide**

<table>
<thead>
<tr>
<th>c: Eve</th>
<th>Dx: x is a dog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Px: x is a person</td>
<td>Bx: x is blond</td>
</tr>
<tr>
<td>Cx: x is a cat</td>
<td>Lxy: x loves y</td>
</tr>
</tbody>
</table>

transcription of:

- Someone loves Eve.
- All cats love Eve.
- Everyone loves Eve.
- Eve loves somebody.
- Eve loves everyone.
- Somebody loves some dog.
- Somebody is neither a cat nor a dog.
- Someone blond loves Eve.
- Some cat is blond.
- Someone loves all cats.
- No cat is a dog.
- Someone loves someone.
- Everyone loves everyone.
- Everybody loves someone.
- Someone is loved by everyone.
- Everyone loves someone.
- Everyone is loved by somebody.

**4-2. Transcribing from English into Logic**

Transcribing into the language of predicate logic can be extremely difficult. Actually, one can do logic perfectly well without getting very good at transcription. But transcriptions into logic provide one of predicate logic's
important uses. This is because, when it comes to quantification, English is often extremely confusing, ambiguous, and even downright obscure. Often we can become clearer about what is being said if we put a statement into logic. Sometimes transcribing into logic is a must for clarity and precision. For example, how do you understand the highly ambiguous sentence, 'All of the boys didn't kiss all of the girls.' I, for one, am lost unless I transcribe into logic.

Before we get started, I should mention a general point. Just as in the case of sentence logic, if two predicate logic sentences are logically equivalent they are both equally good (or equally bad!) transcriptions of an English sentence. Two logically equivalent sentences share the same truth value in all possible cases (understood as all interpretations), and in this sense two logically equivalent sentences "say the same thing." But if two predicate logic sentences say the same thing, then to the extent that one of them says what an English sentence says, then so does the other.

We are going to be looking at quite a few examples, so let's agree on a transcription guide:

Transcription Guide

<table>
<thead>
<tr>
<th>a:</th>
<th>Px: x is a person</th>
</tr>
</thead>
<tbody>
<tr>
<td>J:</td>
<td>Rx: x is a registered voter</td>
</tr>
<tr>
<td>Ax:</td>
<td>Vx: x has the right to vote</td>
</tr>
<tr>
<td>Bx:</td>
<td>Kxy: x kissed y</td>
</tr>
<tr>
<td>Cx:</td>
<td>Lxy: x loves y</td>
</tr>
<tr>
<td>Dx:</td>
<td>Mxy: x is married to y</td>
</tr>
<tr>
<td>Fx:</td>
<td>Oxy: x owns y</td>
</tr>
<tr>
<td>Gx:</td>
<td>Txy: x is a tail of y</td>
</tr>
<tr>
<td>Hx:</td>
<td>x is at home</td>
</tr>
</tbody>
</table>

Take note of the fact that in giving you a transcription guide, I have been using open sentences to indicate predicates. For example, I am using the open sentence 'Px' to indicate the predicate 'is a person.' The idea of using an open sentence to indicate a predicate will soon become very useful.

To keep us focused on the new ideas, I will often use subscripts on restricted quantifiers. However, you should keep in mind that complete transcriptions require you to rewrite the subscripts, as explained in the last section.

Now let's go back and start with the basics. '(\forall x)(Cx \supset Fx)' transcribes 'all cats are furry,' 'Every cat is furry,' 'Any cat is furry,' and 'Each cat is furry.' This indicates that

Usually, the words 'all', 'every', 'any', and 'each' signal a universal quantifier.

Let's make a similar list for the existential quantifier. '(\exists x)(Cx \& Fx)' transcribes 'Some cat is furry,' 'Some cats are furry,' 'At least one cat is furry,' 'There is a furry cat,' and 'There are furry cats':

Usually, the expressions 'some', 'at least one', 'there is', and 'there are' signal an existential quantifier.

These lists make a good beginning, but you must use care. There are no hard and fast rules for transcribing English quantifier words into predicate logic. For starters, 'a' can easily function as a universal or an existential quantifier. For example, 'A car can go very fast.' is ambiguous. It can be used to say either that any car can go very fast or that some car can go very fast.

To make it clearer that 'a' can function both ways, consider the following examples. You probably understand 'A man is wise.' to mean that some man is wise. But most likely you understand 'A dog has four legs.' to mean that all dogs have four legs. Actually, both of these sentences are ambiguous. In both sentences, 'a' can correspond to 'all' or 'some'. You probably didn't notice that fact because when we hear an ambiguous sentence we tend to notice only one of the possible meanings. If a sentence is obviously true when understood with one of its meanings and obviously false when understood with the other, we usually hear the sentence only as making the true statement. So if all the men in the world were wise, we would take 'A man is wise.' to mean that all men are wise, and if only one dog in the world had four legs we would take 'A dog has four legs.' to mean that some dog has four legs.

It is a little easier to hear 'A car can go very fast.' either way. This is because we interpret this sentence one way or the other, depending on how fast we take 'fast' to be. If 'fast' means 30 miles an hour (which is very fast by horse and buggy standards), it is easy to hear 'A car can go very fast.' as meaning that all cars can go very fast. If "fast' means 180 miles an hour it is easy to hear 'a car can go very fast.' as meaning that some car can go very fast.

'A' is not the only treacherous English quantifier word. 'Anyone' usually gets transcribed with a universal quantifier. But not always. Consider:

(3) If anyone is at home, the lights will be on.
(4) If anyone can run a 3:45 mile, Adam can.

We naturally hear (3), not as saying that if everyone is at home the lights will be on, but as saying that if someone is at home the lights will be on. So a correct transcription is

(3a) (\exists x) Hx \supset J
Likewise, by (4), we do not ordinarily mean that if everyone can run a 3:45 mile, Adam can. We mean that if someone can run that fast, Adam can:

\[(\exists x)pFx \supset Fa\]

At least that's what one would ordinarily mean by (4). However, I think that (4) actually is ambiguous. I think 'anyone' in (4) could be understood as 'everyone'. This becomes more plausible if you change the '3:45 mile' to '10-minute mile'. And it becomes still more plausible after you consider the following example: 'Anyone can tie their own shoe laces. And if anyone can, Adam can.'

Going back to (3), one would think that if (4) is ambiguous, (3) should be ambiguous in the same way. I just can't hear an ambiguity in (3). Can you?

'Someone' can play the reverse trick on us. Usually, we transcribe it with an existential quantifier. But consider

\[(\forall x)p(Rx \supset Vx)\]

As in the case of (4), which uses 'anyone', we can have ambiguity in sentences such as (5), which uses 'someone'. If you don't believe me, imagine that you live in a totalitarian state, called Totalitarania. In Totalitarania, everyone is a registered voter. But voter registration is a sham. In fact, only one person, the boss, has the right to vote. As a citizen of Totalitarania, you can still truthfully say that someone who is a registered voter (namely, the boss) has the right to vote. (You can make this even clearer by emphasizing the word 'someone': 'someone who is a registered voter has the right to vote.') In this context we hear the sentence as saying

\[(\exists x)p(Rx & Vx)\]

Ambiguity can plague transcription in all sorts of ways. Consider an example traditional among linguists:

\[(\forall x)p(\forall y)cKxy\]

But it can also mean that each of the boys kissed some girls so that, finally, each and every girl got kissed by some boy:

\[(\forall x)p(\exists y)cKxy & (\forall y)p(\exists x)cKxy\]

If you think that was bad, things get much worse when multiple quantifiers get tangled up with negations. Consider

\[(7) \text{ All the boys didn't kiss all the girls.}\]

Everytime I try to think this one through, I blow a circuit. Perhaps the most natural transcription is to take the logical form of the English at face value and take the sentence to assert that of each and every boy it is true that he did not kiss all the girls; that is, for each and every boy there is at least one girl not kissed by that boy:

\[(7a) (\forall x)p(\neg(\forall y)cKxy), \text{ or } (\forall x)p(\exists y)cKxy\]

But one can also take the sentence to mean that each and every boy refrained from kissing each and every girl, that is, didn't kiss the first girl and didn't kiss the second girl and not the third, and so on. In yet other words, this says that for each and every boy there was no girl whom he kissed, so that nobody kissed anybody:

\[(7b) (\forall x)p(\neg(\exists y)cKxy), \text{ or } (\forall x)p(\neg(\forall y)cKxy, \text{ or } (\exists x)p(\exists y)cKxy\]

We are still not done with this example, for one can also use (7) to mean that not all the boys kissed every single girl—that is, that some boy did not kiss all the girls, in other words that at least one of the boys didn't kiss at least one of the girls:

\[(7c) (\exists x)p(\neg(\forall y)cKxy), \text{ or } (\exists x)p(\neg(\exists y)cKxy), \text{ or } (\exists x)p(\exists y)cKxy\]

It's worth an aside to indicate how it can happen that an innocent-looking sentence such as (7) can turn out to be so horribly ambiguous. Modern linguistics postulates that our minds carry around more than one representation of a given sentence. There is one kind of structure that represents the logical form of a sentence. Another kind of structure represents sentences as we speak and write them. Our minds connect these (and other) representations of a given sentence by making all sorts of complicated transformations. These transformations can turn representations of different logical forms into the same representation of a spoken or written sentence. Thus one sentence which you speak or write can correspond to two, three, or sometimes quite a few different structures that carry very different meanings. In particular, the written sentence (7) corresponds to (at least!) three different logical forms. (7a), (7b), and (7c)
don't give all the details of the different, hidden structures that can be transformed into (7). But they do describe the differences which show up in the language of predicate logic.

You can see hints of all this if you look closely at (7), (7a), (7b), and (7c). In (7) we have two universal quantifier words and a negation. But since the quantifier words appear on either side of 'kissed', it's really not all that clear where the negation is meant to go in relation to the universal quantifiers. We must consider three possibilities. We could have the negation between the two universal quantifiers. Indeed, that is what you see in (7a), in the first of its logically equivalent forms. Or we could have the negation coming after the two universal quantifiers, which is what you find in the first of the logically equivalent sentences in (7b). Finally, we could have the negation preceding both universal quantifiers. You see this option in (7c). In sum, we have three similar, but importantly different, structures. Their logical forms all have two universal quantifiers and a negation, but the three differ, with the negation coming before, between, or after the two quantifiers. The linguistic transformations in our minds connect all three of these structures with the same, highly ambiguous English sentence, (7).

Let's get back to logic and consider some other words which you may find especially difficult to transcribe. I am always getting mixed up by sentences which use 'only', such as 'Only cats are furry.' So I use the strategy of first transcribing a clear case (it helps to use a sentence I know is true) and then using the clear case to figure out a formula. I proceed in this way: Transcribe

(8) Only adults can vote.

This means that anyone who is not an adult can't vote, or equivalently (using the law of contraposition), anyone who can vote is an adult. So either of the following equivalent sentences provides a correct transcription:

(8a) (Vx)(¬Ax ∨ ¬Vx)
(8b) (Vx)(Vx ⊃ Ax)

This works in general. (In the following I used boldface capital P and Q to stand for arbitrary predicates.) Transcribe

(9) Only Ps are Qs.

either as

(9a) (Vx)(¬Px ⊃ ¬Qx)

(9b) (Vx)(Qx ⊃ Px)

Thus 'Only cats are furry' becomes (Vx)(Fx ⊃ Cx).

