Contents

0 What is Logic? 1
- 0.1 Arguments . . . 3
- 0.2 Sentences and Propositions . . . 4
- 0.3 Logical Consequence . . . 6
- 0.4 Logical Form . . . 9
- 0.5 Other Logical Notions . . . 11

1 Propositional Logic 15
- 1.1 Sentence Letters . . . 16
- 1.2 The Sentential Operators . . . 17
- 1.3 Negation . . . 19
- 1.4 Conjunction . . . 21
- 1.5 Disjunction . . . 24
- 1.6 The Material Conditional . . . 25
- 1.7 The Material Biconditional . . . 28
- 1.8 Unless . . . 30
- 1.9 Well-Formed Sentences . . . 30
- 1.10 Metalinguistic Abbreviation . . . 37

2 Logical Consequence 43
- 2.1 Truth-Functional Connectives . . . 43
- 2.2 Complete Truth Tables . . . 44
- 2.3 Using Truth Tables . . . 47
- 2.4 Partial Truth Tables . . . 52
- 2.5 Evaluating English Arguments in SL . . . 54
- 2.6 Satisfaction . . . 57
- 2.7 Entailment . . . 57
- 2.8 Tautologies . . . 58
- 2.9 Weakening . . . 59
- 2.10 Unsatisfiable . . . 60
- 2.11 Consistency . . . 61

3 Natural Deduction Proofs in SL 62
- 3.1 Premises and Assumptions . . . 63
- 3.2 Reiteration . . . 65
- 3.3 Conjunction . . . 66

3.4 Disjunction . . . 68
3.5 Conditional Introduction . . . 71
3.6 Conditional Elimination . . . 73
3.7 The Biconditional . . . 75
3.8 Negation . . . 76
3.9 Subproofs . . . 79
3.10 Proof Strategy . . . 82
3.11 Proofs and Provability . . . 83
3.12 Provability and Entailment . . . 84
3.13 Logical Analysis . . . 85
3.14 Derived Rules . . . 85
3.15 Schemata . . . 92

4 The Soundness of QD **93**
4.1 Soundness . . . 94
4.2 SD Rules . . . 95
4.3 Substitution and Model Lemmas . . . 105
4.4 QD Rules . . . 108
4.5 Conclusion . . . 114

5 The Completeness of QD **115**
5.1 Introduction . . . 115
5.2 Extensions . . . 117
5.3 Henkin Model . . . 122
5.4 Satisfiability . . . 128
5.5 Compactness . . . 130

6 Midterm Review **131**

7 Quantifier Logic **132**
7.1 The Expressive Limitations of SL . . . 132
7.2 Primitive Expressions in QL . . . 134
7.3 The Well-Formed Formulas of QL . . . 142
7.4 Quantifier Scope . . . 144
7.5 Regimentation in QL . . . 146
7.6 Paraphrasing Pronouns . . . 148
7.7 Ambiguous Predicates . . . 149
7.8 Multiple Quantifiers . . . 151

8 A Semantics for QL **154**
8.1 Predicate Extensions . . . 154
8.2 QL Models . . . 157
8.3 Variable Assignments . . . 158
8.4 Semantics for QL . . . 159
8.5 Satisfaction and Entailment . . . 161

8.6 Minimal Models . . . 162
8.7 Reasoning About all Models . . . 166
8.8 Constants and Quantifiers . . . 169
8.9 Particular Models . . . 171
8.10 Conclusion . . . 172

9 Identity **173**
9.1 Identity and Logic . . . 174
9.2 The Syntax for QL$^=$ . . . 178
9.3 The Semantics for QL$^=$ . . . 180
9.4 Uniqueness . . . 182
9.5 Definite Descriptions . . . 183
9.6 Quantities . . . 185
9.7 Leibniz's Law . . . 188

10 Natural Deduction in QL$^=$ **190**
10.1 Substitution Instances . . . 190
10.2 Universal Elimination . . . 192
10.3 Existential Introduction . . . 192
10.4 Universal Introduction . . . 194
10.5 Existential Elimination . . . 196
10.6 Quantifier Exchange Rules . . . 198
10.7 Identity . . . 201
10.8 Proofs and Provability in QL$^=$ . . . 203
10.9 Soundness and Completeness of QD . . . 203

11 The Soundness of QD **205**
11.1 Soundness . . . 206
11.2 SD Rules . . . 207
11.3 Substitution and Model Lemmas . . . 217
11.4 QD Rules . . . 220
11.5 Conclusion . . . 226

12 The Completeness of QD **227**
12.1 Introduction . . . 227
12.2 Extensions . . . 229
12.3 Henkin Model . . . 234
12.4 Satisfiability . . . 240
12.5 Compactness . . . 242

A Other Symbolic Notation **243**

B Quick Reference **246**

Chapter 1

Propositional Logic

This chapter introduces an artificial language $\mathcal{L}^{\text{PL}}$ for *Propositional Logic*. The basic units of this language are complete sentences which, given an interpretation, express propositions. Since we will only be concerned with whether a proposition is obtains or does not, interpreting $\mathcal{L}^{\text{PL}}$ will amount to assigning the sentences of $\mathcal{L}^{\text{PL}}$ to either truth or falsity which we will represent by '1' and '0'. How this goes will be the topic of the next chapter, focusing for the time being on the construction of the sentences of $\mathcal{L}^{\text{PL}}$ and translating English sentences and arguments into $\mathcal{L}^{\text{PL}}$. This brings us to our first definition:

> A REGIMENTATION of an English sentence in $\mathcal{L}^{\text{PL}}$ is any sentence in $\mathcal{L}^{\text{PL}}$ which captures (some amount of) the logical form of that English sentence.

This definition is vague, and necessarily so. As we will see, there will typically be more than one way to regiment a sentence in English, and different regimentations may capture more or less of the English sentence's logical form. Rather than admitting a mathematical definition, regimentation is like any translation an imprecise matter where some regimentations are better than others, and others may be on a par with each other. We may then say:

> A REGIMENTATION of an argument in English is an argument in $\mathcal{L}^{\text{PL}}$ whose sentences regiment the sentences of the argument in English.

Recall from before that an argument in English is a sequence of declarative sentences. We will see a number of examples of arguments and their regimentations throughout this chapter. However vague, it is good to have the definitions above in mind as you read, considering other ways that you might regiment the sentences and arguments that we consider.

1.1 Sentence Letters

In $\mathcal{L}^{\text{PL}}$, the capital Roman letters 'A', 'B', 'C', ... with or without natural numbers for subscripts are the SENTENCE LETTERS of $\mathcal{L}^{\text{PL}}$. These are the basic building blocks from which complex sentences will be constructed. Since a sentence letter could regiment any English sentence, it is important to provide a SYMBOLIZATION KEY which specifies which sentence letters represent which English sentences. For example, consider this argument:

A1. Today is New Year's Day.
A2. If today is New Year's Day, then people are swimming in English Bay.
A3. People are swimming in English Bay.

This is a valid argument in English. In regimenting it, we want to preserve the logical form of the argument which makes it valid. What happens if we replace each sentence with a letter? Our symbolization key would look like this:

A: Today is New Year's Day.
B: If today is New Year's Day, then people are swimming in English Bay.
C: People are swimming in English Bay.

We could then regiment the argument in this way:

B1. A
B2. B
B3. C

This is a regimentation of the argument, but it's not a very interesting one. In particular, our regimentation does not encode any logical connection between A1, A2, and A3. What was compelling about the original argument has been lost in translation. After all, 'A', 'B', and 'C' could be any sentences whatsoever. Just because 'A' and 'B' are true (on a given interpretation), it does not follow that 'C' is also true (on that interpretation).[1]

The symbolization key provided above is by no means the only symbolization key that we could have provided. It is important to observe that A2 is not just *any* sentence. Rather, A2 contains the A1 and A3 *as parts*. Thus, our symbolization key for the New Year's argument only needs to include the following sentences since we can build A2 from just these pieces. Consider the following alternative to the symbolization key given above:

[1] We will provide a formal definition of validity for $\mathcal{L}^{\text{PL}}$ in Chapters 2 and 8.

T: Today is New Year's Day.
S: People are swimming in English Bay.

Although it is often convenient to use letters corresponding to the sentences' subject matter as in the example above, no such requirement is built into the rules of $\mathcal{L}^{\text{PL}}$. We may now use the key given above to provide a more interesting regimentation of the argument:

C1. T
C2. If T, then S.
C3. S

By making use of the English expression 'If... then...', we have managed to preserve enough of the logical structure of the argument in English to provide a valid regimentation. For our formal language, we ultimately want to replace all of the English expressions with logical notation, but this is a good start.

The English sentences that can only be regimented in $\mathcal{L}^{\text{PL}}$ by sentence letters are called ATOMIC SENTENCES. As we will see in later chapters, the internal structure of atomic sentences may be encoded in a formal language which includes predicates and singular terms. However, for the time being, we do not have these expressive resources at our disposal. Instead, atomic sentences are the smallest logical joints at which we may carve while regimenting English sentences in $\mathcal{L}^{\text{PL}}$. Accordingly, the internal structure that an English sentence might have (i.e., its sub-sentential logical form) is lost when regimented by a sentence letter. From the point of view of $\mathcal{L}^{\text{PL}}$, the sentence letters are as basic as it gets. Although the sentence letters can be used to build up more complex sentences, they cannot be taken apart.

