Chapter 1
Syntax: The Grammar of Symbols

The goal of this chapter is to develop the formal language of First-Order Logic from scratch. At the same time, we introduce some terminology of the so-called meta-language, which is the language we use when we speak *about* the formal language (*e.g.*, when we like to express that two strings of symbols are equal). In the meta-language, we shall use some notions of N A I V E S E T T H E O R Y like *sets* or the *membership relation* "$\in$". We would like to emphasise that these notions are not part of the language of formal logic.

Alphabet

Like any other written language, First-Order Logic is based on an *alphabet*, which consists of the following *symbols*:

(a) **Variables** such as $x, y, v_0, v_1, \dots$, which are place holders for objects of the *domain* under consideration (which can for example be the elements of a group, natural numbers, or sets). We use mainly lower case Latin letters (with or without subscripts) for variables.
(b) **logical operators** which are "$\neg$" (*not*), "$\wedge$" (*and*), "$\vee$" (*or*), and "$\rightarrow$" (*implies*).
(c) **Logical quantifiers** which are the *existential quantifier* "$\exists$" (*there is* or *there exists*) and the *universal quantifier* "$\forall$" (*for all* or *for each*), where quantification is restricted to objects only and not to formulae or sets of objects (but the objects themselves may be sets).
(d) **Equality symbol** "$=$", which stands for the particular binary *equality relation*.
(e) **Constant symbols** like the number $\mathtt{0}$ in Peano Arithmetic, or the neutral element $\mathtt{e}$ in Group Theory. Constant symbols stand for fixed individual objects in the domain.
(f) **Function symbols** such as $\circ$ (the operation in Group Theory), or $+, \cdot, \mathtt{s}$ (the operations in Peano Arithmetic). Function symbols stand for fixed functions taking objects as arguments and returning objects as values. With each function

symbol we associate a positive natural number, its co-called "arity" (*e.g.*, "$\circ$" is a 2-ary or binary function, and the successor operation "$\mathtt{s}$" is a 1-ary or unary function). More formally, to each function symbol F we adjoin a fixed FINITE string of place holders $\mathsf{x}\cdots\mathsf{x}$ and write $F\mathsf{x}\cdots\mathsf{x}$.

(g) **Relation symbols** or **predicate constants** (such as $\in$ in Set Theory) stand for fixed relations between (or properties of) objects in the domain. Again we associate an "arity" with each relation symbol (*e.g.*, "$\in$" is a binary relation). More formally, to each relation symbol R we adjoin a fixed FINITE string of place holders $\mathsf{x}\cdots\mathsf{x}$ and write $R\mathsf{x}\cdots\mathsf{x}$.

The symbols in (a)–(d) form the core of the alphabet and are called **logical symbols**. The symbols in (e)–(g) depend on the specific topic we are investigating and are called **non-logical symbols**. The set of non-logical symbols which are used in order to formalise a certain mathematical theory is called the **signature** (or **language**) of this theory, denoted by $\mathscr{L}$, and *formulae* which are formulated in a language $\mathscr{L}$ are usually called $\mathscr{L}$-formulae. For example if we investigate groups, then the only non-logical symbols we use are "$\mathtt{e}$" and "$\circ$", thus, $\mathscr{L} = \{\mathtt{e}, \circ\}$ is the language of Group Theory.

Terms & Formulae

With the symbols of our alphabet we can now start to compose names. In the language of First-Order Logic, these names are called called *terms*.

Terms. A string of symbols is a **term**, if it results from applying FINITELY many times the following rules:

(T0) Each variable is a term.
(T1) Each constant symbol is a term.
(T2) If $\tau_1, \ldots, \tau_n$ are any terms which we have already built and $F\mathsf{x}\cdots\mathsf{x}$ is an n-ary function symbol, then $F\tau_1\cdots\tau_n$ is a term (each place holder x is replaced with a term).

