Department of Mathematics, Government Arts College (Autonomous), Coimbatore-18

Year	Subject Title	Sem.	Sub Code
2018 -19 Onwards	DISCRETE MATHEMATICS FOR COMPUTER SCIENCE	II	

UNIT: I

MATHEMATICAL LOGIC: Propositions and Logical Operators – Truth table – Tautology – Contradiction – Equivalence and Implication – Normal forms (DNF, CNF, PDNF and PCNF).

(Chapter I - Sections: 1.1 to 1.3)

1

MATHEMATICAL LOGIC

INTRODUCTION

One of the main aims of logic is to provide rules by which one can determine whether any particular argument or reasoning is valid (correct).

Logic is concerned with all kinds of reasonings, whether they be legal arguments or mathematical proofs or conclusions in a scientific theory based upon a set of hypotheses. Because of the diversity of their application, these rules, called rules of inference, must be stated in general terms and must be independent of any particular argument or discipline involved. These rules should also be independent of any particular language used in the arguments. More precisely, in logic we are concerned with the forms of the arguments rather than with the arguments themselves. Like any other theory in science, the theory of inference is formulated in such a way that we should be able to decide about the validity of an argument by following the rules mechanically and independently of our own feelings about the argument. Of course, to proceed in this manner requires that the rules be stated unambiguously.

Any collection of rules or any theory needs a language in which these rules or theory can be stated. Natural languages are not always precise enough. They

are also ambiguous and, as such, are not suitable for this purpose. It is therefore necessary first to develop a formal language called the *object language*. A formal language is one in which the syntax is well defined. In fact, every scientific discipline develops its own object language which consists of certain well-defined terms and well-specified uses of these terms. The only difference between logic and other disciplines is that in other disciplines we are concerned with the use of the object language while in logic we are as interested in analyzing our object language as we are in using it. In fact, in the first half of this chapter we shall be concerned with the development and analysis of an object language without considering its use in the theory of inference. This study has important applications in the design of computers and several other two-state devices, as is shown in Sec. 1-2.15. We emphasize this part of logic because the study of formal languages constitutes an important part in the development of means of communication with computing machines. This study is followed by the study of inference theory in Sec. 1-4. It soon becomes apparent that the object language developed thus far is very limited, and we cannot include some very simple argument forms in our inference theory. Therefore, in Sec. 1-5 we expand our object language to include predicates, and then in Sec. 1-6 we discuss the inference theory of predicate logic.

In order to avoid ambiguity, we use symbols which have been clearly defined in the object languages. An additional reason to use symbols is that they are easy to write and manipulate. Because of this use of symbols, the logic that we shall study is also called *symbolic logic*. Our study of the object language requires the use of another language. For this purpose we can choose any of the natural languages. In this case our choice is English, and so the statements about the object language will be made in English. This natural language (English) will then be called our *metalanguage*. Certain inherent difficulties in this procedure could be anticipated, because we wish to study a precise language while using another language which is not so precise.

1-1 STATEMENTS AND NOTATION

In this section we introduce certain basic units of our object language called primary (primitive, atomic) statements. We begin by assuming that the object language contains a set of declarative sentences which cannot be further broken down or analyzed into simpler sentences. These are the primary statements. Only those declarative sentences will be admitted in the object language which have one and only one of two possible values called "truth values." The two truth values are *true* and *false* and are denoted by the symbols T and F respectively. Occasionally they are also denoted by the symbols 1 and 0. The truth values have nothing to do with our feelings of the truth or falsity of these admissible sentences because these feelings are subjective and depend upon context. For our purpose, it is enough to assume that it is *possible* to assign one and only one of the two possible values to a declarative sentence. We are concerned in our study with the *effect* of assigning any particular truth value to declarative sentences rather than with the *actual* truth value of these sentences. Since we admit only two possible truth values, our logic is sometimes called a *two-valued logic*.

We develop a mechanism by which we can construct in our object language other declarative sentences having one of the two possible truth values. Note that we do not admit any other types of sentence, such as exclamatory, interrogative, etc., in the object language.

Declarative sentences in the object language are of two types. The first type includes those sentences which are considered to be primitive in the object language. These will be denoted by distinct symbols selected from the capital letters $A, B, C, \ldots, P, Q, \ldots$, while declarative sentences of the second type are obtained from the primitive ones by using certain symbols, called connectives, and certain punctuation marks, such as parentheses, to join primitive sentences. In any case, all the declarative sentences to which it is possible to assign one and only one of the two possible truth values are called *statements*. These statements which do not contain any of the connectives are called *atomic* (*primary, primitive*) *statements*.

We shall now give examples of sentences and show why some of them are not admissible in the object language and, hence, will not be symbolized.

1 Canada is a country.
2 Moscow is the capital of Spain.
3 This statement is false.
4 $1 + 101 = 110$.
5 Close the door.
6 Toronto is an old city.
7 Man will reach Mars by 1980.

Obviously Statements (1) and (2) have truth values *true* and *false* respectively. Statement (3) is not a statement according to our definition, because we cannot properly assign to it a definite truth value. If we assign the value *true*, then Sentence (3) says that Statement (3) is false. On the other hand, if we assign it the value *false*, then Sentence (3) implies that Statement (3) is true. This example illustrates a semantic paradox. In (4) we have a statement whose truth value depends upon the context; viz., if we are talking about numbers in the decimal system, then it is a false statement. On the other hand, for numbers in binary, it is a true statement. The truth value of a statement often depends upon its context, which is generally unstated but nonetheless understood. We shall soon see that we are not going to be preoccupied with the actual truth value of a statement. We shall be interested only in the fact that it *has* a truth value. In this sense (4), (6), and (7) are all statements. Note that Statement (6) is considered true in some parts of the world and false in certain other parts. The truth value of (7) could be determined only in the year 1980, or earlier if a man reaches Mars before that date. But this aspect is not of interest to us. Note that (5) is not a statement; it is a command.

Once we know those atomic statements which are admissible in the object language, we can use symbols to denote them. Methods of constructing and analyzing statements constructed from one or more atomic statements are discussed in Sec. 1-2, while the method of symbolizing atomic statements will be described here after we discuss some conventions regarding the use and mention of names in statements.

It is customary to use the *name* of an object, not the object itself, when making a statement about the object. As an example, consider the statement

8 This table is big.

The expression "this table" is used as a name of the object. The actual object, namely a particular table, is not used in the statement. It would be inconvenient to put the actual table in place of the expression "this table." Even for the case of small objects, where it may be possible to insert the actual object in place of its name, this practice would not permit us to make two simultaneous statements about the same object without using its name at one place or the other. For this reason it may be agreed that a statement about an object would contain never the object itself but only its name. In fact, we are so familiar with this convention that we take it for granted.

Consider, now, a situation in which we wish to discuss something about a *name*, so that the name is the object about which a statement is to be made. According to the rule just stated, we should use not the name itself in the statement but some name of that name. How does one give a name to a name? A usual method is to enclose the name in quotation marks and to treat it as a name for the name. For example, let us look at the following two statements.

9 Clara is smart.

10 "Clara" contains five letters.

In (9) something is said about a person whose name is Clara. But Statement (10) is not about a person but about a name. Thus "Clara" is used as a name of this name. By enclosing the name of a person in quotation marks it is made clear that the statement made in (10) is about a name and not about a person.

This convention can be explained alternatively by saying that we *use* a certain word in a sentence when that word serves as the name of an object under consideration. On the other hand, we *mention* a word in a sentence when that word is acting not as the name of an object but as the name of the word itself. To "mention" a word means that the word itself has been converted into an object of our consideration.

Throughout the text we shall be making statements not only about what we normally consider objects but also about other statements. Thus it would be necessary to name the statements under consideration. The same device used for naming names could also be used for naming statements. A statement enclosed in quotation marks will be used as the *name* of the statement. More generally, any expression enclosed in quotation marks will be used as the name of that expression. In other words, any expression that is mentioned is placed in quotation marks. The following statement illustrates the above discussion.

11 "Clara is smart" contains "Clara."

Statement (11) is a statement about Statement (9) and the word "Clara." Here Statement (9) was named first by enclosing it in quotation marks and then by using this name in (11) along with the name "Clara"!

In this discussion we have used certain other devices to name statements. One such device is to display a statement on a line separated from the main text. This method of display is assumed to have the same effect as that obtained

by using quotation marks to delimit a statement within the text. Further, we have sometimes numbered these statements by inserting a number to the left of the statement. In a later reference this number is used as a name of the statement. This number is written within the text without quotation marks. Such a display and the numbering of statements permit some reduction in the number of quotation marks. Combinations of these different devices will be used throughout the text in naming statements. Thus the statement

12 "Clara is smart" is true.

could be written as "(9) is true," or equivalently,

12a (9) is true.

A particular person or an object may have more than one name. It is an accepted principle that one may substitute for the name of an object in a given statement any other name of the same object without altering the meaning of the statement. This principle was used in Statements (12) and (12*a*).

We shall be using the name-forming devices just discussed to form the names of statements. Very often such distinctions are not made in mathematical writings, and generally the difference between the name and the object is assumed to be clear from the context. However, this practice sometimes leads to confusion.

A situation analogous to the name-object concept just discussed exists in many programming languages. In particular, the distinction between the name of a variable and its value is frequently required when a procedure (function or subroutine) is invoked (called). The arguments (also called actual parameters) in the statement which invokes the procedure are associated with the (formal) parameters of the procedure either by name or by value. If the association is made by value, then only the value of an argument is passed to its corresponding parameter. This procedure implies that we cannot change the value of the argument from within the function since it is not known where this argument is stored in the computer memory. On the other hand, a call-by-name association makes the name or address of the argument available to the procedure. Such an association allows the value of an argument to be changed by instructions in the procedure. We shall now discuss how call-by-name and call-by-value associations are made in a number of programming languages.

In certain versions of FORTRAN compilers (such as IBM's FORTRAN H and G) the name of an argument, not its value, is passed to a function or subroutine. This convention also applies to the case of an argument's being a constant. The address of a constant (stored in some symbol table of the compiler) is passed to the corresponding parameter of the function. This process could lead to catastrophic results. For example, consider the simple function FUN described by the following sequence of statements:

```
INTEGER FUNCTION FUN(I)
I=5
FUN=I
RETURN
END
```

Suppose that the main program, which invokes FUN, consists of the trivial statements

```
      K = 3
      J = FUN(K) * 3
      L = FUN(3) * 3
      PRINT 10, J, L
   10 FORMAT(1H , I3, I3)
      STOP
      END
```

This program yields values of 15 and 25 for variables J and L respectively. In the evaluation of J, the address of K is known within the function. K is changed in the function to a value of 5 by the statement I = 5. The functional value returned by the function is 5, and a value of 15 for J results. The computation of L, however, is quite different. The address of 3 is passed to the function. Since the corresponding parameter I is changed to 5, the value of 3 in the symbol table in the main program will also be changed to 5. Note that since all references to the symbol table entry for constant 3 were made at compile time, all such future references in the remainder of the main program still refer to that entry or location, but the value will now be 5, not 3. More specifically, the name 3 in the right operand of the multiplication of L has a value of 5.

In other versions of FORTRAN compilers, such results are prevented by creating a dummy variable for each argument that is a constant. These internal (dummy) variables are not accessible to the programmer. A change in parameter corresponding to a dummy variable changes the value of that variable, but it does not change the value of the original argument from which it was constructed.

WATFIV permits the passing of arguments by value by merely enclosing such arguments in slashes. For example, in the function call

```
TEST(I,/K/,5)
```

the value of K is passed to the function TEST.

In PL/I arguments can be passed by value or by name. An argument is passed by value if it is enclosed within parentheses; otherwise it is passed by name. In the function call

```
TEST(I,(K),5)
```

the arguments I and K are passed by name and by value respectively.

As mentioned earlier, we shall use the capital letters A, B, ..., P, Q, ... (with the exception of T and F) as well as subscripted capital letters to represent statements in symbolic logic. As an illustration, we write

13 P: It is raining today.

In Statement (13) we are including the information that "P" is a statement in symbolic logic which corresponds to the statement in English, "It is raining today." This situation is similar to the translation of the same statement into French as "Aujourd'hui il pleut." Thus "P" in (13)—"It is raining today"—and "Aujourd'hui il pleut" are the names of the same statement. Note that "P" and not P is used as the name of a statement.