'Nothing' and 'not everything' often confuse me also. We must carefully distinguish

(10) Nothing is furry: (Vx)¬Fx, or ¬(∃x)Fx

and

(11) Not everything is furry: ¬(Vx)Fx, or (∃x)¬Fx

(The alternative transcriptions given in (10) and (11) are logically equivalent, by the rules ¬(Vx) and ¬(∃x) for logical equivalence introduced in section 3–4.) 'Not everything' can be transcribed literally as 'not all x . . .'. 'Nothing' means something different and much stronger. 'Nothing' means 'everything is not . . .' Be careful not to confuse 'nothing' with 'not everything.' If the distinction is not yet clear, make up some more examples and carefully think them through.

'None' and 'none but' can also cause confusion:

(12) None but adults can vote: (Vx)(¬Ax ⊃ ¬Vx)
(13) None love Adam: (Vx)¬Lxe

'None but' simply transcribes as 'only.' When 'none' without the 'but' fits in grammatically in English you will usually be able to treat it as you do 'nothing'. 'Nothing' and 'none' differ in that we tend to use 'none' when there has been a stated or implied restriction of domain: "How many cats does Adam love? He loves none." In this context a really faithful transcription of the sentence 'Adam loves none.' would be (Vx)(¬Lax), or, rewriting the subscript, (Vx)(Cx ⊃ ¬Lax).

Perhaps the most important negative quantifier expression in English is 'no', as in

(14) No cats are furry.

To say that no cats are furry is to say that absolutely all cats are not furry, so that we transpose (18) as

(15) (Vx)(¬Fx), that is, (Vx)(Cx ⊃ ¬Fx)

In general, transcribe

(16) No Ps are Qx.
as

\[ (\forall x)_p \neg Q, \text{ that is, } (\forall x)(P \lor \neg Q) \]

**EXERCISES**

4-4. Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Transcription Guide**

- a: Adam
- e: Eve
- Ax: x is an animal
- Bx: x is blond
- Cx: x is a cat
- Dx: x is a dog
- Fx: x is furry
- Px: x is a person
- Qx: x purrs
- Lxy: x loves y
- Sxy: x is a son of y

**Exercises**

Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Exercises**

4-4. Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Exercises**

4-4. Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Exercises**

4-4. Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Exercises**

4-4. Transcribe the following English sentences into the language of predicate logic. Use subscripts if you find them helpful in figuring out your answers, but no subscripts should appear in your final answers.

**Exercises**

4-5. Give alternative transcriptions which show the ways in which the following sentences are ambiguous. In this problem you do not have to eliminate subscripts. (It is sometimes easier to study the ambiguity if we write these sentences in the compact subscript notation.)

a) Everyone loves someone.

b) Someone loves everyone.

c) Something is a cat if and only if Adam loves it.

d) All cats are not furry.

e) Not anyone loves Adam.

4-6. In this section I discussed ambiguities connected with words such as 'a', 'someone', and 'anyone.' In fact, English has a great many sorts of ambiguity arising from the ways in which words are connected with each other. For example, 'I won't stay at home to please you.' can mean that if I stay at home, I won't do it in order to please you. But it can also mean that I will go out because going out will please you. 'Eve asked Adam to stay in the house.' can mean that Eve asked Adam to remain in a certain location, and that location is the house. It can also mean that Eve asked Adam to remain in some unspecified location, and that she made her request in the house.

For the following English sentences, provide alternative transcripts showing how the sentences are ambiguous. Use the transcription guides given for each sentence.

a) Flying planes can be dangerous. (Px: x is a plane. Fx: x is flying. Dx: x can be dangerous. Ax: x is an act of flying a plane.)

b) All wild animal keepers are blond. (Kxy: x keeps y. Wx: x is wild. Ax: x is an animal. Bx: x is blond.)

c) Adam only relaxes on Sundays. (a: Adam. Rxy: x relaxes on day y. Lxy: x relaxes ("is lazy") all day long on day y. Sx: x is Sunday.)

d) Eve dressed and walked all the dogs. (e: Eve. Cxy: x dressed y. Dx: x is a dog. Wxy: x walked y.)

Linguists use the expression *Structural Ambiguity* for the kind of ambiguity in these examples. This is because the ambiguities have to do with alternative ways in which the grammatical structure of the
sentences can be correctly analyzed. Structural ambiguity contrasts with \textit{Lexical Ambiguity}, which has to do with the ambiguity in the meaning of isolated words. Thus the most obvious ambiguity of 'I took my brother's picture yesterday.' turns on the ambiguity of the meaning of 'took' (stole vs. produced a picture). The ambiguity involved with quantifier words such as 'a', 'someone', and 'anyone' is actually structural ambiguity, not lexical ambiguity. We can see a hint of this in the fact that '(3x)(Hx \supset J)' is logically equivalent to '(\forall x)(Hx \supset J)' and the fact that '(\forall x)(Hx \supset J)' is logically equivalent to '(\exists x)(Hx \supset J)', as you will prove later on in the course.

4-3. TRANSCRIPTION STRATEGIES

I'm going to turn now from particularly hard cases to general strategy. If you are transcribing anything but the shortest of sentences, don't try to do it all at once. Transcribe parts into logic, writing down things which are part logic and part English. Bit by bit, transcribe the parts still in English into logic until all of the English is gone.

Let's do an example. Suppose we want to transcribe

(18) Any boy who loves Eve is not a furry cat.

(18) says of any boy who loves Eve that he is not a furry cat; that is, it says of all things, x, of which a first thing is true (that x is a boy who loves Eve) that a second thing is true (x is not a furry cat). So the sentence has the form \((\forall x)(P x \supset Q)\):

(18a) \((\forall x)(x \text{ is a boy who loves Eve} \supset x \text{ is not a furry cat})\)

Now all you have to do is to fashion transcriptions of 'x is a boy who loves Eve' and of 'x is not a furry cat' and plug them into (18a):

(18b) x is a boy who loves Eve: Bx & Lxe

(18c) x is not a furry cat: \((-Fx \& Cx)\)

(Something which is not a furry cat is not both furry and a cat. Such a thing could be furry, or a cat, but not both.) Now we plug (18b) and (18c) into (18a), getting our final answer:

(18d) \((\forall x)((Bx \& Lxe) \supset \neg(\exists x \& Cx))\]

Here is another way you could go about the same problem. Think of the open sentence 'Bx & Lxe' as indicating a complex one place predicate. The open sentence 'Bx & Lxe' presents something which might be true of an object or person such as Adam. For example, if the complex predicate is true of Adam, we would express that fact by writing in 'a' for 'x' in 'Bx & Lxe', giving 'Ba & Lae'. Now, thinking of 'Bx & Lxe' as a predicate, we can use the method of quantifier subscripts which we discussed in section 4-1. (18) is somewhat like a sentence which asserts that everything is not a furry cat. But (18) asserts this, not about absolutely everything, but just about all those things which have the complex property Bx & Lxe. So we can write (18) as a universally quantified sentence with the universal quantifier restricted by the predicate 'Bx & Lxe':

(18e) \((\forall x)((Bx \& Lxe) \supset \neg(Fx \& Cx))\)

Now you simply use the rule for rewriting subscripts on universal quantifiers, giving (18d).

In yet a third way of working on (18), you could first use the method of subscripting quantifiers before transcribing the complex predicates into logic. Following this route you would first write.

(18f) \((\forall x)x \text{ is a boy who loves Eve} \supset (x \text{ is not a furry cat})\)

Now transcribe the English complex predicates as in (18b) and (18c), plug the results into (18f), giving (18e). Then you rewrite the subscript, giving (18d) as before. You have many alternative ways of proceeding.

Generally, it is very useful to think of complex descriptions as complex predicates. In particular, this enables us to use two place predicates to construct one place predicates. We really took advantage of this technique in the last example. 'Lxy' is a two place predicate. By substituting a name for 'y', we form a one place predicate, for example, 'Lxe'. 'Lxe' is a one place predicate which is true of anything which loves Eve.

Here is another useful way of constructing one place predicates from two place predicates. Suppose we need the one place predicate 'is married', but our transcription guide only gives us the two place predicate 'Mxy', meaning that x is married to y. To see how to proceed, consider what it means to say that Adam, for example, is married. This is to say that there is someone to whom Adam is married. So we can say Adam is married with the sentence '(3y)May'. We could proceed in the same way to say that Eve, or anyone else, is married. In short, the open sentence '(3y)Mxy' expresses the predicate 'x is married'.

Here's another strategy point: When 'who' or 'which' comes after a predicate they generally transcribe as 'and'. As you saw in (18), the complex predicate 'x is a boy who loves Eve' becomes 'Bx & Lxe'. The complex predicate 'x is a dog which is not furry but has a tail' becomes 'Dx & \((-Fx \& (3y)Tyx))'.

When 'who' or 'which' comes after a quantifier word, they indicate a subscript on the quantifier: 'Anything which is not furry but has a tail' should be rendered as \((\forall x)(\neg Fx \& (3y)Tyx)\). When the quantifier word itself
calls for a subscript, as does ‘someone’, you need to combine both these ideas for treating ‘who’: ‘Someone who loves Eve’ is the subscripted quantifier ‘(∃x)Px & Lxe’.

Let’s apply these ideas in another example. Before reading on, see if you can use only ‘Cx’ for ‘x is a cat’, ‘Lxy’ for ‘x loves y’, and ‘Oxy’ for ‘x owns y’ and transcribe

\begin{align*}
(19) \text{ Some cat owner loves everyone who loves themselves.}
\end{align*}

Let’s see how you did. (19) says that there is something, taken from among the cat owners, and that thing loves everyone who loves themselves. Using a subscript and the predicates ‘x is a cat owner’ and ‘x loves everyone who loves themselves’, (19) becomes

\begin{align*}
(19a) \ (\exists x) (x \text{ is a cat owner}) & (x \text{ loves everyone who loves themselves})
\end{align*}

Now we have to fashion transcriptions for the two complex English predicates used in (19a). Someone (or something) is a cat owner just in case there is a cat which they own:

\begin{align*}
(19b) \text{ x is a cat owner: } (\exists y)(Cy \& Oxy)
\end{align*}

To say that x loves everyone who loves themselves is to say that x loves, not absolutely everyone, but everyone taken from among those that are, first of all people, and second, things which love themselves. So we want to say that x loves all y, where y is restricted to be a person, Py, and restricted to be a self-lover, Lyy:

\begin{align*}
(19c) \text{ x loves everyone who loves themselves: } (\forall y)(Py \& Ly)(Lxy)
\end{align*}

Putting the results of (19b) and (19c) into (19a), we get

\begin{align*}
(19d) \ (\exists x)(\exists y)(Cy \& Oxy) & (\forall y)(Py \& Ly)(Lxy)
\end{align*}

Discharging first the subscript of ‘(\exists x)’ with an ‘&’ and then the subscript of ‘(\forall y)’ with a ‘\lor’, we get

\begin{align*}
(19e) \ (\exists x)(\exists y)(Cy \& Oxy) & (\forall y)(Py \lor Ly)(Lxy)
(19f) \ (\exists x)(\exists y)(Cy \& Oxy) & (\forall y)(Py \lor Ly \lor Lxy)
\end{align*}

This looks like a lot of work, but as you practice, you will find that you can do more and more of this in your head and it will start to go quite quickly.