1.2 The Sentential Operators

Sentential operators are used to build complex sentences from sentence letters. Here are five common sentential operators which we will be able to express in $\mathcal{L}^{\text{PL}}$:

symbol	what it is called	rough translation
$\neg$	negation	'It is not the case that...'
$\wedge$	conjunction	'Both... and ...'
$\vee$	disjunction	'Either... or ... (or both)'
$\rightarrow$	conditional	'If ... then ...'
$\leftrightarrow$	biconditional	'... if and only if ...'

Natural languages like English are vague and imprecise, and carry many complex subtleties of meaning. Providing a descriptive theory of these complexities belongs to linguistics, not logic. In contrast to English, our formal language $\mathcal{L}^{\text{PL}}$ will be clear and precise, defined by explicit

rules that hold without exception. This precision and universality has many advantages, but also comes at a cost: our language is artificial insofar as it's conventions are entirely stipulated, and who is to say which stipulations are the right ones to make?

The question of which logic to accept for which applications is a deep and controversial issue within philosophical logic. Rather than attempting to settle that question here, it will be enough for our purposes here to appeal to one method by which we may evaluate competing logics: abduction. By contrast to inductive arguments, or the deductive arguments with which we will primarily be concerned, abductive arguments draw support from the results that a theory yields. The reason that classical logic (i.e., the propositional and first-order logics that we will be considering) holds the majority among logicians and philosophers is due to its strength and simplicity, making it of great utility for a wide range of applications.

To take just one example, mathematics is almost entirely conducted in a first-order theory. For instance, set theory may be articulated with the expressive resources that we will provide. Nevertheless, the logics that we will consider also have their limits. For instance, the modal logics that you would learn about in an intermediate logic course have also been shown to have powerful applications within linguistics, computer science, and philosophy, and may be naturally combined with the logics with which we will be concerned. Rather than any kind of stopping point, the logics covered in this course make for a natural place to begin.

Despite the advantages afforded by the classical logics we will be studying, these logics will be rather artificial by comparison to the informal patterns of reasoning in English with which you are already familiar. Consequently, the "translations" provided by the table above are only approximate. We'll see some of the differences come out below.

It is also important to mention that although the conventions we will use here are common, there are other conventions used elsewhere. Here is a table with some alternatives that you might come across elsewhere (and should avoid using here):

symbols used here	symbols used elsewhere
$\neg$	$\sim$
$\wedge$	$\&$
$\vee$	$\mid$
$\rightarrow$	$\supset$
$\leftrightarrow$	$\equiv$

Although these symbols would do just as well, they are not quite as common in modern texts or else have come to take on other meanings. They can also be somewhat harder to write on the blackboard with the exception of $\sim$ which we will use in class in order to ease exposition.

It is also worth noting that just because the same symbol that we will use has been used elsewhere does not mean that it picks out the same thing. In general, formal texts like this define their own conventions and you should assume the same for other texts. Nevertheless, the conventions used here are extremely common.

1.3 Negation

Consider how we might regiment these sentences:

D1. Logic is hard.
D2. It is false that logic is hard.
D3. Logic isn't hard.

In order to regiment sentence D1, we will need one sentence letter as below:

H: Logic is hard.

Since sentence D2 is obviously related to sentence D1, we do not want to introduce a different sentence letter since this would obscure their logical relationship. To put it partly in English, D2 may be partially regimented as 'It is not the case that H.' In order to regiment D2 along these lines, we will use the symbol '$\neg$' for negation. Thus we may regiment D2 as '$\neg H$'. A sentence of this type— one that begins with a '$\neg$' symbol— is called a NEGATION. The sentence it negates— in this case 'H'— is called the NEGAND.

What of D3? It says that logic isn't hard, which is just another way of negating D3. Accordingly, we can regiment D3 in the same way that we regimented D3 with '$\neg H$'.

When regimenting English sentences in $\mathcal{L}^{\text{PL}}$, the word 'not' is usually a pretty good clue that '$\neg$' will be an appropriate symbol to use. But it's important to think about the actual meaning of the sentence, and not rely too much on which words appear in it.

> NEGATION TEST: For any sentence φ, a sentence can be regimented by $\neg\varphi$ if it can be paraphrased in English as 'It is not the case that φ'.

Consider the following examples:

E1. Rodrigo is mortal.
E2. Rodrigo is immortal.
E3. Rodrigo is not immortal.

Suppose we let 'R' regiment E1. What about sentence E2? Since being immortal is the same as not being mortal, we may take E2 to be the negation of E1, regimenting it with '$\neg R$'.

Sentence E3 can be paraphrased as 'It is not the case that Rodrigo is immortal'. Using negation twice, we may regiment E3 by '$\neg\neg R$'. The two negations in a row each work as

negations, so the sentence means 'It is not the case that, it is not the case that R'. It is the negation of the negation of 'R'. One can negate any sentence of $\mathcal{L}^{\text{PL}}$— even a negation— by putting the '$\neg$' symbol in front of it. In the case of '$\neg\neg R$', this is a negation whose negand is '$\neg R$', which in turn is a negation whose negand is 'R'.

But sometimes things are not quite a simple as we might initially expect. Here is an example that illustrates some of the complexities to look out for:

F1. Elliott is happy.
F2. Elliott is unhappy.

Suppose we take 'H' to F1. We might be tempted to regiment sentence F2 by '$\neg H$'. But is being unhappy the same thing as not being happy? Here the answer is, 'No'. For instance, Elliott might simply be meh. If you find out that someone is not happy, you cannot infer that they are unhappy (though in some cases this inference might well make sense). Hence, we shouldn't treat F2 as the negation of F1. So long as we are allowing 'unhappy' to mean something besides 'not happy', then we need to use a new sentence letter to regiment F2.

Although we will have more to say about this in the following chapter, it is important to provide a preliminary sense of when a negation is true. In particular, for any sentence φ, if φ is true, then $\neg\varphi$ is false. Similarly, if φ is false, then $\neg\varphi$ is true. Using '1' in place of 'true' and '0' in place of 'false', we can summarize this in the following TRUTH TABLE for negation:

φ	$\neg\varphi$
1	0
0	1

The left column shows the possible truth-values for the negand and the right column shows the corresponding truth-value of the negation. Accordingly, the truth table given above specifies the *truth-conditions* for '$\neg$', i.e., the conditions under which a negated sentence is true (similarly false). By using numerals, we avoid any clash with our sentence letters.[2]

Since 1 and 0, are the only possible values that we will consider in this course, the truth table above defines a function from the truth-value of the negand to the truth-value of the negation. We will refer to such functions from truth-values to truth-values as TRUTH-FUNCTIONS. You can think of these as providing a meaning for the logical terms that we will use where these meanings are held fixed across all interpretations of our language. It is for this reason that the sentential operators are also sometimes called LOGICAL CONSTANTS. We will have much more to say about all of this in the following chapter. For now we may continue to introduce the remaining sentential operators that we will include in the language $\mathcal{L}^{\text{PL}}$.

[2]Note that *Carnap* will not have this virtue: truth tables will use 'T' for *true* and 'F' for *false*.

1.4 Conjunction

Consider the following sentences:

G1. Jessica is strong.
G2. Luke is strong.
G3. Jessica is strong and Luke is also strong.

At the very least, we will need separate sentence letters for G1 and G2:

J: Jessica is strong.
L: Luke is strong.

Sentence G3 can be paraphrased as 'J and L'. In order to fully regiment this sentence, we will use the CONJUNCTION symbol '$\wedge$' for 'and'. We may then take '$J \wedge L$' to regiment G3. Given any conjunction, the sentences to the left and right of '$\wedge$' are referred to as CONJUNCTS. In the case of G3, both 'J' and 'L' are conjuncts.

Notice that we make no attempt to provide a distinct symbol for 'also' as it occurs in sentence G3. Words like 'both' and 'also' function to draw our attention to the fact that two sentences are being conjoined but are not doing any further logical work. Thus we do not need to represent 'both' and 'also' in $\mathcal{L}^{\text{PL}}$. Note that Sentence G3 would have meant the same thing had it simply said 'Jessica is strong and Luke is strong'.

Here are some more examples:

H1. Jessica is strong and grumpy.
H2. Jessica and Matt are strong.
H3. Although Luke is strong, he is not grumpy.
H4. Matt is strong, but Jessica is stronger than Matt.

Sentence H1 is a conjunction. Since the sentence says two things about Jessica, it is natural to use her name only once. It might be tempting to try this when regimenting H1. Letting 'J' regiment 'Jessica is strong', one might attempt to begin by paraphrasing H1 as 'J and grumpy', but this would be a mistake. After all, 'J' is just a sentence letter and $\mathcal{L}^{\text{PL}}$ doesn't keep track of the fact that it was intended to be about Jessica. Moreover, 'grumpy' is not a sentence, and so on its own it is neither true nor false. Instead, we may paraphrase sentence H2 as the conjunction '$J \wedge G_1$' where 'G_1' regiments 'Jessica is grumpy'. More generally:

> CONJUNCTION TEST: A sentence can be regimented by $(\varphi \wedge \psi)$ if it can be paraphrased in English as 'Both φ and ψ' where each conjunct is a sentence.

