2.4. Semantics for these Languages

An *interpretation* I of a formal language $\mathcal{L}$ for truth-functors consists of:

(1) an assignment of a truth-value, either T or F, to each sentence-letter in $\mathcal{L}$. This assignment is arbitrary, i.e. *any* such assignment is allowed.

(2) an assignment of truth-values to all the remaining formulae in $\mathcal{L}$, which is not arbitrary, but is calculated from the values assigned to the sentence-letters in accordance with the truth-tables of the truth-functors involved. For example, if the truth-functors of $\mathcal{L}$ are just $\neg,\wedge,\vee$, then the interpretation of the remaining formulae is determined by the rules

$$
\begin{array}{lll}
|\neg\phi| = \mathrm{T} & \text{iff} & |\phi| = \mathrm{F} \\
|\phi\wedge\psi| = \mathrm{T} & \text{iff} & |\phi| = \mathrm{T} \text{ and } |\psi| = \mathrm{T} \\
|\phi\vee\psi| = \mathrm{T} & \text{iff} & |\phi| = \mathrm{T} \text{ or } |\psi| = \mathrm{T}.
\end{array}
$$

Occasionally it will be useful to consider a non-standard interpretation, which does not obey the stipulations (1) and (2) above. In such a case we shall distinguish the interpretations obeying (1) and (2) as the *standard* interpretations. But 'interpretation' will mean 'standard interpretation' unless there is some indication to the contrary.

The definitions of entailment, inconsistency, and validity for languages for truth-functors are as given in the previous chapter (Section 1.3). For example, a set of formulae Γ is inconsistent iff there is no (standard) interpretation of any language for truth-functors in which all of those formulae are interpreted as true. But it is perhaps easier to think of it in a slightly different way. We will say that the *language* of a set Γ of formulae is the language which has as its vocabulary just the sentence-letters and truth-functors that occur in Γ. Then the set Γ is inconsistent iff in every interpretation of the language of Γ some formula in Γ is interpreted as false. Similarly Γ entails ϕ

iff, in every interpretation of the language of $\Gamma \cup \{\phi\}$, either some formula in Γ is false or the formula ϕ is true. Similarly again, ϕ is valid iff, in every interpretation of the language of $\{\phi\}$, ϕ is true. The point that we are here relying on is this: provided that we restrict attention to interpretations which do interpret every sentence-letter, and every truth-functor, in the formulae we are concerned with, then we may take it for granted that what is not true in that interpretation is false, and that what is not false is true. Moreover, we shall never need to consider interpretations which include more letters, or more truth-functors, than occur in the formulae under consideration. For the interpretation of the extra vocabulary cannot in any way affect the truth-values of formulae which lack that vocabulary.

The most straightforward test for validity or entailment or inconsistency is a truth-table test. So long as we may confine our attention to finite sets of formulae, this test can always be applied and will always yield a definite result. One begins by listing all the letters in the formula or formulae to be tested, and then all the different ways of assigning truth-values to those letters. For each assignation we shall construct a separate line of the truth-table, so when we have 2 letters to consider there will be 4 lines in the table, when we have 3 letters there will be 8 lines, and in general when we have n letters there will be 2^n lines. Each line thus represents one of the possible interpretations for the language of the formula or formulae to be tested, and we simply calculate the resulting truth-value for each whole formula in that interpretation, using the tables already stipulated for the various functors involved. The method of calculation is to work up from the shorter subformulae to the longer ones.

Here is a simple example to show that $P \rightarrow (\neg P \rightarrow Q)$ is a valid formula. The table is

P	Q	P	$\rightarrow$	$(\neg P$	$\rightarrow$	$Q)$
T	T	T	T	F	T	T
T	F	T	T	F	T	F
F	T	F	T	T	T	T
F	F	F	T	T	F	F

We have just two letters to consider, P and Q, so we begin by writing these on the left, and underneath them the four possible interpretations. Then on the right we write the formula we are interested in, and we begin by considering its shortest subformulae, which are the letters P and Q again. Under the first occurrence of P, and under the occurrence of Q, we have simply repeated the value they receive in each assignment. Under the second occurrence of P we have written nothing, because in this case we have at once put in the values of the next longer subformula $\neg P$. These values are written under the main truth-functor of $\neg P$, namely $\neg$. Using the truth-table for $\rightarrow$ we can now calculate the values, in each line, of the next longer subformula $\neg P \rightarrow Q$, and again we write these values under its main truth-functor, namely $\rightarrow$. Finally, we are now in a position to calculate the values of the whole formula, which we write under its main truth-functor, namely the first occurrence of $\rightarrow$ in the formula. For ease of reading, this column is sidelined, since it is the goal of the calculation. It turns out that only the value T occurs in this column, which is to say that in every interpretation of the language of the formula that formula is true, i.e. that the formula is valid.

Here is another example, to verify the entailment

$$P \rightarrow R, Q \rightarrow R \quad \models \quad (P \vee Q) \rightarrow R.$$

The relevant table is

P	Q	R	$P \rightarrow R$	$Q \rightarrow R$	$(P \vee Q) \rightarrow R$
T	T	T	T	T	T
T	T	F	F	F	F
T	F	T	T	T	T
T	F	F	F	T	F
F	T	T	T	T	T
F	T	F	T	F	F
F	F	T	T	T	T
F	F	F	T	T	T

In this table we have saved ink by not bothering to repeat the value of a single letter under that letter, but otherwise the procedure is just the same: the value of each formula is calculated from the values of its shorter subformulae. The table shows that whenever the two premiss-formulae are both true—i.e. in lines 1, 3, 5, 7, 8—the conclusion formula is true too; or equivalently (and easier to check) that whenever the conclusion formula is false—i.e. in lines 2, 4, 6—then at least one of the premiss-formulae is also false. This shows that the proposed entailment is indeed correct.

When setting out truth-tables, it is standard practice always to consider the various interpretations in the order illustrated in my last two examples. So, for instance, in a truth-table of 16 lines the column under the first letter would consist of 8 occurrences of T followed by 8 occurrences of F; the column under the second letter would have 4 occurrences of T alternating with 4 occurrences of F; the column under the third letter would have pairs of T alternating with pairs of F; and the column under the last letter would have T alternating with F. Each column begins with T and ends with F, so that the first interpretation considered is that in which all the letters are interpreted as true, and the last considered is that in which they are all interpreted as false. A sequent which can be shown to be correct by the truth-table test is called a *tautologous* sequent, and a single formula which can be shown to be valid by this test is called a *tautology*.

It is obvious that the task of writing out a full truth-table can become very laborious, especially when there are many letters to be considered. So it is natural to seek for some short cuts. One method which is often handy is this: seek to construct just *one* line of the truth-table, which will falsify the suggested entailment. If the construction succeeds, then obviously the entailment is not correct; if the construction fails, then this will be because it runs up against some obstacle which shows that *no* such construction could succeed. In that case, there is no falsifying line, and therefore the entailment is correct. To apply this method, one works in the opposite direction to that of the truth-tables, i.e. one calculates the values of subformulae from the values of the longer formulae that contain them.

Here is an example, testing an entailment which involves four letters, and which therefore would have 16 lines in its full truth-table. The finished diagram which records the reasoning is this:

$$\begin{array}{cccccccccccccc}
P & \to & R, & Q & \to & S & \models & (P & \lor & Q) & \to & (R & \lor & S) \\
\mathrm{F} & \mathrm{T} & \mathrm{F} & \mathrm{F} & \mathrm{T} & \mathrm{F} & & & \mathrm{T} & & \mathrm{F} & \mathrm{F} & \mathrm{F} & \mathrm{F} \\
\underline{\underline{5}} & 1 & 4 & \underline{\underline{5}} & 1 & 4 & & & \underline{\underline{2}} & & 1 & 3 & 2 & 3
\end{array}$$

(The numbers on the bottom line are put in only to help in the explanation of the reasoning.) We begin by writing down the entailment to be tested, which has two premiss-formulae and one conclusion-formula. Then our first step is to suppose that this suggested entailment is not correct, i.e. we suppose that there is an interpretation which makes both the premiss-formulae true, and the conclusion-formula false. We therefore put T under the main functor of each premiss-formula, and F under the main functor of the conclusion-formula. These three entries are labelled '1' on the diagram. Now there is nothing more that we can do with the premiss-formulae for the time being, so for our second step we just consider the conclusion. We observe that if the whole conditional $(P \lor Q) \to (R \lor S)$ is to be false, then its antecedent $P \lor Q$ must be true and its consequent $R \lor S$ must be false, so we write in T under the main functor of the first, and F under the main functor of the second. These two entries are labelled '2' on the diagram. The third step then notes that if the disjunction $R \lor S$ is to be false, then both R and S must be false, so F is entered twice more at the entries labelled '3'. We have now discovered that if there is an interpretation which falsifies our entailment, it must be one in which both R and S are interpreted as false. So our fourth step just writes in this information for the two premiss-formulae, and allows us to complete the argument. For if the premiss $P \rightarrow R$ is to be true, and the consequent R is now given as false, then the antecedent P must be false as well. So as a fifth step we can write F under P, and by the same reasoning we can also write F under Q. But now we have reached an impossibility, for if P is false, and Q is false, then the disjunction $P \vee Q$ must also be false, and yet we had said at step 2 that it would have to be true. We therefore underline the conflicting truth-values, and draw our conclusion: the proposed entailment must be correct. For the attempt to find a falsifying interpretation has run into a contradiction.