I’m going to give you one more piece of advice on transcribing. Suppose you start with an English sentence and you have tried to transcribe it into logic. In many cases you can catch mistakes by transcribing your logic sentence back into English and comparing your retranscription with the original sentence. This check works best if you are fairly literal minded in retranscribing. Often the original and the retranscribed English sentences will be worded differently. But look to see if they still seem to say the same thing. If not, you have almost certainly made a mistake in transcribing from English into logic.

Here is an illustration. Suppose I have transcribed

\begin{align*}
(20) \text{ If something is a cat, it is not a dog.}
\end{align*}

as

\begin{align*}
(20a) \ (\exists x)(Cx \lor \sim Dx)
\end{align*}

To check, I transcribe back into English, getting

\begin{align*}
(20b) \text{ There is something such that if it is a cat, then it is not a dog.}
\end{align*}

Now compare (20b) with (20). To make (20b) true it is enough for there to be one thing which, if a cat, is not a dog. The truth of (20b) is consistent with there being a million cat-dogs. But (20) is not consistent with there being any cat-dogs. I conclude that (20a) is a wrong transcription. Having seen that (20) is stronger than (20a), I try

\begin{align*}
(20c) \ (\forall x)(Cx \lor \sim Dx)
\end{align*}

Transcribing back into English this time gives me

\begin{align*}
(20d) \text{ Everything which is a cat is not a dog.}
\end{align*}

which does indeed seem to say what (20) says. This time I am confident that I have transcribed correctly.

(20) is (20) ambiguous in the same way that (5) was? I don’t think so!

Here is another example. Suppose after some work I transcribe

\begin{align*}
(21) \text{ Cats and dogs have tails.}
\end{align*}

as

\begin{align*}
(21a) \ (\forall x)((Cx \& Dx) \lor (\exists y)Txy)
\end{align*}

To check, I transcribe back into English:

\begin{align*}
(21b) \text{ Everything is such that if it is both a cat and a dog, then it has a tail.}
\end{align*}

Obviously, something has gone wrong, for nothing is both a cat and a dog. Clearly, (21) is not supposed to be a generalization about such imag-
inary cat-dogs. Having noticed this, I see that (21) is saying one thing about cats and then the same thing about dogs. Thus, without further work, I try the transcription

\[(21c) \quad (\forall x)(Cx \supset (\exists y)Txy) \land (\forall x)(Dx \supset (\exists y)Txy)\]

To check (21c), I again transcribe back into English, getting

\[(21d) \quad \text{If something is a cat, then it has a tail, and if something is a dog, then it has a tail.}\]

which is just a long-winded way of saying that all cats and dogs have tails—in other words, (21). With this check, I can be very confident that (21c) is a correct transcription.

**EXERCISES**

Use this transcription guide for exercises 4-7 and 4-8:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
<th>k</th>
<th>l</th>
</tr>
</thead>
<tbody>
<tr>
<td>a: Adam</td>
<td>Fx: x is furry</td>
<td>e: Eve</td>
<td>Px: x is a person</td>
<td>Ax: x is an animal</td>
<td>Qx: x purrs</td>
<td>Bx: x is blond</td>
<td>Lxy: x loves y</td>
<td>Cx: x is a cat</td>
<td>Sxy: x is a son of y</td>
<td>Dx: x is a dog</td>
<td>Txy: x is a tail of y</td>
</tr>
</tbody>
</table>

4-7. Transcribe the following sentences into English:

a) \((\exists x)(\exists y)(Pxy \land Py \land Sxy)\)
b) \((\forall x)(Pxy \land Ax)\)
c) \((\forall x)[Qxy \supset (Fx \land Cx)]\)
d) \((\exists x)[Qxy \land (Fx \land Cx)]\)
e) \((\forall x)[Pxy \land (Lxa \land Lxe)]\)
f) \((\forall x)[Pxy \land (Lxa \land Lxe)]\)
g) \((\forall x)(Vy)([Dxy \land Cy] \supset Lxy)\)
h) \((\forall x)(Vy)([Dxy \land (Cy \supset Lxy)]\)
i) \((\forall x)[Pxy \land (\exists z)(Py \land Szy \land Lxz)]\)
j) \((\exists x)[Pxy \land (\exists z)(Pzy \land Szy \land Lxz)]\)
k) \((\forall x)[[Bx \land (\exists y)(Fxy \land Txy)] \supset (\exists z)(Czy \land Txz)]\)
l) \((\forall x)(\exists y)(Sxy \supset (\exists z)(Czy \land Lxz) = (\exists z)(Dzy \land Lxz))\)

4-8. Transcribe the following sentences into predicate logic. I have included some easy problems as a review of previous sections along with some real brain twisters. I have marked the sentences which seem to me clearly ambiguous, and you should give different transcriptions for these showing the different ways of understanding the English. Do you think any of the sentences I haven't marked are also ambiguous? You should have fun arguing out your intuitions about ambiguous cases with your classmates and instructor.

a) All furry cats purr.
b) Any furry cat purrs.
c) No furry cats purr.
d) None of the furry cats purr. (Ambiguous?)
e) None but the furry cats purr.
f) Some furry cats purr.
g) Some furry cats do not purr.
h) Some cats and dogs love Adam.
i) Except for the furry ones, all cats purr.
j) Not all furry cats purr.
k) If a cat is furry, it purrs.
l) A furry cat purrs. (Ambiguous)
m) Only furry cats purr.
n) Adam is not a dog or a cat.
o) Someone is a son.
p) Some sons are blond.
q) Adam loves a blond cat, and Eve loves one too.
r) Adam loves a blond cat and so does Eve. (Ambiguous)
s) Eve does not love everyone.
t) Some but not all cats are furry.
u) Cats love neither Adam nor Eve.
v) Something furry loves Eve.
w) Only people love someone.
x) Some people have sons.
y) Any son of Adam is a son of Eve.
z) Adam is a son and everybody loves him.
aa) No animal is furry, but some have tails.
b) Any furry animal has a tail.
cc) No one has a son.
dd) Not everyone has a son.
e) Some blonds love Eve, some do not.
ff) Adam loves any furry cat.
gg) All blonds who love themselves love Eve.
hh) Eve loves someone who loves himself.
ii) Anyone who loves no cats loves no dogs.
jj) Cats love Eve if they love anyone. (Ambiguous)
k) If anyone has a son, Eve loves Adam. (Ambiguous)
l) If anyone has a son, that person loves Adam.
Anyone who has a son loves Eve.
If someone has a son, Adam loves Eve.
If someone has a son, that person loves Adam.
Someone who has a son loves Adam. (Ambiguous)
All the cats with sons, except the furry ones, love Eve.
Anyone who loves a cat loves an animal.
Anyone who loves a person loves no animal.
Adam has a son who is not furry.
If Adam's son has a furry son, so does Adam.
A son of Adam is a son of Eve. (Ambiguous)
If the only people who love Eve are blond, then nobody loves Eve.
No one loves anyone. (Ambiguous)
Everyone loves no one.
Everyone doesn't love everyone. (Ambiguous!)
Nobody loves nobody. (Ambiguous?)
Except for the furry ones, every animal loves Adam.
Everyone loves a lover. (Ambiguous)
None but those blonds who love Adam own cats and dogs.
No one who loves no son of Adam loves no son of Eve.
Only owners of dogs with tails own cats which do not love Adam.
None of Adam's sons are owners of furry animals with no tails.
Anyone who loves nothing without a tail owns nothing which is loved by an animal.
Only those who love neither Adam nor Eve are sons of those who own none of the animals without tails.
Anyone who loves all who Eve loves loves someone who is loved by all who love Eve.

No one likes Professor Snarf.
Any dog can hear better than any person.
Neither all Republicans nor all Democrats are honest.
Some movie stars are better looking than others.
None of the students who read A Modern Formal Logic Primer failed the logic course.
Only people who eat carrots can see well in the dark.
Not only people who eat carrots can see as well as people who eat strawberries.
Peter likes all movies except for scary ones.

Some large members of the cat family can run faster than any horse.
Not all people with red hair are more temperamental than those with blond hair.
Some penny on the table was minted before any dime on the table.
No pickle tastes better than any strawberry.
John is not as tall as anyone on the basketball team.
None of the pumpkins at Smith's fruit stand are as large as any of those on MacGreggor's farm.
Professors who don't prepare their lectures confuse their students.
Professor Snarf either teaches Larry or teaches someone who teaches Larry.
Not only logic teachers teach students at Up State U.
Anyone who lives in Boston likes clams more than anyone who lives in Denver. (Ambiguous)
Except for garage mechanics who fix cars, no one has greasy pants.
Only movies shown on channel 32 are older than movies shown on channel 42.
No logic text explains logic as well as some professors do.
The people who eat, drink, and are merry are more fun than those who neither smile nor laugh.

CHAPTER SUMMARY EXERCISES

In reviewing this chapter make a short summary of the following to ensure your grasp of these ideas:
a) Restricted Quantifiers
b) Rule ~\(V\)
c) Rule ~\(E\)
d) Transcription Guide
e) Words that generally transcribe with a universal quantifier
f) Word that generally transcribe with an existential quantifier
g) Negative Quantifier Words
h) Ambiguity
i) Give a summary of important transcription strategies
Let's get back to the problem of demonstrating argument validity. You know how to construct derivations which demonstrate the validity of valid sentence logic arguments. Now that you have a basic understanding of quantified sentences and what they mean, you are ready to extend the system of sentence logic derivations to deal with quantified sentences.

Let's start with a short review of the fundamental concepts of natural deduction: To say that an argument is valid is to say that in every possible case in which the premises are true, the conclusion is true also. The natural deduction technique works by applying truth preserving rules. That is, we use rules which, when applied to one or two sentences, license us to draw certain conclusions. The rules are constructed so that in any case in which the first sentence or sentences are true, the conclusion drawn is guaranteed to be true also. Certain rules apply, not to sentences, but to subderivations. In the case of these rules, a conclusion which they license is guaranteed to be true if all the sentences reiterated into the subderivation are true.

A derivation begins with no premises or one or more premises. It may include subderivations, and any subderivation may itself include a subderivation. A new sentence, or conclusion, may be added to a derivation if one of the rules of inference licenses us to draw the conclusion from previous premises, assumptions, conclusions, or subderivations. Because these rules are truth preserving, if the original premises are true in a case, the first conclusion drawn will be true in that case also. And if this first conclusion is true, then so will the next. And so on. Thus, altogether, in any case in which the premises are all true, the final conclusion will be true.

The only further thing you need to remember to be able to write sentence logic derivations are the rules themselves. If you are feeling rusty, please refresh your memory by glancing at the inside front cover, and review chapters 5 and 7 of Volume I, if you need to.