Sentence H2 says one thing about two different subjects. It says of both Jessica and Matt that they are strong, and in English we use the word 'strong' only once. In regimenting sentences in $\mathcal{L}^{\text{PL}}$, we want to make sure each conjunct is a sentence on its own, and so we may paraphrase H2 by repeating the predicate: 'Jessica is strong and Matt is strong.' Once we add a new sentence letter 'M' for 'Matt is strong', we may take '$J \wedge M$' to regiment H2.[3]

Sentence H3 is a bit more complicated. The word 'although' tends to suggest a kind of contrast between the first part of the sentence and the second part. Nevertheless, the sentence is still telling us two things: Luke is strong and he is not grumpy. So we can paraphrase sentence H3 as, 'Luke is strong and Luke is not grumpy.' The second conjunct contains a negation, so we may paraphrase further: 'Luke is strong and it is not the case that Luke is grumpy'. Letting 'G_2' regiment 'Luke is grumpy', we can take '$L \wedge \neg G_2$' to regiment H3.

Once again, this is an imperfect translation of the English sentence H3. Whereas H3 suggests that there is a contrast between Luke begin strong and not grumpy, our regimentation merely says that Luke is strong and not grumpy. Nevertheless, our regimentation preserves some of the important features of the original sentence, specifically the logical features of that sentence. That is, the sentences says that Luke is strong and that Luke is not grumpy.

The word 'but' in H4 indicates a similar contrast between its conjuncts. Since contrasts like this are irrelevant for the purpose of regimenting sentences in $\mathcal{L}^{\text{PL}}$, we can paraphrase the sentence as 'Matt is strong and Jessica is stronger than Matt. It remains to say how to regiment the second conjunct. We already have the sentence letters 'J' and 'M', but neither of these says anything comparative. Thus we need an entirely new sentence letter. Letting 'S' regiment 'Jessica is stronger than Matt', we may take '$M \wedge S$' to regiment H4.[4]

Here is the resulting symbolization key:

J: Jessica is strong.
G_1: Jessica is grumpy.
M: Matt is strong.
G_2: Luke is grumpy.
S: Jessica is stronger than Matt.

It is important to keep in mind that the sentence letters 'J', 'G_1', 'M', 'G_2', and 'S' have no meaning beyond their truth-values. We used 'J' and 'M' to regiment different English sentences that are about people being strong, but this similarity is completely lost when we regiment sentences in $\mathcal{L}^{\text{PL}}$. Nor does $\mathcal{L}^{\text{PL}}$ recognize any similarity between 'G_1' and 'G_2'. Such connections will be preserved in $\mathcal{L}^{\text{FOL}}$, but for the time being we will accept these limitations.

For any two sentences φ and ψ of $\mathcal{L}^{\text{PL}}$, the conjunction $\varphi \wedge \psi$ is true if and only if both of its conjuncts— φ and ψ— are true. We can summarize this in the truth table for conjunction:

[3] One might contest this claim by instead taking 'Jessica and Matt' to name a plurality where 'are strong' is a plural predicate. We will be overlooking such complexities which are better handled in plural logics.

[4] In chapter 7, we will learn an even better way to regiment relations.

φ	ψ	$\varphi \wedge \psi$
1	1	1
1	0	0
0	1	0
0	0	0

The two left columns indicate the truth-values of the conjuncts. Since there are four possible combinations of truth-values, there are four rows. The conjunction is true when both conjuncts are true, and false otherwise. Whereas negation takes one truth-value as input, and returns one truth-value as output, conjunction takes two truth-values as inputs (the truth-values of its conjuncts) and returns a single truth-value as an output (the truth-value of the conjunction). Put otherwise, conjunction is a *binary operator* and negation is a *unary operator*. Despite this difference, the truth table given above specifies the truth-conditions for conjunction in a similar manner to the way we specified truth-conditions for negation.

Note that conjunction is commutative: $\varphi \wedge \psi$ always has the same truth-value as $\psi \wedge \varphi$. We may then ask: is 'and' in English commutative? Consider the following claims:

I1. Dan went home and took a shower.
I2. Dan took a shower and went home.

Do these sentences say the same thing? Not really. Often the order of the conjuncts suggests the temporal order of the events. Accordingly, we may take I1 to claim that *first* Dan went home, and only *then* did he take a shower, whereas I2 says that these events happened in the opposite order. But what of our truth table for conjunction?

Although $\mathcal{L}^{\text{PL}}$ helps to identify certain logical features in English, it cannot recover everything that we might want to recover. This doesn't mean that one cannot provide a non-commutative theory of conjunction, but we won't be providing such a theory in this course. Rather we stipulate that conjunction for the purposes of this course is commutative where the truth table for conjunction encodes this stipulation. Put otherwise, you can think of the type of conjunction we will be concerned with as an abstraction from the complexities that conjunction in English may include. This doesn't make the commutative theory of conjunction less interesting than a non-commutative theory. After all, simplicity is a good thing, providing for broad applications where a non-commutative notion of conjunction may get in the way.

It is worth considering an example to make these claims more concrete. For instance, consider mathematical claims. Assuming that all claims in pure mathematics are eternal— they are true at all times if they are true at any time— we do not need to keep track of temporal order when engaging in mathematical reasoning. Moreover, even if one were concerned to encode temporal order, getting to grips with $\mathcal{L}^{\text{PL}}$ and $\mathcal{L}^{\text{FOL}}$ will provide an important foundation.

1.5 Disjunction

Consider the following sentences and symbolization key:

J1. Denison will climb with me or he will watch movies.
J2. Either Denison or Ellery will climb with me.

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

Sentence J1 may be paraphrased 'Either D or M'. To fully regiment J1, we will introduce the DISJUNCTION symbol '$\vee$' where the sentences to its left and right are called DISJUNCTS. We may then take the disjunction '$D \vee M$' to regiment J1 where and D and M are the disjuncts. It is important to stress that '$\vee$' expresses an *inclusive* reading of disjunction which requires *at least one disjunct* to be true, admitting the possibility where both disjuncts are true.

Sentence I2 is only slightly more complicated. Although there are two subjects, the English sentence only includes the verb once. We can paraphrase I2 as 'Either Denison will climb with me or Ellery will climb with me.' We may then take '$D \vee E$' to regiment I2.

> DISJUNCTION TEST: A sentence can be regimented by $(\varphi \vee \psi)$ if it can be paraphrased in English as 'Either φ or ψ' where each disjunct is a sentence.

Since '$\vee$' is a symbol that we have introduced, we get to *stipulate* its truth-conditions. And we stipulate that we are using inclusive or, so that $A \vee B$ is true provided that at least one disjunct is true, including the scenario where both are true. Is this a good stipulation for how to treat 'or'? In a study of the semantics of English, it would be appropriate to pursue this question much further. But this book is about logic, not linguistics. Rather than concerning ourselves with describing the exact patterns by which 'or' is used in English, we will be concerned to identify a formal analogue which is simpler and more consistent in its use. Thus we have good reason to stipulate that '$\vee$' is inclusive even if 'or' in English is not.

So '$D \vee E$' is true if 'D' is true, if 'E' is true, or if both 'D' and 'E' are true. It is false only if both 'D' and 'E' are false. We can summarize this with the truth table for disjunction:

φ	ψ	$\varphi \vee \psi$
1	1	1
1	0	1
0	1	1
0	0	0

Like conjunction, disjunction is a binary operator which takes two truth-values as inputs and returns a single truth-value as output. The truth table above stipulates a meaning for '$\vee$' by indicating the truth-conditions for $\varphi \vee \psi$ where φ and ψ are any sentences whatsoever. We may succinctly restate the truth-conditions for conjunction by observing that $\varphi \vee \psi$ is false when φ and ψ are false, and true otherwise. This makes disjunction inclusive.

Consider the following sentences and symbolization key:

K1. Either you will not have soup or you will not have salad.
K2. You will have neither soup nor salad.
K3. You will have soup or salad, but not both.

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

Sentence K1 may be paraphrased by 'Either it is not the case that you will have soup, or it is not the case that you get salad.' Regimenting this claim requires both disjunction and negation since it is the disjunction of two negations '$\neg S_1 \vee \neg S_2$'.

Sentence K2 also requires negation. It can be paraphrased as, 'It is not the case that: either that you get soup or that you get salad.' In other words, it is a negation of a disjunction. We need some way of indicating that the negation does not just negate the right or left disjunct, but rather the entire disjunction. In order to do this, we will put parentheses around the disjunction as in '$\neg(S_1 \vee S_2)$'.[5] The parentheses are doing important work, since the sentence '$\neg S_1 \vee S_2$' regiments 'Either you will not have soup or you will have salad', which is different.

Since K3 has a more complicated structure, we can avoid making mistakes by break it into two parts. The first part says that you will have soup or you will have salad. We regiment this by '$(S_1 \vee S_2)$', including parentheses so that we don't get mixed up later on. The second part says that you will not have both soup and salad. We can paraphrase this as, 'It is not the case that you will have soup and you will have salad' which we may regiment with '$\neg(S_1 \wedge S_2)$'. In order to put these two parts together we may recall that 'but' is typically regimented by conjunction. Thus we may take '$(S_1 \vee S_2) \wedge \neg(S_1 \wedge S_2)$' to regiment K3.

1.6 The Material Conditional

Consider the following sentences and symbolization key:

L1. If you cut the red wire, then the bomb will explode.