Reasoning by this method does not always work out quite so straightforwardly. Here is another example, which in this case does work out, but which begins to show how problems may arise. Let us see whether the entailment we have just tested also holds the other way round. In this case, our diagram works out like this:

$$\begin{array}{ccccccccccccccc}
(P & \vee & Q) & \rightarrow & (R & \vee & S) & \models & (P & \rightarrow & R) & \wedge & (Q & \rightarrow & S) \\
\mathrm{T} & \mathrm{T} & & \mathrm{T} & \mathrm{F} & \mathrm{T} & \mathrm{T} & & \mathrm{T} & \mathrm{F} & \mathrm{F} & \mathrm{F} & & \mathrm{T} & \mathrm{T} \\
 & & & & & & & & & \mathrm{A} & & & & & \\
4 & 5 & & 1 & 4 & 6 & 7 & & 3 & 2 & 3 & 1 & & 9 & 8
\end{array}$$

Step 1 enters T for the left formula, and F for the right formula, as before. But these values in fact do not determine any further values at all, so in order to get any further we must now make an *assumption*. There are several assumptions that one could make. We shall explore later a method which makes assumptions about the values of the individual sentence-letters. (This is Quine's method of truth-value analysis, introduced in Section 2.11.) But it is more in keeping with the present method to make an assumption about the value of a longer formula, in fact of a longest subformula that is not yet assigned a value. So we choose to consider the conclusion-formula $(P \rightarrow R) \wedge (Q \rightarrow S)$. Our initial supposition is that this conjunction is false. So it follows that one or other of the two conjuncts, $P \rightarrow R$ and $Q \rightarrow S$, is false, but we do not know which. We shall assume, then, that it is $P \rightarrow R$ that is false, marking this in on the diagram as step 2, but also labelling it 'A' for 'Assumption'. Now suppose that in following out the consequences of this assumption we meet an impossibility, as we did with the last example. Then what we have to do is to run the test again, this time making the alternative assumption that $Q \rightarrow S$ is false. If *both* assumptions lead to an impossible distribution of truth-values, then we know that there cannot be any falsifying interpretation. But that is not what happens with the present example, as we soon see. The assumption at step 2, that $P \rightarrow R$ is false, allows us to mark P as true and R as false at step 3. Step 4 then carries this information across to the left formula, and step 5 then notes that, since P is true, $P \vee Q$ must be true too. Combining this with the initial supposition that the left formula as a whole is true, we deduce in step 6 that $R \vee S$ must be true, and hence in step 7 that S must be true. Step 8 then carries this information across to the other occurrence of S, and step 9 infers that the clause $Q \rightarrow S$ must therefore be true. At this point we may stop. Admittedly, the value of Q has not yet been determined, and for completeness we should make some assumption about it. But it is easily seen that we could make either assumption, and the rest of the diagram will not be affected. So we have not run up against any impossibility. On the contrary, we have succeeded in constructing a line of the truth-table in which the premiss-formula is true and the conclusion-formula is false, thus showing that the proposed entailment is *not* correct.

As will be evident from these two examples, using this method requires more thought than constructing a full truth-table, but it can save a great deal of time. The method can also become rather complicated, if we are forced to make several different assumptions, one after the other, in testing the same entailment. But I shall not say any more about it at this stage, for in fact the basic idea that we are using here will be developed into an elegant and foolproof procedure in Chapter 4, with a simple technique for handling a number of assumptions.

EXERCISES

2.4.1. Use truth-tables to determine whether the following sequents are correct. (We use '$\phi \dashv\models \psi$' as short for '$\phi \models \psi$' and '$\psi \models \phi$'.)

(*a*) $\neg(P \rightarrow P) \models (P \rightarrow \neg P)$
(*b*) $\models (P \rightarrow Q) \vee (P \rightarrow \neg Q)$
(*c*) $\models (P \rightarrow Q) \vee (R \rightarrow P)$
(*d*) $(P \rightarrow Q) \wedge (P \rightarrow \neg Q) \models$
(*e*) $(P \leftrightarrow Q) \wedge (P \leftrightarrow \neg Q) \models$
(*f*) $P \rightarrow (Q \vee R) \quad \dashv\models \quad (P \rightarrow Q) \vee (P \rightarrow R)$
(*g*) $(Q \vee R) \rightarrow P \quad \dashv\models \quad (Q \rightarrow P) \vee (R \rightarrow P)$
(*h*) $P \leftrightarrow (Q \vee R) \quad \dashv\models \quad (P \wedge Q) \vee (P \wedge R) \vee (\neg P \wedge \neg Q \wedge \neg R)$
(*i*) $P \leftrightarrow (Q \vee R) \quad \dashv\models \quad (P \wedge Q \wedge R) \vee (\neg P \wedge \neg Q) \vee (\neg P \wedge \neg R)$.

2.4.2. Without writing out a full truth-table, determine whether the following sequents are correct. (Indicate your reasoning.)

(*a*) $P \rightarrow Q, Q \rightarrow R, R \rightarrow S \vDash P \rightarrow S$
(*b*) $P \vee Q, \neg(P \wedge R), \neg(Q \wedge S) \vDash \neg(R \wedge S)$
(*c*) $P \rightarrow (Q \vee R), R \rightarrow (P \rightarrow S), \neg(S \wedge P) \vDash Q \rightarrow P$
(*d*) $P \rightarrow (Q \vee R), R \rightarrow (P \rightarrow S), \neg(S \wedge P) \vDash P \rightarrow Q.$

3

Quantifiers

3.1. Names and Extensionality

The logic to be studied in this chapter is standardly called 'predicate logic', as the logic of the last chapter is standardly called 'propositional logic'. But a much better name for it is the logic of quantifiers, or, more fully, the logic of 'elementary' or 'first-order' quantifiers. However, there is a more basic notion to be considered before we can come to the quantifiers, and that is the idea of a *name*, or more generally of a *logical subject*. For the elementary, or first-order, quantifiers take the place of names, and sentences containing these quantifiers are most easily understood in terms of the simpler sentences that result from them, when the quantifiers are dropped and names restored in their place. However, there is no agreement amongst philosophers on what is to count as a name, or logical subject, and this is not the place to explore the issues involved. I therefore offer only a bare outline of what are, from the logician's point of view, the crucial assumptions.

The paradigm of a name is an ordinary proper name, written (in English) with an initial capital, whose role is to make a singular reference to a particular object of some kind, e.g. the name of a person, a place, a ship, a hurricane, or whatever. What is important about names, for logical purposes, is this job of singular reference that they perform. So we generalize the idea and say that any other expression too may be counted as a name, for our purposes, if it too performs the same job. I think there would be general agreement that demonstrative expressions, such as 'this book' or 'that chair', may be so counted, though, of course, they are not ordinarily thought of as 'names'. But there is no agreement on other examples. In particular there is a category of expressions called 'definite descriptions' on which philosophers are deadlocked. I shall return to this topic in Chapter 8, but meanwhile I simply leave the question open. The important thing about a name is that it is an expression used to make a singular reference to a particular object; just which expressions do play this role must for the time being remain undetermined.

An expression that is commonly used as a name may nevertheless be functioning in some different way in a particular context. To take a simple example, consider the sentence

'Horace' rhymes with 'Doris' and with 'Maurice'.

The three expressions here quoted are standardly used as names that refer to people, but it is clear that that is not their role here. On the contrary, we say that in this sentence the name 'Horace' is not being *used* at all, but only mentioned, and the different name '"Horace"' is being used to refer to it. In other words, '"Horace"' is functioning here, as it usually does, as a name of 'Horace', whereas 'Horace' is not functioning as a name at all. The important point is that for our purposes we do not count an expression as a name unless it is actually functioning as a name in whatever context is under consideration, i.e. unless it is, in that context, being used to refer to a particular object. By an extension of the same idea, we do not count anything as a repetition of the *same* name unless it is, in the context, being used to refer to the same object. And this implies that an expression is not counted as a name unless it *succeeds* in referring to an object, i.e. unless there really is an object to which it refers.

This is the first assumption that we make about names, i.e. that they always do refer. (In Chapter 8 I shall consider abandoning this assumption, but for the time being it is imposed.) There is also a second assumption, which may naturally be regarded as an extension of the first: for logical purposes, it is only the object referred to that is important, and not the name that is used to refer to it. To put this more precisely: the truth-value of a proposition which contains a name will (usually) depend upon which object that name refers to, but it will never depend upon which name is being used to refer to it; consequently, any other name with the same reference could have been used instead, and the truth-value of the whole would not be affected. That is, if two names each refer to the same object, then in any proposition which contains either of them the other may always be substituted in its place, and the truth-value of the proposition will be unaltered. This assumption is called the *Principle of Extensionality.*

Any natural language will abound with counter-examples to this principle. To adapt what must count as the case best known to philosophers (see Frege 1892/1960), it is true (given a suitable understanding of 'Ancient') that

> The Ancient Egyptians did not know that the Morning Star and the Evening Star are the same heavenly body.

Now in fact the expressions 'the Morning Star' and 'the Evening Star' are names of the same heavenly body, to wit the planet Venus. So, according to the principle of extensionality, it should always be possible to substitute either for the other without affecting truth-value. Accordingly, it should be equally true that

> The Ancient Egyptians did not know that the Morning Star and the Morning Star are the same heavenly body.

But this conclusion is manifestly absurd. Nor can one say that this apparent counter-example arises only because we have taken as names the expressions 'the Morning Star' and 'the Evening Star', whereas these expressions are not really names (but, say, disguised descriptions). For with *any* two names 'a' and 'b', which are in fact names of the same object, it will surely be possible to know that $a = a$ without knowing that $a = b$. What is causing the trouble in this example is not the particular names involved, but the kind of claim that is being made by the sentence as a whole.