Now we are ready to extend our system of natural deduction for sentence logic to the quantified sentences of predicate logic. Everything you have already learned will still apply without change. Indeed, the only fundamental conceptual change is that we now must think in terms of an expanded idea of what constitutes a case. For sentence logic derivations, truth preserving rules guarantee that if the premises are true for an assignment of truth values to sentence letters, then conclusions drawn will be true for the same assignment. In predicate logic we use the same overall idea, except that for a "case" we use the more general idea of an interpretation instead of an assignment of truth values to sentence letters. Now we must say that if the premises are true in an interpretation, the conclusions drawn will be true in the same interpretation.

Since interpretations include assignment of truth values to any sentence letters that might occur in a sentence, everything from sentence logic applies as before. But our thinking for quantified sentences now has to extend to include the idea of interpretations as representations of the case in which quantified sentences have a truth value.

You will remember each of our new rules more easily if you understand why they work. You should understand why they are truth preserving by thinking in terms of interpretations. That is, you should try to understand why, if the premises are true in a given interpretation, the conclusion licensed by the rule will inevitably also be true in that interpretation.

Predicate logic adds two new connectives to sentence logic: the universal and existential quantifiers. So we will have four new rules, an introduction and elimination rule for each quantifier. Two of these rules are easy and two are hard. Yes, you guessed it! I'm going to introduce the easy rules first.

### 5-2. The Universal Elimination Rule

Consider the argument

$$
\frac{\text{Everyone is blond.} \quad (\forall x)Bx}{\text{Adam is blond.} \quad Ba}
$$

5-2. The Universal Elimination Rule: Consider the argument
Intuitively, if everyone is blond, this must include Adam. So if the premise is true, the conclusion is going to have to be true also. In terms of interpretations, let’s consider any interpretation you like which is an interpretation of the argument’s sentences and in which the premise, ‘(Vx)Bx’, is true. The definition of truth of a universally quantified sentence tells us that ‘(Vx)Bx’ is true in an interpretation just in case all of its substitution instances are true in the interpretation. Observe that ‘Ba’ is a substitution instance of ‘(Vx)Bx’. So in our arbitrarily chosen interpretation in which ‘(Vx)Bx’ is true, ‘Ba’ will be true also. Since ‘Ba’ is true in any interpretation in which ‘(Vx)Bx’ is true, the argument is valid.

(In this and succeeding chapters I am going to pass over the distinction between someone and something, as this complication is irrelevant to the material we now need to learn. I could give examples of things instead of people, but that makes learning very dull.)

The reasoning works perfectly generally:

**Universal Elimination Rule:** If X is a universally quantified sentence, then you are licensed to conclude any of its substitution instances below it. Expressed with a diagram, for any name, a, and any variable, u,

\[
\begin{align*}
(Vu)(\ldots u \ldots) \\
(\ldots s \ldots) & \quad \text{VE}
\end{align*}
\]

Remember what the box and the circle mean: If on a derivation you encounter something with the form of what you find in the box, the rule licenses you to conclude something of the form of what you find in the circle.

Here is another example:

Everyone loves Eve. \((Vx)Lxe\) 1 \((Vx)Lxe\) 1, VE

Adam loves Eve. \(La\) 2 \(La\) 1, VE

In forming the substitution instance of a universally quantified sentence, you must be careful always to put the same name everywhere for the substituted variable. Substituting ‘a’ for ‘x’ in ‘(Vx)Lxx’, we get ‘Laa’, not ‘Lxa’. Also, be sure that you substitute your name only for the occurrences of the variable which are free after deleting the initial quantifier. Using the name ‘a’ again, the substitution instance of ‘(Vx)(Bx ⊃ (Vx)Lxe)’ is ‘Ba ⊃ (Vx)Lxe’. The occurrence of ‘x’ in ‘Lxe’ is bound by the second ‘(Vx)’, and so is still bound after we drop the first ‘(Vx)’. If you don’t understand this example, you need to review bound and free variables, and substitution instances, discussed in chapter 3.

When you feel confident that you understand the last example, look at one more:

\[
\begin{align*}
(Vx)(Gx ⊃ Kx) \\
Gf & \quad P \\
Kf & \quad P \\
& \quad 1, \text{VE} \\
Kf & \quad 2, 3, \supset \text{E}
\end{align*}
\]

**EXERCISES**

5–1. Provide derivations which demonstrate the validity of these arguments. Remember to work from the conclusion backward, seeing what you will need to get your final conclusions, as well as from the premises forward. In problem (d) be sure you recognize that the premise is a universal quantification of a conditional, while the conclusion is the very different conditional with a universally quantified antecedent.

\[
\begin{align*}
a) & \quad (Vx)(Px & Dx) \\
\quad & \quad Pk \\
\quad & \quad Pd & Dk \\
\quad & \quad (Vx)Dx \\
\quad & \quad Ka \\
\quad & \quad (Vx)(Fx v Hx) \\
\quad & \quad (Vx)(Fx & Dx) \\
\quad & \quad (Vx)(Fx v Hx) & (Vx)(Fx & Dx) \\
\quad & \quad Db & Db \\
\quad & \quad (Vx)(Gx ⊃ Lx) \\
\quad & \quad (Vx)(R xx v R xx) \\
\quad & \quad \sim Lmh \\
\quad & \quad (Vx)∼ Ryk \\
\quad & \quad \sim (Vx)Lxx \\
\quad & \quad Rcc & Rff
\end{align*}
\]

5–3. THE EXISTENTIAL INTRODUCTION RULE

Consider the argument

\[
\begin{align*}
\text{Adam is blond.} & \quad \text{Ba} \\
\text{Someone is blond.} & \quad (3x)Bx
\end{align*}
\]
Intuitively, this argument is valid. If Adam is blond, there is no help for it: Someone is blond. Thinking in terms of interpretations, we see that this argument is valid according to our new way of making the idea of validity precise. Remember how we defined the truth of an existentially quantified sentence in an interpretation: '(∃x)Bx' is true in an interpretation if and only if at least one of its substitution instances is true in the interpretation. But 'Ba' is a substitution instance of '(∃x)Bx'. So, in any interpretation in which 'Ba' is true, '(∃x)Bx' is true also, which is just what we mean by saying that the argument 'Ba. Therefore '(∃x)Bx.' is valid.

You can probably see the form of reasoning which is at play here: From a sentence with a name we can infer what we will call an Existential Generalization of that sentence. '(∃x)Bx' is an existential generalization of 'Ba'. We do have to be a little careful in making this notion precise because we can get tripped up again by problems with free and bound variables. What would you say is a correct existential generalization of '(∀x)Lax'? In English: If Adam loves everyone, then we know that someone loves everyone. But we have to use two different variables to transcribe 'Someone loves everyone': '(∃y)(∀x)Lyx'. If I start with '(∀x)Lax', and replace the 'a' with 'x', my new occurrence of 'x' is bound by that universal quantifier. I will have failed to generalize existentially on 'a'.

Here is another example for you to try: Existentially generalize

(i) Ba ⊃ (∀x)Lax

If I drop the 'a' at 2 and 4, write in 'x', and preface the whole with '(∃x)', I get

(ii) (∃x)(Bx ⊃ (∀x)Lxx) Wrong

The 'x' at 4, which replaced one of the 'a's, is bound by the universally quantified 'x' at 3, not by the existentially quantified 'x' at 1, as we intend in forming an existential generalization. We have to use a new variable. A correct existential generalization of 'Ba ⊃ (∀x)Lax' is

(iii) (∃y)(By ⊃ (∀x)Lyx)

as are

(iv) (∃y)(By ⊃ (∀x)Lax)

and

(v) (∃y)(Ba ⊃ (∀x)Lyx)

1 2 3 45

Here is how you should think about this problem: Starting with a closed sentence, (. . . s . . .), which uses a name, s, take out one or more of the occurrences of the name s. For example, take out the 'a' at 4 in (i). Then look to see if the vacated spot is already in the scope of one (or more) quantifiers. In (i) to (v), the place marked by 4 is in the scope of the '(∀x)' at 3. So you can't use 'x'. You must perform your existential generalization with some variable which is not already bound at the places at which you replace the name. After taking out one or more occurrences of the name, s, in (. . . s . . .), replace the vacated spots with a variable (the same variable at each spot) which is not bound by some quantifier already in the sentence.

Continuing our example, at this point you will have turned (i) into

(vi) Ba ⊃ (∀x)Lya

You will have something of the form (. . . u . . .) in which u is free: 'y' is free in (vi). At this point you must have an open sentence. Now, at last, you can apply your existential quantifier to the resulting open sentence to get the closed sentence (∃u)(. . . u . . .).

To summarize more compactly:

(∃u)(. . . u . . .) is an Existential Generalization of (. . . s . . .) with respect to the name s if and only if (∃u)(. . . u . . .) results from (. . . s . . .) by

a) Deleting any number of occurrences of s in (. . . s . . .),

b) Replacing these occurrences with a variable, u, which is free at these occurrences, and

c) Applying (∃u) to the result.

(In practice you should read (a) in this definition as “Deleting one or more occurrences of s in (. . . s . . .).” I have expressed (a) with "any number of" so that it will correctly treat the odd case of vacuous quantifiers, which in practice you will not need to worry about. But if you are interested, you can figure out what is going on by studying exercise 3–3.)

It has taken quite a few words to set this matter straight, but once you see the point you will no longer need the words.

With the idea of an existential generalization, we can accurately state the rule for existential introduction:

**Existential Introduction Rule:** From any sentence, X, you are licensed to conclude any existential generalization of X anywhere below. Expressed with a diagram,
Let's look at a new example, complicated only by the feature that it involves a second name which occurs in both the premise and the conclusion:

Adam loves Eve.  Lae
Adam loves someone.  (Ex)Lax

'(Ex)Lax' is an existential generalization of 'Lae'. So Ex applies to make the following a correct derivation:

1         2          3      4
Lae       (Ex)Lax    1, Ex
P          P          P          P

To make sure you have the hang of rule Ex, we'll do one more example. Notice that in this example, the second premise has an atomic sentence letter as its consequent. Remember that predicate logic is perfectly free to use atomic sentence letters as components in building up sentences.

Kx         1         2
(Ex)Kx ⊃ P  (Ex)Kx ⊃ P
P

In line 4 I applied De to lines 2 and 3. De applies here in exactly the same way as it did in sentence logic. In particular De and the other sentence logic rules apply to sentences the components of which may be quantified sentences as well as sentence logic sentences.

Now let's try an example which applies both our new rules:

(Ex)Ax       1
(Ex)Tx     2

In addition to illustrating both new rules working together, this example illustrates something else we have not yet seen. In past examples, when I applied Ve I instantiated a universally quantified sentence with a name which already occurred somewhere in the argument. In this case no name occurs in the argument. But if a universally quantified sentence is true in an interpretation, all of its substitution instances must be true in the interpretation. And every interpretation must have at least one object in it. So a universally quantified sentence must always have at least one substitution instance true in an interpretation. Since a universally quantified sentence always has at least one substitution instance, I can introduce a name into the situation with which to write that substitution instance, if no name already occurs.

To put the point another way, because every interpretation always has at least one object in it, I can always introduce a name to refer to some object in an interpretation and then use this name to form my substitution instance of the universally quantified sentence.