Terms of the form (T0) or (T1) are the most basic terms, expressions we have, and since every term is built up from such terms, they are called **atomic terms**. In order to define rule (T2) we had to use variables for terms, but since the variables of our alphabet stand just for objects of the domain and not for terms or other objects of the formal language, we had to introduce new symbols. For these new symbols, which do not belong to the alphabet of the formal language, we have chosen Greek letters. In fact, we shall mainly use Greek letters for variables which stand for objects of the formal language, also to emphasise the distinction between the formal language and the metalanguage However, we shall use the Latin letters F & R as variables for function and relation symbols respectively.

Note that this recursive definition of terms allows us to use the following principle: If we want to prove that all terms satisfy some property Φ, then one has to prove

- All variables satisfy Φ.
- Each constant symbol satisfies Φ.
- If some terms $\tau_1, \dots, \tau_n$ satisfy Φ, then so does $F\tau_1, \cdots, \tau_n$ for every n-ary function symbol F.

We call this principle **induction on the term construction**.

To make terms, relations, and other expressions in the formal language easier to read, it is convenient to introduce some more symbols, like brackets and commas, to our alphabet. For example we usually write $F(\tau_1, \dots, \tau_n)$ rather than $F\tau_1 \cdots \tau_n$.

To some extent, terms correspond to names, since they denote objects of the domain under consideration. Like real names, they are not statements and cannot express or describe possible relations between objects. So, the next step is to build sentences, or more precisely *formulae*, with these terms.

Formulae. A string of symbols is called a **formula**, if it results from applying F I N I T E L Y many times the following rules:

(F0) If τ_1 and τ_2 are terms, then $\tau_1 = \tau_2$ is a formula.
(F1) If $\tau_1, \dots, \tau_n$ are any terms and $R\mathsf{x}\cdots\mathsf{x}$ is any non-logical n-ary relation symbol, then $R\tau_1 \cdots \tau_n$ is a formula.
(F2) If φ is any formula which we have already built, then $\neg\varphi$ is a formula.
(F3) If φ and ψ are formulae which we have already built, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, and $(\varphi \rightarrow \psi)$ are formulae. (To avoid the use of brackets one could write these formulae for example in *Polish notation*, *i.e.*, $\wedge\varphi\psi$, $\vee\varphi\psi$, *et cetera*.)
(F4) If φ is a formula which we have already built, and ν is an arbitrary variable, then $\exists\nu\varphi$ and $\forall\nu\varphi$ are formulae.

Formulae of the form (F0) or (F1) are the most basic expressions we have, and since every formula is a logical connection or a quantification of these formulae, they are called **atomic formulae**.

In the same way as for terms, a property Φ is satisfied by all formulae, if we check the follwoing:

- All atomic formulae satisfy Φ.
- If φ and ψ satisfy Φ and ν is a variable, then so do $\neg\varphi, \varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$, $\exists\nu\varphi$ and $\forall\nu\varphi$.

In accordance with the corresponding principle for terms, we denote this as **induction on the term construction**.

For binary relations $R\mathsf{xx}$ it is convenient to write xRy instead of $R(x, y)$. For example we write $x \in y$ instead of $\in(x, y)$, and we write $x \notin y$ rather than $\neg(x \in y)$.

If a formula φ is of the form $\exists x\psi$ or of the form $\forall x\psi$ (for some formula ψ) and x occurs in ψ, then we say that x is in the *range* of a logical quantifier. Every occurrence of a variable x in a formula φ is said to be **bound** by the innermost quantifier in whose range it occurs. If an occurrence of x is not in the range of a quantifier, it is said to be **free**. Notice that it is possible that a variable occurs in a given formula at a certain place bound and at another place free. For example, in the formula $\exists z(x = z) \wedge \forall x(x = y)$, the variable x occurs bound and free, whereas z occurs just bound and y occurs just free. However, one can always rename the bound variables occurring in a given formula φ such that each variable in φ is either bound or free (the rules for this procedure are given later). For a formula φ, the set of variables occurring free in φ is denoted by $\mathrm{free}(\varphi)$. A formula φ is a **sentence** (or a **closed formula**) if it contains no free variables (*i.e.*, $\mathrm{free}(\varphi) = \emptyset$). For example $\forall x(x = x)$ is a sentence but $(x = x)$ is just a formula.