1-2 CONNECTIVES

The notions of a statement and of its truth value have already been introduced. In the case of simple statements, their truth values are fairly obvious. However, it is possible to construct rather complicated statements from simpler statements by using certain connecting words or expressions known as "sentential connectives." Several such connectives are used in the English language. Because they are used with a variety of meanings, it is necessary to define a set of connectives with definite meanings. It is convenient to denote these new connectives by means of symbols. We define these connectives in this section and then develop methods to determine the truth values of statements that are formed by using them. Various properties of these statements and some relationships between them are also discussed. In addition, we show that the statements along with the connectives define an algebra that satisfies a set of properties. These properties enable us to do some calculations by using statements as objects. The algebra developed here has interesting and important applications in the field of switching theory and logical design of computers, as is shown in Sec. 1-2.15. Some of these results are also used in the theory of inference discussed in Sec. 1-4.

The statements that we consider initially are simple statements, called *atomic* or *primary statements*. As already indicated, new statements can be formed from atomic statements through the use of sentential connectives. The resulting statements are called *molecular* or *compound statements*. Thus the atomic statements are those which do not have any connectives.

In our everyday language we use connectives such as "and," "but," "or," etc., to combine two or more statements to form other statements. However, their use is not always precise and unambiguous. Therefore, we will not symbolize these connectives in our object language; however, we will define connectives which have some resemblance to the connectives in the English language.

The idea of using the capital letters P, Q, ..., P_1, P_2, ... to denote statements was already introduced in Sec. 1-1. Now the same symbols, namely, the capital letters with or without subscripts, will also be used to denote arbitrary statements. In this sense, a statement "P" either denotes a particular statement or serves as a placeholder for any statement whatsoever. This dual use of the same symbol to denote either a definite statement, called a *constant*, or an arbitrary statement, called a *variable*, does not cause any confusion as its use will be clear from the context. The truth value of "P" is the truth value of the actual statement which it represents. It should be emphasized that when "P" is used as a statement variable, it has no truth value and as such does not represent a statement in symbolic logic. We understand that if it is to be replaced, then its replacement must be a statement. Then the truth value of P could be determined. It is convenient to call "P" in this case a "statement formula." We discuss the notion of "statement formula" in Sec. 1-2.4. However, in the sections that follow, we often abbreviate the term "statement formula" simply by "statement." This abbreviation keeps our discussion simple and emphasizes the meaning of the connectives introduced.

As an illustration, let

P: It is raining today.

Q: It is snowing.

and let R be a statement variable whose possible replacements are P and Q. If no replacement for R is specified, it remains a statement variable and has no truth value. On the other hand, the truth values of P and Q can be determined because they are statements.

1-2.1 Negation

The *negation* of a statement is generally formed by introducing the word "not" at a proper place in the statement or by prefixing the statement with the phrase "It is not the case that." If "P" denotes a statement, then the negation of "P" is written as "$\neg P$" and read as "not P." If the truth value of "P" is T, then the truth value of "$\neg P$" is F. Also if the truth value of "P" is F, then the truth value of "$\neg P$" is T. This definition of the negation is summarized by Table 1-2.1.

Notice that we have not used the quotation marks to denote the names of the statements in the table. This practice is in keeping with the one adopted earlier, when a statement was separated from the main text and written on a separate line. From now on we shall drop the quotation marks even within the text when we use symbolic names for the statements, except in the case where this practice may lead to confusion. We now illustrate the formation of the negation of a statement.

Consider the statement

P: London is a city.

Then $\neg P$ is the statement

$\neg P$: It is not the case that London is a city.

Normally $\neg P$ can be written as

$\neg P$: London is not a city.

Although the two statements "It is not the case that London is a city" and "London is not a city" are not identical, we have translated both of them by $\neg P$. The reason is that both these statements have the same meaning in English. A given statement in the object language is denoted by a symbol, and it may correspond to several statements in English. This multiplicity happens because in a natural language one can express oneself in a variety of ways.

As an illustration, if a statement is

P: I went to my class yesterday.

then P is any one of the following

I I did not go to my class yesterday.

Table 1-2.1 TRUTH TABLE FOR NEGATION

P	$\neg P$
T	F
F	T

2 I was absent from my class yesterday.
3 It is not the case that I went to my class yesterday.

The symbol "¬" has been used here to denote the negation. Alternate symbols used in the literature are "~," a bar, or "*NOT*," so that ¬*P* is written as ~*P*, $\bar{P}$, or *NOT P*. Note that a negation is called a connective although it only modifies a statement. In this sense, negation is a unary operation which operates on a single statement or a variable. The word "operation" will be explained in Chap. 2. For the present it is sufficient to note that an operation on statements generates other statements. We have chosen "¬" to denote negation because this symbol is commonly used in the textbooks on logic and also in several programming languages, one of which will be used here.

1-2.2 Conjunction

The *conjunction* of two statements *P* and *Q* is the statement $P \wedge Q$ which is read as "*P* and *Q*." The statement $P \wedge Q$ has the truth value *T* whenever both *P* and *Q* have the truth value *T*; otherwise it has the truth value *F*. The conjunction is defined by Table 1-2.2.

EXAMPLE 1 Form the conjunction of

P: It is raining today.

Q: There are 20 tables in this room.

SOLUTION It is raining today and there are 20 tables in this room. ////

Normally, in our everyday language the conjunction "and" is used between two statements which have some kind of relation. Thus a statement "It is raining today and $2 + 2 = 4$" sounds odd, but in logic it is a perfectly acceptable statement formed from the statements "It is raining today" and "$2 + 2 = 4$."

EXAMPLE 2 Translate into symbolic form the statement

Jack and Jill went up the hill.

SOLUTION In order to write it as a conjunction of two statements, it is necessary first to paraphrase the statement as

Jack went up the hill and Jill went up the hill.

Table 1-2.2 TRUTH TABLE FOR CONJUNCTION

P	*Q*	$P \wedge Q$
T	*T*	*T*
T	*F*	*F*
F	*T*	*F*
F	*F*	*F*

If we now write

P: Jack went up the hill.

Q: Jill went up the hill.

then the given statement can be written in symbolic form as $P \wedge Q$. ////

So far we have seen that the symbol $\wedge$ is used as a translation of the connective "and" appearing in English. However, the connective "and" is sometimes used in a different sense, and in such cases it cannot be translated by the symbol $\wedge$ defined above. In order to see this difference, consider the statements:

1 Roses are red and violets are blue.
2 He opened the book and started to read.
3 Jack and Jill are cousins.

In Statement (1) the conjunction "and" is used in the same sense as the symbol $\wedge$. In (2) the word "and" is used in the sense of "and then," because the action described in "he started to read" occurs after the action described in "he opened the book." In (3) the word "and" is not a conjunction. Note that our definition of conjunction is symmetric as far as P and Q are concerned; that is to say, the truth values of $P \wedge Q$ and of $Q \wedge P$ are the same for specific values of P and Q. Obviously the truth value of (1) will not change if we write it as

Violets are blue and roses are red.

On the other hand, we cannot write (2) as

He started to read and opened the book.

These examples show that the symbol $\wedge$ has a specific meaning which corresponds to the connective "and" in general, although "and" may also be used with some other meanings. Some authors use the symbol &, or a dot, or "*AND*" to denote the conjunction. Note that the conjunction is a binary operation in the sense that it connects two statements to form a new statement.

1-2.3 Disjunction

The *disjunction* of two statements P and Q is the statement $P \vee Q$ which is read as "P or Q." The statement $P \vee Q$ has the truth value F only when both P and Q have the truth value F; otherwise it is *true*. The disjunction is defined by Table 1-2.3.

Table 1-2.3 TRUTH TABLE FOR DISJUNCTION

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

The connectives $\neg$ and $\wedge$ defined earlier have the same meaning as the words "not" and "and" in general. However, the connective $\vee$ is not always the same as the word "or" because of the fact that the word "or" in English is commonly used both as an "exclusive OR" and as an "inclusive OR." For example, consider the following statements:

1. I shall watch the game on television or go to the game.
2. There is something wrong with the bulb or with the wiring.
3. Twenty or thirty animals were killed in the fire today.

In Statement (1), the connective "or" is used in the exclusive sense; that is to say, one or the other possibility exists but not both. In (2) the intended meaning is clearly one or the other or both. The connective "or" used in (2) is the "inclusive OR." In (3) the "or" is used for indicating an approximate number of animals, and it is not used as a connective.

From the definition of disjunction it is clear that $\vee$ is "inclusive OR." The symbol $\vee$ comes from the Latin word "vel" which is the "inclusive OR." It is not necessary to introduce a new symbol for "exclusive OR," since there are other ways to express it in terms of the symbols already defined. We demonstrate this point in Sec. 1-2.14.

Normally in our everyday language, the disjunction "or" is used between two statements which have some kind of relationship between them. It is not necessary in logic that there be any relationship between them according to the definition of disjunction. The truth value of $P \vee Q$ depends only upon the truth values of P and Q. As before, it may be necessary to paraphrase given statements in English before they can be translated into symbolic form. Similarly, translations of statements from symbolic logic into statements in English may require paraphrasing in order to make them grammatically acceptable.

1-2.4 Statement Formulas and Truth Tables

We have defined the connectives $\neg$, $\wedge$, and $\vee$ so far. Other connectives will be defined subsequently. We shall occasionally distinguish between two types of statements in our symbolic language. Those statements which do not contain any connectives are called *atomic* or *primary* or *simple statements*. On the other hand, those statements which contain one or more primary statements and some connectives are called *molecular* or *composite* or *compound statements*. As an example, let P and Q be any two statements. Some of the compound statements formed by using P and Q are

$$\neg P \qquad P \vee Q \qquad (P \wedge Q) \vee (\neg P) \qquad P \wedge (\neg Q) \tag{1}$$

The compound statements given above are statement formulas derived from the statement variables P and Q. Therefore, P and Q may be called the components of the statement formulas. Observe that in addition to the connectives we have also used parentheses in some cases in order to make the formula unambiguous. We discuss the rules of constructing statement formulas in Sec. 1-2.7.

Recall that a statement formula has no truth value. It is only when the statement variables in a formula are replaced by definite statements that we get

a statement. This statement has a truth value which depends upon the truth values of the statements used in replacing the variables.

In the construction of formulas, the parentheses will be used in the same sense in which they are used in elementary arithmetic or algebra or sometimes in a computer programming language. This usage means that the expressions in the innermost parentheses are simplified first. With this convention in mind, $\neg(P \wedge Q)$ means the negation of $P \wedge Q$. Similarly $(P \wedge Q) \vee (Q \wedge R)$ means the disjunction of $P \wedge Q$ and $Q \wedge R$. $((P \wedge Q) \vee R) \wedge (\neg P)$ means the conjunction of $\neg P$ and $(P \wedge Q) \vee R$, while $(P \wedge Q) \vee R$ means the disjunction of $P \wedge Q$ and R.

In order to reduce the number of parentheses, we will assume that the negation affects as little as possible of what follows. Thus $\neg P \vee Q$ is written for $(\neg P) \vee Q$, and the negation means the negation of the statement immediately following the symbol $\neg$. On the other hand, according to our convention, $\neg(P \wedge Q) \vee R$ stands for the disjunction of $\neg(P \wedge Q)$ and R. The negation affects $P \wedge Q$ but not R.

Truth tables have already been introduced in the definitions of the connectives. Our basic concern is to determine the truth value of a statement formula for each possible combination of the truth values of the component statements. A table showing all such truth values is called the *truth table* of the formula. In Table 1-2.1 we constructed the truth table for $\neg P$. There is only one component or atomic statement, namely P, and so there are only two possible truth values to be considered. Thus Table 1-2.1 has only two rows. In Tables 1-2.2 and 1-2.3 we constructed truth tables for $P \wedge Q$ and $P \vee Q$ respectively. These statement formulas have two component statements, namely P and Q, and there are 2^2 possible combinations of truth values that must be considered. Thus each of the two tables has 2^2 rows. In general, if there are n distinct components in a statement formula, we need to consider 2^n possible combinations of truth values in order to obtain the truth table.

Two methods of constructing truth tables are shown in the following examples.

EXAMPLE 1 Construct the truth table for the statement formula $P \vee \neg Q$.