Good. Let's try yet another example:

Notice that although the rules permit me to apply Ex to line 2, doing so would not have gotten me anywhere. To see how I came up with this derivation, look at the final conclusion. You know that it is an existentially quantified sentence, and you know that Ex permits you to derive such a sentence from an instance, such as 'Md'. So you must ask yourself: Can I derive such an instance from the premises? Yes, because the first premise says about everything that if it is C, then it is M. And the second premise says that d, in particular, is C. So applying Ve to 1 you can get 3, which, together with 2, gives 4 by De.

**EXERCISES**

5-2. Provide derivations which demonstrate the validity of the following arguments:

a) Na   b) (Ex)(Kx & Px)   c) (Ex)(Hx ⊃ ¬Dx)
(Ex)(Nx v Gx)  (Ex)(Kx & (Ex)Fx)  (Ex)(Nx v (Ex)x)
(Ex)(Ax v (Ex)Tx)  (Ex)(Fx v (Ex)x)  (Ex)(Sx v (Ex)x)
5-4. The Existential Elimination and Universal Introduction Rules: Background in Informal Argument

Now let's go to work on the two harder rules. To understand these rules, it is especially important to see how they are motivated. Let us begin by looking at some examples of informal deductive arguments which present the kind of reasoning which our new rules will make exact. Let's start with this argument:

Everyone likes either rock music or country/western.
Someone does not like rock.
Someone likes country/western.

Perhaps this example is not quite as trivial as our previous examples. How can we see that the conclusion follows from the premises? We commonly argue in the following way. We are given the premise that someone does not like rock. To facilitate our argument, let us suppose that this someone (or one of them if there are more than one) is called Doe. (Since I don't know this person's name, I'm using 'Doe' as the police do when they book a man with an unknown name as 'John Doe.') Now, since according to the first premise, everyone likes either rock or country/western, this must be true, in particular, of Doe. That is, either Doe likes rock, or he or she likes country/western. But we had already agreed that Doe does not like rock. So Doe must like country/western. Finally, since Doe likes country/western, we see that someone likes country/western. But that was just the conclusion we were trying to derive.

What you need to focus on in this example is how I used the name 'Doe'. The second premise gives me the assumption that someone does not like rock. So that I can talk about this someone, I give him or her a name: 'Doe'. I don't know anything more that applies to just this person,

but I do have a fact, the first premise, which applies to everyone. So I can use this fact in arguing about Doe, even though I really don't know who Doe is. I use this general fact to conclude that Doe, whoever he or she might be, does like country/western. Finally, before I am done, I acknowledge that I really don't know who Doe is, in essence by saying: Whoever this person Doe might be, I know that he or she likes country/western. That is, what I really can conclude is that there is someone who likes country/western.

Now let's compare this argument with another:

(1) Everyone either likes rock or country/western.
(2) Anyone who likes country/western likes soft music.
(3) Anyone who doesn't like rock likes soft music.

This time I have deliberately chosen an example which might not be completely obvious so that you can see the pattern of reasoning doing its work.

The two premises say something about absolutely everyone. But it's hard to argue about 'everyone'. So let us think of an arbitrary example of a person, named 'Arb', to whom these premises will then apply. My strategy is to carry the argument forward in application to this arbitrarily chosen individual. I have made up the name 'Arb' to emphasize the fact that I have chosen this person (and likewise the name) perfectly arbitrarily. We could just as well have chosen any person named by any name.

To begin the argument, the first premise tells us that

(4) Either Arb likes rock, or Arb likes country/western.

The second premise tells us that

(5) If Arb does like country/western, then Arb likes soft music.

Now, let us make a further assumption about Arb:

(6) (Further Assumption): Arb doesn't like rock.

From (6) and (4), it follows that

(7) Arb likes country/western.

And from (7) and (5), it follows that

(8) Arb likes soft music.

Altogether we see that Arb's liking soft music, (8), follows from the further assumption, (6), with the help of the original premises (1) and (2) as
applied through this application to Arb, in (4) and (5). Consequently, from the original premises it follows that

(9) If Arb doesn’t like rock, then Arb likes soft music.

All this is old hat. Now comes the new step. The whole argument to this point has been conducted in terms of the person, Arb. But Arb could have been anyone, or equally, we could have conducted the argument with the name of anyone at all. So the argument is perfectly general. What (9) says about Arb will be true of anyone. That is, we can legitimately conclude that

(3) Anyone who doesn’t like rock likes soft music.

which is exactly the conclusion we were trying to reach.

We have now seen two arguments which use “stand-in” names, that is, names that are somehow doing the work of “someone” or of “anyone”. Insofar as both arguments use stand-in names, they seem to be similar. But they are importantly different, and understanding our new rules turns on understanding how the two arguments are different. In the second argument, Arb could be anyone—absolutely anyone at all. But in the first argument, Doe could not be anyone. Doe could only be the person, or one of the people, who does not like rock. ‘Doe’ is “partially arbitrary” because we are careful not to assume anything we don’t know about Doe. But we do know that Doe is a rock hater and so is not just anyone at all. Arb, however, could have been anyone.

We must be very careful not to conflate these two ways of using stand-in names in arguments. Watch what happens if you do conflate the ways:

Someone does not like rock. (Invalid)
Everyone does not like rock.

The argument is just silly. But confusing the two functions of stand-in names could seem to legitimate the argument, if one were to argue as follows: Someone does not like rock. Let’s call this person ‘Arb’. So Arb does not like rock. But Arb could be anyone, so everyone does not like rock. In such a simple case, no one is going to blunder in this way. But in more complicated arguments it can happen easily.

To avoid this kind of mistake, we must find some way to clearly mark the difference between the two kinds of argument. I have tried to bring out the distinction by using one kind of stand-in name, ‘Doe’, when we are talking about the existence of some particular person, and another kind of stand-in name, ‘Arb’, when we are talking about absolutely any arbitrary individual. This device works well in explaining that a stand-in name can function in two very different ways. Unfortunately, we cannot incorporate this device in natural deduction in a straightforward way simply by using two different kinds of names to do the two different jobs.

Let me try to explain the problem. (You don’t need to understand the problem in detail right now; detailed understanding will come later. All you need at this point is just a glimmer of what the problem is.) At the beginning of a derivation a name can be arbitrary. But then we might start a subderivation in which the name occurs, and although arbitrary from the point of view of the outer derivation, the name might not be arbitrary from the point of view of the subderivation. This can happen because in the original derivation nothing special, such as hating rock, is assumed about the individual. But inside the subderivation we might make such a further assumption about the individual. While the further assumption is in effect, the name is not arbitrary, although it can become arbitrary again when we discharge the further assumption of the subderivation. In fact, exactly these things happened in our last example. If, while the further assumption (6) was in effect, I had tried to generalize on statements about Arb, saying that what was true of Arb was true of anyone, I could have drawn all sorts of crazy conclusions. Look back at the example and see if you can figure out for yourself what some of these conclusions might be.

Natural deduction has the job of accurately representing valid reasoning which uses stand-in names, but in a way which won’t allow the sort of mistake or confusion I have been pointing out. Because the confusion can be subtle, the natural deduction rules are a little complicated. The better you understand what I have said in this section, the quicker you will grasp the natural deduction rules which set all this straight.

EXERCISES

5–3. For each of the two different uses of stand-in names discussed in this section, give a valid argument of your own, expressed in English, which illustrates the use.

5–5. THE UNIVERSAL INTRODUCTION RULE

Here is the intuitive idea for universal introduction, as I used this rule in the soft music example: If a name, as it occurs in a sentence, is completely arbitrary, you can Universally Generalize on the name. This means that you rewrite the sentence with a variable written in for all occurrences of the arbitrary name, and you put a universal quantifier, written with the same
variable, in front. To make this intuition exact, we have to say exactly when a name is arbitrary and what is involved in universal generalization. We must take special care because universal generalization differs importantly from existential generalization.

Let's tackle arbitrariness first. When does a name not occur arbitrarily? Certainly not if some assumption is made about (the object referred to by) the name. If some assumption is made using a name, then the name can't refer to absolutely anything. If a name occurs in a premise or assumption, the name can refer only to things which satisfy that premise or assumption. So a name does not occur arbitrarily when the name appears in a premise or an assumption, and it does not occur arbitrarily as long as such a premise or assumption is in effect.

The soft music example shows these facts at work. I'll use 'Rx' for 'x likes rock', 'Cx' for 'x likes countrywestern', and 'Sx' for 'x likes soft music.' Here are the formalized argument and derivation which I am going to use to explain these ideas:

\[(\forall x)(Rx \lor Cx)\]
\[(\forall x)(Cx \supset Sx)\]
\[(\forall x)(\neg Rx \supset Sx)\]

Read through this copy of the derivation and make sure you understand why the hat occurs where it does and why it does not occur where it doesn't. If you have a question, reread the previous paragraph, remembering that a hat on a name just means that the name occurs arbitrarily at that place.

I want to be sure that you do not misunderstand what the hat means. A name with a hat on it is not a new kind of name. A name is a name is a name, and two occurrences of the same name, one with and one without a hat, are two occurrences of the same name. A hat on a name is a kind of flag to remind us that at that point the name is occurring arbitrarily. Whether or not a name occurs arbitrarily is not really a fact just about the name. It is a fact about the relation of the name to the derivation in which it occurs. If, at an occurrence of a name, the name is governed by a premise or assumption which uses the same name, the name does not occur there arbitrarily. It is not arbitrary there because the thing it refers to has to satisfy the premise or assumption. Only if a name is not governed by any premise or assumption using the same name is the name arbitrary, in which case we mark it by dressing it with a hat.

Before continuing, let's summarize the discussion of arbitrary occurrence with an exact statement:

Suppose that a sentence, X, occurs in a derivation or subderivation. That occurrence of X is governed by a premise or assumption, Y, if and only if Y is a premise or assumption of X's derivation, or of any outer derivation of X's derivation (an outer derivation, or outer-outer derivation, and so on). In particular, a premise or assumption is always governed by itself.
A name occurs arbitrarily in a sentence of a derivation if that occurrence of the sentence is not governed by any premise or assumption in which the name occurs. To help us remember, we mark an arbitrary occurrence of a name by writing it with a hat.

The idea for the universal introduction rule was that we would universally generalize on a name that occurs arbitrarily. We have discussed arbitrary occurrence. Now on to universal generalization.

The idea of a universal generalization differs in one important respect from the idea of an existential generalization. To see the difference, you must be clear about what we want out of a generalization: We want a new quantified sentence which follows from a sentence with a name.

For the existential quantifier, $(\exists x) Lxx$, $(\exists x) Lax$, and $(\exists x) Lxa$ all follow from ‘La’. From the fact that Adam loves himself, it follows that Adam loves someone, someone loves Adam, and someone loves himself.

Now suppose that the name ‘a’ occurs arbitrarily in ‘La’. We know that “Adam” loves himself, where Adam now could be just anybody at all. What universal fact follows? Only that $(\forall x) Lax$, that everyone loves himself. It does not follow that $(\forall x) Lax$ or $(\forall x) Lxa$. That is, it does not follow that Adam loves everyone or everyone loves Adam. Even though ‘Adam’ occurs arbitrarily, $(\forall x) Lax$ and $(\forall x) Lxa$ make it sound as if someone (“Adam”) loves everyone and as if someone (“Adam”) is loved by everyone. These surely do not follow from ‘La’. But $\exists a$ would license us to infer these sentences, respectively, from $(\forall x) Lax$ and from $(\forall x) Lxa$.