There are many other examples of this phenomenon in an ordinary language. They are cases where we have a name which is apparently being used to refer to something, but where it makes a difference if we substitute some other name with the same reference. In such a case, the function that the name is performing cannot *just* be to refer to that object; it must be playing some other role as well, which a different name might fail to play, even though it too referred to the same object. So the name is not being used *purely* to refer. We shall say, in such a case, that the name does not occur *extensionally*. Where, however, the substitution of any other name with the same reference must preserve the same truth-value, there the name does occur extensionally. For logical purposes, it is only the extensional occurrences of names that we shall count as being occurrences of names. (I observe in parenthesis that it is common to find names occurring non-extensionally where they occur in a clause which specifies what is going on in someone's mind—e.g. what he knows, believes, fears, doubts, is thinking of, and so on. The same phenomenon also occurs when we are concerned with what is necessary, or possible, or probable; or with the question of how some fact is to be explained, or some conduct justified; and in other cases too.)

EXERCISES

3.1.1. The names in italics in the following sentences do not occur there as names, from a logical point of view. Explain why not.

(*a*) Many people have painted pictures of *Pegasus*.
(*b*) It is not likely that *King Arthur* existed.
(*c*) *The Pope* is usually an Italian.
(*d*) Trieste is no *Vienna*.
(*e*) *Giorgione* was so-called because of his size.

3.1.2. Discuss whether the names in italics in the following sentences do occur there as names, from a logical point of view. (Be prepared to find that some examples are ambiguous, and that we can accept them as using names when taken in one way, but not when taken in another.)

(*a*) *George Eliot* was Mary Ann Evans.
(*b*) At the time, no one knew that *George Eliot* was Mary Ann Evans.
(*c*) There are some who believe that Shakespeare was *Bacon*.
(*d*) Oedipus did not know that *Laius* was his father.
(*e*) No one is afraid of *Dr Jekyll*.
(*f*) It is most improbable that *Dr Jekyll* is a murderer.
(*g*) *The Morning Star* and the Evening Star might have turned out to be different planets.
(*h*) The police took no action because it was *Prince Charles* who was driving.

3.2. **Predicates, Variables, Quantifiers**

A simple way of approaching the modern[1] notion of a predicate is this: given any sentence which contains a name, the result of dropping that name and leaving only a gap in its place is a predicate. Given a suitable context, the sentence as a whole will express some proposition, true or false, about some object. The name refers to the object, and the rest of the sentence, i.e. the predicate, says something about it, something that will be either *true of* the object, or *false of* the object, as the case may be. (And as we commonly say, for short, that sentences themselves are true or false, so we shall similarly speak of predicates themselves being true of certain objects and false of others.) It should be noted that this way of speaking already presupposes that the name we began with occurred extensionally, for we do not allow for the idea that a predicate may be 'true of' an object under one name, but 'false of' it under another. On the contrary, the predicate is either true or false of the object itself, without room for any further qualification. So, if we remove from a sentence a *non-extensional* occurrence of a name, then what is left is *not* to be counted as a predicate, for our purposes.

In fact it will not quite do to say, as I have just done, that a predicate is just a sentence with a gap in it. Some simple *one*-place predicates may be regarded in this way, without creating any problem, but more will need to be said as soon as we move on to consider sentences with several gaps in them. These can arise either because we began with a sentence containing several names, and dropped more than one of them; or because we began with a sentence containing the same name several times over, and dropped that name at more than one of its occurrences; or, of course, from any mixture of these two reasons. To avoid the ambiguity that can result, we shall never in fact write a simple gap, but will always mark that gap by writing in it a letter that is called a *variable*. As variables we introduce the following alphabet

$$u,v,w,x,y,z,u_1,\ldots$$

The point of marking a gap in this way is that two gaps which are each marked with the same variable must each be filled by the same name, if we are to form a sentence containing the predicate in question; whereas gaps which are marked by different variables may be filled by different names.

A sentence which contains some variables in place of names is called an *open sentence*, so in practice we shall use open sentences to represent our

[1] Note that the use of the words 'subject' and 'predicate' from Aristotle to Kant was very different from that introduced here.

predicates. An open sentence which contains just one variable, occurring one or more times, represents a one-place (or monadic) predicate, true or false of single objects; one obtains a genuine sentence from it and a single name, upon substituting that name for all occurrences of its variable. An open sentence which contains just two variables replacing names represents a two-place (or dyadic) predicate—also called a dyadic relation—true or false of ordered pairs of objects; one obtains a genuine sentence from it and a pair of names, by substituting the first name of the pair for all occurrences of the first variable, and the second name for all occurrences of the second variable. (Note that, as a special case, the first name of the pair might happen to be the *same* name as the second.) The generalization is obvious: an open sentence containing just *n* distinct variables replacing names[2] represents an *n*-place predicate, true or false of ordered *n*-tuples of objects. Two special notes may be added to this statement. (1) It is convenient to count the generalization as covering *zero-place* predicates; these are represented by 'open' sentences with *no* variables replacing names, i.e. they are just ordinary sentences. (2) There is no need to distinguish between the 'one-tuple' of an object and the object itself, so we shall not bother to do so.

As schematic letters to take the place of names we shall use the alphabet

$$a,b,c,d,e,a_1,\ldots$$

As schematic letters for predicates we shall use

$$F,G,H,P,Q,R,S,T,F_1,\ldots$$

A predicate-letter will be followed by one or more name-letters, as in '*Fa*' or '*Gab*', to stand in for a sentence containing those names; or by one or more variables, as in '*Fx*' or '*Gxy*', to stand in for the corresponding open sentences; or by a mixture of the two, as in '*Gxb*'. For official purposes each predicate-letter is regarded as having some definite numeral superscripted, so that F^n represents only the *n*-place predicates, needing to be followed by *n*-tuples of name-letters or variables.[3] But in practice we shall always omit these superscripts, for since a predicate-letter is always followed by some definite number of name-letters and variables, this itself allows us to reconstruct its superscript. There is, however, one point arising from the omission to which it is worth drawing attention. Both '*Fa*' and '*Fab*' will be counted as formulae, and therefore so also is '$Fa \wedge Fab$'. But, despite appearances, this formula does not contain the *same* predicate-letter twice over. For the first

[2] The qualification 'replacing names' is essential. It restricts attention to *free* occurrences of variables. (See further pp. 79–80).

[3] An *n*-tuple is a series of *n* items, not necessarily distinct (cf. p. 16 above). E.g. $\langle a,b,a,c,a\rangle$ is a 5-tuple.

occurrence of '*F*' is short for 'F^1', and the second is short for 'F^2', and these are different predicate-letters.

We have now introduced names, variables, and predicates, which form the auxiliary vocabulary needed to express quantification, so we can proceed to the quantifiers themselves. For the time being[4] we shall recognize only two quantifiers, namely the universal quantifier '$\forall$', which does the work of the English 'all' or 'every' or 'any', and the existential quantifier '$\exists$', which does the work of the English 'some' or 'a'. (The inverted 'A' is intended to suggest 'All'; the reversed 'E' is intended to suggest the Existence expressed by 'there is a' or 'there are some'.) But from a grammatical point of view our quantifiers $\forall$ and $\exists$ do not work in the same way as their English counterparts.

If we begin with an open sentence, containing (say) just the one variable x and no other, then in English one could form an ordinary sentence from it in two main ways: either one could replace the variable x by a name, or one could replace it by a quantifying expression such as 'everything' or 'something' or 'all men' or 'some girls'. But in logic we never write a quantifier in the same position as one could write a name. On the contrary, we do not replace the variable x by a quantifier, but we *prefix* a quantifier which, as we say, *binds* that variable. (The point of this departure from natural languages is that it makes clear what the *scope* of the quantifier is.) We show which variable a quantifier is binding by writing that variable itself immediately after the quantifying expression $\forall$ or $\exists$. Thus one writes '$\forall x$' for the universal quantifier binding the occurrences of 'x' in what follows, or '$\exists y$' for the existential quantifier binding the variable 'y' in what follows; and so on. The closest analogue to this in English is to prefix a phrase such as 'Everything is such that' or 'Something is such that', and then to replace the subsequent occurrences of the relevant variable by occurrences of the pronoun 'it', all governed by the opening prefix. So it comes about that English sentences such as

All men are mortal
Some girls wear blue stockings

are rephrased in our logical notation as

$\forall x$(if x is a man then x is mortal)
$\exists x$(x is a girl and x wears blue stockings)

and are represented in our schematic language by formulae such as

[4] The situation will alter in Section 8.1.

$\forall x(Fx \rightarrow Gx)$
$\exists x(Fx \wedge Gx)$.

As in the case of the truth-functors, there is plenty of room for debate over the relation between the quantifiers of logic and their analogues in ordinary languages, but that is not a topic for the present book.

EXERCISE

3.2.1. Briefly explore the topic that is not for this book. Let us label the four statements thus:

(1) All men are mortal.
(2) Some girls wear blue stockings.
(3) $\forall x$(x is a man $\rightarrow$ x is mortal).
(4) $\exists x$(x is a girl $\wedge$ x wears blue stockings).