SOLUTION It is necessary to consider all possible truth values of P and Q. These values are entered in the first two columns of Table 1-2.4 for both methods. In the table which is arrived at by method 1, the truth values of $\neg Q$ are entered

Table 1-2.4*a*

P	Q	$\neg Q$	$P \vee \neg Q$
T	T	F	T
T	F	T	T
F	T	F	F
F	F	T	T

Method 1

Table 1-2.4*b*

P	Q	P	$\vee$	$\neg$	Q
T	T	T	T	F	T
T	F	T	T	T	F
F	T	F	F	F	T
F	F	F	T	T	F
Step Number		1	3	2	1

Method 2

in the third column, and the truth values of $P \vee \neg Q$ are entered in the fourth column. In method 2, as given in Table 1-2.4*b*, a column is drawn for each statement as well as for the connectives that appear. The truth values are entered step by step. The step numbers at the bottom of the table show the sequence followed in arriving at the final step. ////

EXAMPLE 2 Construct the truth table for $P \wedge \neg P$.

SOLUTION See Table 1-2.5. Note that the truth value is F for every possible truth value of P. In this special case, the truth value of $P \wedge \neg P$ is independent of the truth value of P. ////

EXAMPLE 3 Construct the truth table for $(P \vee Q) \vee \neg P$.

SOLUTION See Table 1-2.6. In this case the truth value of the formula $(P \vee Q) \vee \neg P$ is independent of the truth values of P and Q. This independence is due to the special construction of the formula, as we shall see in Sec. 1-2.8. ////

Table 1-2.5

P	$\neg P$	$P \wedge \neg P$
T	F	F
F	T	F

Method 1

P	P	$\wedge$	$\neg$	P
T	T	F	F	T
F	F	F	T	F
Step Number	1	3	2	1

Method 2

Table 1-2.6

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

Method 1

P	Q	$(P$	$\vee$	$Q)$	$\vee$	$\neg$	P
T	T	T	T	T	T	F	T
T	F	T	T	F	T	F	T
F	T	F	T	T	T	T	F
F	F	F	F	F	T	T	F
Step Number		1	2	1	3	2	1

Method 2

Observe that if the truth values of the component statements are known, then the truth value of the resulting statement can be readily determined from the truth table by reading along the row which corresponds to the correct truth values of the component statements.

EXERCISES 1-2.4

1 Using the statements

R: Mark is rich.

H: Mark is happy.

write the following statements in symbolic form:

(*a*) Mark is poor but happy.

(*b*) Mark is rich or unhappy.

(*c*) Mark is neither rich nor happy.

(*d*) Mark is poor or he is both rich and unhappy.

2 Construct the truth tables for the following formulas.

(*a*) $\neg(\neg P \vee \neg Q)$

(*b*) $\neg(\neg P \wedge \neg Q)$

(*c*) $P \wedge (P \vee Q)$

(*d*) $P \wedge (Q \wedge P)$

(*e*) $(\neg P \wedge (\neg Q \wedge R)) \vee (Q \wedge R) \vee (P \wedge R)$

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

3 For what truth values will the following statement be true? "It is not the case that houses are cold or haunted and it is false that cottages are warm or houses ugly." (*Hint*: There are four atomic statements.)

4 Given the truth values of P and Q as T and those of R and S as F, find the truth values of the following:

(*a*) $P \vee (Q \wedge R)$

(*b*) $(P \wedge (Q \wedge R)) \vee \neg((P \vee Q) \wedge (R \vee S))$

(*c*) $(\neg(P \wedge Q) \vee \neg R) \vee (((\neg P \wedge Q) \vee \neg R) \wedge S)$

1-2.5 Logical Capabilities of Programming Languages

In this section we discuss the logical connectives available in certain programming languages and how these connectives can be used to generate a truth table for a statement formula. The logical connectives discussed thus far are available in most programming languages. In PL/I, the connectives $\wedge$, $\vee$, and $\neg$ are written as &, |, and $\neg$ respectively. The truth values T and F are written as '1'B and '0'B respectively. In ALGOL the connectives are represented as we have written them, while T and F are written as **true** and **false** respectively. FORTRAN also permits the use of logical variables and expressions, and it is these facilities which are to be discussed in this section.

In FORTRAN, the truth values T and F are denoted by the logical constants .TRUE. and .FALSE. respectively. Logical variables and expressions in the language assume only one of the logical values at any given time. All logical variables must be explicitly declared as in the statement

```
LOGICAL P, Q, R
```

which declares the three variables P, Q, and R.

The statement that a relation exists between arithmetic expressions is itself an expression that has a truth value; in FORTRAN, these expressions are formulated from the following *relational operators*:

.LT.(<) .LE.(≤) .EQ.(=) .GE.(≥) .GT.(>) .NE.(≠)

For example, if P has been declared LOGICAL, the statement

```
P = 5 * 2 .LT. 17
```

assigns a value of .TRUE. to P. Similarly, if Q has been appropriately declared, the statement

```
Q = A + 5 .GE. C + D
```

assigns the value .TRUE. to Q if A + 5 is greater than or equal to C + D when the statement is executed, and the value .FALSE. otherwise.

From the truth values arising, for example, from relations, more complex logical expressions can be obtained in FORTRAN by using one or more of the three logical connectives previously discussed. The logical operators .AND., .OR., and .NOT. correspond to the symbolic logical operators $\wedge$, $\vee$, and $\neg$ respectively. The statement

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

is equivalent to the FORTRAN statement

```
P .OR. (.NOT. (Q .AND. R))
```

Unnecessary parentheses are avoided in FORTRAN by using the following precedence scheme. The arithmetic operators, with their usual order of precedence, are the highest in rank and are consequently evaluated first. All relational operators have the same rank and are evaluated after the arithmetic operators. The logical operators are the last to be evaluated, and .NOT., .AND., and .OR. is their decreasing order of precedence. Of two or more binary operators having the same precedence value in an expression, the leftmost is evaluated first; for unary operators, it is the rightmost which is evaluated first. Thus, .NOT. P .AND. Q means (.NOT. P) .AND. Q; and A + B + 5.0 .LT. C + D means ((A + B) + 5.0) .LT. (C + D).

FORTRAN has a logical "IF statement" whose form is

```
IF (logical expression) statement
```

If the logical expression in the "IF statement" is true, then the statement following the expression is executed; otherwise, it is skipped. For example, when the statement

```
IF (.NOT. P .OR. Q) GO TO 100
```

is executed, it will not transfer control to statement 100 if P and Q have the values .TRUE. and .FALSE. respectively.

Arrays of logical variables can also be used in FORTRAN. The statement

```
LOGICAL CASE(10)
```

declares a one-dimensional array of type LOGICAL consisting of 10 elements. Elements of logical arrays are referenced in the same manner as any other subscripted variable.

Consider the problem of generating all possible assignments of truth values to the logical variables P, Q, and R, as shown in Table 1-2.7. There are $2^3 = 8$ possible assignments. Notice that the truth value of the variable P remains at the same value of T or F for each of four consecutive assignments of logical values. The values of variables Q and R remain at T or F for two assignments and one assignment of logical values respectively. The value of variable R changes more frequently than the value of variable Q, and that of Q more frequently than that of P. The number of times the kth logical variable remains at a constant truth value can be easily computed and is denoted by $BASE[k]$. In the case under discussion, we have three variables, and the values can be computed as

$$BASE[k] = 2^{(3-k)} \qquad k = 1, 2, 3$$

where we have associated $BASE[1]$, $BASE[2]$, and $BASE[3]$ with variables P, Q, and R respectively.

In addition to computing the $BASE$ elements, we also need to know the number of assignments which remain to be generated with a particular logical variable remaining at the same value. For example, if we had already generated the assignments TTT and TTF, then variable P would remain at its present value of T throughout the generation of the next two assignments. This information is stored in an element denoted by $LENGTH[k]$. For variable P, $LENGTH[1]$ would have a value of 2 after generation of TTT and TTF. The $LENGTH$ values associated with variables P, Q, and R are initially the same as their corresponding $BASE$ values. Therefore, initially

$$LENGTH[k] = BASE[k] \qquad k = 1, 2, 3$$

Every time an assignment is generated, each element of $LENGTH$ is decremented by 1. When the $LENGTH$ value associated with a variable becomes zero, then the truth value of that variable is negated, and the $LENGTH$ value is reset to the $BASE$ value. The algorithm for the generation of such assignments can now be precisely formulated.

Table 1-2.7

	P	Q	R	
	T	T	T	$BASE[3]$
$BASE[1]$	T	T	F	
	T	F	T	
	T	F	F	
	F	T	T	
$BASE[2]$	F	T	F	
	F	F	T	
	F	F	F	

Algorithm *NEXT* Given n logical variables having values stored in $CASE[1]$, $CASE[2]$, ..., $CASE[n]$ and two vectors $BASE$ and $LENGTH$ each having n elements, it is required to generate the next assignment of truth values for these variables.

1 [Initialize counter] Set $k \leftarrow 1$.
2 [Decrement $LENGTH[k]$] Set $LENGTH[k] \leftarrow LENGTH[k] - 1$.
3 [Negate variable and reset $LENGTH[k]$?] If $LENGTH[k] = 0$ then set $CASE[k] \leftarrow \neg CASE[k]$ and $LENGTH[k] \leftarrow BASE[k]$.
4 [Increment counter] Set $k \leftarrow k + 1$. If $k \leq n$ then go to step 2; otherwise Exit. ////

A program for algorithm *NEXT* is given in Fig. 1-2.1. The subroutine has the four parameters CASE, N, BASE, and LENGTH. All parameters except N are arrays. The logical array CASE contains an assignment of truth values for the logical variables from which the subroutine is to generate a new assignment of values. For example, for the case of three logical variables, CASE(1), CASE(2), and CASE(3) could be associated with the variable names P, Q, and R respectively. The new assignment of truth values is returned to the main program via the logical array CASE.

Let us now consider the problem of constructing a truth table for a statement formula. The following straightforward algorithm uses the various logical arrays such as $CASE$, $BASE$, and $LENGTH$ which were discussed in algorithm *NEXT*.

Algorithm *TRUTH* Given a statement formula in n variables and subalgorithm *NEXT* which generates a new assignment of truth values, it is required to construct a truth table for the given statement formula.

1 [Initialize] Repeat for $k = 1, 2, \ldots, n$: Set $BASE[k] \leftarrow 2^{(n-k)}$, $LENGTH[k] \leftarrow BASE[k]$, and $CASE[k] \leftarrow F$. Set $i \leftarrow 1$ and print headings for the truth table.
2 [Evaluate statement] Substitute the logical values in array $CASE$ into the statement formula. Print the values in array $CASE$ and the value of the statement.
3 [Obtain next assignment for variables] Invoke subalgorithm *NEXT*.

```
      SUBROUTINE NEXT(CASE,N,BASE,LENGTH)
C  GENERATE THE NEXT ASSIGNMENT OF LOGICAL VALUES.
      LOGICAL CASE(N)
      INTEGER BASE(N),LENGTH(N)
      DO 1 K = 1,N
      LENGTH(K) = LENGTH(K) - 1
      IF(LENGTH(K).NE.0) GO TO 1
      CASE(K) = .NOT.CASE(K)
      LENGTH(K) = BASE(K)
    1 CONTINUE
      END
```

FIGURE 1-2.1 Program for algorithm *NEXT*.

4 [Increment counter] Set $i \leftarrow i + 1$. If $i \leq 2^n$ then go to step 2; otherwise Exit. ////

The FORTRAN program for the algorithm is given in Fig. 1-2.2. As an example, the formula

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

was used in the program. The program consists of a main program, a subroutine, and a function. The subroutine NEXT, given in Fig. 1-2.1, generates an assignment each time it is invoked. The function LOGIC is very simple, and its purpose is to generate a single truth value for the statement formula each time the function is invoked. The number of logical variables in the given statement formula and their associated values are passed to the function LOGIC by using the integer variable N and the logical vector CASE respectively.

For our example, the variables P, Q, and R are denoted in the program by the subscripted variables CASE(1), CASE(2), and CASE(3) respectively.

Each time the main program needs a new assignment of truth values for the variables, it calls on procedure NEXT after which the function LOGIC is invoked to evaluate the statement formula for this new assignment of values. The main program computes the BASE and LENGTH vectors for subroutine NEXT.