Worse, $a$ is still arbitrary in $(\forall x) Lax$. So if we could infer $(\forall x) Lax$ from ‘La’, we could then argue that in $(\forall x) Lax$, ‘a’ could be anyone. We would then be able to infer $(\forall y) (\forall x) Lyx$, that everyone loves everyone! But from ‘La’ we should only be able to infer $(\forall x) Lxx$, that everyone loves himself, not $(\forall y) (\forall x) Lyx$, that everyone loves everyone.

We want to use the idea of existential and universal generalizations to express valid rules of inference. The last example shows that, to achieve this goal, we have to be a little careful with sentences in which the same name occurs more than once. If $s$ occurs more than once in $(\ldots s \ldots)$, we may form an existential generalization by generalizing on any number of the occurrences of $s$. But, to avoid the problem I have just described and to get a valid rule of inference, we must insist that a universal generalization of $(\ldots s \ldots)$, with respect to the name, $s$, must leave no instance of $s$ in $(\ldots s \ldots)$.

In other respects the idea of universal generalization works just like existential generalization. In particular, we must carefully avoid the trap of trying to replace a name by a variable already bound by a quantifier. This idea works exactly as before, so I will proceed immediately to an exact statement:

The sentence $(\forall u)(\ldots u \ldots)$ results by universally generalizing on the name $s$ in $(\ldots s \ldots)$ if and only if one obtains $(\forall u)(\ldots u \ldots)$ from $(\ldots s \ldots)$ by:

a) Deleting all occurrences of $s$ in $(\ldots s \ldots)$,
b) Replacing these occurrences with a variable, $u$, which is free at these occurrences, and
c) Applying $(\forall u)$ to the result.

$(\forall u)(\ldots u \ldots)$ is then said to be the Universal Generalization of $(\ldots s \ldots)$ with Respect to the Name $s$.

With these definitions, we are at last ready for an exact statement of the universal introduction rule:

**Universal Introduction Rule:** If a sentence, $X$, appears in a derivation, and if at the place where it appears a name, $i$, occurs arbitrarily in $X$, then you are licensed to conclude, anywhere below, the sentence which results by universally generalizing on the name $i$ in $X$. Expressed with a diagram:

$$[\ldots s \ldots]$$

$(\forall u_1 \ldots u_n)$

Where $s$ occurs arbitrarily in $(\ldots s \ldots)$ and $(\forall u_1 \ldots u_n)$ is the universal generalization of $(\ldots s \ldots)$ with respect to $s$.

Let’s look at two simple examples to illustrate what can go wrong if you do not follow the rule correctly. The first example is the one we used to illustrate the difference between existential and universal generalization:

Everyone loves themself.
Everyone loves Adam. (Invalid!)

To apply $\forall i$ to 1. ‘a’ is not arbitrary at 1.

The second example will make sure you understand the requirement that $\forall i$ applies only to an arbitrary occurrence of a name:

Adam is blond.
Everyone is blond. (Invalid!)
The problem here is that the premise assumes something special about the thing referred to by 'a', that it has the property referred to by 'B'. We can universally generalize on a name—that is, apply \( \forall I \) only when nothing special is assumed in this way, that is, when the name is arbitrary. You will see this even more clearly if you go back to our last formalization of the soft music example and see what sorts of crazy conclusions you could draw if you were to allow yourself to generalize on occurrences of names without hats.

Let's consolidate our understanding of \( \forall I \) by working through one more example. Before reading on, try your own hand at providing a derivation for

\[
(\forall x)(Lax \land Lxa) \\
(\forall x)(Lax = Lxa)
\]

If you don't see how to begin, use the same overall strategy we developed in chapter 6 of volume I. Write a skeleton derivation with its premise and final conclusion and ask what you need in order to get the final, or target, conclusion.

We could get our target conclusion by \( \forall I \) if we had a sentence of the form \( \forall b \overset{\sim}{=} \forall \bar{a} \). Let's write that in to see if we can make headway in this manner:

\[
(\forall x)(Lax \land Lxa) \\
\forall I \\
(\forall x)(Lax = Lxa)
\]

Notice that 'b' gets a hat wherever it appears in the main derivation. There, 'b' is not governed by any assumption in which 'b' occurs. But 'b' occurs in the assumptions of both subderivations. So in the subderivations 'b' gets no hat. Finally, 'a' occurs in the original premise. That by itself rules out putting a hat on 'a' anywhere in the whole derivation, which includes all of its subderivations.

Back to the question of how we will fill in the subderivations. We need to derive \( \forall b \) in the first and \( \forall x \) in the second. Notice that if we apply \( \forall E \) to the premise, using 'b' to instantiate 'x', we get a conjunction with exactly the two new target sentences as conjuncts. We will be able to apply \( \& E \) to the conjunction and then simply reiterate the conjuncts in the subderivations. Our completed derivation will look like this:
Natural Deduction for Predicate Logic

5-5. The Universal Introduction Rule

Once more, notice that 'b' gets a hat in lines 2, 3, and 4. In these lines no premise or assumption using 'b' is operative. But in lines 5, 6, 8, and 9, 'b' gets no hat, even though exactly the same sentences appeared earlier (lines 3 and 4) with hats on 'b'. This is because when we move into the subderivations an assumption goes into effect which says something special about 'b'. So in the subderivations, off comes the hat. As soon as this special assumption about 'b' is discharged, and we move back out of the subderivation, no special assumption using 'b' is in effect, and the hat goes back on 'b'.

You may well wonder why I bother with the hats in lines like 2, 3, 4, 7, on which I am never going to universally generalize. The point is that, so far as the rules go, I am permitted to universally generalize on 'b' in these lines. In this problem I don't bother, because applying VI to these lines will not help me get my target conclusion. But you need to develop awareness of just when the formal statement of the VI rule allows you to apply it. Hence you need to learn to mark those places at which the rule legitimately could apply.

Students often have two more questions about hats. First, VI permits you to universally generalize on a name with a hat. But you can also apply $\exists i$ to a name with a hat. Now that I have introduced the hats, the last example in section 5-3 should really look like this:

If everyone loves themself, then Arb loves him or herself, whoever Arb may be. But then someone loves themself. When a name occurs arbitrarily, the name can refer to anything. But then it also refers to something. You can apply either VI or $\exists i$ to a hatted name.

It is also easy to be puzzled by the fact that a name which is introduced in the assumption of a subderivation, and thus does not occur arbitrarily there, can occur arbitrarily after the assumption of the subderivation has been discharged. Consider this example:

In the subderivation something is assumed about 'a', namely, that it has the property P. So, from the point of view of the subderivation, 'a' is not arbitrary. As long as the assumption of the subderivation is in effect, 'a' cannot refer to just anything. It can only refer to something which is P. But after the subderivation's assumption has been discharged, 'a' is arbitrary. Why? The rules tell us that 'a' is arbitrary in line 7 because line 7 is not governed by any premises or assumptions in which 'a' occurs. But to make this more intuitive, notice that I could have just as well constructed the same subderivation using the name 'b' instead of 'a', using $\exists i$ to write $\forall b \exists Qb$ on line 7. Or I could have used 'c', 'd', or any other name. This is why 'a' is arbitrary in line 7. I could have arrived at a conditional in line 7 using any name I liked instead of using 'a'.

Some students get annoyed and frustrated by having to learn when to put a hat on a name and when to leave it off. But it's worth the effort to learn. Once you master the hat trick, VI is simple: You can apply VI whenever you have a name with a hat. Not otherwise.

EXERCISES

5-4. There is a mistake in the following derivation. Put on hats where they belong, and write in the justification for those steps which are justified. Identify and explain the mistake.
5–5. Provide derivations which establish the validity of the following arguments. Be sure you don’t mix up sentences which are a quantification of a sentence formed with a ‘&’, a ‘v’, or a ‘3’ with compounds formed with a ‘&’, a ‘v’, or a ‘3’, the components of which are quantified sentences. For example, ‘(Vx)(Px & Qa)’ is a universally quantified sentence to which you may apply VE. ‘(Vx)Px & Qa’ is a conjunction to which you may apply &E but not VE.

a) \[(Vx)(Fx & Gx)\] b) \[(Vx)(Mx v Nx)\] c) \[A\]
   \[
   \begin{align*}
   & (Vx)Fx \\
   & (Vx)Mx \\
   & (Vx)(A v Nx)
   \end{align*}
   \]

5–6. THE EXISTENTIAL ELIMINATION RULE

VE and \(\exists E\) are difficult rules. Many of you will have to work patiently over this material a number of times before you understand them clearly. But if you have at least a fair understanding of VE, we can proceed to \(\exists E\) because ultimately these two rules need to be understood together.

Let’s go back to the first example in section 5–4: Everyone likes either rock music or country/western. Someone does not like rock. So someone likes country/western. I will symbolize this as

\[
(Vx)(Rx v Cx)
\]

\[
(\exists x)\neg Rx
\]

\[
(\exists x)Cx
\]

In informally showing this argument’s validity, I used ‘Doc’, which I will now write just as ‘d’, as a stand-in name for the unknown “someone” who does not like rock. But I must be careful in at least two respects:

i) I must not allow myself to apply VI to the stand-in name, ‘d’. Otherwise, I could argue from ‘(\exists x)\neg Rx’ to ‘\neg Rd’ to ‘(Vx)\neg Rx’. In short, I have to make sure that such a name never gets a hat.

ii) When I introduce the stand-in name, ‘d’, I must not be assuming anything else about the thing to which ‘d’ refers other than that ‘\neg R’ is true of it.

It’s going to take a few paragraphs to explain how we will meet these two requirements. To help you follow these paragraphs, I’ll begin by writing down our example’s derivation, which you should not expect to understand until you have read the explanation. Refer back to this example as you read:

\[
(Vx)(Rx v Cx)
\]

\[
(\exists x)\neg Rx
\]

\[
(\exists x)Cx
\]

I propose to argue from the premise, ‘(\exists x)\neg Rx’, by using the stand-in name, ‘d’. I will say about the thing named by ‘d’ what ‘(\exists x)\neg Rx’ says.
about "someone". But I must be sure that 'd' never gets a hat. How can I guarantee that? Well, names that occur in assumptions can't get hats anywhere in the subderivation governed by the assumption. So we can guarantee that 'd' won't get a hat by introducing it as an assumption of a subderivation and insisting that 'd' never occur outside that subderivation. This is what I did in line 3. '¬Rd' appears as the subderivation's assumption, and the 'd' written just to the left of the scope line signals the requirement that 'd' be an Isolated Name. That is to say, 'd' is isolated in the subderivation the scope line of which is marked with the 'd'. An isolated name may never appear outside its subderivation.

Introducing 'd' in the assumption of a subderivation might seem a little strange. I encounter the sentence, '(∃x)¬Rx', on a derivation. I reason: Let's assume that this thing of which '¬R' is true is called 'd', and let's record this assumption by starting a subderivation with '¬Rd' as its assumption, and see what we can derive. Why could this seem strange? Because if I already know '(∃x)¬Rx', no further assumption is involved in assuming that there is something of which '¬R' is true. But, in a sense, I do make a new assumption in assuming that this thing is called 'd'. It turns out that this sense of making a special assumption is just what we need.