[5] A second, equivalent, way to regiment K2 is '$(\neg S_1 \wedge \neg S_2)$'. We'll see why this is equivalent later on. The equivalence of '$\neg(A \vee B)$' and '$(\neg A \wedge \neg B)$' is an instance of one of De Morgan's laws.

L2. The bomb will explode only if you cut the red wire.
L3. The bomb will explode if you cut the red wire.

R: You cut the red wire.
B: The bomb will explode.

Sentence L1 can be partially regimented by 'If R, then B'. We will use the MATERIAL CONDITIONAL symbol '$\rightarrow$' (or CONDITIONAL for short) where the sentence on the left is the ANTECEDENT and the sentence on the right is the CONSEQUENT. Thus we may take '$R \rightarrow B$' to regiment L1 where 'R' is the antecedent and 'B' is the consequent.

The sentence L2 is also a conditional sentence. Since the word 'if' appears in the second half of the sentence, it might be tempting to regiment this in the same way as sentence L1. However, the conditional '$R \rightarrow B$' says what the partial regimentation '*if* R, *then* B' says or, equivalently, what 'B *if* R' says. Neither of these claims say that your cutting the red wire is the *only* way that the bomb will explode. For instance, someone else might cut the wire, or else the bomb might be on a timer. The sentence '$R \rightarrow B$' does not say anything about what to expect if 'R' is false. By contrast, L2 says that the only conditions under which the bomb will explode are ones where you cut the red wire, i.e., if the bomb explodes, then you have cut the wire. Thus sentence L2 may be regimented by '$B \rightarrow R$'.

Notice that we have the same antecedent/consequent pattern for both sentences: the sentence before the 'only if' is the antecedent and the sentence after the 'only if' is the consequent. Hence, we could also write sentence L1 as 'You cut the red wire only if the bomb will explode', which we could regiment by '$R \rightarrow B$'. We find something different in the sentence L3 where the 'if' occurs in the middle. Sentence L3 can be paraphrased by L1, and so regimented in the same way by '$R \rightarrow B$'. Regimenting conditional claims in English is tricky business and will require considerably more care than regimenting negations, conjunctions, and disjunctions.

It is important to remember that the operator '$\rightarrow$' says only that, if the antecedent is true, then the consequent is true. It says nothing about the *explanatory* connection between the antecedent and consequent. For instance, regimenting sentence L2 as '$B \rightarrow R$' does not mean that the bomb exploding would somehow have caused you to cut the wire. Both sentence L1 and L2 suggest that, if you cut the red wire, your cutting the red wire would explain why the bomb exploded. Nevertheless, they differ in the *logical* connection that they assert. If sentence L2 were true, then an explosion would tell us that you had cut the red wire. Without an explosion, sentence L2 tells us nothing about what you did with the red wire.

> MATERIAL CONDITIONAL TEST: A sentence can be regimented by $(\varphi \rightarrow \psi)$ if it can be paraphrased in English as 'If φ, then ψ', where the antecedent and consequent are both sentences.

For any sentences φ and ψ, if the conditional $\varphi \rightarrow \psi$ is true and the antecedent φ is true, then the consequent ψ is also be true. Hence, if the antecedent φ is true but the consequent

ψ is false, then the conditional $\varphi \to \psi$ is false. But what is the truth-value of $\varphi \to \psi$ when either φ is false or ψ is true? Suppose, for instance, that the antecedent φ happens to be false. It would seem that the conditional $\varphi \to \psi$ does not assert any claim about the truth-value of the consequent ψ, and so— at least in ordinary English— it is unclear what truth-value $\varphi \to \psi$ should have on the assumption that φ is false.

In English, the truth of conditionals often depends on what *would* be the case if the antecedent *were true* even if the antecedent is false. Put otherwise, such reasoning is not truth-functional, i.e., we need to know more about the sentence in question than just its truth-value. This poses a serious challenge for regimenting conditionals in $\mathcal{L}^{\text{PL}}$. In order to consider what the world would be like if R were true, we would need to know something about what R says about the world and we are quickly led into deep questions about the way the world is, laws of nature, and counterfactual reasoning— topics for philosophy of language, metaphysics, and the philosophy of science, but not this class. Rather, we will restrict consideration to the truth-values of sentences, ignoring circumstances in which their truth-values are different.

What we are after is a truth-function which approximates the meaning of conditional claims in English such as L1 and L2. More specifically, we want to specify the truth-value of $\varphi \to \psi$ as a function of the truth-values for φ and ψ, and nothing else. This is a big limitation, since there are not many truth-functions of two values out there. In fact there are only 16. Thus we will choose the best among them to approximate the 'If... then...' construction in English. We will specify this truth-function with the following truth table:

φ	ψ	$\varphi \to \psi$
1	1	1
1	0	0
0	1	1
0	0	1

Observe that when the antecedent φ is false, the conditional $\varphi \to \psi$ is true regardless of the truth-value of the consequent ψ. If both φ and ψ are true, then the conditional $\varphi \to \psi$ is true. In short, $\varphi \to \psi$ is false just in case φ is true and ψ is false.

More than any other sentential operator, the material conditional provides an extremely rough approximation of English conditional claims in $\mathcal{L}^{\text{PL}}$ with some counter-intuitive consequences. For example, a conditional in $\mathcal{L}^{\text{PL}}$ is true whenever the consequent is true, independent of the truth-value of the antecedent. Additionally, a conditional in $\mathcal{L}^{\text{PL}}$ is true any time the antecedent is false, independent of the truth-value of the consequent. These are odd consequence since at least some conditionals in English with true consequents and/or false antecedents seem clearly to be false. For instance, consider the following examples:

M1. If there are no philosophy courses at MIT, then Logic I is a philosophy course at MIT.
M2. If this book has fewer than thirty pages, then it will win the Pulitzer prize.

Both M1 and M2 seem clearly false. But each of them, regimented in $\mathcal{L}^{\text{PL}}$, would come out true. Sentences such as M1 and M2 are sometimes referred to as *paradoxes for the material conditional* since their regimentations in terms of the material conditional are true. However, such sentences are only paradoxical if we take '$\rightarrow$' to mean what 'If ..., then ...' means in English. Rather, the material conditional is a purely artificial piece of formal vocabulary which we have introduced by stipulating how it's truth-value is determined. Although the material conditional only provides a rough approximation of the 'If ..., then ...' in English, who said that English has the best vocabulary for theoretical applications? After all, mathematics is also full of artificial, entirely stipulated definitions which are nevertheless of great utility in part because of the precise nature of their meanings given their explicit definitions.

Despite the oddness of taking the regimentations of sentences such as M1 and M2 to be true, the material conditional preserves many of the most important logical features of conditionals claims in English. Indeed, the material conditional is perfectly adapted to the purposes of mathematics, and this alone covers a very important part of human reasoning. Programming languages also make use of material conditional claims, and these have proved to be profoundly useful. Rather than trying to capture all the subtleties of conditional English constructions, we may take the material conditional to be an approximation of and abstraction from the complexities of a natural language such as English.

Whereas the truth-conditions for material conditional sentences are precisely defined, there is very little in English for which we have clear definitions, and certainly not the word 'if'! Indeed, philosophers, linguists, and logicians have spent over a century developing sophisticated mathematical theories to model the behaviour of 'If ..., then ...' in English, and we are still very far from any kind of conclusive theory. Given how unclear we are about the meaning of conditionals constructions like 'If ..., then ...' in English, it is difficult to rely on these constructions in theoretical applications. By contrast, the material conditional is easy to understand completely even if it pulls apart from similar sounding claims in English.

We will have lots of occasions to observe the utility of the material conditional throughout this course. For now, I'll just ask you to go along with this approach to conditionals even if the material conditional sometimes seems to provide strange results. Believe it or not, the methods of modern logic have been applied for now over a hundred years attempting, among many other things, to provide a fully adequate regimentation of 'if' as it occurs in English. Although considerable progress has been made, conclusive success is still forthcoming.

1.7 The Material Biconditional

Consider the following sentences and symbolization key:

N1. The figure on the board is a triangle only if it has exactly three sides.
N2. The figure on the board is a triangle if it has exactly three sides.

N3. The figure on the board is a triangle if and only if it has exactly three sides.

T: The figure is a triangle.
S: The figure has three sides.

Sentence N1, for reasons discussed above, can be regimented as '$T \to S$'.

Sentence N2 is different. Since N2 can be paraphrased as, 'If the figure has three sides, then it is a triangle', we may take '$S \to T$' to regiment N2.

Sentence N3 says that T is true *if and only if* S is true. Although we could regiment N3 by conjoining two conditional claims, we will introduce the MATERIAL BICONDITIONAL symbol '$\leftrightarrow$' (or BICONDITIONAL for short) for this purpose. Because we could always write $(\varphi \to \psi) \wedge (\psi \to \varphi)$ instead of $\varphi \leftrightarrow \psi$, we do not strictly speaking *need* to introduce a new symbol for the biconditional.[6] Nevertheless, logical languages often include such a symbol, and in our case $\mathcal{L}^{\text{PL}}$ will include one, making it easier to regiment phrases like 'if and only if'.