In analogy to this definition we say that a term is a **closed term** if it contains no variables. Obviously, the only terms which are closed are the constant symbols and the function symbols followed by closed terms.

Sometimes it is useful to indicate explicitly which variables occur free in a given formula φ, and for this we usually write $\varphi(x_1, \dots, x_n)$ to indicate that $\{x_1, \dots, x_n\} \subseteq \mathrm{free}(\varphi)$.

If φ is a formula, ν a variable, and τ a term, then $\varphi(\nu/\tau)$ is the formula we get after replacing all *free* instances of the variable ν by τ. The process to obtain the formula $\varphi(\nu/\tau)$ is called **substitution**. Now, a substitution is **admissible** *iff* no free occurrence of ν in φ is in the range of a quantifier that binds any variable which appears in τ (*i.e.*, for each variable $\tilde{\nu}$ appearing in τ, no place where ν occurs free in φ is in the range of "$\exists\tilde{\nu}$" or "$\forall\tilde{\nu}$"). For example, if $x \notin \mathrm{free}(\varphi)$, then $\varphi(x/\tau)$ is admissible for any term τ. In this case, the formulae φ and $\varphi(x/\tau)$ are identical which we express by $\varphi \equiv \varphi(x/\tau)$. In general, we use the symbol "$\equiv$" in the metalanguage to denote equality of strings of symbols of the formal language. Furthermore, if φ is a formula and the substitution $\varphi(x/\tau)$ is admissible, then we write just $\varphi(\tau)$ instead of $\varphi(x/\tau)$. To express this we write $\varphi(\tau) :\equiv \varphi(x/\tau)$, where we use ":$\equiv$" in the metalanguage to define symbols (or strings of symbols) of the formal language.

So far we have letters, and we can build names and sentences. However, these sentences are just strings of symbols without any inherent meaning. Later we shall interpret formulae in the intuitively natural way by giving the symbols the intended meaning (*e.g.*, "$\wedge$" meaning "and", "$\forall x$" meaning "for all x", *et cetera*). But before we shall do so, let us stay a little bit longer on the syntactical side—nevertheless, one should consider the formulae also from a semantical point of view.

Axioms

Below we shall label certain formulae or types of formulae as **axioms**, which are used in connection with *inference rules* in order to derive further formulae. From a semantical point of view we can think of axioms as "true" statements from which we deduce or prove further results. We distinguish two types of axiom, namely *logical axioms* and *non-logical axioms* (which will be discussed later). A **logical axiom** is a sentence or formula φ which is universally valid (*i.e.*, φ is true in any possible universe, no matter how the variables, constants, *et cetera*, occurring in φ are interpreted). Usually one takes as logical axioms some minimal set of formulae that is sufficient for deriving all universally valid formulae (such a set is given below).

If a symbol is involved in an axiom which stands for an arbitrary relation, function, or even for a first-order formula, then we usually consider the statement as an **axiom schema** rather than a single axiom, since each instance of the symbol represents a single axiom. The following list of axiom schemata is a system of logical axioms.

Let φ, φ_1, φ_2, and ψ, be arbitrary first-order formulae:

L_0: $\varphi \vee \neg\varphi$,
L_1: $\varphi \to (\psi \to \varphi)$,
L_2: $(\psi \to (\varphi_1 \to \varphi_2)) \to ((\psi \to \varphi_1) \to (\psi \to \varphi_2))$,
L_3: $(\varphi \wedge \psi) \to \varphi$,
L_4: $(\varphi \wedge \psi) \to \psi$,
L_5: $\varphi \to (\psi \to (\psi \wedge \varphi))$,
L_6: $\varphi \to (\varphi \vee \psi)$,
L_7: $\psi \to (\varphi \vee \psi)$,
L_8: $(\varphi_1 \to \varphi_3) \to ((\varphi_2 \to \varphi_3) \to ((\varphi_1 \vee \varphi_2) \to \varphi_3))$,
L_9: $\neg\varphi \to (\varphi \to \psi)$.