By making 'd' occur in the assumption of a subderivation, and insisting that 'd' be isolated, that it appear only in the subderivation, I guarantee that 'd' never gets a hat. But this move also accomplishes our other requirement: If 'd' occurs only in the subderivation, 'd' cannot occur in any outer premise or assumption.

Now let's see how the overall strategy works. Look at the argument's subderivation, steps 3–7. You see that, with the help of reiterated premise 1, from '¬Rd' I have derived '(∃x)Cx'. But neither 1 nor the conclusion '(∃x)Cx' uses the name 'd'. Thus, in this subderivation, the fact that I used the name 'd' was immaterial. I could have used any other name not appearing in the outer derivation. The real force of the assumption '¬Rd' is that there exists something of which '¬R' is true (there is someone who does not like rock). But that there exists something of which '¬R' is true has already been given to me in line 2! Since the real force of the assumption of line 3 is that there exists something of which '¬R' is true, and since I am already given this fact in line 2, I don't really need the assumption 3. I can discharge it. In other words, if I am given the truth of lines 1 and 2, I know that the conclusion of the subderivation, 7, must also be true, and I can enter 7 as a further conclusion of the outer derivation.

It is essential, however, that 'd' not appear in line 7. If 'd' appeared in the final conclusion of the subderivation, then I would not be allowed to discharge the assumption and enter this final conclusion in the outer derivation. For if 'd' appeared in the subderivation's final conclusion, I would be relying, not just on the assumption that '¬R' was true of something, but on the assumption that this thing was named by 'd'.

The example's pattern of reasoning works perfectly generally. Here is how we make it precise:

A name is Isolated in a Subderivation if it does not occur outside the subderivation. We mark the isolation of a name by writing the name at the top left of the scope line of its subderivation. In applying this definition, remember that a sub-sub-derivation of a subderivation counts as part of the subderivation.

Existential Elimination Rule: Suppose a sentence of the form (3u)(... u ...) appears in a derivation, as does a subderivation with assumption (... s ...), a substitution instance of (3u)(... u ...). Also suppose that s is isolated in this subderivation. If X is any of the subderivation's conclusions in which s does not occur, you are licensed to draw X as a further conclusion in the outer derivation, anywhere below the sentence (3u)(... u ...) and below the subderivation. Expressed with a diagram:

![Diagram]

When you annotate your application of the ∃E rule, cite the line number of the existentially quantified sentence and the inclusive line numbers of the subderivation to which you appeal in applying the rule.

You should be absolutely clear about three facets of this rule. I will illustrate all three.

Suppose the ∃E rule has been applied, licensing the new conclusion, X, by appeal to a sentence of the form (3u)(... u ...) and a subderivation beginning with assumption (... s ...):

1) s cannot occur in any premise or prior assumption governing the subderivation,
2) s cannot occur in (3u)(... u ...), and
3) s cannot occur in X.

All three restrictions are automatically enforced by requiring s to be isolated in the subderivation. (Make sure you understand why this is cor-
Some texts formulate the $\exists E$ rule by imposing these three requirements separately instead of requiring that $s$ be isolated. If you reach chapter 15, you will learn that these three restrictions are really all the work that the isolation requirement needs to do. But, since it is always easy to pick a name which is unique to a subderivation, I think it is easier simply to require that $s$ be isolated in the subderivation.

Let us see how things go wrong if we violate the isolation requirement in any of these three ways. For the first, consider:

\[
\begin{array}{c|c}
\text{Ca} & 1 \text{ Ca} \\
(\exists x)Bx & 2 (\exists x)Bx \\
(\exists x)(Cx & Bx) & 3 a, Ba \\
& 4 Ca & 1, R \\
& 5 Ca & B & 3, 4, &I \\
& 6 (\exists x)(Cx & Bx) & 5, \exists I \\
& 7 (\exists x)(C & Bx) & \text{Mistaken attempt to apply } \exists E \text{ to 2 and 3-6. } 'a' \text{ occurs in premise 1 and is not isolated in the subderivation.}
\end{array}
\]

From the fact that someone is blond, it will never follow that everyone is blond.

One more example will illustrate the point about a sub-sub-derivation being part of a subderivation. The following derivation is completely correct:

\[
\begin{array}{c|c}
(\forall x)(\exists y)Lxy & 1 (\exists x)Lxx \\
(\exists x)Lxx & (\forall x)(\exists y)Lxy \\
& 2 (\exists x)Lxy \\
& 3 a, Laa \\
& 4 (\exists x)Lxx & 3, \exists I \\
& 5 (\exists x)Lxx & \text{Mistaken attempt to apply } \exists E \text{ to 2 and 3-4. } 'a' \text{ occurs in 2 and is not isolated in the subderivation.}
\end{array}
\]

From the fact that Adam is clever and someone (it may well not be Adam) is blond, it does not follow that any one person is both clever and blond.

Now let's see what happens if one violates the isolation requirement in the second way:

\[
\begin{array}{c|c}
(\forall x)(\exists y)Lxy & 1 (\exists x)Bx \\
(\exists x)Lxx & 2 (\exists x)Lxx \\
& 3 d, Bd \\
& 4 Cd & A \\
& 5 Cd & \exists B & 5, \exists E \\
& 6 Bd & 4, 6, \exists E \\
& 7 Bd & 3, R \\
& 8 Bd & 3, R \\
& 9 \sim Bd & 4-8, \sim I & 3, \exists I \\
& 10 Ce & 9, \exists I \\
& 11 (\exists x)\sim Cx & 2, 3-10, \exists E \\
\end{array}
\]

From the fact that everyone loves someone, it certainly does not follow that someone loves themselves.

And, for violation of the isolation requirement in the third way:

\[
\begin{array}{c|c}
(\exists x)Bx & 1 (\exists x)Bx \\
(\forall x)Bx & 2 a, Ba \\
& 3 Ba & 2, R \\
& 4 Ba & 2, R \\
& 5 (\forall x)Bx & \text{Mistaken attempt to apply } \exists E \text{ to 1 and 2-3. } 'a' \text{ occurs in 4 and is not isolated in the subderivation.}
\end{array}
\]

You might worry about this derivation: If 'd' is supposed to be isolated in subderivation 2, how can it legitimately get into sub-sub-derivation 3?

A subderivation is always part of the derivation in which it occurs, and the same holds between a sub-sub-derivation and the subderivation in which it occurs. We have already encountered this fact in noting that the premises and assumptions of a derivation or subderivation always apply to the derivation's subderivations, its sub-sub-derivations, and so on.
Now apply this idea about parts to the occurrence of ‘d’ in sub-subderivation 3 above: When I say that a name is isolated in a subderivation I mean that the name can occur in the subderivation and all its parts, but the name cannot occur outside the subderivation.

Here is another way to think about this issue: The ‘d’ at the scope line of the second derivation means that ‘d’ occurs to the right of the scope line and not to the left. But the scope line of subderivation 3 is not marked by any name. So the notation permits you to use ‘d’ to the right of this line also.

I hope that you are now beginning to understand the rules for quantifiers. If your grasp still feels shaky, the best way to understand the rules better is to go back and forth between reading the explanations and practicing with the problems. As you do so, try to keep in mind why the rules are supposed to work. Struggle to see why the rules are truth preserving. By striving to understand the rules, as opposed to merely learning them as cookbook recipes, you will learn them better, and you will also have more fun.

EXERCISES

5–6. There is one or more mistakes in the following derivation. Write the hats where they belong, justify the steps that can be justified, and identify and explain the mistake, or mistakes.

1  (Vx)(3x)Lxy  
2  (3x)Lxb  
3  Lab  
4  (Vx)Lay  
5  (3x)(Vx)Lxy  
6  (3x)(Vx)Lxy

5–7. Provide derivations which establish the validity of the following arguments:

a) (3x)(Fx C Gx)  b) (3x)(A C Px)  c) (3x)(Hmx v Gxn)  
   (3x)(Fx)  
   (3x)(A C (3x)Px)  
   (3x)(Fx)  

5–8. Are you bothered by the fact that 3E requires use of a subderivation with an instance of the existentially quantified sentence as its assumption? Good news! Here is an alternate version of 3E which does not require starting a subderivation:

Show that, in the presence of the other rules, this version is exchangeable with the 3E rule given in the text. That is, show that the above is a derived rule if we start with the rules given in the text. And show that if we start with all the rules in the text except for 3E, and if we use the above rule for 3E, then the 3E of the text is a derived rule.
CHAPTER SUMMARY EXERCISES

Here is a list of important terms from this chapter. Explain them briefly and record your explanations in your notebook:

a) Truth Preserving Rule of Inference  
b) Sound  
c) Complete  
d) Stand-in Name  
e) Govern  
f) Arbitrary Occurrence  
g) Existential Generalization  
h) Universal Generalization  
i) Isolated Name  
j) Existential Introduction Rule  
k) Existential Elimination Rule  
l) Universal Introduction Rule  
m) Universal Elimination Rule
Appendix A

Modifications to the *Logic Primer*’s Formalism
A.1 Logical Notation

<table>
<thead>
<tr>
<th>New notation</th>
<th>Old notation</th>
<th>Name</th>
<th>Informal reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>∧</td>
<td>&amp;</td>
<td>conjunction</td>
<td>... and ...</td>
</tr>
<tr>
<td>∨</td>
<td></td>
<td>disjunction</td>
<td>... or ...</td>
</tr>
<tr>
<td>→</td>
<td>⊃</td>
<td>(material) implication</td>
<td>if ... then ...</td>
</tr>
<tr>
<td>↔</td>
<td>≡</td>
<td>biconditional</td>
<td>... if and only if ...</td>
</tr>
<tr>
<td>¬</td>
<td>~</td>
<td>negation</td>
<td>not ...</td>
</tr>
<tr>
<td>∀</td>
<td></td>
<td>universal quantifier</td>
<td>every ... ...</td>
</tr>
<tr>
<td>∃</td>
<td></td>
<td>existential quantifier</td>
<td>some ... ...</td>
</tr>
</tbody>
</table>
A.2 Modified Rules

\[ \begin{array}{c|c}
\text{\textbf{\textbackslash{}VE}} & \text{\textbf{\textbackslash{}I}} \\
\hline
m & A \lor B \\
\vdots & \vdots \\
n & A \\
\vdots & \vdots \\
p & C \\
p + 1 & B \\
\vdots & \vdots \\
q & C \\
q + 1 & C \quad \text{\textbackslash{}VE, } m, \ n-p, \ (p+1)-q \\
\end{array} \]

\[ \begin{array}{c|c}
\text{\textbf{\textbackslash{}I}} & \\
\hline
m & A \\
\vdots & \vdots \\
n & B \\
\vdots & \vdots \\
p & A \\
p + 1 & A \leftrightarrow B \quad \leftrightarrow I, \ m-n, \ (n+1)-p \\
\end{array} \]
A.2. MODIFIED RULES

APPENDIX A. MODIFICATIONS TO THE LOGIC PRIMER’S FORMALISM

\[ \leftrightarrow E \ (1) \]

\[
\begin{array}{c|c}
\vdots & \vdots \\
m & A \leftrightarrow B \\
\vdots & \vdots \\
n & A \\
\vdots & \vdots \\
p & B & \leftrightarrow E, m, n \\
\end{array}
\]

\[ \leftrightarrow E \ (2) \]

\[
\begin{array}{c|c}
\vdots & \vdots \\
m & A \leftrightarrow B \\
\vdots & \vdots \\
n & B \\
\vdots & \vdots \\
p & A & \leftrightarrow E, m, n \\
\end{array}
\]

\[ R \]

\[
\begin{array}{c|c}
\vdots & \vdots \\
m & A \\
\vdots & \vdots \\
n & A & R, m \\
\end{array}
\]

\[ X \]

(Implicit) eXchange rule:

In an application of a rule, the line numbers specified in the justification need not be in increasing order.