Instead of referring to the sentence on the left-hand side of a biconditional as the antecedent and the sentence on the right-hand side as the consequent, we will refer to the sentences on either side of a biconditional as the ARGUMENTS of the biconditional, where this is a general term for the sentences on which a logical constant operates. Whereas negation takes one argument (the negand), all of the other operators in $\mathcal{L}^{\text{PL}}$ take two arguments. It is important to clarify that the arguments of an operator have nothing to do with the arguments consisting of sentences which we will evaluate for logical validity.

> MATERIAL BICONDITIONAL TEST: A sentence can be regimented by $(\varphi \leftrightarrow \psi)$ if it can be paraphrased in English as 'φ if and only if ψ' or 'φ just in case ψ', where the arguments are both sentences.

In order to specify the truth-conditions for sentences like $\varphi \leftrightarrow \psi$, consider the following:

φ	ψ	$\varphi \leftrightarrow \psi$
1	1	1
1	0	0
0	1	0
0	0	1

The truth table above indicates the truth-conditions for the biconditional by taking $\varphi \leftrightarrow \psi$ to be true if φ and ψ have the same truth-value, and false if φ and ψ have different truth-values. Although we know that φ and ψ will have different truth-values if $\varphi \leftrightarrow \psi$ is false, this does not tell us whether φ is true and ψ is false, or *vice versa*. Similarly, knowing that $\varphi \leftrightarrow \psi$ is true does not tell us whether both φ and ψ are true, or whether both are false.

[6] If fact the only truth-function we really need is called 'nand' (not-both), but doing so would be tedious!

1.8 Unless

We have now introduced all of the operators for $\mathcal{L}^{\text{PL}}$. We can use them together to regiment many kinds of sentences. Consider the following examples and associated symbolization key:

O1. Unless you wear a jacket, you will catch a cold.
O2. You will catch a cold unless you wear a jacket.

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

We can paraphrase sentence O1 as 'Unless J, D.' This means that if you do not wear a jacket, then you will catch a cold, and so we may regiment O1 as '$\neg J \to D$.'

This same sentence O1 also means that if you do not catch a cold, then you must have worn a jacket. With this in mind, we might regiment O1 as '$\neg D \to J$'. You might then wonder which of these is the correct regimentation of sentence O1. The answer is that both regimentations are correct. Not only are these regimentations logically equivalent, there no reason to prefer one regimentation to another. Rather, both regimentations are equally natural.

What about O2? We may begin by taking 'Unless you wear a jacket, you will catch a cold' to paraphrase O2, and so either '$\neg J \to D$' or '$\neg D \to J$' may be taken to regiment O2 where neither is better than the other. As a result, O2 is logically equivalent to sentence O1.

When regimenting sentences like sentence O1 and sentence O2, it is easy to get turned around. Since the conditional is not symmetric, it would be wrong to regiment either sentence as '$J \to \neg D$' or '$D \to \neg J$'. Fortunately, there are symmetric ways to regiment O1 and O2. In particular, both sentences say that you will wear a jacket or— if you do not wear a jacket— then you will catch a cold. So we can regiment O1 and O2 as '$J \vee D$'. Although linguistically less natural, this regimentation is easier to remember. It helps that '$Q \vee P$' is logically equivalent to '$P \vee Q$', so if you use disjunction, you don't have to worry about the order.

1.9 Well-Formed Sentences

The sentence 'Apples are not blue' is a sentence of English and '$\neg A$' is a sentence of $\mathcal{L}^{\text{PL}}$. Although we can identify sentences of English when we encounter them, we do not have a precise definition of what counts as an English sentence. Students used to learn grammarians' attempts to formalize some such rules, but contemporary linguists agree that this was a hopeless project. Natural languages like English are just not susceptible to such precisification. By contrast, it is possible to define what counts as a sentence in $\mathcal{L}^{\text{PL}}$ where it is in part for this reason that we introduce an artificial language like $\mathcal{L}^{\text{PL}}$ in the first place.

Whenever a language becomes the object of study, we call the language that is being studied the OBJECT LANGUAGE and the language in which we are conducting our study the METALANGUAGE. The object language we will be concerned with in this chapter is $\mathcal{L}^{\text{PL}}$. In this section, we will provide a precise definition of the sentences of $\mathcal{L}^{\text{PL}}$. The definition itself will be given in the metalanguage which in our case will consist of English enriched with certain amount of mathematical vocabulary, e.g., the schematic variables 'φ' and 'ψ'.

It is vitally important to distinguish between the object language and metalanguage, doing our best to avoid mixing them up. We will be helped by the fact that the sentences of our object language $\mathcal{L}^{\text{PL}}$ are entirely formal, whereas the sentences of our metalanguage are mostly informal, though they may contain some mathematical elements. For instance, the sentence '$\neg A$' is a sentence in the object language $\mathcal{L}^{\text{PL}}$ because it only uses symbols of $\mathcal{L}^{\text{PL}}$. In contrast, the sentence "The expression '$\neg A$' is a sentence of $\mathcal{L}^{\text{PL}}$" is not a sentence of $\mathcal{L}^{\text{PL}}$, but rather a sentence in the metalanguage that we use to talk *about* '$\neg A$' which is a sentence of $\mathcal{L}^{\text{PL}}$.

1.9.1 The Use/Mention Distinction

So far, we have talked a lot *about* sentences and will continue to do so throughout this text. Of course, we have also used sentences to say things, e.g., that there is no precise mathematical definition of the declarative sentences of English. In order to sharpen this contrast, consider these the following sentences:

P1. Kamala Harris is the Democratic Nominee.
P2. 'Kamala Harris' is composed of two uppercase letters and ten lowercase letters.

When we want to talk about Kamala Harris, we *use* her name as in P1. When we want to talk about Kamala Harris' name, we *mention* her name which we do by putting her name in quotation marks as in P2. Similarly, whereas it is true to say that you are learning logic, the expression 'logic' is the name of the subject that you are learning.

In general, when we want to talk about how things are, we *use* expressions in a language. When we want to talk about the expressions of a language, we *mention* those expression. Of course, we need to indicate that we are mentioning expressions rather than using them. To do this, some convention or other is needed. Here is the first convention that we will use:

QUOTES: A quoted expression is the *canonical name* for the expression quoted.

For instance 'ABC' is the name for the expression consisting of the first letter of the alphabet, followed by the second letter of the alphabet, followed by the third letter of the alphabet. Put otherwise, 'ABC' is the result of CONCATENATING the symbols 'A', 'B', and 'C' where speaking loosely, this means that the complex symbol 'ABC' is formed by putting the others together one after the next. Concatenation will play an important role below.

Consider the following sentences:

Q1. 'Kamala Harris' is the Democratic Nominee.
Q2. Kamala Harris is composed of two uppercase letters and ten lowercase letters.

Sentence Q1 says that the expression 'Kamala Harris' is the Democratic Nominee which is false. Rather, it is the *woman* named by the expression 'Kamala Harris' who is the Democratic Nominee, not her *name*. We find a related mistake in P2 which says that the woman Kamala Harris is composed of letters which is also false. Here is another important type of example:

R1. " 'Kamala Harris' " is the name of 'Kamala Harris'.

Whereas on the left we have the name of a name, on the right we have a name. Perhaps this kind of sentence only occurs in logic textbooks, but it is true nonetheless.

It is important to contrast two other uses that quotation marks often have in other contexts: attribution and scare quotes. These uses are connected. For instance, in writing a paper, one might *use* the words of another while nevertheless attributing those words to that author. Here's a concrete example. Say we are discussing Quine's ontology, and we want to say that Quine argues that positing merely possible objects, "offends the aesthetic sense of us who have a taste for desert landscapes." The quoted words belong to Quine, and we want to make this clear to our reader. Nevertheless, we are still *using* Quine's words; we are not merely mentioning them, i.e., naming the string that he wrote.

Along these same lines, one might *use* certain words to make a claim but without wanting to attribute that claim to oneself even though there is no one else to which we might attribute the claim. For instance, one might claim that the song is 'ratchet' to use the slang term but without fully standing by its use. This usage might suggest that others take the song to be ratchet without joining them in doing so. There are many such examples along these and other lines, but this is not the way that we will be using quotation marks in this text.

Although quotes are often helpful in order to indicate that we are talking about an expression rather than using that expression to assert something, quotes are only useful for speaking about the specific expression inside the quotes. In order to speak more effectively about all expressions of $\mathcal{L}^{\text{PL}}$, it will be important to introduce a further device. By way of motivation, the following section will begin to introduce the primitive symbols of $\mathcal{L}^{\text{PL}}$.

1.9.2 Primitive Symbols

In order to define the sentences of $\mathcal{L}^{\text{PL}}$ in a more careful way than we have so far, it will be important to introduce the primitive symbols of $\mathcal{L}^{\text{PL}}$. Some of these we have already seen above. In particular, consider the following definitions for $\mathcal{L}^{\text{PL}}$:

SENTENTIAL OPERATORS: '$\neg$','$\wedge$','$\vee$','$\rightarrow$', and '$\leftrightarrow$'.
PUNCTUATION: '(' and ')'.