Consider the following arguments for saying that (1) does not mean the same as (3), and that (2) does not mean the same as (4):

(*a*) (2) implies that more than one girl wears blue stockings; (4) does not.
(*b*) (2) implies that there are also girls who do not wear blue stockings; (4) does not.
(*c*) (1) implies (or presupposes) that there are men; (3) does not.
(*d*) Indeed, (3) is true if there are no men; (1) is not.
(*e*) (1) is about men and nothing else; (3) is apparently about all objects. So the two have different subject-matters.
(*f*) In fact there is no saying what (3) is about, since its domain is just left unspecified. (And the specification 'all objects' is no help, for 'object' is so vague a word that we do not know what is to count as an object for this purpose.) Thus (3) has no clear domain of quantification, whereas (1) does; for (1) quantifies just over men.

How good are these arguments? Would it be possible to meet them by adopting more complex versions of (3) and (4)?

3.3. Languages for Quantifiers

In order to study the effect of quantifiers upon entailment we shall again concentrate on so-called 'formal' languages—i.e. schematic languages—in which definite quantifiers occur, but no definite names or predicates. We shall allow definite truth-functors to occur too, so that the languages in question are really schematic languages for truth-functors *and* quantifiers, and the languages of the previous chapter will be a special case of those to be introduced now. The vocabulary from which the languages are built therefore consists of

name-letters: $a,b,c,d,e,a_1,\ldots$
variables: $u,v,w,x,y,z,u_1,\ldots$
n-place predicate-letters: $F^n,G^n,H^n,P^n,Q^n,R^n,S^n,T^n,F_1^n,\ldots$ (for $n\geq 0$).

(The zero-place predicate-letters are the sentence-letters of the previous chapter.) It is convenient to add here a further definition:

The *terms* of a language are its name-letters and its variables together.

In addition to this vocabulary the languages may contain any desired truth-functors (which will add brackets to the notation) and either none, or one, or both of the quantifiers $\forall$ and $\exists$.

For definiteness, let us suppose that we wish to include both quantifiers in our language, but of truth-functors only $\neg,\wedge,\vee$. Then the formation rules are:

(1) An n-place predicate-letter, followed by n terms, is a formula.
(2) If ϕ is a formula, so is $\neg\phi$.
(3) If ϕ and ψ are formulae, so are $(\phi\wedge\psi)$ and $(\phi\vee\psi)$.
(4) If ϕ is a formula, and ξ a variable, $\forall\xi\phi$ and $\exists\xi\phi$ are formulae.
(5) There are no other formulae.

Apart from variations in the truth-functors and quantifiers that may be included in the language, we also permit variations in the name-letters, variables, and predicate-letters: the language may contain only some, or perhaps none, of those listed. For example, we may consider a language which has no name-letters (as several authors do), or one which has only the one-place predicate-letters, and so on.

The formation rules just given have some rather unnatural consequences. One might, indeed, look askance even at rule (1), which provides for the *atomic formulae*. For this rule allows as a formula not only schematic expressions such as '*Fa*', which take the place of sentences containing names, but also expressions such as '*Fx*', which take the place of *open* sentences. An open sentence, however, is not a proper sentence; it cannot express any definite proposition, and it makes no sense to assign it a truth-value. Do we want such a thing to be counted as a formula? But a far more serious difficulty arises with rule (4), which says that if ϕ is *any formula whatever* then we can always form another formula by adding a prefix such as '$\forall x$' or '$\exists y$', whether or not there are any further occurrences of 'x' or 'y' in the formula, to be bound by this prefix. So, for example, the following count as formulae: '$\forall xP$', '$\exists yFxx$', '$\forall x\exists x\forall xFa$', and so on. But these are expressions which correspond to no English sentences or open sentences; it is natural to say that they make no sense at all.

To consider this point more accurately, we need some further definitions:

(6) The *scope* of an occurrence of a quantifier (or a truth-functor) is the shortest formula in which it occurs.

(7) An occurrence of a quantifier $\forall$ or $\exists$, immediately followed by an occurrence of the variable ξ, as in $\forall\xi$ or $\exists\xi$, is said to be ξ-*binding*.[5]

(8) An occurrence of a variable ξ in a formula ϕ is *free in* ϕ iff it is not in the scope of any ξ-binding quantifier in ϕ; otherwise it is *bound in* ϕ.

(9) A *closed* formula is one in which no variable occurs free; a formula which is not closed is *open*.

(10) An occurrence of a quantifier $\forall\xi$ or $\exists\xi$ is *vacuous* iff its scope is $\forall\xi\psi$ or $\exists\xi\psi$, and the variable ξ does not occur free in ψ.

A vacuous quantifier, then, is one which is ξ-binding but which fails to bind any occurrence of ξ, except for the occurrence which forms part of the quantifying prefix itself. All the examples of quantifiers in the apparently 'senseless' formulae just noted are vacuous.

In some books the formation rules are so arranged that formulae with free variables, and formulae with vacuous quantifiers, are not counted as formulae at all. It is easy to see how to achieve this. We restrict formation rule (1) to

(1′) An n-place predicate-letter, followed by n *name-letters*, is a formula,

and we allow the introduction of variables only with rule (4), at the same time as we provide for the quantifiers that bind them. Further, to ensure that these quantifiers should not be vacuous, we now phrase rule (4) in this way:

(4′) If ϕ is a formula containing a name α, and if $\phi(\xi/\alpha)$ is what results upon substituting the variable ξ for all occurrences of α in ϕ, then $\forall\xi\phi(\xi/\alpha)$ and $\exists\xi\phi(\xi/\alpha)$ are formulae, *provided that* $\phi(\xi/\alpha)$ is not *itself* a formula.[6]

[5] I use the small Greek letters 'ξ' and 'ζ' ('xi' and 'zeta', with long 'i' and long 'e') as metalogical symbols to stand in for any variables.

[6] I use the small Greek letters 'α','β','γ' ('alpha', 'beta', 'gamma', with a long 'e' in 'beta') as metalogical symbols to stand in for any name-letters.

Notice that the proviso ensures that at least one occurrence of ξ in $\phi(\xi/\alpha)$ is a *free* occurrence, not already bound by any ξ-binding quantifier in $\phi(\xi/\alpha)$, so that the newly prefixed quantifier cannot be vacuous.[7] So on these revised rules no formula is allowed to contain a free variable or a vacuous quantifier.

But there is a price to pay. From one point of view it is convenient not to count open formulae as formulae, namely because they cannot be assigned truth-values. But on the other hand we cannot just ignore open formulae, since one frequently needs to say things about them, and they must be given some name or other. Since in fact the name 'open formula' (corresponding to 'open sentence') is now well established, it seems perverse not to use it. But then one might as well go along with what the name implies, and accept that open formulae are indeed formulae. When necessary, one can always explicitly restrict attention to such formulae as are not open, but closed. There is no similar motivation for accepting formulae with vacuous quantifiers. Apart from one or two very recherché purposes,[8] these are for the most part just a nuisance, for one has to keep remembering that sensible-looking rules which apply to all normal quantifiers may not apply to them. But in order to rule them out, while still accepting open formulae as formulae, it turns out that the rules of formation must be given in a much more complicated form, which it is not worth stating here.[9] I therefore return to rules (1)–(5) as first stated, which do permit vacuous quantifiers, but I adopt a ruling about them which renders them harmless. The ruling is that a vacuous quantifier alters nothing, i.e. if ξ does not occur free in ϕ, then $\forall\xi\phi$ and $\exists\xi\phi$ are each logically equivalent to ϕ.

Having decided what a language for quantifiers is to be, our next task is to say what counts as an interpretation for such a language. This is the task of the next section. Before we come to that I pause here to introduce a notation for substitution that will be convenient in what follows. If ϕ is any formula, α any name, and ξ any variable, then $\phi(\alpha/\xi)$ is to be the formula that results from ϕ upon substituting occurrences of the name α for every occurrence of the variable ξ in ϕ that is *free* in ϕ. If ξ has no free occurrence in ϕ, then $\phi(\alpha/\xi)$ is just ϕ itself. Similarly, $\phi(\xi/\alpha)$ is to be the formula that results from ϕ upon substituting for each occurrence of α in ϕ an occurrence of ξ that is *free* in ϕ. If either there are no occurrences of α in ϕ, or there is one such that, when an

[7] Observe that we cannot explicitly use the notion of a 'free occurrence' when framing rule (4′), for at this stage that notion has not been defined.

[8] For example, vacuous quantifiers in the logic of one-place predicates are a useful model for 'vacuous' modal operators in the modal logic S5.

[9] One must simultaneously define *both* what counts as a formula *and* what counts as a free occurrence in that formula.

occurrence of ξ is substituted for it, that occurrence is bound in ϕ, then in either case $\phi(\xi/\alpha)$ is just ϕ itself. The point is that this notation concerns the substitution of names for *free* variables, and vice versa; if we wish to consider substitution of or for bound variables (as on p. 103), then we must say so explicitly.

EXERCISES

3.3.1. Let us call a formula 'sensible' iff it contains no vacuous quantifiers. Allowing reasonable conventions for omitting brackets, which of the following expressions are (*a*) formulae, (*b*) closed formulae, (*c*) sensible formulae? Give reasons.

(1) Fax	(6) $\exists x\forall yFyx \wedge \forall x\exists zFzy$
(2) $\forall xFax \rightarrow Fax$	(7) $\forall x(Fab \wedge Ga \rightarrow \exists x(Fxb \wedge Gx))$
(3) $\forall x(Fax \rightarrow \exists xFax)$	(8) $\forall x(Fab \wedge Ga \rightarrow \exists xFxb \wedge Gx)$
(4) $\forall x(\exists yFyy \rightarrow Fxy)$	(9) $\exists x\wedge\exists yFxy$
(5) $\forall a(\forall yFyy \vDash Faa)$	(10) $\forall x(P \rightarrow Fx) \vDash P \rightarrow \forall xFx$

3.3.2. Argue in detail that any expression which is counted as a formula on the revised rules (1′) and (4′) of p. 79 is also counted as a formula which is both closed and sensible on the rules originally given on p. 78. [Method: use induction on the length of the expression, noting that length is now to be measured by the number of occurrences of truth-functors *and quantifiers*.]