(e.g., in an application of rule \( \leftrightarrow E \ (1) \), it can be \( m < n \) as well as \( m > n \) – i.e., the formula \( A \) may occur either below or above the formula \( A \leftrightarrow B \))
**APPENDIX A. MODIFICATIONS TO THE LOGIC PRIMER’S FORMALISM**

**A.2. MODIFIED RULES**

**∀I**

\[
\begin{array}{c|c}
\vdash & \vdash \\
- & - \\
m & A[a/x]
\end{array}
\]

\[
\begin{array}{c|c}
- & - \\
- & - \\
m + 1 & \forall x A \quad \forall I, m
\end{array}
\]

* (a does not occur in A, nor in any premises)*

**∃E**

\[
\begin{array}{c|c}
\vdash & \vdash \\
- & - \\
m & \exists x A
\end{array}
\]

\[
\begin{array}{c|c}
- & - \\
- & - \\
n & A[a/x]
\end{array}
\]

\[
\begin{array}{c|c}
- & - \\
- & - \\
p & B
\end{array}
\]

\[
\begin{array}{c|c}
- & - \\
- & - \\
p + 1 & B \quad \exists E, m, n-p
\end{array}
\]

* (a does not occur in A, nor in B, nor in any premises)*
Corrections to A Modern Formal Logic Primer

Here is a list of corrections to the regrettably many printing errors in A Modern Formal Logic Primer and the answer manual, and correction to the one substantive error which have so far come to my attention. Where not otherwise indicated, the changes are indicated by underlining. 'I*' ('II*'') means line (lines) counted up from page bottom. These notes use notation similar to that found in the answer manual: '->' for the conditional, '<->' for the biconditional, (u) for universal quantifier, and (Eu) for existential quantifier.

Volume I

p. 50, on the left of the truth table, there should be single quotes around the hourglass.
p. 54, l. 9, substitute 'conditionals' for 'conditions.'
p. 54, Truth table definition of the biconditional, at page bottom: The annotation at the left of the truth table should have a triple bar, in single quotes.
p. 92, l. 24: set the conclusion T->U as a conclusion of 1, by ->I; and ->I, similarly
p. 103, I* 12: (Traditionally called "Modus Tollens")
p. 111, Problem 7-7-d: Delete last square parenthesis.
p. 132, I* 14: 3-4.
p.144, Problem 9-6 should refer to problem 9-5 instead of 9-4.
Inside back cover: Insert circles around conclusions in rules ~ , ~ , and =l.

Answer Manual

p. 2, Problem 1-4-d: Last line of the truth table should be f f f f f
p. 4, Problem 2-2-t: (Av~A)&[C&(DvE)]
p. 8, II* 6 [Third line of problem 3-6-g]: Annotation should read: D, SLE
p. 14, third line of problem 4-6-c: Annotation should read: C, SLE
p. 49, 8-4-c, line 4: annotation should read '>', not '->' (rule for conditional, not negated conditional).
p. 51, 8-4-h, lines 5 and 6: In both lines the annotation should read '>', not '->' (rule for conditional, not negated conditional).
p. 51, 8-4-i: There is a second counterexample, M&N.
p. 62, 9-2-t: Insert line numbers for lines 19 and 18. Also add '&~A' to the statement of the counterexample.
Volume II

There is an important substantive error on pp. 153 and 154, in the rewrite rule for the (The u) operator. (1a) on p. 153 should read: \((\exists x!)Lxe \& (x)(Lxe \rightarrow Bx)\). The rewrite rule, p. 154, II* 11-13, should read:

Rule for rewriting *Definite Descriptions Using '(The u)': 

\(Q[\text{The } u]P(u)\) is shorthand for \((\exists u!)P(u) \& (u)[P(u) \rightarrow Q(u)]\), where \(P(u)\) and \(Q(u)\) are open formulas with \(u\) the only free variable.

Then on page 155 the corresponding corrections need to be made in (2a): "...rewritten as \((\exists x!)Lxe \& (Ax)(Lxe \rightarrow \neg Bx)\)" and in (2b): "...rewritten as \(\neg[(\exists x!)Lxe \& (Ax)(Lxe \rightarrow Bx)]\)."

This formulation eliminates cases in which, for example there are two Ps and two Qs, but only one thing which is both P and Q. We require that there be exactly one P, and any P be a Q, so that if there is one P, it is Q also.

p. 16, l. 19: fies a list of objects of which the predicate is true. If the predicate is a two

p. 22, problem 2-2-e: \((x)(Bx \vee Lax)\)

p. 18, problem 2-1-f: \((x)(Kx \& Rax) \rightarrow (Ex)(Mx \vee Rcx)\)

p. 44, l. 13 (line 8 of informal proof): Change annotation to read '\(\neg A_s\)'

p. 47, l*10: miles an hour it is easy to hear 'A car can go very fast.' as meaning that

p. 51, l. *1 Change last 'x' to 's'.

p. 54, l*4: (18d) \((x)[(Bx \& Lxe) \rightarrow \neg(\exists x & Cx)]\).

p. 59, problem 4-8,Il: If anyone has a son, that person loves Eve.

p. 75, 1-2. a) should be 'Cc', not 'Ca'

p. 81. l* 12: the same subderivation using the name 'b' instead of 'a', using \(\rightarrow \) to write

p. 82, Second line of 4-2: rule should have 'S

'subscripted' '\(\neg(Eu)\)'s'

p. 84, ll. 13, 17, l* 13, 10, p. 85, l. 1: the 'R's should all be followed by an open variable, 'x'.

p. 85, In box, eliminate the first horizontal line.

p. 86, missing 'l' in n: "(x)(y)[size=32]...", and a missing 'l' in r: "(x)(y)\(\ldots\)"

p. 89, problem 5-7-1: Second premise starts with a universal, not an existential quantifier.
p. 95, l.3: arguments:

p., 96, problem 6-1-q: The conclusion should read: '(x)Px v (x)Qx'.

p. 102, l. 17: Eliminate "and 6-8".

p. 118, exercise 7-3e: In conclusion, 'E' should be reversed.

p. 97, l* 8: will also use a trivial reformulation of the rules ~I and RD expressed in

p. 125, l* 10: contradiction so that X and Y are logically equivalent. If there is an open

p. 135, problem 8-5-f: (x)[(y)Txy & (Ey)-Txy]

p. 139, l*11: With this notation we can give the _! rule:

[Note: Throughout I have used the notation, '(Ex!), etc, putting the exclamation mark AFTER the variable or metavariable. Traditional usage actually puts it after the sign of existential quantification, (Ex)' etc.

p. 139, l* 9. sign for universal quantification missing in ....(Av)P(v)...

p. 141, l. 6: (Add to transcription guide) Lxy: x loves y

p. 144, (Line 6 of the derivation): A Â¬B .-> (FÂ ¬ -> FÂ¬ )

p. 145, l. 15 'R' should be bold faced in '(Ax)R(x,x)'

p.146, Problem 9-5-d: Pa <-> (x)(x=a -> Px)

p. 152, l*3 (Conclusion of problem d): (Ex)(Ey)(f(x) ≠ y) [Insert parenthese around x]

p. 154, l. 17: In both occurrences, the Greek letter iota should be printed upside down.

p. 148, l*5: for writing in one, two, or more arguments (with the arguments separated by commas when there are more than one).

p. 167, l*16: D6': The set Z of sentences is Consistent iff Z A&¬A.

p. 252 (Page facing inside back cover): Should be the same as corresponding page in volume I.

Inside back cover: Insert circles around conclusions in rules ~ , ~ , and =I.

Answer Manual

p. 4, Problem 2-4-i: (R&B)&¬H (R: Roses are red. B: Violets are blue. H: Transcribing this jingle is hard to do.) [The answer orginally given did not make the negation explicit, and so was not as detailed a transcription as we can give.]
p. 76, problem 2-2-i: First substitution instance is true, not false.

p. 77, problem 2-3-i: True, not false

78, problem 2-6-e: D = \{e\}; ¬Be & Lee.

p. 82, In problem 4-1 and 4-2 'S' has to be read as a subscript when immediately following a quantifier (No subscript was available on the program that prepared the answer manual!) Also in problem 4-2, line 6 add 'DN' to the annotation. And in line 7 the annotation should read ∼(Eu)

p. 83, ll. 2-3: Exercises g) and h) have been reversed.

p. 83, problem 4-3-n: Remove an extra left parenthesis: (x)[Px -> (Ey)(Py & Lxy)].

p. 83, Problem 4-4-j: (x)(Cx -> ¬Dx)

p. 83, problem 4-4-y: Insert a left parenthesis: (x)[(Px & ...]

p. 83, problem 4-4-aa: Insert a left parenthesis: ∼(x)[(Px & ...]

p. 84, Problem 4-7-g: All dogs love all cats.

p. 84, Problem 4-7-h: All dogs love all cats.

p. 85, problem 4-8-hh: (Ex)(Px & Lxx & Lex)

p. 88, problem 5-2-d: Throughout 'Tx' should read Txd' and 'Tâ' should read Tâd'.

p. 103, problem 6-1-q: Line 14 should read: Pa v Qb. Line 15 should read: Qb. (Put hats on 'b')

p. 110, problem 6-6-c: l. 4: 5-7. b) and u).

p. 124, problem 7-3-g: The counterexample is Ha & Gb & ¬Ga & ¬Hb

p. 135, problem 8-1-i: The counterexamples should read: Na & Ma; Ma: ¬Na

p. 136, problem 8-1-k, first part: There are three more counterexamples, ¬Ga; Sa, Ka

p. 152, problem 9-1-c: (Ex)(Ey)(Ez)(Cx & Cy & Cz & x\#y & y\#z & x\#z)

p. 152, problem 9-1-e: (x)(y)(z)(w)[(Cx & .....]

p. 152, problem 9-2-h: Pa & ¬Sae & (x)[Px & x\#a & x\#e) -> Sxe]

p. 152, problem 9-2-j: (x)(Px -> (Ey)(Ez)[Myx & ...

p. 165, problem 9-11h. Line 3 should read: (z)(g(ˆb,z)=a)

p. 170, problem 9-12-c: (Ex)[Fx & (y)(Fyc -> y =x) & a=x] (delete '& (y)(Fyc')