Quotes have been used above in order to explicitly indicate the symbols that are taken to be sentential operators and parentheses, respectively. The sentential operators are also commonly called CONNECTIVES since they connect sentences to form new sentences of greater complexity. In our case, the sentential operators above are all truth-functional since the truth-value of a sentence with any of the sentential operators given above as its main operator is determined by the truth-values of its arguments. Accordingly, the sentential operators indicated above are also sometimes called the TRUTH-FUNCTIONAL CONNECTIVES (or TRUTH-FUNCTIONS) as well as EXTENSIONAL OPERATORS. For simplicity, we will refer to the sentential operators above as the OPERATORS of $\mathcal{L}^{\text{PL}}$ since we will not consider non-extensional operators. Additionally, the parentheses indicated above provide the necessary punctuation needed to specify the order of operations in complex sentences in which multiple operators occur.

Given that there are a finite number of operators and parentheses, it is straightforward to indicate them individually using quotation marks. The same cannot be said, however, for the sentence letters which may include infinitely many different subscripts. For instance, consider the following attempt to specify the sentence letters of $\mathcal{L}^{\text{PL}}$:

Attempt 1: 'φ_x' is a SENTENCE LETTER of $\mathcal{L}^{\text{PL}}$ whenever φ is a capital letter of the English alphabet and x is a numeral for a natural number.
Attempt 2: φ_x is a SENTENCE LETTER of $\mathcal{L}^{\text{PL}}$ whenever φ is a capital letter of the English alphabet and x is a numeral for a natural number.

The problem with the first attempt is that even assuming that 'φ' is schematic variable whose values range over all capital letters of the English alphabet and 'x' is a variable ranging over natural numbers, 'φ_x' names one and the same expression every time. In particular, 'φ_x' names the expression which occurs within the quotation marks which consists of a lowercase Greek letter subscripted by a lowercase English letter. By the lights of the first attempt, there is only once sentence letter $\mathcal{L}^{\text{PL}}$, i.e., 'φ_x', contrary to what we intended.

In response to the shortcomings of this first attempt, the second attempt given above removes the quotation marks altogether. Given that φ is any capital letter of the English alphabet and x is any natural number, we may now expect there to be many different sentence letters of $\mathcal{L}^{\text{PL}}$, thereby avoiding the problem that we faced before. In the instance where φ has the first capital letter of the English alphabet 'A' as its value and x has the numeral for the first nonzero natural number '1' as its value, the second attempt asserts that A_1 is a sentence letter of $\mathcal{L}^{\text{PL}}$. Similarly, B_3, H_0, W_{27}, and so on are all taken to be sentence letters by the lights of the second attempt. Although this may seem to cut considerably closer to what we want, the second attempt falls short by *using* the sentence letters that it aims to introduce rather than *mentioning* them as we might otherwise intend.

In order to press the previous point, it is worth considering one more attempt to define all of the sentence letters of $\mathcal{L}^{\text{PL}}$. Rather than using variables, one might provide paradigm cases:

Attempt 3: 'A_0','A_1', ...,'B_0','B_1', ...,'Z_0','Z_1', ... are the SENTENCE LETTERS of $\mathcal{L}^{\text{PL}}$.

All of the sentence letters indicated above are mentioned rather than used, where this is just what we want in order to complete our specification of the primitive symbols of $\mathcal{L}^{\text{PL}}$. Whereas A_1 is not a sentence letter of $\mathcal{L}^{\text{PL}}$, 'A_1' is a sentence letter of $\mathcal{L}^{\text{PL}}$. Although it is reasonably clear how to continue the list of partial lists indicated above, this definition still leaves something to be desired. Besides answering the immediate question of *which* sentence letters $\mathcal{L}^{\text{PL}}$ includes, there is also the methodological question of *how* to specify all of the sentence letters included in $\mathcal{L}^{\text{PL}}$ without relying on our readers ability to extend the partial lists indicated above. Even if the answer to the first question is clear enough, it is the answer to the second question that will have a number of further applications below. It is for this reason that we will improve on the third attempt however pedantic this may seem.

1.9.3 Corner Quotes

Continuing with our previous ambition to specify the sentence letters of $\mathcal{L}^{\text{PL}}$ in an accurate, explicit, and exhaustive way, consider the following somewhat long-winded alternative:

Attempt 4: The symbol to result from subscripting φ by x is a SENTENCE LETTER of $\mathcal{L}^{\text{PL}}$ whenever φ is a capital letter of the English alphabet and x is a numeral for a natural number.

This definition succeeds where the others failed. The only remaining issues that we may raise is just how cumbersome the construction 'The symbol to result from subscripting...' is, lacking the syntactic simplicity and generality that we might otherwise want.

Enter CORNER QUOTES, also called QUINE QUOTES on account of W.V. Quine. Instead of using the long-winded construction given above, we may stipulate the following:

SENTENCE LETTERS: $\ulcorner\varphi_x\urcorner$ for any capital English letter φ and natural numeral x.

Whereas standard quotation marks are used to name the string of symbols that they contain, corner quotes are used to refer to the complex symbol that results when the schematic variables are replaced with explicit values. One way to put the point is to say that $\ulcorner\varphi_x\urcorner$ is the result of concatenating φ with a subscript x. So long as 'φ' and 'x' are both schematic variables which have symbols as values, concatenating φ with a subscript x is also a symbol, e.g., 'A_1'. It is in this way that we may explicitly specify all sentence letters of $\mathcal{L}^{\text{PL}}$.

1.9.4 Expressions

Given the definitions above, we may now define the PRIMITIVE SYMBOLS of $\mathcal{L}^{\text{PL}}$ to include the operators, punctuation, and sentence letters for $\mathcal{L}^{\text{PL}}$ given above, and nothing besides. The EXPRESSIONS of $\mathcal{L}^{\text{PL}}$ may the be defined recursively as follows:

1. Every primitive symbol of $\mathcal{L}^{\text{PL}}$ is an expression of $\mathcal{L}^{\text{PL}}$.
2. For any expressions φ and ψ of $\mathcal{L}^{\text{PL}}$, $\ulcorner\varphi\psi\urcorner$ is an expression of $\mathcal{L}^{\text{PL}}$.
3. Nothing else is an expression of $\mathcal{L}^{\text{PL}}$.

In addition to taking the primitive symbols of $\mathcal{L}^{\text{PL}}$ to be expressions of $\mathcal{L}^{\text{PL}}$, the result of concatenating any two expressions of $\mathcal{L}^{\text{PL}}$ is an expression of $\mathcal{L}^{\text{PL}}$, and nothing else besides. This definition specifies the expressions of $\mathcal{L}^{\text{PL}}$ in an explicit and exhaustive way.

1.9.5 Well-Formed Sentences

Since any sequence of symbols is an expression, many expressions of $\mathcal{L}^{\text{PL}}$ will fail to be candidates for interpretation. That is, not only do they fail to mean something on a particular interpretation, there is no good way to interpret them at all. In particular, there is no good way to assign them truth-values. For example, consider the following expressions:

S1. $\neg\neg\neg\neg$
S2. B_3A_0
S3. $))\leftrightarrow$
S4. $A_4\ \vee$

In order to interpret $\mathcal{L}^{\text{PL}}$, we need to say which expressions are candidates for interpretation, where we may expect the expressions above to be excluded. Put otherwise, we may ask which expressions of $\mathcal{L}^{\text{PL}}$ are grammatical, or as we will soon say, *well-formed sentences*. For ease of exposition, we will use the acronym 'wfs' throughout what follows where the plural is 'wfss'.

Sentence letters like 'A_1' and 'G_{13}' are certainly wfss. We can form further wfss out of these by using the various operators. Using negation, we can get '$\neg A_1$' and '$\neg G_{13}$'. Using conjunction, we can get '$A_1 \wedge G_{13}$', '$G_{13} \wedge A_1$', '$A_1 \wedge A_1$', and '$G_{13} \wedge G_{13}$'. We could also apply negation repeatedly to get '$\neg\neg A_1$' or apply negation along with conjunction to get wfss like '$\neg(A_1 \wedge G_{13})$' and '$\neg(G_{13} \wedge \neg G_{13})$'. The possible combinations are endless, even starting with just these two sentence letters rather than infinitely many sentence letters as above.

Although there is no point in trying to list all the wfss, we can define all WELL-FORMED SENTENCES OF $\mathcal{L}^{\text{PL}}$ by way of the following recursive clauses:

1. Every sentence letter of $\mathcal{L}^{\text{PL}}$ is a wfs of $\mathcal{L}^{\text{PL}}$.
2. For any expressions φ and ψ of $\mathcal{L}^{\text{PL}}$, if φ and ψ are wfss of $\mathcal{L}^{\text{PL}}$, then:
   (a) $\ulcorner\neg\varphi\urcorner$ is a wfs of $\mathcal{L}^{\text{PL}}$;
   (b) $\ulcorner(\varphi \wedge \psi)\urcorner$ is a wfs of $\mathcal{L}^{\text{PL}}$;
   (c) $\ulcorner(\varphi \vee \psi)\urcorner$ is a wfs of $\mathcal{L}^{\text{PL}}$;
   (d) $\ulcorner(\varphi \rightarrow \psi)\urcorner$ is a wfs of $\mathcal{L}^{\text{PL}}$; and
   (e) $\ulcorner(\varphi \leftrightarrow \psi)\urcorner$ is a wfs of $\mathcal{L}^{\text{PL}}$.
3. Nothing else is a wfs of $\mathcal{L}^{\text{PL}}$.