3.3.3. Compare rule (4′) with this rule:

> (4″) If ϕ is a formula containing a name-letter α, and if $\phi(\xi/\alpha)$ is what results upon substituting the variable ξ for all occurrences of α in ϕ, then $\forall\xi\phi(\xi/\alpha)$ and $\exists\xi\phi(\xi/\alpha)$ are formulae, *provided that* ξ does not already occur in ϕ.

(Several books adopt (1′) and (4″), in place of (1′) and (4′).) Show that there are expressions which are permitted as formulae by (4′) but not by (4″), and discuss whether they should be permitted.

3.3.4. The *main* logical symbol in a formula is that symbol in it, either truth-functor or quantifier, which has the whole formula as its scope. Prove that in any formula, apart from atomic formulae, there is always one and only one such symbol. [Method: adapt Exercise 2.8.2.]

3.4. Semantics for these Languages

What is to count as an *interpretation* of a language which contains names, predicates, and quantifiers, as well as truth-functors, is very much more complicated than it was when only the truth-functors needed to be considered. There are also, as we shall see, two rather different methods of providing such interpretations. The method that I give first is I think the simpler: it concentrates just on truth-values, as in the previous chapter, and consequently it ignores open formulae altogether. It is this method that I shall use in subsequent discussions. But since most books these days use a different method, I shall give a brief outline of this too.

We may start with what is common to both methods. An interpretation for a language of the kind that we are now interested in will always begin with the selection of a *domain*, to be what is called 'the domain of the interpretation'. (We abbreviate this to $\mathcal{D}$.) It is also called 'the domain of discourse', since it is thought of as containing all the things that the language in question can speak about. What we choose as the domain is arbitrary; it can be any set of things whatever, finite or infinite, with this one proviso: it cannot be *empty*. I comment briefly on some consequences of this proviso in this section and the next, but I shall not examine its merits until Chapter 8. For the present it is simply a stipulation on what is to count as a (standard) interpretation.

The next step is to provide an interpretation, *on that domain*, of the name-letters and predicate-letters in our language. This means that for each name-letter in the language we must specify some object of the domain for it to be the name of. Since one of our central assumptions about names was that a name must succeed in naming something, we are not permitted to interpret any name-letter as lacking in denotation, but otherwise there are no restrictions. For example, we may, if we wish, interpret all the names as naming the *same* object in the domain, or we may interpret them as all naming different objects, and so on. Similarly, we interpret the predicate-letters on the domain by saying which objects in the domain they count as true of. More accurately, we interpret a zero-place predicate-letter, i.e. a sentence-letter, just by assigning it a truth-value, T or F. We interpret a one-place predicate-letter by assigning it some set of objects from the domain which it is true of. (It is then also specified as false of all other objects in the domain.) Any set of objects from the domain is permitted; in particular we may include *all* the objects in the domain, which interprets the predicate-letter as true of everything; or we may interpret it as true of nothing, or anything in between. We interpret a two-place predicate-letter by assigning to it some set of the ordered pairs that can be formed from members of the domain; and so on.

In order to have a brief notation, for any symbol σ let us write $|\sigma|$ for the 'semantic value' assigned to the symbol σ by whatever interpretation is in question. (If we need to distinguish different interpretations, say I and $\mathcal{J}$, we shall add suitable subscripts as in $|\sigma|_I$ and $|\sigma|_{\mathcal{J}}$.) Also, where $\mathcal{D}$ is the domain of the interpretation, let us write $\mathcal{D}^n$ (for $n>0$) for the set of all n-tuples that can be formed from the objects of that domain. Then we can say: an interpretation I for a language $\mathcal{L}$ for quantifiers, will always specify

(1) A non-empty domain $\mathcal{D}$.
(2) For each name-letter α in $\mathcal{L}$ some object in $\mathcal{D}$ as its denotation (i.e. what it names). Thus $|\alpha| \in \mathcal{D}$.
(3) For each zero-place predicate-letter Φ^0 in $\mathcal{L}$ a truth-value.[10] Thus $|\Phi^0| = \text{T}$ or $|\Phi^0| = \text{F}$.[11]
(4) For each n-place predicate-letter Φ^n in $\mathcal{L}$ ($n>0$), a set of n-tuples formed from the objects in $\mathcal{D}$ as its extension (i.e. what it is true of). Thus $|\Phi^n| \subseteq \mathcal{D}^n$.

These clauses (1)–(4) concern the interpretation of the non-logical vocabulary of $\mathcal{L}$, which will be different from one interpretation to another. We now need to consider the interpretation of the logical vocabulary of $\mathcal{L}$, and this is not chosen arbitrarily, but is designed to conform to the intended meaning of the logical signs. It is therefore the same for all interpretations of the same language.

Let us suppose, again, that we are dealing with a language which contains just $\neg,\wedge,\vee$ as truth-functors, and $\forall$ and $\exists$ as quantifiers. There is also another piece of 'logical vocabulary' that needs explanation, and that is the significance of writing name-letters immediately after a predicate-letter in an atomic formula. So we also need a clause which tells us how atomic formulae are to be evaluated. In the first method that I give, we confine attention throughout to *closed* formulae, so that the atomic formulae that are relevant are just those that contain name-letters but no variables. The obvious clause is this:

(5) $|\Phi^n\alpha\beta...\gamma| = \text{T} \quad \text{iff} \quad \langle|\alpha|,|\beta|,...,|\gamma|\rangle \in |\Phi^n|$.[12]

[10] I use the capital Greek letter 'Φ' ('phi') as a metalogical symbol to stand in for any predicate-letter. (I add that '$\in$' abbreviates 'is a member of' and '$\subseteq$' abbreviates 'is a subset of'.)

[11] We *could* avoid a special clause for zero-place predicates in this way. We may suppose that $\mathcal{D}^0$ is the set of 0-tuples that can be formed from members of $\mathcal{D}$, and that there is just one such 0-tuple, namely 'the empty tuple' (i.e. the empty sequence) which is represented by $\langle\ \rangle$. Then $\mathcal{D}^0 = \{\langle\ \rangle\}$, and if $|\Phi^0| \subseteq \mathcal{D}^0$ then either $|\Phi^0| = \{\langle\ \rangle\}$ or $|\Phi^0| = \{\ \}$ (i.e. the empty set). For a continuation of this approach, see n. 12.

[12] If this clause is intended to include zero-place predicate-letters (cf. n. 11), then in their case it is interpreted as meaning

$$|\Phi^0| = \text{T} \quad \text{iff} \quad \langle\ \rangle \in |\Phi^0|.$$

Thus a sentence-letter is true if its value is $\{\langle\ \rangle\}$, and false if its value is $\{\ \}$. (I am indebted to Dana Scott for this ingenious suggestion.)

In words, this says: an atomic formula consisting of an n-place predicate-letter followed by n names counts as true (in the interpretation in question) iff the n-tuple formed by taking the objects which are the denotations of the names (in that interpretation), in the order in which the names occur in the formula, is a member of the set of n-tuples which is the extension of the predicate-letter (in that interpretation). It is long-winded to say it, but the thought is surely very simple. We may add to this the expected clauses for the truth-functors, namely (in the present case)

(6) $|\neg\phi| = \mathrm{T}$ iff $|\phi| \neq \mathrm{T}$

(7)(*a*) $|\phi\wedge\psi| = \mathrm{T}$ iff $|\phi| = \mathrm{T}$ and $|\psi| = \mathrm{T}$

(*b*) $|\phi\vee\psi| = \mathrm{T}$ iff $|\phi| = \mathrm{T}$ or $|\psi| = \mathrm{T}$.

This brings us, then, to the problem of what we are to say about the quantifiers.

Let us suppose that $\forall x\phi(x)$ and $\exists x\phi(x)$ are closed and sensible formulae, so that x, but only x, occurs free in $\phi(x)$. Then the basic idea is evidently this: $\forall x\phi(x)$ is to count as true iff the predicate represented by the open sentence $\phi(x)$ is *true of* all the objects in the domain; and similarly $\exists x\phi(x)$ is to count as true iff that predicate is *true of* some object in the domain. But this way of putting things evidently introduces a difficulty. In clauses (5)–(7) we have been working towards a definition of *true* for our language, but have not said anything about *being true of*. Either, then, we must think of some way of explaining the truth-values of quantified sentences in terms of the *truth* of their simpler instances, or we must go back and revise clauses (5)–(7) so that they are concerned, not—or not only—with truth, but also with being true of. My first method takes the first course, and my second method takes the second.

The first method starts from the thought that if $\forall x\phi(x)$ is true, then so is every simpler formula $\phi(\alpha)$ obtained from it by substituting some name α for the occurrences of x that are free in $\phi(x)$. Provided that the interpretation we are concerned with has assigned a name to each object in the domain, then we can also say conversely that if every formula $\phi(\alpha)$ is true, then so is $\forall x\phi(x)$. But this proviso is not something that we can take for granted. In many cases it is not fulfilled, and in some cases it could not be, since there may be *more* objects in the domain than there are name-letters.[13] The solution to this problem is not to try to ensure an adequate supply of names, but just to think of the many ways of interpreting a single name-letter. The idea

[13] There are as many name-letters as there are natural numbers; Cantor proved that there are more real numbers than there are natural numbers.

is, roughly speaking, that $\forall x\phi(x)$ is true iff $\phi(\alpha)$ is true *for every way of interpreting* the name-letter α. Let us try to put this more precisely.