If τ is a term, ν a variable, and the substitution which leads to $\varphi(\nu/\tau)$ is admissible, then:

L_{10}: $\forall\nu\varphi(\nu) \to \varphi(\tau)$,
L_{11}: $\varphi(\tau) \to \exists\nu\varphi(\nu)$.

If ψ is a formula and ν a variable such that $\nu \notin \mathrm{free}(\psi)$, then:

L_{12}: $\forall\nu(\psi \to \varphi(\nu)) \to (\psi \to \forall\nu\varphi(\nu))$,
L_{13}: $\forall\nu(\varphi(\nu) \to \psi) \to (\exists\nu\varphi(\nu) \to \psi)$.

What is not covered yet is the symbol "$=$", so, let us have a closer look at the binary equality relation. The defining properties of equality can already be found in Book VII, Chapter 1 of Aristotle's *Topics* [2], where one of the rules to decide whether two things are the same is as follows: ... *you should look at every possible predicate of each of the two terms and at the things of which they are predicated and see whether there is any discrepancy anywhere. For anything which is predicated of the one ought also to be predicated of the other, and of anything of which the one is a predicate the other also ought to be a predicate*.

In our formal system, the binary equality relation is defined by the following three axioms.

If $\tau, \tau_1, \ldots, \tau_n, \tau'_1, \ldots, \tau'_n$ are any terms, R an n-ary relation symbol (*e.g.*, the binary relation symbol "$=$"), and F an n-ary function symbol, then:

L_{14}: $\tau = \tau$,
L_{15}: $(\tau_1 = \tau'_1 \wedge \cdots \wedge \tau_n = \tau'_n) \to (R(\tau_1, \ldots, \tau_n) \to R(\tau'_1, \ldots, \tau'_n))$,
L_{16}: $(\tau_1 = \tau'_1 \wedge \cdots \wedge \tau_n = \tau'_n) \to (F(\tau_1, \ldots, \tau_n) = F(\tau'_1, \ldots, \tau'_n))$.

Finally, we define the logical operator "$\leftrightarrow$" and the binary relation symbol "$\neq$" by stipulating

$$\varphi \leftrightarrow \psi \quad :\Longleftrightarrow \quad (\varphi \to \psi) \wedge (\psi \to \varphi)$$

$$\tau \neq \tau' \quad :\Longleftrightarrow \quad \neg(\tau = \tau')$$

where we use "$:\Longleftrightarrow$" in the metalanguage to define relations between symbols (or strings of symbols) of the formal language (*i.e.*, "$\leftrightarrow$" & "$\neq$" are just abbreviations).

This completes the list of our logical axioms. In addition to these axioms, we are allowed to state arbitrarily many *sentences*. In logic, such a (possibly empty) set of sentences is also called a **theory**, or, when the signature $\mathscr{L}$ is specified, an **$\mathscr{L}$-theory**. The elements of a theory are called **non-logical axioms**. Notice that non-logical axioms are sentences (*i.e.*, closed formulae). Examples of theories (*i.e.*, of sets of non-logical axioms) which will be discussed in this book are the axioms of Set Theory (see Part IV), the axioms of *Peano Arithmetic* PA (also known as *Number Theory*), and the axioms of *Group Theory* GT, which we discuss first.

GT: The language of Group Theory is $\mathscr{L}_{\mathsf{GT}} = \{\mathsf{e}, \circ\}$, where "e" is a constant symbol and "$\circ$" is a binary function symbol.

GT_0: $\forall x \forall y \forall z (x \circ (y \circ z) = (x \circ y) \circ z)$ (*i.e.*, "$\circ$" is *associative*)
GT_1: $\forall x (\mathsf{e} \circ x = x)$ (*i.e.*, "e" is a *left-neutral* element)
GT_2: $\forall x \exists y (y \circ x = \mathsf{e})$ (*i.e.*, every element has a *left-inverse*)

PA: The language of **Peano Arithmetic** is $\mathscr{L}_{\mathsf{PA}} = \{\mathsf{0}, \mathsf{s}, +, \cdot\}$, where "0" is a constant symbol, "s" is a unary function symbol, and "+" & "$\cdot$" are binary function symbols.