Initially, all logical variables are set to false, which enables subroutine NEXT to obtain the next assignment. Note that all variables could have been set to true instead. This assignment, of course, would have produced a truth table with the same information as shown in the sample output but in a different order.

1-2.6 Conditional and Biconditional

If P and Q are any two statements, then the statement $P \rightarrow Q$ which is read as "If P, then Q" is called a *conditional* statement. The statement $P \rightarrow Q$ has a truth value F when Q has the truth value F and P the truth value T; otherwise it has the truth value T. The conditional is defined by Table 1-2.8.

The statement P is called the *antecedent* and Q the *consequent* in $P \rightarrow Q$. Again, according to the definition, it is not necessary that there be any kind of relation between P and Q in order to form $P \rightarrow Q$.

Table 1-2.8 TRUTH TABLE FOR CONDITIONAL

P	Q	$P \rightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

EXAMPLE 1 Express in English the statement $P \rightarrow Q$ where

P: The sun is shining today.

Q: $2 + 7 > 4$.

SOLUTION If the sun is shining today, then $2 + 7 > 4$. ////

The conditional often appears very confusing to a beginner, particularly when one tries to translate a conditional in English into symbolic form. A variety of expressions are used in English which can be appropriately translated by the symbol $\rightarrow$. It is customary to represent any one of the following expressions by $P \rightarrow Q$:

1 Q is necessary for P.
2 P is sufficient for Q.
3 Q if P.
4 P only if Q.
5 P implies Q.

We shall avoid the translation "implies." Although, in mathematics, the statements "If P, then Q" and "P implies Q" are used interchangeably, we want to use the word "implies" in a different way.

In our everyday language, we use the conditional statements in a more restricted sense. It is customary to assume some kind of relationship or implication or feeling of cause and effect between the antecedent and the consequent in using the conditional. For example, the statement "If I get the book, then I shall read it tonight" sounds reasonable because the second statement "I shall read it (the book) tonight" refers to the book mentioned in the first part of the statement. On the other hand, a statement such as "If I get the book, then this room is red" does not make sense to us in our conventional language. However, according to our definition of the conditional, the last statement is perfectly acceptable and has a truth value which depends on the truth values of the two statements being connected.

The first two entries in Table 1-2.8 are similar to what we would expect in our everyday language. Thus, if P is true and Q is true, then $P \rightarrow Q$ is true. Similarly, if P is true and Q is false, then "If P, then Q" appears to be false. Consider, for example, the statement "If I get the money, then I shall buy the car." If I actually get the money and buy the car, then the statement appears to be correct or true. On the other hand, if I do not buy the car even though I get the money, then the statement is false. Normally, when a conditional statement is made, we assume that the antecedent is true. Because of this convention in English, the first two entries in the truth table do not appear strange. Referring to the above statement again, if I do not get the money and I still buy the car, it is not so clear whether the statement made earlier is true or false. Also, if I do not buy the car and I do not get the money, then it is not intuitively clear whether the statement made is true or false. It may be possible to justify entries in the last two rows of the truth table by considering special examples or even by emphasizing certain aspects of the statements given in the above examples. However, it is best to consider Table 1-2.8 as the definition of the conditional in which the entries in the last two rows are arbitrarily assigned in order to avoid any am-

biguity. Any other choice for the last two entries would correspond to some other connective which has either been defined or will be defined. In general, the use of "If ..., then ..." in English has only partial resemblance to the use of the conditional $\rightarrow$ as defined here.

EXAMPLE 2 Write the following statement in symbolic form.

If either Jerry takes Calculus or Ken takes Sociology, then Larry will take English.

SOLUTION Denoting the statements as

J: Jerry takes Calculus.

K: Ken takes Sociology.

L: Larry takes English.

the above statement can be symbolized as

$$(J \vee K) \rightarrow L \qquad ////$$

EXAMPLE 3 Write in symbolic form the statement

The crop will be destroyed if there is a flood.

SOLUTION Let the statements be denoted as

C: The crop will be destroyed.

F: There is a flood.

Note that the given statement uses "if" in the sense of "If ..., then ..." It is better to rewrite the given statement as "If there is a flood, then the crop will be destroyed." Now it is easy to symbolize it as

$$F \rightarrow C \qquad ////$$

EXAMPLE 4 Construct the truth table for $(P \rightarrow Q) \wedge (Q \rightarrow P)$.

SOLUTION See Table 1-2.9. Note that the given formula has the truth value T whenever both P and Q have identical truth values. ////

If P and Q are any two statements, then the statement $P \rightleftarrows Q$, which is read as "P if and only if Q" and abbreviated as "P iff Q," is called a *biconditional statement*. The statement $P \rightleftarrows Q$ has the truth value T whenever both P and

Table 1-2.9

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

Table 1-2.10 TRUTH TABLE FOR BICONDITIONAL

P	Q	$P \rightleftarrows Q$
T	T	T
T	F	F
F	T	F
F	F	T

Table 1-2.11

P	Q	$P \wedge Q$	$\neg(P \wedge Q)$	$\neg P$	$\neg Q$	$\neg P \vee \neg Q$	$\neg(P \wedge Q) \rightleftarrows (\neg P \vee \neg Q)$
T	T	T	F	F	F	F	T
T	F	F	T	F	T	T	T
F	T	F	T	T	F	T	T
F	F	F	T	T	T	T	T

Q have identical truth values. Table 1-2.10 defines the biconditional. The statement $P \rightleftarrows Q$ is also translated as "P is necessary and sufficient for Q." Note that the truth values of $(P \rightarrow Q) \wedge (Q \rightarrow P)$ given in Table 1-2.9 are identical to the truth values of $P \rightleftarrows Q$ defined here.

EXAMPLE 5 Construct the truth table for the formula

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

SOLUTION See Table 1-2.11. Note that the truth values of the given formula are T for all possible truth values of P and Q. ////

EXERCISES 1-2.6

1 Show that the truth values of the following formulas are independent of their components.
   (a) $(P \wedge (P \rightarrow Q)) \rightarrow Q$
   (b) $(P \rightarrow Q) \rightleftarrows (\neg P \vee Q)$
   (c) $((P \rightarrow Q) \wedge (Q \rightarrow R)) \rightarrow (P \rightarrow R)$
   (d) $(P \rightleftarrows Q) \rightleftarrows ((P \wedge Q) \vee (\neg P \wedge \neg Q))$

2 Construct the truth tables of the following formulas.
   (a) $(Q \wedge (P \rightarrow Q)) \rightarrow P$
   (b) $\neg(P \vee (Q \wedge R)) \rightleftarrows ((P \vee Q) \wedge (P \vee R))$

3 A connective denoted by $\overline{\vee}$ is defined by Table 1-2.12. Find a formula using P, Q, and the connectives $\wedge$, $\vee$, and $\neg$ whose truth values are identical to the truth values of $P \overline{\vee} Q$.

4 Given the truth values of P and Q as T and those of R and S as F, find the truth values of the following:
   (a) $(\neg(P \wedge Q) \vee \neg R) \vee ((Q \rightleftarrows \neg P) \rightarrow (R \vee \neg S))$
   (b) $(P \rightleftarrows R) \wedge (\neg Q \rightarrow S)$
   (c) $(P \vee (Q \rightarrow (R \wedge \neg P))) \rightleftarrows (Q \vee \neg S)$

Table 1-2.12

P	Q	$P \bar{\vee} Q$
T	T	F
T	F	T
F	T	T
F	F	F

1-2.7 Well-formed Formulas

The notion of a statement formula has already been introduced. A statement formula is not a statement (although, for the sake of brevity, we have often called it a statement); however, a statement can be obtained from it by replacing the variables by statements. A *statement formula* is an expression which is a string consisting of variables (capital letters with or without subscripts), parentheses, and connective symbols. Not every string of these symbols is a formula. We shall now give a recursive definition of a statement formula, often called a well-formed formula (wff). A *well-formed formula* can be generated by the following rules:

1 A statement variable standing alone is a well-formed formula.

2 If A is a well-formed formula, then $\neg A$ is a well-formed formula.

3 If A and B are well-formed formulas, then $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, and $(A \rightleftarrows B)$ are well-formed formulas.

4 A string of symbols containing the statement variables, connectives, and parentheses is a well-formed formula, iff it can be obtained by finitely many applications of the rules 1, 2, and 3.

According to this definition, the following are well-formed formulas:

$$\neg(P \wedge Q) \qquad \neg(P \vee Q) \qquad (P \rightarrow (P \vee Q)) \qquad (P \rightarrow (Q \rightarrow R))$$

$$(((P \rightarrow Q) \wedge (Q \rightarrow R)) \rightleftarrows (P \rightarrow R))$$

The following are not well-formed formulas.

1 $\neg P \wedge Q$. Obviously P and Q are well-formed formulas. A wff would be either $(\neg P \wedge Q)$ or $\neg(P \wedge Q)$.

2 $(P \rightarrow Q) \rightarrow (\wedge Q)$. This is not a wff because $\wedge Q$ is not.

3 $(P \rightarrow Q$. Note that $(P \rightarrow Q)$ is a wff.

4 $(P \wedge Q) \rightarrow Q)$. The reason for this not being a wff is that one of the parentheses in the beginning is missing. $((P \wedge Q) \rightarrow Q)$ is a wff, while $(P \wedge Q) \rightarrow Q$ is still not a wff.

It is possible to introduce some conventions so that the number of parentheses used can be reduced. In fact, there are conventions which, when followed, allow one to dispense with all the parentheses. We shall not discuss these conventions here. For the sake of convenience we shall omit the outer parentheses. Thus we write $P \wedge Q$ in place of $(P \wedge Q)$, $(P \wedge Q) \rightarrow Q$ in place of $((P \wedge Q) \rightarrow Q)$, and $((P \rightarrow Q) \wedge (Q \rightarrow R)) \rightleftarrows (P \rightarrow R)$ instead of $(((P \rightarrow Q) \wedge (Q \rightarrow R)) \rightleftarrows (P \rightarrow R))$. Since the only formulas we will encounter are well-formed formulas, we will refer to well-formed formulas as formulas.

1-2.8 Tautologies

Well-formed formulas have been defined. We also know how to construct the truth table of a given formula. Let us consider what a truth table represents. If definite statements are substituted for the variables in a formula, there results a statement. The truth value of this resulting statement depends upon the truth values of the statements substituted for the variables. Such a truth value appears as one of the entries in the final column of the truth table. Observe that this entry will not change even if any of the definite statements that replace particular variables are themselves replaced by other statements, as long as the truth values associated with all variables are unchanged. In other words, an entry in the final column depends only on the truth values of the statements assigned to the variables rather than on the statements themselves. Different rows correspond to different sets of truth value assignments. A truth table is therefore a summary of the truth values of the resulting statements for all possible assignments of values to the variables appearing in a formula. It must be emphasized that a statemen' formula does not have a truth value. In our discussion which follows we shall, for the sake of simplicity, use the expression "the truth value of a statement formula" to mean the entries in the final column of the truth table of the formula.

In general, the final column of a truth table of a given formula contains both T and F. There are some formulas whose truth values are always T or always F regardless of the truth value assignments to the variables. This situation occurs because of the special construction of these formulas. We have already seen some examples of such formulas.

Consider, for example, the statement formulas $P \vee \neg P$ and $P \wedge \neg P$ in Table 1-2.13. The truth values of $P \vee \neg P$ and $P \wedge \neg P$, which are T and F respectively, are independent of the statement by which the variable P may be replaced.

A statement formula which is true regardless of the truth values of the statements which replace the variables in it is called a *universally valid formula* or a *tautology* or a *logical truth*. A statement formula which is false regardless of the truth values of the statements which replace the variables in it is called a *contradiction*. Obviously, the negation of a contradiction is a tautology. We may say that a statement formula which is a tautology is *identically true* and a formula which is a contradiction is *identically false*.

A straightforward method to determine whether a given formula is a tautology is to construct its truth table. This process can always be used but often becomes tedious, particularly when the number of distinct variables is large or when the formula is complicated. Recall that the numbers of rows in a truth table is 2^n, where n is the number of distinct variables in the formula. Later,

Table 1-2.13

P	$\neg P$	$P \vee \neg P$	$P \wedge \neg P$
T	F	T	F
F	T	T	F

alternative methods will be developed that will be able to determine whether a statement formula is a tautology without having to construct its truth table.