We are trying to settle the truth-values of quantified sentences in an interpretation I. To do this we need also to consider other interpretations which are *variants* of I. In particular, for any name α, let us say that I_α is an α-variant of I iff I_α does interpret the name α, and it differs from I either not at all or just on the interpretation that it assigns to α and in no other way. This may be because I does not assign any interpretation to α whereas I_α does, or because the two assign different interpretations. In all other ways, however, the two interpretations are to be the same. It should be noted here that, for any name α, and any interpretation I, there always is at least one α-variant interpretation I_α. This would not be true if we had permitted the domain of an interpretation to be empty. For if we have a language which contains no name-letters, then it can be interpreted on an empty domain; truth-values may be assigned arbitrarily to its sentence-letters, and all other predicate-letters are assigned the empty set as their extension. But this is an interpretation which has no α-variant interpretation for any name α. For an α-variant interpretation *does* assign a denotation to the name α, which cannot be done if at the same time the domain has to be kept empty. As things are, however, we are debarring the empty domain, so this problem does not arise.

The idea, then, is to specify the truth-value of a quantified formula $\forall\xi\phi$ in terms of the truth-values of its singular instances $\phi(\alpha/\xi)$, not only in the interpretation I that we began with, but also in variant interpretations which treat the substituted name differently.[14] We must, then, specify that the substituted name should be one that does not *already* occur in $\forall\xi\phi$. For we want to hold all the symbols in $\forall\xi\phi$ to their *existing* interpretation while nevertheless considering other interpretations for the name that is introduced in place of the quantified variable. There is no problem about this, for no single formula can already contain all the name-letters that there are. This leads us, then, to adopt the following clauses for the quantifiers:

(8)(*a*) $|\forall\xi\phi|_I = \mathrm{T}$ iff for every name α not in ϕ, and every α-variant interpretation I_α, $|\phi(\alpha/\xi)|_{I_\alpha} = \mathrm{T}$.

(*b*) $|\exists\xi\phi|_I = \mathrm{T}$ iff for some name α not in ϕ, and some α-variant interpretation I_α, $|\phi(\alpha/\xi)|_{I_\alpha} = \mathrm{T}$.

An alternative formulation, which is quite easily seen to yield the same results, is this:

[14] This method is used in Mates (1972: 60 ff.).

(8′) Let β be the alphabetically earliest name that is not in ϕ. Then

(*a*) $|\forall\xi\phi|_I = \mathrm{T}$ iff for every β-variant interpretation I_β, $|\phi(\beta/\xi)|_{I_\beta} = \mathrm{T}$.

(*b*) $|\exists\xi\phi|_I = \mathrm{T}$ iff for some β-variant interpretation I_β, $|\phi(\beta/\xi)|_{I_\beta} = \mathrm{T}$.

For the fact is that for any two names α and β, neither of which occur in ϕ, the α-variant interpretations of $\phi(\alpha/\xi)$ exactly match the β-variant interpretations of $\phi(\beta/\xi)$, and all of the first will be true (or false) iff all of the second are also. (I give a proof of this claim in the next section, as 3.5.C.)

This completes the account of an interpretation, according to the first method. Clauses (1)–(8) have specified what an interpretation is in a way that ensures that the interpretation assigns a definite truth-value, T or F, to every *closed* formula of the language being interpreted. In this method, open formulae are simply ignored. They cannot significantly be assigned truth-values, and no other kinds of values have been considered for them. (Because it concentrates entirely on truth-values, the method is said to give a *recursion on truth.*[15]) I now pass on to the second method, which does assign values of a kind to open formulae.

As I explained the problem initially it was this. A simple quantification creates a closed formula from an open formula. So apparently the truth-value of the quantification should be determined by the 'value' of the open formula that is quantified. But an open formula simply does not have a *truth*-value. What kind of value, then, does it have?

Well, the suggestion that we shall pursue is basically this: an open formula can be regarded as having a truth-value if at the same time we artificially treat its free variables as if they were names. Of course there will be many ways of so treating the variables, i.e. of assigning them denotations. But if we can specify what value the formula has for *each* possible way of assigning denotations to its free variables, then this can be regarded as assigning a non-arbitrary value to the formula itself. In effect, it assigns to the formula an *extension*, for to speak of those ways of assigning objects to the variables that make the formula true is much the same as to speak of those n-tuples of objects that the formula counts as true of. But it is not quite the same. For our technique will specify extensions in a way which *also* allows us to calculate the extensions of complex formulae from the extensions of their simpler components. A simple illustration will make clear how this works.

[15] It is called a *recursion* because—very roughly—it determines the truth-values of complex formulae by *going back* to the truth-values of their simpler components (or instances).

Let us take a simple two-place predicate-letter F. The interpretation will specify an extension for this letter; let us suppose it is just the pair of objects $\langle a,b \rangle$ and nothing else. Then it is natural to say that this pair $\langle a,b \rangle$ can equally well be regarded as the extension, in this interpretation, of the open formula '*Fxy*'. But it is then equally natural to say that the same pair is the extension of the open formula '*Fyx*'. Considered as open formulae, which may be true of, or satisfied by, pairs of objects, there is surely no significant difference between '*Fxy*' and '*Fyx*'. For it does not matter *which* variables we use to mark the gaps in an open sentence; all that is significant is whether the various gaps are filled by the same variables or different variables. But then, if we are to say that the extensions of '*Fxy*' and '*Fyx*' are the same, we cannot suppose that the extensions of the two conjuncts of a conjunction determine the extension of the conjunction itself. For clearly the conjunctions '$Fxy \wedge Fxy$' and '$Fxy \wedge Fyx$' need not have the same extensions as one another. On the contrary, in the interpretation given, the first has the extension $\langle a,b \rangle$ again, while the second has a null extension (assuming that $a \neq b$). To keep track of this kind of thing, our ways of assigning objects to variables will not lose sight of which variables are involved where. So we shall have one assignment which assigns a to 'x' and b to 'y', and this assignment satisfies '*Fxy*' but not '*Fyx*'. There will also be another assignment which assigns b to 'x' and a to 'y', and this assignment satisfies '*Fyx*' but not '*Fxy*'. But there will be no assignment of objects to variables that satisfies both '*Fxy*' and '*Fyx*', and hence no assignment that satisfies the conjunction '$Fxy \wedge Fyx$'. Let us now put this idea more precisely.

We suppose given an interpretation I, which specifies a domain, and the interpretation on that domain of the name-letters and predicate-letters in our language $\mathcal{L}$. In fact, let us begin with the simplifying assumption that $\mathcal{L}$ contains no name-letters, so that all formulae are built up just from predicate-letters and variables. We now introduce the idea of an assignment s of denotations to the variables of $\mathcal{L}$, i.e. a function which, for each variable ξ of $\mathcal{L}$ as input, yields as output some object $s(\xi)$ of the domain $\mathcal{D}$ of the interpretation I. We shall speak of such an assignment s in I as *satisfying* a formula, meaning by this that the formula comes out true (in the interpretation I) when each variable ξ that occurs free in that formula is taken as denoting $s(\xi)$.[16] We give a recursive definition of satisfaction (abbreviating 'satisfies' to 'sats') which starts in the expected way:

[16] An alternative method, adopted in Tarski's pioneering work of 1931, and still employed in many books, is this. We take the infinitely many variables of $\mathcal{L}$ to be alphabetically ordered, and we consider the infinite sequences (allowing repetitions) of objects from the domain. We then stipulate that the nth variable of the alphabet is *always* to be taken as denoting the nth member of any sequence, and with this

(1) s sats $\Phi^n\xi_1\xi_2...\xi_n$ iff $\langle s(\xi_1),s(\xi_2),...,s(\xi_n)\rangle \in |\Phi^n|$.[17]
(2) s sats $\neg\phi$ iff not (s sats ϕ).
(3) s sats $\phi\wedge\psi$ iff (s sats ϕ) and (s sats ψ).
s sats $\phi\vee\psi$ iff (s sats ϕ) or (s sats ψ).

To state the clause for the quantifiers we now introduce the idea of one assignment being a *variant* on another. In particular, given an assignment s, and a particular variable ξ, an assignment s_ξ will be a ξ-variant on s iff either it is s or it differs from s just in the denotation it assigns to ξ and in no other way. Then the quantifier clauses are

(4) s sats $\forall\xi\phi$ iff for every ξ-variant s_ξ on s, s_ξ sats ϕ
s sats $\exists\xi\phi$ iff for some ξ-variant s_ξ on s, s_ξ sats ϕ.

These clauses specify what it is for any assignment s to satisfy any formula ϕ of $\mathcal{L}$, whether ϕ is closed or open. In fact they ensure that a closed formula is either satisfied by all assignments or by none, so we can now complete our account of the interpretation I by adding: for any closed formula ϕ

$|\phi|_I = \mathrm{T}$ iff for every s in I, s sats ϕ.