PA_0: $\neg \exists x (\mathsf{s}x = \mathsf{0})$
PA_1: $\forall x \forall y (\mathsf{s}x = \mathsf{s}y \to x = y)$,
PA_2: $\forall x (x + \mathsf{0} = x)$
PA_3: $\forall x \forall y (x + \mathsf{s}y = \mathsf{s}(x + y))$
PA_4: $\forall x (x \cdot \mathsf{0} = \mathsf{0})$
PA_5: $\forall x \forall y (x \cdot \mathsf{s}y = (x \cdot y) + x)$

If φ is any $\mathscr{L}_{\mathsf{PA}}$-formula with $x \in \text{free}(\varphi)$, then:

PA_6: $\big(\varphi(\mathsf{0}) \wedge \forall x(\varphi(x) \to \varphi(\mathsf{s}(x)))\big) \to \forall x \varphi(x)$

Notice that PA_6 is an axiom schema, known as the **induction schema**, and not just a single axiom like PA_0–PA_5.

It is often convenient to add certain *defined symbols* to a given language so that the expressions get shorter or at least are easier to read. For example in Peano Arithmetic—which is an axiomatic system for the natural numbers—we usually replace for example the expression $\mathtt{s0}$ with $\mathtt{1}$ and $\mathtt{ss0}$ with $\mathtt{2}$. More formally, we define

$$\mathtt{1} :\equiv \mathtt{s0} \qquad \text{and} \qquad \mathtt{2} :\equiv \mathtt{ss0}\,.$$

Obviously, all that can be expressed in the language $\mathscr{L}_{\mathsf{PA}} \cup \{\mathtt{1}, \mathtt{2}\}$ can also be expressed in $\mathscr{L}_{\mathsf{PA}}$.

Formal Proofs

So far we have a set of logical and non-logical axioms in a certain language and can define, if we wish, as many new constants, functions, and relations as we like. However, we are still not able to deduce anything from the given axioms, since until now, we do not have *inference rules* which allow us for example to infer a certain sentence from a given set of axioms.

Surprisingly, just two **inference rules** are sufficient, namely:

MODUS PONENS (MP): $\dfrac{\varphi \to \psi,\ \varphi}{\psi}$ and GENERALISATION ($\forall$): $\dfrac{\varphi}{\forall \nu \varphi}$,

where ν is a variable which does not occur free in any non-logical axiom.

In the former case we say that ψ is obtained from $\varphi \to \psi$ and φ by MODUS PONENS, abbreviated (MP), and in the latter case we say that $\forall \nu \varphi$ (where ν can be any variable) is obtained from φ by GENERALISATION, abbreviated ($\forall$).

Using these two inference rules, we are now able to define the notion of **formal proof**: Let $\mathscr{L}$ be a signature (*i.e.*, a possibly empty set of non-logical symbols) and let $\mathbf{\Phi}$ be a possibly empty set of $\mathscr{L}$-formulae (*e.g.*, a set of axioms). An $\mathscr{L}$-formula ψ is **provable** from $\mathbf{\Phi}$ (or provable in $\mathbf{\Phi}$), denoted $\mathbf{\Phi} \vdash \psi$, if there is a F I N I T E sequence $\varphi_0, \dots, \varphi_n$ of $\mathscr{L}$-formulae such that $\varphi_n \equiv \psi$ (*i.e.*, the formulae φ_n and ψ are identical), and for all i with $i \leq n$ we have:

- φ_i is a logical axiom, or
- $\varphi_i \in \mathbf{\Phi}$, or
- there are $j, k < i$ such that $\varphi_j \equiv \varphi_k \to \varphi_i$, or
- there is a $j < i$ such that $\varphi_i \equiv \forall \nu\, \varphi_j$ for some variable ν.