A simple fact about tautologies is that the conjunction of two tautologies is also a tautology. Let us denote by A and B two statement formulas which are tautologies. If we assign any truth values to the variables of A and B, then the truth values of both A and B will be T. Thus the truth value of $A \wedge B$ will be T, so that $A \wedge B$ will be a tautology.

A formula A is called a *substitution instance* of another formula B if A can be obtained from B by substituting formulas for some variables of B, with the condition that the same formula is substituted for the same variable each time it occurs. We now illustrate this concept. Let

$$B\colon P \rightarrow (J \wedge P)$$

Substitute $R \rightleftarrows S$ for P in B, and we get

$$A\colon (R \rightleftarrows S) \rightarrow (J \wedge (R \rightleftarrows S))$$

Then A is a substitution instance of B. Note that

$$(R \rightleftarrows S) \rightarrow (J \wedge P)$$

is not a substitution instance of B because the variable P in $J \wedge P$ was not replaced by $R \rightleftarrows S$. It is possible to substitute more than one variable by other formulas, provided that all substitutions are considered to occur simultaneously. For example, substitution instances of $P \rightarrow \neg Q$ are

1 $(R \wedge \neg S) \rightarrow \neg(J \vee M)$
2 $(R \wedge \neg S) \rightarrow \neg(R \wedge \neg S)$
3 $(R \wedge \neg S) \rightarrow \neg P$
4 $Q \rightarrow \neg(P \wedge \neg Q)$

In (2) both P and Q have been replaced by $R \wedge \neg S$. In (4), P is replaced by Q and Q by $P \wedge \neg Q$.

Next, consider the following formulas which result from $P \rightarrow \neg Q$.

1 Substitute $P \vee Q$ for P and R for Q to get the substitution instance $(P \vee Q) \rightarrow \neg R$.

2 First substitute $P \vee Q$ for P to obtain the substitution instance $(P \vee Q) \rightarrow \neg Q$. Next, substitute R for Q in $(P \vee Q) \rightarrow \neg Q$, and we get $(P \vee R) \rightarrow \neg R$. This formula is a substitution instance of $(P \vee Q) \rightarrow \neg Q$, but it is not a substitution instance of $P \rightarrow \neg Q$ under the substitution $(P \vee Q)$ for P and R for Q. This statement is true because we did not substitute simultaneously as we did in (1).

It may be noted that in constructing substitution instances of a formula, substitutions are made for the atomic formula and never for the molecular formula. Thus $P \rightarrow Q$ is not a substitution instance of $P \rightarrow \neg R$, because it is R which must be replaced and not $\neg R$.

The importance of the above concept lies in the fact that any substitution instance of a tautology is a tautology. Consider the tautology $P \vee \neg P$. Regardless of what is substituted for P, the truth value of $P \vee \neg P$ is always T. Therefore, it we substitute any statement formula for P, the resulting formula will be

a tautology. Hence the following substitution instances of $P \vee \neg P$ are tautologies.

$$(R \to S) \vee \neg(R \to S)$$
$$((P \vee S) \wedge R) \vee \neg((P \vee S) \wedge R)$$
$$(((P \vee \neg Q) \to R) \rightleftarrows S) \vee \neg(((P \vee \neg Q) \to R) \rightleftarrows S)$$

Thus, if it is possible to detect whether a given formula is a substitution instance of a tautology, then it is immediately known that the given formula is also a tautology. Similarly, one can start with a tautology and write a large number of formulas which are substitution instances of this tautology and hence are themselves tautologies.

EXERCISES 1-2.8

1 From the formulas given below select those which are well-formed according to the definition in Sec. 1-2.7, and indicate which ones are tautologies or contradictions.
   (*a*) $(P \to (P \vee Q))$
   (*b*) $((P \to (\neg P)) \to \neg P)$
   (*c*) $((\neg Q \wedge P) \wedge Q)$
   (*d*) $((P \to (Q \to R)) \to ((P \to Q) \to (P \to R)))$
   (*e*) $((\neg P \to Q) \to (Q \to P)))$
   (*f*) $((P \wedge Q) \rightleftarrows P)$

2 Produce the substitution instances of the following formulas for the given substitutions.
   (*a*) $(((P \to Q) \to P) \to P)$; substitute $(P \to Q)$ for P and $((P \wedge Q) \to R)$ for Q.
   (*b*) $((P \to Q) \to (Q \to P))$; substitute Q for P and $(P \wedge \neg P)$ for Q.

3 Determine the formulas which are substitution instances of other formulas in the list and give the substitutions.
   (*a*) $(P \to (Q \to P))$
   (*b*) $((((P \to Q) \wedge (R \to S)) \wedge (P \vee R)) \to (Q \vee S))$
   (*c*) $(Q \to ((P \to P) \to Q))$
   (*d*) $(P \to ((P \to (Q \to P)) \to P))$
   (*e*) $((((R \to S) \wedge (Q \to P)) \wedge (R \vee Q)) \to (S \vee P))$

1-2.9 Equivalence of Formulas

Let A and B be two statement formulas and let $P_1, P_2, \ldots, P_n$ denote all the variables occurring in both A and B. Consider an assignment of truth values to $P_1, P_2, \ldots, P_n$ and the resulting truth values of A and B. If the truth value of A is equal to the truth value of B for every one of the 2^n possible sets of truth values assigned to $P_1, P_2, \ldots, P_n$, then A and B are said to be *equivalent*. Assuming that the variables and the assignment of truth values to the variables appear in the same order in the truth tables of A and B, then the final columns in the truth tables for A and B are identical if A and B are equivalent.

Here are some examples of formulas which are equivalent. Verify their equivalence by truth tables.

1 $\neg\neg\neg P$ is equivalent to P.
2 $P \vee P$ is equivalent to P.

3 $(P \wedge \neg P) \vee Q$ is equivalent to Q.
4 $P \vee \neg P$ is equivalent to $Q \vee \neg Q$.

In the definition of equivalence of two formulas, it is not necessary to assume that they both contain the same variables. This point is illustrated in the examples given in (3) and (4) above. It may, however, be noted that if two formulas are equivalent and a particular variable occurs in only one of them, then the truth value of this formula is independent of this variable. For example, in (3) the truth value of $(P \wedge \neg P) \vee Q$ is independent of the truth value of P. Similarly in (4), the truth values of $P \vee \neg P$ and $Q \vee \neg Q$ are each independent of P and Q.

Recalling the truth table (Table 1-2.10) in the definition of the biconditional, it is clear that $P \rightleftarrows Q$ is true whenever both P and Q have the same truth values. Therefore the statement formulas A and B are equivalent provided $A \rightleftarrows B$ is a tautology; and, conversely, if $A \rightleftarrows B$ is a tautology, then A and B are equivalent. We shall represent the equivalence of two formulas, say A and B, by writing "$A \Leftrightarrow B$," which is read as "A is equivalent to B." Note that the expression "$A \Leftrightarrow B$" which can also be displayed as

$$A \Leftrightarrow B$$

should be written as

$$\text{“}A\text{”} \Leftrightarrow \text{“}B\text{”}$$

according to the rules given earlier (in Sec. 1-1) regarding the use and mention of expressions. Observe that "$A \Leftrightarrow B$" is a statement in English (the metalanguage) and not in the object language. Also the symbol "$\Leftrightarrow$" is not a connective but a symbol in the metalanguage. Having noted this, we shall often drop the quotation marks because this will not lead to any ambiguity.

Equivalence is a symmetric relation; that is, "A is equivalent to B" is the same as "B is equivalent to A." Also if $A \Leftrightarrow B$ and $B \Leftrightarrow C$, then $A \Leftrightarrow C$. This relationship may also be expressed by saying that the equivalence of statement formulas is transitive.

As in the case of tautologies, one method to determine whether any two statement formulas are equivalent is to construct their truth tables. All combinations of truth values associated with the variables appearing in both formulas are presented in the table, and the final columns (for the two formulas) are compared.

EXAMPLE 1 Prove $(P \rightarrow Q) \Leftrightarrow (\neg P \vee Q)$.

SOLUTION See Table 1-2.14. Note that the truth values in the columns for $P \rightarrow Q$ and $\neg P \vee Q$ are identical, and so the biconditional will have the truth value T. To compare columns, it is not necessary to form the biconditional; thus the last column could have been avoided. ////

A list of some basic equivalent formulas which will be found useful is given in Table 1-2.15. In order to make the list complete, we use **T** and **F** as special variables in the sense that **T** can be replaced by only a tautology and **F** by only a contradiction.

Table 1-2.14

P	Q	$P \to Q$	$\neg P$	$\neg P \vee Q$	$(P \to Q) \rightleftarrows (\neg P \vee Q)$
T	T	T	F	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

In view of the associative laws, we can write $(P \vee Q) \vee R$ as $P \vee Q \vee R$, and $(P \wedge Q) \wedge R$ as $P \wedge Q \wedge R$.

In Table 1-2.15 we note that pairs of equivalent formulas are arranged two to a line such as

$$A_1 \Leftrightarrow B_1 \qquad A_2 \Leftrightarrow B_2$$

For each pair A_1, B_1 there is a corresponding pair A_2, B_2 in which $\vee$ is replaced by $\wedge$, $\wedge$ by $\vee$, **T** by **F**, and **F** by **T**. A_1 and A_2 are said to be duals of each other, and so are B_1 and B_2. Duality is discussed in Sec. 1-2.10.

In constructing substitution instances of a statement formula, we are allowed to substitute only for the variables appearing in the formula. Furthermore, the same formula is to be substituted for every occurrence of a particular variable. This rule ensures that substitution instances of a tautology are also tautologies. Consider now another process, called a *replacement process*, in which we replace any part of a statement formula which is itself a formula, be it atomic or molecular, by any other formula. For example, in the formula $(P \wedge Q) \to P$ we replace $(P \wedge Q)$ by $R \to (S \wedge \neg M)$ and the second P by $(P \wedge R) \to (\neg S \vee M)$ to obtain $(R \to (S \wedge \neg M)) \to ((P \wedge R) \to (\neg S \vee M))$. In general, a replacement yields a new formula, but it may not always be an interesting formula. However, if we impose the restriction that any part of a given formula that is to be replaced by another formula must be equivalent to that other formula, then the result is equivalent to the original formula. By this process one can obtain new formulas which are equivalent to the original formula. For example, we can replace P in $P \wedge Q$ by the formula $P \vee P$, since $P \vee P \Leftrightarrow P$, to get $(P \vee P) \wedge Q$ which is equivalent to $P \wedge Q$. Consequently, if we replace any part or parts of a tautology by formulas that are equivalent to these parts, we again get a tautology.

EXAMPLE 2 Show that $P \to (Q \to R) \Leftrightarrow P \to (\neg Q \vee R) \Leftrightarrow (P \wedge Q) \to R$.

SOLUTION Recall from Example 1 that $Q \to R \Leftrightarrow \neg Q \vee R$. Replacing $Q \to R$ by $\neg Q \vee R$, we get $P \to (\neg Q \vee R)$, which is equivalent to $\neg P \vee (\neg Q \vee R)$ by the same rule. Now

$$\neg P \vee (\neg Q \vee R) \Leftrightarrow (\neg P \vee \neg Q) \vee R \Leftrightarrow \neg(P \wedge Q) \vee R$$
$$\Leftrightarrow (P \wedge Q) \to R$$

using associativity of $\vee$, De Morgan's law, and the previously used rule. ////

Table 1-2.15 EQUIVALENT FORMULAS

$P \vee P \Leftrightarrow P$	$P \wedge P \Leftrightarrow P$	(Idempotent laws)	(1)
$(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$	$(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$	(Associative laws)	(2)
$P \vee Q \Leftrightarrow Q \vee P$	$P \wedge Q \Leftrightarrow Q \wedge P$	(Commutative laws)	(3)
$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$	$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$	(Distributive laws)	(4)
$P \vee \mathbf{F} \Leftrightarrow P$	$P \wedge \mathbf{T} \Leftrightarrow P$		(5)
$P \vee \mathbf{T} \Leftrightarrow \mathbf{T}$	$P \wedge \mathbf{F} \Leftrightarrow \mathbf{F}$		(6)
$P \vee \neg P \Leftrightarrow \mathbf{T}$	$P \wedge \neg P \Leftrightarrow \mathbf{F}$		(7)
$P \vee (P \wedge Q) \Leftrightarrow P$	$P \wedge (P \vee Q) \Leftrightarrow P$	(Absorption laws)	(8)
$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$	$\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$	(De Morgan's laws)	(9)

EXAMPLE 3 Show that $(\neg P \wedge (\neg Q \wedge R)) \vee (Q \wedge R) \vee (P \wedge R) \Leftrightarrow R$.