Now let us go back to remove the simplification imposed earlier, that the language should contain no name-letters. If we do have name-letters to take into consideration, it turns out that the simplest way of doing so is to enlarge our assignments of denotations to variables so that they also include assignments of denotations to names. But, of course, there will be this very clear difference: within a given interpretation I, *every* assignment s of denotations will assign the *same* denotation to each name-letter α interpreted by I, namely $|\alpha|$; but they will differ from one another by assigning different denotations $s(\xi)$ to the variables ξ. Each assignment s, then, is to be a function from the *terms* of the language (i.e. its name-letters and its variables) to the objects in the domain of the interpretation. To each term τ it assigns a denotation $s(\tau)$ but in such a way that[18]

for a name-letter α, always $s(\alpha) = |\alpha|$;
for a variable ξ, $s(\xi)$ is an arbitrary member of $\mathcal{D}$.

convention we can speak directly of a formula being satisfied by a sequence of objects. (But the sequences in question are infinitely long.) See Tarski (1956, ch. 8), which contains an English translation including corrections.

[17] For the case of zero-place predicate-letters, see nn. 11 and 12 earlier. We have
s sats Φ^0 iff $\langle\ \rangle \in |\Phi^0|$.

[18] I use the small Greek letter 'τ' ('tau') as a metalogical symbol to stand in for any term.

To accommodate this ruling into the overall scheme, we therefore generalize clause (1) above so that it deals with *all* atomic formulae, both those containing names and those containing variables:

(1′) s sats $\Phi^n\tau_1 \ldots \tau_n$ iff $\langle s(\tau_1),\ldots,s(\tau_n)\rangle \in |\Phi^n|$.

The other clauses (and the definition of truth in terms of satisfaction) remain unchanged.

EXERCISES

Throughout these exercises suppose that we are given some interpretation *I* which is specified in the second way, with a recursion on satisfaction, as on pp. 86–9. (This set of exercises is the only part of the book that will work with interpretations specified in this way.) To abbreviate labour, assume that the only logical symbols in the language of *I* are $\neg, \wedge, \forall$.

3.4.1. Let $\phi(\tau_1)$ and $\phi(\tau_2)$ be any formulae which result from one another upon replacing some free occurrences of τ_1 by free occurrences of τ_2, or vice versa. (If τ_i is a name-letter, every occurrence counts vacuously as 'free'.) Let s be an assignment in *I* of denotations to terms in which $s(\tau_1) = s(\tau_2)$. Prove

$$\text{s sats } \phi(\tau_1) \quad \text{iff} \quad \text{s sats } \phi(\tau_2).$$

[Method: use induction on the length of $\phi(\tau_1)$. The inductive hypothesis should be that the result holds for all formulae $\psi(\tau_1)$ shorter than $\phi(\tau_1)$ *and all assignments* s in *I*. It may help to compare the analogous result for a semantics specified in the first way, by a recursion on truth, which is proved in the next section as 3.5.B, but you will find that the present proof is much simpler than that one.]

3.4.2. Prove that, if ϕ is a closed formula, then ϕ is satisfied either by all assignments in I or by none. [Method: induction on the length of ϕ. In the clause for $\forall$ you will need to introduce a new name in place of the quantified variable, so that the inductive hypothesis can be brought to bear, and to use Exercise 3.4.1.]

3.4.3. Let I^* be an interpretation specified in the first way, with a recursion on truth rather than satisfaction, as on pp. 84–6. Suppose that I and I^* have the same domain, and the same interpretation on that domain of all name-letters and predicate-letters in ϕ. Using the result of Exercise 3.4.2 prove that

If ϕ is closed, then $|\phi|_I = |\phi|_{I^*}$.

3.5. Some Lemmas on these Semantics

From now on I shall assume that our semantics is specified in the first way, by a recursion on truth rather than on satisfaction. This seems to me to be the more natural and direct approach. But even so it is quite tricky to work with, so I here insert a short section which illustrates in some detail how this is done. I prove three lemmas which will be put to use in the next section. You will see that the claims to be argued for are very straightforward, but in the second case the argument is quite complex.

The first two arguments will proceed by induction on the length of a formula, and as you will know (unless you have skipped the exercises) the length of a formula is now defined as the number of occurrences of truth-functors *and quantifiers* that it contains. To save labour, I shall therefore assume that we are dealing with a language that contains only $\neg,\wedge,\forall$ as its logical symbols. You will find that further cases can easily be argued in the same way as these. (Or you may rely upon the fact that other truth-functors and quantifiers may be defined in terms of these.)

My first lemma is something which was obvious enough to go without saying in languages for truth-functors, but now deserves a proper treatment:

3.5.A. Lemma on interpretations. If two interpretations I and $\mathcal{J}$ have the same domain, and the same interpretations (on that domain) of all the name-letters and predicate-letters in a (closed) formula ϕ, then they also assign the same value to ϕ.

The proof, as I have said, is by induction on the length of the formula ϕ, which we assume to be a closed formula. So assume also that I and $\mathcal{J}$ are interpretations with the same domain, which assign the same values to all the letters in ϕ. We must show that they assign the same value to ϕ, i.e. that $|\phi|_{\mathcal{I}} = |\phi|_{\mathcal{J}}$. The hypothesis of induction is

> For all closed formulae ψ shorter than ϕ, and all interpretations $\mathcal{K}_1$ and $\mathcal{K}_2$ with the same domain, which assign the same values to all the letters in ψ, $|\psi|_{\mathcal{K}_1} = |\psi|_{\mathcal{K}_2}$.

We have four cases to consider.

Case (1): ϕ is atomic, i.e. it consists of an n-place predicate-letter followed by n name-letters. Since $\mathcal{I}$ and $\mathcal{J}$ assign the same values to all these letters, the result follows at once from the clause for evaluating atomic formulae (i.e. clause (5) on p. 83).

Case (2): ϕ is $\neg\psi$. Since $\mathcal{I}$ and $\mathcal{J}$ assign the same values to all the letters in ϕ, they also assign the same values to all the letters in ψ, which is shorter than ϕ. Hence by the inductive hypothesis we have $|\psi|_{\mathcal{I}} = |\psi|_{\mathcal{J}}$. So the result follows by the clause for evaluating negations (i.e. clause (6) on p. 84).

Case (3): ϕ is $\psi\wedge\chi$. As in case (2), the inductive hypothesis implies $|\psi|_{\mathcal{I}} = |\psi|_{\mathcal{J}}$ and $|\chi|_{\mathcal{I}} = |\chi|_{\mathcal{J}}$, so the result follows by the clause for conjunctions (i.e. clause (7) on p. 84).

Case (4): ϕ is $\forall\xi\psi$. Assume first that $|\forall\xi\psi|_{\mathcal{I}} = \text{F}$. By the clause for quantifications (i.e. clause (8) on p. 85), this means that for some name α not in ψ, and some α-variant interpretation $\mathcal{I}_\alpha$ of $\mathcal{I}$, we have $|\psi(\alpha/\xi)|_{\mathcal{I}_\alpha} = \text{F}$. Now let $\mathcal{K}$ be an interpretation which agrees in all respects with $\mathcal{J}$, except that it interprets the name α as $\mathcal{I}_\alpha$ does. Since $\mathcal{I}$ and $\mathcal{J}$ agree on all letters in $\psi(\alpha/\xi)$, except possibly on α, it follows that $\mathcal{I}_\alpha$ and $\mathcal{K}$ agree on all letters in $\psi(\alpha/\xi)$, and so by the inductive hypothesis we have $|\psi(\alpha/\xi)|_{\mathcal{K}} = \text{F}$. But also, $\mathcal{K}$ is an α-variant on $\mathcal{J}$, and so by the clause for quantifications we also have $|\forall\xi\psi|_{\mathcal{J}} = \text{F}$.

Thus if $|\phi|_{\mathcal{I}} = \text{F}$ then $|\phi|_{\mathcal{J}} = \text{F}$. By an exactly similar argument, if $|\phi|_{\mathcal{J}} = \text{F}$ then $|\phi|_{\mathcal{I}} = \text{F}$. This establishes the desired result $|\phi|_{\mathcal{I}} = |\phi|_{\mathcal{J}}$.

This completes the induction.

My second lemma shows that our semantics does obey the principle of extensionality discussed earlier, in Section 3.1. That is, if α and β are name-letters which are assigned the same denotation in some interpretation $\mathcal{I}$, then either may be substituted for the other, at one or more places in a formula, without changing the value of that formula in $\mathcal{I}$. To state this more concisely, let $\phi(\alpha)$ be any closed formula containing the name α, and let $\phi(\beta)$

result from it upon replacing some occurrences of α in $\phi(\alpha)$ by occurrences of β.[20] Then the lemma is

3.5.B. Lemma on extensionality. If $|\alpha|_I = |\beta|_I$ then $|\phi(\alpha)|_I = |\phi(\beta)|_I$.

The proof is by induction on the length of the formula $\phi(\alpha)$. The inductive hypothesis is

> For all formulae $\psi(\alpha)$ shorter than $\phi(\alpha)$, for all name-letters α and β, and for all interpretations $\mathcal{J}$: if $|\alpha|_{\mathcal{J}} = |\beta|_{\mathcal{J}}$, then $|\psi(\alpha)|_{\mathcal{J}} = |\psi(\beta)|_{\mathcal{J}}$.

We have four cases to consider, but the argument for the last case has to be somewhat roundabout. We assume $|\alpha|_I = |\beta|_I$.

Case (1): $\phi(\alpha)$ is atomic; it might be for example $\Phi^3\alpha\beta\gamma$. Then $\phi(\beta)$ is $\Phi^3\beta\beta\gamma$. Since we are given that $|\alpha|_I = |\beta|_I$, it evidently follows that

$$\langle|\alpha|_I,|\beta|_I,|\gamma|_I\rangle = \langle|\beta|_I,|\beta|_I,|\gamma|_I\rangle.$$

Applying the clause for atomic formulae, it at once follows that $|\Phi^3\alpha\beta\gamma|_I = |\Phi^3\beta\beta\gamma|_I$. The same reasoning evidently applies to any atomic formula, so we may conclude that $|\phi(\alpha)|_I = |\phi(\beta)|_I$ as desired.