If a formula ψ is not provable from $\mathbf{\Phi}$, *i.e.*, if there is no formal proof for ψ which uses just formulae from $\mathbf{\Phi}$, then we write $\mathbf{\Phi} \nvdash \psi$.

Formal proofs, even of very simple statements, can get quite long and tricky. Nevertheless, we shall give two examples:

Example 1.1. For every formula φ we have:

$$\vdash \varphi \to \varphi$$

A formal proof of $\varphi \to \varphi$ is given by

φ_0:	$(\varphi \to ((\varphi \to \varphi) \to \varphi)) \to ((\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))$	instance of L_2
φ_1:	$\varphi \to ((\varphi \to \varphi) \to \varphi)$	instance of L_1
φ_2:	$(\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi)$	from φ_0 and φ_1 by (MP)
φ_3:	$\varphi \to (\varphi \to \varphi)$	instance of L_1
φ_4:	$\varphi \to \varphi$	from φ_2 and φ_3 by (MP)

Example 1.2. We give a formal proof of $\mathsf{PA} \vdash \mathtt{1} + \mathtt{1} = \mathtt{2}$. Recall that we have defined $\mathtt{1} :\equiv \mathtt{s0}$ and $\mathtt{2} :\equiv \mathtt{ss0}$, so we need to prove $\mathsf{PA} \vdash \mathtt{s0} + \mathtt{s0} = \mathtt{ss0}$.

φ_0:	$\forall x \forall y (x + \mathtt{s}y = \mathtt{s}(x + y))$	instance of PA_3
φ_1:	$\forall x \forall y (x + \mathtt{s}y = \mathtt{s}(x + y)) \to \forall y (\mathtt{s0} + \mathtt{s}y = \mathtt{s}(\mathtt{s0} + y))$	instance of L_{10}
φ_2:	$\forall y (\mathtt{s0} + \mathtt{s}y = \mathtt{s}(\mathtt{s0} + y))$	from φ_1 and φ_0 by (MP)
φ_3:	$\forall y (\mathtt{s0} + \mathtt{s}y = \mathtt{s}(\mathtt{s0} + y)) \to \mathtt{s0} + \mathtt{s0} = \mathtt{s}(\mathtt{s0} + \mathtt{0})$	instance of L_{10}
φ_4:	$\mathtt{s0} + \mathtt{s0} = \mathtt{s}(\mathtt{s0} + \mathtt{0})$	from φ_3 and φ_2 by (MP)
φ_5:	$\forall x (x + \mathtt{0} = x)$	instance of PA_2
φ_6:	$\forall x (x + \mathtt{0} = x) \to \mathtt{s0} + \mathtt{0} = \mathtt{s0}$	instance of L_{10}
φ_7:	$\mathtt{s0} + \mathtt{0} = \mathtt{s0}$	from φ_6 and φ_5 by (MP)
φ_8:	$\mathtt{s0} + \mathtt{0} = \mathtt{s0} \to \mathtt{s}(\mathtt{s0} + \mathtt{0}) = \mathtt{ss0}$	instance of L_{16}
φ_9:	$\mathtt{s}(\mathtt{s0} + \mathtt{0}) = \mathtt{ss0}$	from φ_8 and φ_7 by (MP)
φ_{10}:	$\mathtt{s0} + \mathtt{s0} = \mathtt{s0} + \mathtt{s0}$	instance of L_{14}
φ_{11}:	$\varphi_{10} \to (\varphi_9 \to (\varphi_{10} \wedge \varphi_9))$	instance of L_5
φ_{12}:	$\varphi_9 \to (\varphi_{10} \wedge \varphi_9)$	from φ_{11} and φ_{10} by (MP)
φ_{13}:	$\varphi_{10} \wedge \varphi_9$	fromφ_{12} and φ_9 by (MP)
φ_{14}:	$(\varphi_{10} \wedge \varphi_9) \to (\mathtt{s0} + \mathtt{s0} = \mathtt{s}(\mathtt{s0} + \mathtt{0}) \to \mathtt{s0} + \mathtt{s0} = \mathtt{ss0})$	instance of L_{15}
φ_{15}:	$\mathtt{s0} + \mathtt{s0} = \mathtt{s}(\mathtt{s0} + \mathtt{0}) \to \mathtt{s0} + \mathtt{s0} = \mathtt{ss0}$	from φ_{14} and φ_{13} by (MP)
φ_{16}:	$\mathtt{s0} + \mathtt{s0} = \mathtt{ss0}$	from φ_{15} and φ_4 by (MP)

In Chapter 2 we will introduce some techniques which allow us to simplify formal proofs such as the one presented above.