SOLUTION

$$
\begin{aligned}
&(\neg P \wedge (\neg Q \wedge R)) \vee (Q \wedge R) \vee (P \wedge R) \\
&\quad \Leftrightarrow (\neg P \wedge (\neg Q \wedge R)) \vee ((Q \vee P) \wedge R) && (4)\\
&\quad \Leftrightarrow ((\neg P \wedge \neg Q) \wedge R) \vee ((Q \vee P) \wedge R) && (2)\\
&\quad \Leftrightarrow ((\neg P \wedge \neg Q) \vee (Q \vee P)) \wedge R && (4)\\
&\quad \Leftrightarrow (\neg (P \vee Q) \vee (P \vee Q)) \wedge R && (9), (3)\\
&\quad \Leftrightarrow \mathbf{T} \wedge R && (7)\\
&\quad \Leftrightarrow R && (5)
\end{aligned}
$$

The basic equivalent statement formulas used are denoted by the numbers on the right-hand side which correspond to numbers in Table 1-2.15. ////

EXAMPLE 4 Show that $((P \vee Q) \wedge \neg(\neg P \wedge (\neg Q \vee \neg R))) \vee (\neg P \wedge \neg Q) \vee (\neg P \wedge \neg R)$ is a tautology.

SOLUTION Using De Morgan's laws, we obtain

$$
\begin{aligned}
\neg P \wedge \neg Q \Leftrightarrow \neg(P \vee Q) \qquad \neg P \wedge \neg R \Leftrightarrow \neg(P \vee R)\\
(\neg P \wedge \neg Q) \vee (\neg P \wedge \neg R) \Leftrightarrow \neg(P \vee Q) \vee \neg(P \vee R)\\
\Leftrightarrow \neg((P \vee Q) \wedge (P \vee R))
\end{aligned}
$$

Also

$$
\begin{aligned}
\neg(\neg P \wedge (\neg Q \vee \neg R) &\Leftrightarrow \neg(\neg P \wedge \neg(Q \wedge R))\\
&\Leftrightarrow P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)\\
(P \vee Q) \wedge ((P \vee Q) \wedge (P \vee R)) &\Leftrightarrow (P \vee Q) \wedge (P \vee R)
\end{aligned}
$$

Consequently, the given formula is equivalent to

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

which is a substitution instance of $P \vee \neg P$. ////

The equivalences given in Table 1-2.15 also describe the properties of the operators $\wedge$, $\vee$, and $\neg$ on the set of statements in symbolic logic. It is shown in Chap. 4 that the set of all statements under the operations $\wedge$, $\vee$, and $\neg$ is an algebra called the *statement algebra* which is a particular example of a Boolean algebra. A comparison of the statement algebra and the set algebra is given in Chap. 2.

1-2.10 Duality Law

In this section we shall consider formulas which contain the connectives $\wedge$, $\vee$, and $\neg$. There is no loss of generality in restricting our consideration to these connectives since we shall see later that any formula containing any other connective can be replaced by an equivalent formula containing only these three connectives.

Two formulas, A and A^*, are said to be *duals* of each other if either one can be obtained from the other by replacing $\wedge$ by $\vee$ and $\vee$ by $\wedge$. The connectives $\wedge$ and $\vee$ are also called *duals* of each other. If the formula A contains the special variables **T** or **F**, then A^*, its dual, is obtained by replacing **T** by **F** and **F** by **T** in addition to the above-mentioned interchanges.

EXAMPLE 1 Write the duals of (*a*) $(P \vee Q) \wedge R$; (*b*) $(P \wedge Q) \vee \mathbf{T}$; (*c*) $\neg(P \vee Q) \wedge (P \vee \neg(Q \wedge \neg S))$.

SOLUTION The duals are (*a*) $(P \wedge Q) \vee R$, (*b*) $(P \vee Q) \wedge \mathbf{F}$, and (*c*) $\neg(P \wedge Q) \vee (P \wedge \neg(Q \vee \neg S))$. ////

The following theorem shows the equivalence of a formula and one that is obtained from its dual.

Theorem 1-2.1 Let A and A^* be dual formulas and let $P_1, P_2, \ldots, P_n$ be all the atomic variables that occur in A and A^*. That is to say, we may write A as $A(P_1, P_2, \ldots, P_n)$ and A^* as $A^*(P_1, P_2, \ldots, P_n)$. Then through the use of De Morgan's laws

$$P \wedge Q \Leftrightarrow \neg(\neg P \vee \neg Q) \qquad P \vee Q \Leftrightarrow \neg(\neg P \wedge \neg Q)$$

we can show

$$\neg A(P_1, P_2, \ldots, P_n) \Leftrightarrow A^*(\neg P_1, \neg P_2, \ldots, \neg P_n) \tag{1}$$

Thus the negation of a formula is equivalent to its dual in which every variable is replaced by its negation. As a consequence of this fact, we also have

$$A(\neg P_1, \neg P_2, \ldots, \neg P_n) \Leftrightarrow \neg A^*(P_1, P_2, \ldots, P_n) \tag{2}$$

EXAMPLE 2 Verify equivalence (1) if $A(P, Q, R)$ is $\neg P \wedge \neg(Q \vee R)$.

SOLUTION Now $A^*(P, Q, R)$ is $\neg P \vee \neg(Q \wedge R)$, and $A^*(\neg P, \neg Q, \neg R)$ is $\neg\neg P \vee \neg(\neg Q \wedge \neg R) \Leftrightarrow P \vee (Q \vee R)$. On the other hand, $\neg A(P, Q, R)$ is $\neg(\neg P \wedge \neg(Q \vee R)) \Leftrightarrow P \vee (Q \vee R)$. ////

We shall now prove an interesting theorem which states that if any two formulas are equivalent, then their duals are also equivalent to each other. In other words, if $A \Leftrightarrow B$, then $A^* \Leftrightarrow B^*$.

Theorem 1-2.2 Let $P_1, P_2, \ldots, P_n$ be all the atomic variables appearing in the formulas A and B. Given that $A \Leftrightarrow B$ means "$A \rightleftarrows B$ is a tautology," then the following are also tautologies.

$$A(P_1, P_2, \ldots, P_n) \rightleftarrows B(P_1, P_2, \ldots, P_n)$$
$$A(\neg P_1, \neg P_2, \ldots, \neg P_n) \rightleftarrows B(\neg P_1, \neg P_2, \ldots, \neg P_n)$$

Using (2), we get

$$\neg A^*(P_1, P_2, \ldots, P_n) \rightleftarrows \neg B^*(P_1, P_2, \ldots, P_n)$$

Hence $A^* \Leftrightarrow B^*$.

This theorem explains why in Table 1-2.15 we have for every pair of equivalent formulas an equivalent pair of formulas consisting of duals of the first pair.

EXAMPLE 3 Show that

(*a*) $\neg(P \wedge Q) \rightarrow (\neg P \vee (\neg P \vee Q)) \Leftrightarrow (\neg P \vee Q)$
(*b*) $(P \vee Q) \wedge (\neg P \wedge (\neg P \wedge Q)) \Leftrightarrow (\neg P \wedge Q)$

SOLUTION

(*a*)
$$\begin{aligned} &\neg(P \wedge Q) \rightarrow (\neg P \vee (\neg P \vee Q)) \\ &\Leftrightarrow (P \wedge Q) \vee (\neg P \vee (\neg P \vee Q)) \qquad (3) \\ &\Leftrightarrow (P \wedge Q) \vee (\neg P \vee Q) \\ &\Leftrightarrow (P \wedge Q) \vee \neg P \vee Q \\ &\Leftrightarrow ((P \vee \neg P) \wedge (Q \vee \neg P)) \vee Q \\ &\Leftrightarrow (Q \vee \neg P) \vee Q \Leftrightarrow Q \vee \neg P \Leftrightarrow \neg P \vee Q \end{aligned}$$

From (3) it follows that

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

Writing the duals, we obtain by Theorem 1-2.2 that

$$(P \vee Q) \wedge (\neg P \wedge (\neg P \wedge Q)) \Leftrightarrow \neg P \wedge Q \qquad ////$$

1-2.11 Tautological Implications

Recall the definition of the conditional as given in Table 1-2.8. The connectives $\wedge$, $\vee$, and $\rightleftarrows$ are symmetric in the sense that $P \wedge Q \Leftrightarrow Q \wedge P$, $P \vee Q \Leftrightarrow Q \vee P$, and $P \rightleftarrows Q \Leftrightarrow Q \rightleftarrows P$. On the other hand, $P \rightarrow Q$ is not equivalent to $Q \rightarrow P$.

For any statement formula $P \rightarrow Q$, the statement formula $Q \rightarrow P$ is called its *converse*, $\neg P \rightarrow \neg Q$ is called its *inverse*, and $\neg Q \rightarrow \neg P$ is called its *contrapositive*.

From Table 1-2.16 it is clear that

$$P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P \qquad Q \rightarrow P \Leftrightarrow \neg P \rightarrow \neg Q$$

A statement A is said to *tautologically imply* a statement B if and only if $A \rightarrow B$ is a tautology. We shall denote this idea by $A \Rightarrow B$ which is read as "A implies B." Similar to the case with $\Leftrightarrow$, we note that $\Rightarrow$ is not a connective nor is $A \Rightarrow B$ a statement formula. Just as $A \Leftrightarrow B$ states that A and B are equiv-

Table 1-2.16

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

alent or that $A \rightleftarrows B$ is a tautology, in a similar manner $A \Rightarrow B$ states that $A \rightarrow B$ is a tautology or A tautologically implies B.

We have avoided using the expression "imply" to translate the conditional, so that we shall abbreviate "tautologically imply" simply as "imply." Obviously $A \Rightarrow B$ guarantees that B has the truth value T whenever A has the truth value T.

One can determine whether $A \Rightarrow B$ by constructing the truth tables of A and B in the same manner as was done in the determination of $A \Leftrightarrow B$.

The implications in Table 1-2.17 have important applications. All of them can be proved by truth table or by other methods.

In order to show any of the given implications, it is sufficient to show that an assignment of the truth value T to the antecedent of the corresponding conditional leads to the truth value T for the consequent. This procedure guarantees that the conditional becomes a tautology, thereby proving the implication. In (9), if we assume that $\neg Q \wedge (P \rightarrow Q)$ has the truth value T, then both $\neg Q$ and $P \rightarrow Q$ have the truth value T, which means that Q has the value F. $P \rightarrow Q$ has the truth value T, and hence P must have the value F. Therefore the consequent $\neg P$ must have the value T.

In (12), we assume that the antecedent is true. This assumption means that $P \vee Q$, $P \rightarrow R$, and $Q \rightarrow R$ are true. If P is true, then R must be true because $P \rightarrow R$ is true. If Q is true, then R must also be true. But at least one of P or Q is true by our assumption that $P \vee Q$ is true, and so R is true.

Another method to show $P \Rightarrow Q$ is to assume that the consequent Q has the value F and then show that this assumption leads to P's having the value F. Then $P \rightarrow Q$ must have the value T.

In (9) assume that $\neg P$ is false, so that P is true. Then $\neg Q \wedge (P \rightarrow Q)$ must be false. This statement holds because if Q is true, then $\neg Q$ is false, while if Q is false, then $P \rightarrow Q$ is false. Hence the implication in (9) is shown.

Example 4 in Sec. 1-2.6 shows the equivalence of the statements $P \rightleftarrows Q$ and $(P \rightarrow Q) \wedge (Q \rightarrow P)$; it is easy to verify that ($P \Rightarrow Q$ and $Q \Rightarrow P$) iff $P \Leftrightarrow Q$. This statement is an alternative definition of the equivalence of two formulas. If each of the two formulas A and B implies the other, then A and B are equivalent.