As in the definition of the expressions of $\mathcal{L}^{\text{PL}}$, the definition of the wfss of $\mathcal{L}^{\text{PL}}$ is recursive on account of calling on the wfss of $\mathcal{L}^{\text{PL}}$ in (2) in order to define further wfss of $\mathcal{L}^{\text{PL}}$. If you have not seen recursive definitions before, you might worry that this is all rather circular.

In order assuage any doubts that the recursive definition of the wfss of $\mathcal{L}^{\text{PL}}$ given above is in good standing it will help to think of building the set of all wfss of $\mathcal{L}^{\text{PL}}$ as follows. In stage 0, we add all the sentence letters, calling this set Λ_0. Then in stage 1, we take any wfss from Λ_0 and substitute them for φ and ψ in the COMPOSITION RULES given in condition (2) above, adding the results to a new set Λ_0'. We do this in all the ways that we can, adding as much to Λ_0' as possible while limiting ourselves to the ingredients included in Λ_0. Once we stop getting anything new, we may take $\Lambda_1 = \Lambda_0 \cup \Lambda_0'$ which contains all and only the wfss which belong to either Λ_0 or Λ_0'. We then repeat the process to build Λ_2 from Λ_1 in the same way. More generally, given any Λ_n we may build Λ_{n+1} by the same procedure. Finally we consider the union which gathers together the members from each Λ_n for all $n \in \mathbb{N}$.

$$\Lambda = \bigcup_{n \in \mathbb{N}} \Lambda_n.$$

A simpler way to describe the same thing is to take the set Λ of wfs of $\mathcal{L}^{\text{PL}}$ to be the smallest set— this corresponds to condition (3) above— to satisfy conditions (1) and (2) above. By imposing condition (3), we are making sure that nothing else ends up a wfs of $\mathcal{L}^{\text{PL}}$ like the number 2 or the Eiffel tower, thereby specifying a unique set of wfs of $\mathcal{L}^{\text{PL}}$.

Note that the definition of the wfss of $\mathcal{L}^{\text{PL}}$ is purely *syntactic*. Each composition rule specifies which expressions of $\mathcal{L}^{\text{PL}}$ are to be considered wfs of $\mathcal{L}^{\text{PL}}$. These rules will matter a great deal in the semantic and metalogical portions of this text. The resulting definition provides exactly what linguists have given up attempting to provide for English: a complete specification once and for all of which syntactic constructions are grammatical sentences— i.e., the wfss— of $\mathcal{L}^{\text{PL}}$. It is important to stress that the definition of the wfss $\mathcal{L}^{\text{PL}}$ does not tell us what the sentences of $\mathcal{L}^{\text{PL}}$ *mean* or what truth-values they have. To do so, we will provide a semantics for $\mathcal{L}^{\text{PL}}$ in Chapter 2. For now, our concern is limited to the rules for writing sequences of symbols in $\mathcal{L}^{\text{PL}}$ which count as well-formed sentences and nothing more.

1.9.6 Main Operator

It is worth computing whether an expression is a wfss of $\mathcal{L}^{\text{PL}}$ or not. For instance, suppose that we want to know whether or not '$\neg\neg\neg D$' is a wfs of $\mathcal{L}^{\text{PL}}$. Looking at the second clause of the definition, we know that '$\neg\neg\neg D$' is a wfs *if* '$\neg\neg D$' is a wfs. So now we need to ask whether or not '$\neg\neg D$' is a wfs. Again looking at the second clause of the definition, '$\neg\neg D$' is a wfs *if* '$\neg D$' is. Again, '$\neg D$' is a wfs *if* 'D' is a wfs. Now 'D' is a sentence letter, so we know that 'D' is a wfs by the first clause of the definition. Thus '$\neg\neg\neg D$' is in fact a wfs.

The operator that you look to first in decomposing a sentence is called the MAIN OPERATOR of that sentence. For example, the main operator of '$\neg(E \vee (F \rightarrow G))$' is negation, and the main operator of '$(\neg E \vee (F \rightarrow G))$' is disjunction. Conversely, if you're building up a wfs from simpler sentences, the operator introduced by the last composition rule you apply is the main operator of the resulting sentence. It is the operator that governs the interpretation of the entire sentence. Being able to identify the main operator of sometimes convoluted sentences in $\mathcal{L}^{\text{PL}}$ is going to be an essential skill in your logical tool kit.

1.10 Metalinguistic Abbreviation

So far we have been careful to be precise. This is sort of precision is important when introducing a formal theory of syntax for the first time. However, maintaining this level of precision is both tedious and can often be hard to read, and so it is common to introduce certain simplifications as a convenient shorthand. For instance, we will often suppress the subscript '$_0$', writing 'A' and 'B' in place of 'A_0' and 'B_0', and so on for the other twenty-four sentence letters in which '$_0$' is the subscript. In this way, subscripts are rarely required, avoiding quite a bit of trouble by simply using different capital letters.

However simple, suppressing the subscript '$_0$' in sentence letters provides a paradigm case of the kinds of simplifications that we will introduce in this section. It is important to specify that these notational conventions do not change the official definitions. For instance, just because we have started writing 'A' instead of 'A_0' does not mean that 'A' is officially a wfss of $\mathcal{L}^{\text{PL}}$ and 'A_0' is not. Rather, we have simply provided a shorthand which can be done away with, recovering the official sentence letters of $\mathcal{L}^{\text{PL}}$. More generally, our notational conventions provide a convenient shorthand that eases the expression of the sentences of $\mathcal{L}^{\text{PL}}$ given a clear understanding of the official definitions. Accordingly, these notational conventions are called METALINGUISTIC ABBREVIATIONS since they do not happen in our object language, but rather in the metalanguage that we use for talking *about* elements of the object language in a natural and abbreviated way. It is nevertheless important to be clear about which conventions we will permit throughout what follows to avoid introducing confusion.

Before indulging in the simplifications indicated below, it is important to ensure that: (1) you understand the official definitions in both spirit and letter; (2) the simplifications that

you employ have been specifically permitted in this text; and (3) you do not make any simplifications that lead to ambiguities that cannot be removed or else permitted without harm (more on this soon). For instance, the following section will explain how to drop parentheses. However, until you are confident about the use of parentheses in the definition of the wfss of $\mathcal{L}^{\text{PL}}$, it is good advice to stick to the official definition of the wfss of $\mathcal{L}^{\text{PL}}$. The notational conventions that we provide are a way to skip steps when writing things down, but they will not help you master the official definitions. If you're unsure about whether it's OK to take a certain shortcut, the safest thing is to continue to use the official definitions. With this health warning in place, we may proceed with caution.

1.10.1 Parentheses

Given the definition of the wfs of $\mathcal{L}^{\text{PL}}$, we are now in a position to observe that neither of the following are wfss of $\mathcal{L}^{\text{PL}}$, and for good reason:

T1. $Q \wedge R$.
T2. $C \vee D \wedge E$.

The reason that T1 is not a wfs is the boring but crucial reason that it lacks outermost parentheses. Officially, a wfs like '$Q \wedge R$' must have outermost parentheses because we might want to use this sentence to construct further, more complicated sentences which have this sentence as a part. For instance, if we negate '$(Q \wedge R)$', we get '$\neg(Q \wedge R)$'. If we just had '$Q \wedge R$' without the parentheses and put a negation in front of it, we would have '$\neg Q \wedge R$' which it would be hard to distinguish from '$(\neg Q \wedge R)$', something very different than '$\neg(Q \wedge R)$'. The sentence '$\neg(Q \wedge R)$' means that it is not the case that both 'Q' and 'R' are true; 'Q' might be false or 'R' might be false, but the sentence does not tell us which. The sentence '$(\neg Q \wedge R)$' means specifically that 'Q' is false and that 'R' is true. This highlights the critical role that parentheses will play when we go on to interpret the wfss of $\mathcal{L}^{\text{PL}}$.

Officially, a conjunction is a sentence of the form $\ulcorner(\varphi \wedge \psi)\urcorner$ for any sentences φ and ψ, not a sentence of the form $\ulcorner\varphi \wedge \psi\urcorner$. Nevertheless, dropping the outermost parentheses is both permissible and common when it does not lead to any ambiguities, i.e., when there is a unique wfs $\mathcal{L}^{\text{PL}}$ which is easy to recover by adding an outermost pair, or else when the ambiguity that results does not make any substantial difference (more on this in the next chapter).

For example, we may recover the following wfs from T1 without any ambiguity whatsoever:

U1. $(Q \wedge R)$.

The same cannot be said for T2. Rather, we must choose between the following candidates:

V1. $((C \vee D) \wedge E)$.
V2. $(C \vee (D \wedge E))$.

Apart from anything to do with their meaning, the two wfss above are different. Whereas neither T1 nor T2 are officially wfss of $\mathcal{L}^{\text{PL}}$, both V1 and V2 are wfss of $\mathcal{L}^{\text{PL}}$ in the most official sense. Of course, we could have defined the wfss not to include outermost parentheses, so what's the reason for this stipulation? As it will turn out, V1 and V2 will have different truth-conditions, providing good reason to distinguish between them syntactically.

Without including any parentheses at all, our syntax would fail to keep track of the order in which a sentence has been constructed, and as a result, would run together sentences like V1 and V2 above, turning both of them back into the sentence T2. Although it will be convenient to sometimes drop parentheses to make things easier for ourselves, it is important to do this carefully so that we do not lose important information that we need in order to get back to the original form of the sentence. We will do this in several ways.