Tautologies & Logical Equivalence

We say that two formulae φ and ψ are **logically equivalent** (or just **equivalent**), denoted $\varphi \Leftrightarrow \psi$, if $\vdash \varphi \leftrightarrow \psi$. More formally:

$$\varphi \Leftrightarrow \psi \quad :\Longleftrightarrow \quad \vdash \varphi \leftrightarrow \psi$$

In other words, if $\varphi \Leftrightarrow \psi$, then—from a logical point of view—φ and ψ state exactly the same, and therefore we could call $\varphi \leftrightarrow \psi$ a tautology, which means *saying the same thing twice*. Indeed, in logic, a formula φ is a **tautology** if $\vdash \varphi$. Thus, the formulae φ and ψ are equivalent if and only if $\varphi \leftrightarrow \psi$ is a tautology.

Example 1.3. For every formula φ we have:

$$\varphi \Leftrightarrow \varphi$$

This follows directly from Example 1.1, since $\varphi \leftrightarrow \varphi$ is simply an abbreviation for $(\varphi \to \varphi) \wedge (\varphi \to \varphi)$:

φ_0:	$\varphi \to \varphi$	Example 1.1
φ_1:	$(\varphi \to \varphi) \to ((\varphi \to \varphi) \to (\varphi \leftrightarrow \varphi))$	instance of L_5
φ_2:	$(\varphi \to \varphi) \to (\varphi \leftrightarrow \varphi)$	from φ_0 and φ_1 by (MP)
φ_3:	$\varphi \leftrightarrow \varphi$	from φ_0 and φ_2 by (MP)

Example 1.4. For every formula φ we have

$$\varphi \Leftrightarrow \neg\neg\varphi.$$

By applying L_5 as in Example 1.3 one can easily check that it suffices to prove separately that $\varphi \to \neg\neg\varphi$ and $\neg\neg\varphi \to \varphi$ are tautologies. We only prove the first statement, the second one is handled in Example 2.1.

φ_0:	$(\neg\varphi \to (\varphi \to \neg\neg\varphi) \to ((\neg\neg\varphi \to (\varphi \to \neg\neg\varphi)) \to$ $((\neg\varphi \vee \neg\neg\varphi) \to (\varphi \to \neg\neg\varphi)))$	instance of L_8
φ_1:	$\neg\varphi \to (\varphi \to \neg\neg\varphi)$	instance of L_9
φ_2:	$(\neg\neg\varphi \to (\varphi \to \neg\neg\varphi)) \to ((\neg\varphi \vee \neg\neg\varphi) \to (\varphi \to \neg\neg\varphi))$	from φ_0 and φ_1 by (MP)
φ_3:	$\neg\neg\varphi \to (\varphi \to \neg\neg\varphi)$	instance of L_1
φ_4:	$(\neg\varphi \vee \neg\neg\varphi) \to (\varphi \to \neg\neg\varphi)$	from φ_2 and φ_2 by (MP)
φ_5:	$\neg\varphi \vee \neg\neg\varphi$	instance of L_0
φ_6:	$\varphi \to \neg\neg\varphi$	from φ_4 and φ_5 by (MP)

Example 1.5. Commutativity and associativity of $\wedge$ and $\vee$ are tautological, i.e. $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$ and $\varphi \wedge (\psi \wedge \chi) \Leftrightarrow (\varphi \wedge \psi) \wedge \chi$; and similarly for $\vee$. Again, we omit the proof since it will be trivial once we have proved the DEDUCTION THEOREM (THEOREM 2.1). This legitimizes the notations $\varphi_0 \wedge \ldots \wedge \varphi_n$ and $\varphi_0 \vee \ldots \vee \varphi_n$ respectively for $\varphi_0 \wedge (\varphi_1 \wedge (\ldots \wedge \varphi_n) \ldots)$ and $\varphi_0 \vee (\varphi_1 \vee (\ldots \vee \varphi_n) \ldots)$ respectively.