Table 1-2.17 IMPLICATIONS

$P \wedge Q \Rightarrow P$	(1)
$P \wedge Q \Rightarrow Q$	(2)
$P \Rightarrow P \vee Q$	(3)
$\neg P \Rightarrow P \rightarrow Q$	(4)
$Q \Rightarrow P \rightarrow Q$	(5)
$\neg(P \rightarrow Q) \Rightarrow P$	(6)
$\neg(P \rightarrow Q) \Rightarrow \neg Q$	(7)
$P \wedge (P \rightarrow Q) \Rightarrow Q$	(8)
$\neg Q \wedge (P \rightarrow Q) \Rightarrow \neg P$	(9)
$\neg P \wedge (P \vee Q) \Rightarrow Q$	(10)
$(P \rightarrow Q) \wedge (Q \rightarrow R) \Rightarrow P \rightarrow R$	(11)
$(P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R) \Rightarrow R$	(12)

There are several important facts about implication and equivalence that should be observed. If a formula is equivalent to a tautology, then it must be a tautology. Similarly, if a formula is implied by a tautology, then it is a tautology.

Both implication and equivalence are transitive. To say that equivalence is transitive means if $A \Leftrightarrow B$ and $B \Leftrightarrow C$, then $A \Leftrightarrow C$. This statement follows from the definition of equivalence. To show that the implication is also transitive, assume that $A \Rightarrow B$ and $B \Rightarrow C$. Then $A \rightarrow B$ and $B \rightarrow C$ are tautologies. Hence $(A \rightarrow B) \wedge (B \rightarrow C)$ is also a tautology. But from (11), $(A \rightarrow B) \wedge (B \rightarrow C) \Rightarrow (A \rightarrow C)$. Hence $A \rightarrow C$ is a tautology.

The transitivity of implications can also be applied in several stages. In order to show that $A \Rightarrow C$, it may be convenient to introduce a series of formulas $B_1, B_2, \ldots, B_m$ such that $A \Rightarrow B_1$, $B_1 \Rightarrow B_2, \ldots, B_{m-1} \Rightarrow B_m$, and $B_m \Rightarrow C$.

Another important property of implication is that if $A \Rightarrow B$ and $A \Rightarrow C$, then $A \Rightarrow (B \wedge C)$. By our assumption, if A is true, then B and C are both true. Thus $B \wedge C$ is true, and hence $A \rightarrow (B \wedge C)$ is true.

We can extend our notion of implication $P \Rightarrow Q$ to the case where several formulas, say $H_1, H_2, \ldots, H_m$, jointly imply a particular formula Q; that is, $H_1, H_2, \ldots, H_m \Rightarrow Q$ means $(H_1 \wedge H_2 \wedge \cdots \wedge H_m) \Rightarrow Q$.

An important theorem which is used in Sec. 1-4.1 follows.

Theorem 1-2.3 If $H_1, H_2, \ldots, H_m$ and P imply Q, then $H_1, H_2, \ldots, H_m$ imply $P \rightarrow Q$.

PROOF From our assumption we have

$$(H_1 \wedge H_2 \wedge \cdots \wedge H_m \wedge P) \Rightarrow Q$$

This assumption means $(H_1 \wedge H_2 \wedge \cdots \wedge H_m \wedge P) \rightarrow Q$ is a tautology. Using the equivalence (see Example 2, Sec. 1-2.9)

$$P_1 \rightarrow (P_2 \rightarrow P_3) \Leftrightarrow (P_1 \wedge P_2) \rightarrow P_3$$

we can say that

$$(H_1 \wedge H_2 \wedge \cdots \wedge H_m) \rightarrow (P \rightarrow Q)$$

is a tautology. Hence the theorem. ////

EXERCISES 1-2.11

1 Show the following implications.
(*a*) $(P \wedge Q) \Rightarrow (P \rightarrow Q)$
(*b*) $P \Rightarrow (Q \rightarrow P)$
(*c*) $(P \rightarrow (Q \rightarrow R)) \Rightarrow (P \rightarrow Q) \rightarrow (P \rightarrow R)$

2 Show the following equivalences.
(*a*) $P \rightarrow (Q \rightarrow P) \Leftrightarrow \neg P \rightarrow (P \rightarrow Q)$
(*b*) $P \rightarrow (Q \vee R) \Leftrightarrow (P \rightarrow Q) \vee (P \rightarrow R)$
(*c*) $(P \rightarrow Q) \wedge (R \rightarrow Q) \Leftrightarrow (P \vee R) \rightarrow Q$
(*d*) $\neg(P \rightleftarrows Q) \Leftrightarrow (P \vee Q) \wedge \neg(P \wedge Q)$

3 Show the following implications without constructing the truth tables.
(a) $P \rightarrow Q \Rightarrow P \rightarrow (P \wedge Q)$
(b) $(P \rightarrow Q) \rightarrow Q \Rightarrow P \vee Q$
(c) $((P \vee \neg P) \rightarrow Q) \rightarrow ((P \vee \neg P) \rightarrow R) \Rightarrow (Q \rightarrow R)$ (see Sec. 1-6.3)
(d) $(Q \rightarrow (P \wedge \neg P)) \rightarrow (R \rightarrow (P \wedge \neg P)) \Rightarrow (R \rightarrow Q)$ (see Sec. 1-6.3)

4 Show that P is equivalent to the following formulas.

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

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

5 Show the following equivalences.
(a) $\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$
(b) $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$
(c) $\neg(P \rightarrow Q) \Leftrightarrow P \wedge \neg Q$
(d) $\neg(P \rightleftarrows Q) \Leftrightarrow (P \wedge \neg Q) \vee (\neg P \wedge Q)$

1-2.12 Formulas with Distinct Truth Tables

Using all the connectives defined so far and the rules for constructing well-formed formulas, it is possible to construct an unlimited number of statement formulas. We shall try to determine how many of these formulas have distinct truth tables.

Let us consider all possible truth tables that can be obtained when the formulas involve only one variable P. These possible truth tables are shown in Table 1-2.18.

Any formula involving only one variable will have one of these four truth tables. Obviously the simplest formulas corresponding to the entries under 1, 2, 3, and 4 are P, $\neg P$, $P \vee \neg P$, and $P \wedge \neg P$ respectively. Every other formula depending upon P alone would then be equivalent to one of these four formulas.

If we consider formulas obtained by using two variables P and Q and any of the connectives defined, we also obtain an unlimited number of formulas. The number of distinct truth tables for formulas involving two variables is given by $2^{2^2} = 2^4 = 16$. Since there are 2^2 rows in the truth table and since each row could have any of the two entries T or F, we have 2^{2^2} possible tables, as shown in Table 1-2.19.

Any formula involving only two variables will have one of these 16 truth tables. All those formulas which have one of these truth tables are equivalent to each other.

A statement formula containing n variables must have as its truth table one of the 2^{2^n} possible truth tables, each of them having 2^n rows. This fact suggests that there are many formulas which may look very different from one another but are equivalent.

Table 1-2.18

P	1	2	3	4
T	T	F	T	F
F	F	T	T	F

Table 1-2.19

P	*Q*	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
T	*T*	*T*	*T*	*T*	*T*	*T*	*T*	*T*	*T*	*F*	*F*	*F*	*F*	*F*	*F*	*F*	*F*
T	*F*	*T*	*T*	*T*	*T*	*F*	*F*	*F*	*F*	*T*	*T*	*T*	*T*	*F*	*F*	*F*	*F*
F	*T*	*T*	*T*	*F*	*F*	*T*	*T*	*F*	*F*	*T*	*T*	*F*	*F*	*T*	*T*	*F*	*F*
F	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*	*T*	*F*

One method to determine whether two statement formulas A and B are equivalent is to construct their truth tables and compare them. This method is very tedious and difficult to implement even on a computer because the number of entries increases very rapidly as n increases (note that $2^{10} \simeq 1{,}000$). A better method would be to transform A and B to some standard forms A' and B' such that a simple comparison of A' and B' should establish whether $A \Leftrightarrow B$. This method is feasible; the standard forms are called canonical forms or normal forms and are discussed in Sec. 1-3.

1-2.13 Functionally Complete Sets of Connectives

So far we have defined the connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, and $\rightleftarrows$. We introduce some other connectives in Sec. 1-2.14 because of their usefulness in certain applications. On the other hand, we show in this section that not all the connectives defined thus far are necessary. In fact, we can find certain proper subsets of these connectives which are sufficient to express any formula in an equivalent form. Any set of connectives in which every formula can be expressed in terms of an equivalent formula containing the connectives from this set is called a *functionally complete* set of connectives. It is assumed that such a functionally complete set does not contain any redundant connectives, i.e., a connective which can be expressed in terms of the other connectives.

In order to arrive at a functionally complete set, we first examine the following equivalence:

$$P \rightleftarrows Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$$

This equivalence suggests that in any formula we can replace the part (here "part" means any part which is itself a formula) containing the biconditional by an equivalent formula not containing the biconditional. Thus all the biconditionals can be replaced in a formula.

EXAMPLE 1 Write an equivalent formula for $P \wedge (Q \rightleftarrows R) \vee (R \rightleftarrows P)$ which does not contain the biconditional.

SOLUTION

$$P \wedge (Q \rightleftarrows R) \vee (R \rightleftarrows P)$$
$$\Leftrightarrow P \wedge ((Q \rightarrow R) \wedge (R \rightarrow Q)) \vee ((R \rightarrow P) \wedge (P \rightarrow R))$$

Thus the equivalent formula is $P \wedge ((Q \rightarrow R) \wedge (R \rightarrow Q)) \vee ((R \rightarrow P) \wedge (P \rightarrow R))$. ////

Next we now consider the equivalence $P \rightarrow Q \Leftrightarrow \neg P \vee Q$. This equivalence suggests that the conditionals can also be eliminated by replacing those parts which contain conditionals by their equivalents.

EXAMPLE 2 Write an equivalent formula for $P \wedge (Q \rightleftarrows R)$ which contains neither the biconditional nor the conditional.

SOLUTION

$$P \wedge (Q \rightleftarrows R) \Leftrightarrow P \wedge ((Q \rightarrow R) \wedge (R \rightarrow Q))$$
$$\Leftrightarrow P \wedge ((\neg Q \vee R) \wedge (\neg R \vee Q))$$

Thus the required formula is $P \wedge (\neg Q \vee R) \wedge (\neg R \vee Q)$. ////

Notice that from De Morgan's laws we have

$$P \wedge Q \Leftrightarrow \neg(\neg P \vee \neg Q) \qquad P \vee Q \Leftrightarrow \neg(\neg P \wedge \neg Q)$$

This first equivalence means that it is also possible to obtain a formula which is equivalent to a given formula in which conjunctions are eliminated. A similar procedure is possible for the elimination of disjunctions.

If we implement all the steps suggested above, we can first replace all biconditionals, then the conditionals, and finally all the conjunctions or all the disjunctions in any formula to obtain an equivalent formula which contains either the negation and disjunction only or the negation and conjunction only. This fact means that the sets of connectives $\{\wedge, \neg\}$ and $\{\vee, \neg\}$ are functionally complete.

One can show that $\{\neg\}$, $\{\wedge\}$, or $\{\vee\}$ are not functionally complete and neither is $\{\wedge, \vee\}$.

From the five connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\rightleftarrows$ we have obtained at least two sets of functionally complete connectives. A question arises whether there is any one connective which is functionally complete. The answer to such a question is no if only the above five connectives are considered. There are some connectives, which are defined in Sec. 1-2.14, that are functionally complete. The question of finding a functionally complete set with fewer connectives is not as theoretical as it may appear, because in physical two-state devices, which are described in Sec. 1-2.15, the connectives correspond to certain physical elements of the device. From the point of view of maintenance and economical production, it is sometimes necessary not to use a variety of different elements.

Note that if a given formula is replaced by an equivalent formula in which the number of different connectives is reduced, the resulting formula may become more complex. This is why we use a larger number of connectives than are needed. In Sec. 1-2.14 we define some more connectives which will be found useful.