Case (2): $\phi(\alpha)$ is $\neg\psi(\alpha)$. By inductive hypothesis we have $|\psi(\alpha)|_I = |\psi(\beta)|_I$, and so the result follows by the clause for negations.

Case (3): $\phi(\alpha)$ is $\psi(\alpha) \wedge \chi(\alpha)$, where α may perhaps be present in only one of the conjuncts. Then by inductive hypothesis we have $|\psi(\alpha)|_I = |\psi(\beta)|_I$ and $|\chi(\alpha)|_I = |\chi(\beta)|_I$, where again β may perhaps be present in only one of these formulae. This makes no difference to the argument, for in either case the result follows by the clause for conjunctions.

Case (4): $\phi(\alpha)$ is $\forall\xi\psi(\alpha)$. Assume $|\forall\xi\psi(\alpha)|_I = F$. That is to say: there is some name γ, which is not in $\psi(\alpha)$, and some γ-variant interpretation I_γ of I, such that $|\psi(\alpha)(\gamma/\xi)|_{I_\gamma} = F$. We have two cases to consider, according as the name γ is or is not in $\psi(\beta)$. Suppose first that it is. In that case γ can only be β itself (since it is in $\psi(\beta)$ but not in $\psi(\alpha)$). It follows that β is not in $\psi(\alpha)$, and what we are given is that there is a β-variant interpretation I_β of I such that

(*a*) $|\psi(\alpha)(\beta/\xi)|_{I_\beta} = F$.

We shall show that in that case there is *also* a suitable γ-variant, using a new name γ other than β, so that this case reduces to the other case first distinguished.

[20] Thus $\phi(\alpha)$ is $\phi'(\alpha/\xi)$ and $\phi(\beta)$ is $\phi'(\beta/\xi)$ for some formula ϕ' which contains free occurrences of ξ just where α and β are to be interchanged.

Consider a new name γ, not in $\psi(\alpha)$ or $\psi(\beta)$, and a new interpretation $I_{\beta\gamma}$ which agrees with I_β on the interpretation of all the symbols in the language of I_β, but which also interprets the name γ so that $|\gamma|_{I_{\beta\gamma}} = |\beta|_{I_\beta}$, and hence $|\gamma|_{I_{\beta\gamma}} = |\beta|_{I_{\beta\gamma}}$. Now $I_{\beta\gamma}$ and I_β agree on the interpretation of all the symbols in $\psi(\alpha)(\beta/\xi)$. Hence by lemma 3.5.A on interpretations we have

(*b*) $|\psi(\alpha)(\beta/\xi)|_{I_\beta} = |\psi(\alpha)(\beta/\xi)|_{I_{\beta\gamma}}$.

And by the inductive hypothesis we have

(*c*) $|\psi(\alpha)(\beta/\xi)|_{I_{\beta\gamma}} = |\psi(\alpha)(\gamma/\xi)|_{I_{\beta\gamma}}$.

Further, let I_γ be an interpretation which is just like $I_{\beta\gamma}$ except that in I_γ the name β is interpreted as in I and not as in I_β. Now we are given that β does not occur in $\psi(\alpha)$, nor therefore in $\psi(\alpha)(\gamma/\xi)$. Hence I_γ and $I_{\beta\gamma}$ agree on all the symbols in that formula, and so by the lemma on interpretations we have

(*d*) $|\psi(\alpha)(\gamma/\xi)|_{I_{\beta\gamma}} = |\psi(\alpha)(\gamma/\xi)|_{I_\gamma}$.

Thus I_γ is the required interpretation. For by construction it is a γ-variant on I, and if we put together the four equations (*a*)–(*d*), we at once have

(*e*) $|\psi(\alpha)(\gamma/\xi)|_{I_\gamma} = \text{F}$.

It is now easy to complete the proof. In either of the cases first distinguished, equation (*e*) holds for some name γ not in $\phi(\beta)$. And since I_γ agrees with I on the denotations of α and β we have $|\alpha|_{I_\gamma} = |\beta|_{I_\gamma}$, and so by inductive hypothesis

(*f*) $|\psi(\alpha)(\gamma/\xi)|_{I_\gamma} = |\psi(\beta)(\gamma/\xi)|_{I_\gamma}$.

Putting (*e*) and (*f*) together we have: there is a name γ not occurring in $\phi(\beta)$, and a γ-variant interpretation I_γ of I, such that

(*g*) $|\psi(\beta)(\gamma/\xi)|_{I_\gamma} = \text{F}$.

But in view of the clause for the universal quantifier, that is to say

$$|\forall\xi\psi(\beta)|_I = \text{F}.$$

Conditionalizing this whole argument so far, what we have shown is

$$\text{If } |\phi(\alpha)|_I = \text{F} \text{ then } |\phi(\beta)|_I = \text{F}.$$

Interchanging α and β throughout the argument, we equally have

$$\text{If } |\phi(\beta)|_I = \text{F} \text{ then } |\phi(\alpha)|_I = \text{F}.$$

And therefore finally

$$|\phi(\alpha)|_I = |\phi(\beta)|_I.$$

This completes the induction.

The awkward part of this last argument was to show that if we are given some result with one new name in place of a quantified variable, then we can always obtain the same result with any other new name instead. It is worth recording this point explicitly. Perhaps the most useful way of doing so is by noting that it allows us to give an alternative statement of the truth-conditions for quantified formulae, in terms of their singular instances:[21]

3.5.C. Lemma on alternative semantics for the quantifiers.
$|\forall\xi\phi|_I = \mathrm{T}$ iff, for *some* name-letter α not in ϕ, for all α-variants I_α of I, $|\phi(\alpha/\xi)|_{I_\alpha} = \mathrm{T}$.
$|\exists\xi\phi|_I = \mathrm{T}$ iff, for *every* name-letter α not in ϕ, for some α-variant I_α of I, $|\phi(\alpha/\xi)|_{I_\alpha} = \mathrm{T}$.

I shall show that these semantics for $\forall$ are equivalent to those first given (in clause (8) on p. 85), leaving the application to $\exists$ as an exercise. Now it is obvious that if $\forall\xi\phi$ is true according to the original semantics, then it is also true according to these alternative semantics, since it is obvious that what holds for all name-letters not in ϕ must also hold for some. (This depends upon the point that there must be some name-letters that are not in ϕ, whatever formula ϕ may be.) Suppose, then, that $\forall\xi\phi$ is false in some interpretation I according to the original semantics. That is to say: suppose that there is some name-letter, say β, not occurring in ϕ, and some β-variant interpretation I_β of I, such that

(*a*) $|\phi(\beta/\xi)|_{I_\beta} = \mathrm{F}$.

We have to show that $\forall\xi\phi$ is also false according to the alternative semantics, i.e. that for *any* name-letter α not in ϕ there is an α-variant interpretation I_α such that $|\phi(\alpha/\xi)|_{I_\alpha} = \mathrm{F}$.

Let α be any name not in ϕ other than β. Let $I_{\beta\alpha}$ be an interpretation exactly the same as I_β, except that it assigns to α the same denotation as I_β assigns to β. Now since α is not in ϕ, it is not in $\phi(\beta/\xi)$ either, so I_β and $I_{\beta\alpha}$ agree on all the symbols in $\phi(\beta/\xi)$. Hence by lemma 3.5.A on interpretations we have

(*b*) $|\phi(\beta/\xi)|_{I_\beta} = |\phi(\beta/\xi)|_{I_{\beta\alpha}}$.

Further, since α and β are assigned the same denotation in $I_{\beta\alpha}$, by lemma 3.5.B. on extensionality we have

[21] Compare the further alternative given on p. 86. (It is clear that the present argument also establishes the correctness of that alternative.)

(*c*) $|\phi(\beta/\xi)|_{I_{\beta\alpha}} = |\phi(\alpha/\xi)|_{I_{\beta\alpha}}$.

Finally, let I_α be an interpretation exactly the same as $I_{\beta\alpha}$, except that it assigns to β whatever denotation (if any) it was assigned in I. Then since β is not in ϕ, it is not in $\phi(\alpha/\xi)$ either, so I_α and $I_{\beta\alpha}$ agree on all the symbols in $\phi(\alpha/\xi)$. Hence by the lemma on interpretations we have

(*d*) $|\phi(\alpha/\xi)|_{I_{\beta\alpha}} = |\phi(\alpha/\xi)|_{I_\alpha}$.

Moreover, I_α is by construction an α-variant of I. So putting (*a*)–(*d*) together, I_α is an α-variant of I such that

(*e*) $|\phi(\alpha/\xi)|_{I_\alpha} = \text{F}$.

This completes the proof.

We are now ready to move on to some principles of entailment for our languages with quantifiers which extend those given in Section 2.5 for languages with truth-functors.

EXERCISES

3.5.1. Extend the argument for 3.5.A, by adding new clauses to the induction, so that the result is proved also for languages containing $\vee$ and $\exists$.

3.5.2. The argument for clause (4) of the induction proving 3.5.B establishes that

$$|\forall\xi\psi(\alpha)|_I = |\forall\xi\psi(\beta)|_{I'}.$$

Use the equivalence noted below as 3.6.E to show how that argument also establishes the result for $\exists$ in place of $\forall$.

3.5.3. In a similar way, extend the argument given for 3.5.C to cover $\exists$ as well as $\forall$.