In Appendix 17 there is a list of tautologies which will be frequently used in formal proofs. Note that it follows from Exercise 1.4 that $\Leftrightarrow$ defines an equivalence relation on all $\mathscr{L}$-formulae for some giveen signature $\mathscr{L}$. Moreover, it even defines a congruence relation, *i.e.*equivalence is closed under all logical operations. More precisely, if $\varphi \Leftrightarrow \varphi'$ and $\psi \Leftrightarrow \psi'$, then

$$\neg\varphi \Leftrightarrow \neg\varphi'$$
$$\varphi \circ \psi \Leftrightarrow \varphi' \circ \psi'$$
$$\exists\!\!\!\forall x\varphi \Leftrightarrow \exists\!\!\!\forall x\varphi',$$

where $\circ$ stands for either $\wedge$, $\vee$, or $\to$, and $\exists\!\!\!\forall$ stands for either $\exists$ or $\forall$. A proof of these statements will be easier once we have proved THEOREM 2.1.

The observation above enables us to replace subformulae of a given formula φ by equivalent formulae so that the resulting formula is equivalent to φ.

THEOREM 1.1 (SUBSTITUTION THEOREM). *Let φ be a formula and let α be a subformula of φ. Let ψ be the formula obtained from φ by replacing one or multiple occurences of α by some formula β. Then we have*

$$\alpha \Leftrightarrow \beta \quad \Longrightarrow \quad \varphi \Leftrightarrow \psi.$$

Proof. We prove the theorem by induction on the recursive construction of the formula φ. If φ is an atomic formula or if α is φ, then the statement is trivial. If φ is a composite formula, then we use the observation that $\Leftrightarrow$ is a congruence relation: For example, if φ is of the form $\neg\varphi'$, then, since α is not φ, there is a formula ψ' such that $\psi \equiv \neg\psi'$. Now inductively we may assume that $\varphi' \Leftrightarrow \psi'$. But then $\neg\varphi' \Leftrightarrow \neg\psi'$ as desired. The other cases can be checked in a similar way. $\dashv$

THEOREM 1.2 (3-SYMBOLS). *For every each φ there is an equivalent formula ψ which contains only "$\neg$" and "$\wedge$" as logical operators and "$\exists$" as quantifier.*

Proof. By THEOREM 1.1, it suffices to prove the equivalences

$$\begin{aligned} \varphi \vee \psi &\Leftrightarrow \neg(\neg\varphi \wedge \neg\psi) \\ \varphi \rightarrow \psi &\Leftrightarrow \neg\varphi \vee \psi \\ \forall x\varphi &\Leftrightarrow \neg\exists x\neg\varphi. \end{aligned}$$

The proof of these equivalences is left as an exercise (see Exercise 1.2). Note that THEOREM 2.1 as well as the methods of proof introduced in Chapter 2 will simplify the proofs to a great extent. $\dashv$

As a consequence of THEOREM 1.2 one could simply both the alphabet and the logical axioms. Nevertheless, we do not wish to do so, since this would also decrease the readability of formulae.

EXERCISES

1.0 Something with terms.

1.1 The equality relation is transitive.

1.2 Prove the equivalences in the proof of Theorem 1.2.
Hint: Prove first the tautologies (K.0), (L.0), and (T)

1.3 Show that propositional formulae can be written only with nand/nor.

1.4 Show that $\Leftrightarrow$ defines an equivalence relation.

1.5 Something with the conjunctive (and/or disjunctive) normal form.

1.6 L0 ist unabhaengig von L1-L9