First, we understand that '$Q \wedge R$' is short for '$(Q \wedge R)$'. As a matter of convention, we can leave off parentheses that occur *around the entire sentence*. Even though we don't always write out the outermost parentheses, we know that they really should be there. It is important to stress that this is only possible when '$Q \wedge R$' occurs by itself, and not as a part of some more complex sentence. If we were able to drop parentheses even when some sentence occurs as a part of a bigger sentence, we would be able to turn both V1 and V2 into T2, and that is not what we want because then there would be no way to determine the main operator.

Second, it can sometimes be confusing to look at long sentences with nested sets of parentheses. We adopt the convention of using square brackets '[' and ']' in place of parenthesis. There is no logical difference between '$(P \vee Q)$' and '$[P \vee Q]$', for example. The unwieldy sentence '$(((H \to I) \vee (I \to H)) \wedge (J \vee K))$' could be written in the following way, omitting the outermost parentheses and using square brackets to make the structure of the sentence clear:

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

Note that *Carnap* will always add the outer parentheses which can make things a bit harder to parse. So be careful when using that system.

Third, we will sometimes want to regiment the conjunction or disjunction of three or more sentences. For instance, consider the following sentence and symbolization key:

W1. Alice, Bob, and Candice all went to the party.
W2. Either Alice, Bob, and Candice went to the party.

A: Alice went to the party.
B: Bob went to the party.
C: Candice went to the party.

The composition rules only allow us to form a conjunction out of two sentences, so we can regiment W1 as either '$(A \wedge B) \wedge C$' or '$A \wedge (B \wedge C)$'. However, there is no reason to distinguish between these regimentation since the two are logically equivalent, or put otherwise, the two are true in exactly the same interpretations. So we might as well just write '$A \wedge B \wedge C$'. As a matter of convention, we can leave out parentheses when we conjoin three or more sentences.

A similar situation arises in disjoining multiple sentences. For instance, W2 can be regimented as either '$(A \vee B) \vee C$' or '$A \vee (B \vee C)$'. Since these two regimentations are logically equivalent, we may write '$A \vee B \vee C$'. These two conventions only apply to multiple conjunctions or multiple disjunctions, not any combination of conjunctions and disjunctions. If a series of operators includes both disjunctions and conjunctions, then the parentheses are essential, as with '$(A \wedge B) \vee C$' and '$A \wedge (B \vee C)$'. The parentheses are also required if there is a series of conditionals or biconditionals, as with '$(A \to B) \to C$' and '$A \leftrightarrow (B \leftrightarrow C)$'.

If we had given a different definition of the wfss of $\mathcal{L}^{\text{PL}}$, strings of conjunctions or disjunctions could have counted as wfss of $\mathcal{L}^{\text{PL}}$, obviating the need for the conventions indicated above. For instance, we might have permitted $\ulcorner(\varphi \wedge \ldots \wedge \psi)\urcorner$ to be a wfs of $\mathcal{L}^{\text{PL}}$ whenever $\varphi, \ldots, \psi$ are all wfss of $\mathcal{L}^{\text{PL}}$. This would have made it a little easier to regiment some English sentences, but it would have also come at the cost of making our the language $\mathcal{L}^{\text{PL}}$ much more complicated. In particular, we would have to keep this more complex definition in mind when we provide a semantics and proof system for $\mathcal{L}^{\text{PL}}$. Rather, we want $\mathcal{L}^{\text{PL}}$ to be as simple as possible without reducing its expressive power. Since no expressive power is added by permitted $\ulcorner(\varphi \wedge \ldots \wedge \psi)\urcorner$ to be a wfs of $\mathcal{L}^{\text{PL}}$, there is little reason to indulge in this complication, keeping the semantics and proof theory as simple as possible. Nevertheless, insisting that conjunctions and disjunctions have exactly two conjuncts or disjuncts on the official definition does raise certain syntactic redundancies that are not motivated by corresponding semantic differences, e.g., between the equivalent regimentations of W1 and W2. Adopting notational conventions is a compromise between the competing desires to avoid complicating the semantics and proof theory of $\mathcal{L}^{\text{PL}}$ on the one hand and to avoid needless syntactic redundancies on the other.

Strictly speaking, '$A \vee B \vee C$' and '$A \wedge B \wedge C$' are not sentences of $\mathcal{L}^{\text{PL}}$. We write them this way for the sake of convenience, but really these sentences are ambiguous. The only reason we can get away with this is that the ambiguities do not amount to any logical differences since any way of adding parentheses will amount to sentences with the same truth-conditions.

1.10.2 Dropping Quotes

We have taken subscripted uppercase letters in English to be sentence letters of $\mathcal{L}^{\text{PL}}$:

$$A, B, C, Z, A_1, B_4, A_{25}, J_{375}, \ldots$$

Although we could have added quotes around each of the letters above, it is clear in this context that we mean to be mentioning and not using these sentences. After all, this text is written in the metalanguage of mathematical English and the sentences above are sentences of the object language $\mathcal{L}^{\text{PL}}$ that we intend to be discussing. Consider the following examples:

X1. D is a sentence letter of $\mathcal{L}^{\text{PL}}$.
X2. 'D' is a sentence letter of $\mathcal{L}^{\text{PL}}$.

Since 'D' is a sentence of $\mathcal{L}^{\text{PL}}$ and not our metalanguage, X1 is nonsense. Put flatly, the sentences of $\mathcal{L}^{\text{PL}}$ do not belong to our metalanguage mathematical English. Rather, it is the canonical names for the sentences of $\mathcal{L}^{\text{PL}}$ which belong to our metalanguage, and for this we must use quotes as before. Thus whereas X1 is gibberish, X2 is true.

We may then compare:

Y1. Schnee ist weiß is a German sentence.
Y2. 'Schnee ist weiß' is a German sentence.

Whereas Y1 is again gibberish belonging to no single natural language, Y2 is a perfectly intelligible sentence of English which happens to mention a sentence of German.

Although quotes are officially required for Y2 to be true, it is nevertheless pretty clear what this sentence intends. Matters are even clearer in the case of X1 and X2 above. Accordingly, it is permissible to drop the quotes in X2, using X1 in its place for ease. Since we will be talking about the sentences of $\mathcal{L}^{\text{PL}}$ a lot and the quotes can get cumbersome, we will often drop the quotes, relying on context and the reader's competence to know where they belong.

In addition to a convenience that we will indulge in throughout this course, dropping quotes is common practice throughout logic, and so it is good to get some practice with these conventions. This applies as much to corner quotes as it does to standard quotes. For instance, in defining the wfss of a language, it is common practice to do so *without* using corner quotes by relying on the reader's competence to know where they belong. Since this is an introductory text, the definitions above have been made explicit. Nevertheless, when we go on to setup further languages such as $\mathcal{L}^{\text{FOL}}$ below, we will have occasion to indulge in certain simplifications, leaving off the corner quotes.

In making such simplifications, what matters is that the conveniences that we indulge do not lead to any real ambiguities. Additionally, we should maintain quotes when clarity is improved. After all, the goal is to make things to read, not harder, so bear this in mind if you choose to drop quotes. Certainly in some contexts using quotes is invaluable.

1.10.3 Conventions for Arguments

One of the main purposes of using $\mathcal{L}^{\text{PL}}$ is to study arguments. In English, the premises of an argument are often expressed by individual sentences, and the conclusion by a further sentence. Since we can regiment English sentences in $\mathcal{L}^{\text{PL}}$, we can also regiment English arguments using $\mathcal{L}^{\text{PL}}$ by regimenting each of the sentences used in an English argument. Sometimes

English arguments run all the sentences together where clarity is helped by dividing things up when we go about giving our regimentation. Even so, $\mathcal{L}^{\text{PL}}$ itself has no way to flag some sentence as the conclusion and the other sentences as the premises. By contrast, English uses words like 'so', 'therefore', etc., to mark which sentence is the conclusion.

In Chapter 0, we specified that we will arrange arguments so that the conclusion is the final sentence, separating the premises with a horizontal line. Although we will maintain this convention below, it is worth emphasizing that $\mathcal{L}^{\text{PL}}$ does not include any such horizontal lines. Rather, these presentational details are taking place in the metalanguage in order to help us specify which sentences are the premises and conclusion. Nevertheless, this convention has the downside that it makes use of vertical space, and so cannot be present in line. Thus it will occasionally be helpful to use the symbol '$\therefore$' in order to indicate that what follows is the conclusion. For instance, suppose we want to regiment an argument with the premises and conclusion with ψ. We may write '$\varphi_1, \ldots, \varphi_n \therefore \psi$' without breaking the paragraph.

Since the symbol '$\therefore$' belongs to the metalanguage not $\mathcal{L}^{\text{PL}}$, one might think that we would need to put quotes around the $\mathcal{L}^{\text{PL}}$-sentences which flank it, referring to the result as an argument with $\mathcal{L}^{\text{PL}}$ sentences. That is a sensible thought, but adding these quotes would only make things more cumbersome and rather than easier to read. While we are stipulating conventions, we can simply stipulate that quotes around arguments are unnecessary. For instance, we may say that $A, A \to B \therefore B$ is an argument in $\mathcal{L}^{\text{PL}}$ whose premises are A and $A \to B$ and whose conclusion is B, leaving the quotes off for ease.