EXERCISES 1-2.13

1 Write formulas which are equivalent to the formulas given below and which contain the connectives $\wedge$ and $\neg$ only.
   (*a*) $\neg(P \rightleftarrows (Q \rightarrow (R \vee P)))$
   (*b*) $((P \vee Q) \wedge R) \rightarrow (P \vee R)$
2 For each column in Table 1-2.19 write a formula, involving two variables P and Q, whose truth table corresponds to the truth values in the column chosen.
3 Show that $\{\wedge, \vee\}$, $\{\vee\}$, and $\{\neg\}$ are not functionally complete. (*Hint*: Write a formula which is a tautology.)

1-2.14 Other Connectives

It was shown earlier that not all connectives defined thus far are necessary for the description of the statement calculus. For any formula of the statement calculus, there exists an equivalent formula in which appear only those connectives belonging to one of the functionally complete sets. In spite of this fact, we did define other connectives because, by using them, some of the formulas become simpler. There are other connectives which serve similar purposes, and these will be defined in this section.

Let P and Q be any two formulas. Then the formula $P \overline{\vee} Q$, in which the connective $\overline{\vee}$ is called an *exclusive OR*, is true whenever either P or Q, but not both, is true. The exclusive OR is defined by Table 1-2.20. The exclusive OR is also called the *exclusive disjunction*. The following equivalences follow from its definition.

1	$P \overline{\vee} Q \Leftrightarrow Q \overline{\vee} P$	(symmetric)
2	$(P \overline{\vee} Q) \overline{\vee} R \Leftrightarrow P \vee (Q \overline{\vee} R)$	(associative)
3	$P \wedge (Q \overline{\vee} R) \Leftrightarrow (P \wedge Q) \overline{\vee} (P \wedge R)$	(distributive)
4	$(P \overline{\vee} Q) \Leftrightarrow (P \wedge \neg Q) \vee (\neg P \wedge Q)$	
5	$(P \overline{\vee} Q) \Leftrightarrow \neg(P \rightleftarrows Q)$	

One can also prove that if $P \overline{\vee} Q \Leftrightarrow R$, then $P \overline{\vee} R \Leftrightarrow Q$ and $Q \overline{\vee} R \Leftrightarrow P$, and $P \overline{\vee} Q \overline{\vee} R$ is a contradiction.

Given a formula in which $\overline{\vee}$ appears, it is possible to obtain an equivalent formula in which only the connectives $\wedge$, $\vee$, and $\neg$ appear by using the equivalence of the formulas in (4).

Other connectives which have useful applications in the design of computers are called *NAND* and *NOR*. The word *"NAND"* is a combination of *"NOT"* and *"AND,"* where *"NOT"* stands for negation and *"AND"* for the conjunction. The connective *NAND* is denoted by the symbol $\uparrow$. For any two formulas P and Q

$$P \uparrow Q \Leftrightarrow \neg(P \wedge Q)$$

Another connective, useful in a similar context, is known as *NOR*, a combination of *NOT* and *OR*, where *"OR"* stands for the disjunction. The connective *NOR* is denoted by the symbol $\downarrow$. For any two formulas P and Q

$$P \downarrow Q \Leftrightarrow \neg(P \vee Q)$$

The connectives $\uparrow$ and $\downarrow$ have been defined in terms of the connectives $\wedge$, $\vee$, and $\neg$. Therefore, for any formula containing the connectives $\uparrow$ or $\downarrow$,

Table 1-2.20

P	Q	$P \overline{\vee} Q$
T	T	F
T	F	T
F	T	T
F	F	F

one can obtain an equivalent formula containing the connectives $\wedge$, $\vee$, and $\neg$ only. Note that $\uparrow$ and $\downarrow$ are duals of each other. Therefore, in order to obtain the dual of a formula which includes $\uparrow$ or $\downarrow$, we should interchange $\uparrow$ and $\downarrow$ in addition to making the other interchanges mentioned earlier.

We now show that each of the connectives $\uparrow$ and $\downarrow$ is functionally complete. In order to do this, it is sufficient to show that the sets of connectives $\{\wedge, \neg\}$ and $\{\vee, \neg\}$ can be expressed either in terms of $\uparrow$ alone or in terms of $\downarrow$ alone. The following equivalences express $\neg$, $\wedge$, and $\vee$ in terms of $\uparrow$ alone.

$$P \uparrow P \Leftrightarrow \neg(P \wedge P) \Leftrightarrow \neg P \vee \neg P \Leftrightarrow \neg P$$
$$(P \uparrow Q) \uparrow (P \uparrow Q) \Leftrightarrow \neg(P \uparrow Q) \Leftrightarrow P \wedge Q$$
$$(P \uparrow P) \uparrow (Q \uparrow Q) \Leftrightarrow \neg P \uparrow \neg Q \Leftrightarrow \neg(\neg P \wedge \neg Q) \Leftrightarrow P \vee Q$$

In a similar manner, the following equivalences express $\neg$, $\vee$, and $\wedge$ in terms of $\downarrow$ alone

$$P \downarrow P \Leftrightarrow \neg(P \vee P) \Leftrightarrow \neg P \wedge \neg P \Leftrightarrow \neg P$$
$$(P \downarrow Q) \downarrow (P \downarrow Q) \Leftrightarrow \neg(P \downarrow Q) \Leftrightarrow P \vee Q$$
$$(P \downarrow P) \downarrow (Q \downarrow Q) \Leftrightarrow \neg P \downarrow \neg Q \Leftrightarrow P \wedge Q$$

Because a single operator *NAND* or *NOR* is functionally complete, we call each of the sets $\{\uparrow\}$ and $\{\downarrow\}$ a *minimal functionally complete set*, or, in short, a *minimal set*.

The idea of a minimal set of connectives is useful in the economics of the design of electronic circuits. It may be noted that although we can express any formula by an equivalent formula containing a single connective $\uparrow$ or $\downarrow$, we seldom do so because such formulas are often complicated. This fact explains why programming languages as well as our symbolic language have a number of connectives available.

We now list some of the basic properties of the connectives *NAND* and *NOR*.

$$P \uparrow Q \Leftrightarrow Q \uparrow P \qquad P \downarrow Q \Leftrightarrow Q \downarrow P \qquad \text{(commutative)} \tag{1}$$
$$P \uparrow (Q \uparrow R) \Leftrightarrow P \uparrow \neg(Q \wedge R) \Leftrightarrow \neg(P \wedge \neg(Q \wedge R)) \Leftrightarrow \neg P \vee (Q \wedge R) \tag{2}$$

while

$$(P \uparrow Q) \uparrow R \Leftrightarrow (P \wedge Q) \vee \neg R$$

Thus the connective $\uparrow$ is not associative. Similarly

$$P \downarrow (Q \downarrow R) \Leftrightarrow \neg P \wedge (Q \vee R) \qquad (P \downarrow Q) \downarrow R \Leftrightarrow (P \vee Q) \wedge \neg R$$

It is possible to define $P \uparrow Q \uparrow R \Leftrightarrow \neg(P \wedge Q \wedge R)$. However, $P \uparrow Q \uparrow R$ is not equivalent to $P \uparrow (Q \uparrow R)$ or to $(P \uparrow Q) \uparrow R$ or to $Q \uparrow (P \uparrow R)$. This nonassociativity of the connectives *NAND* and *NOR* creates some difficulty in using them.

$$P \uparrow Q \Leftrightarrow \neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q \Leftrightarrow (\neg P \wedge Q) \vee (P \wedge \neg Q) \vee (\neg P \wedge \neg Q) \tag{3}$$

Similarly

$$P \downarrow Q \Leftrightarrow \neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q \Leftrightarrow (\neg P \vee Q) \wedge (P \vee \neg Q) \wedge (\neg P \vee \neg Q) \tag{4}$$

EXERCISES 1-2.14

1 If $A(P, Q, R)$ is given by $P \uparrow (Q \wedge \neg(R \downarrow P))$, find its dual $A^*(P, Q, R)$. Also find formulas which are equivalent to A and A^*, but which contain the connectives $\wedge$, $\vee$, and $\neg$ only.

2 Express $P \rightarrow (\neg P \rightarrow Q)$ in terms of $\uparrow$ only. Express the same formula in terms of $\downarrow$ only.

3 Express $P \uparrow Q$ in terms of $\downarrow$ only.

1-2.15 Two-state Devices and Statement Logic

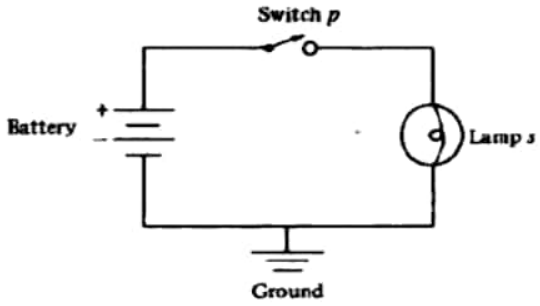
The statement logic that we have discussed so far is called *two-valued logic*, because we admit only those statements having a truth value of true or false. A similar situation exists in various electrical and mechanical devices which are assumed to be in one of two possible configurations; for this reason, they are called two-state devices. We first give several examples of such commonly known devices and then show their connection to two-valued logic.

An electric switch which is used for turning "on" and "off" an electric light is a two-state device. Normally, such a switch is operated manually; however, if it is operated automatically by electric power, we say the switch is relay-operated. A vacuum tube or a transistor is another two-state device in which the current is either passing (conducting) or not passing (nonconducting). A mechanical clutch can be engaged or disengaged. A small doughnut-shaped metal disc with a wire coil wrapped around it (called a magnetic core in computers) may be magnetized in one direction if the current is passed through the coil in one way and may be magnetized in the opposite direction if the current is reversed. Many other examples of two-state devices can be cited. A general discussion of such devices can be given by replacing the word "switch" by the word "gate" to mean a device which permits or stops the flow of not only electric current but any quantity that can go through the device, such as water, information, persons, etc.

Let us first consider the example of an electric lamp controlled by a mechanical switch. Such a circuit is displayed in Fig. 1-2.3. When the switch p is open, there is no current flowing in the circuit and the lamp s is "off." When p is closed, the lamp s is "on." The state of the switch and the lamp can be represented by the table of combinations in Fig. 1-2.3. Let us denote the statements as

P: The switch p is closed.

S: The lamp s is on.



State of switch p	State of light s
closed	on
open	off

FIGURE 1-2.3 A switch as a two-state device.

Then we can rewrite the table of Fig. 1-2.3 as

$p(P)$	$s(S)$
1	1
0	0

Throughout this section we shall denote the truth values of statements $P, Q, \ldots$ by 1 and 0 in place of T and F respectively. At the same time, the input switches such as p or the output indicator such as the lamp s will be assigned the values 1 and 0 to correspond to the states when the current is flowing or not flowing. In such cases the table shown here can be understood to be either a truth table or a table that relates the input and the output values.

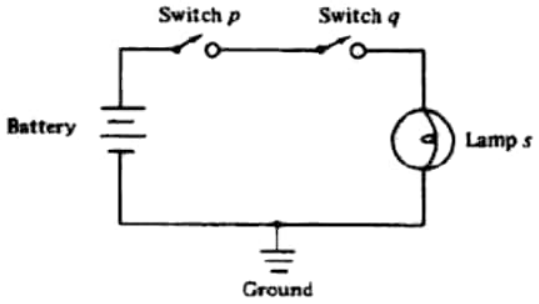
Next, consider an extension of the preceding circuit in which we have two switches p and q in series. The lamp s is turned on whenever both the switches p and q are closed. Such a circuit with its table of combinations is shown in Fig. 1-2.4. In the table, we have used the statements

P: The switch p is closed.

Q: The switch q is closed.

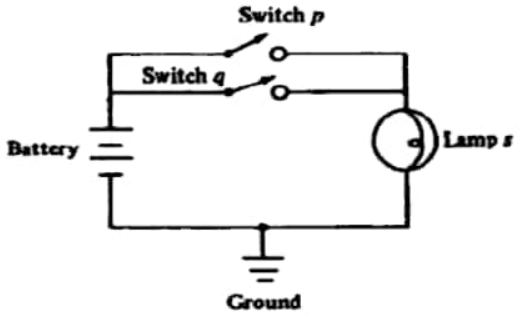
S: The light s is on.

From the table it is clear that $P \wedge Q \Leftrightarrow S$.



P	Q	S
1	1	1
1	0	0
0	1	0
0	0	0

FIGURE 1-2.4 A two-state device for AND logic.



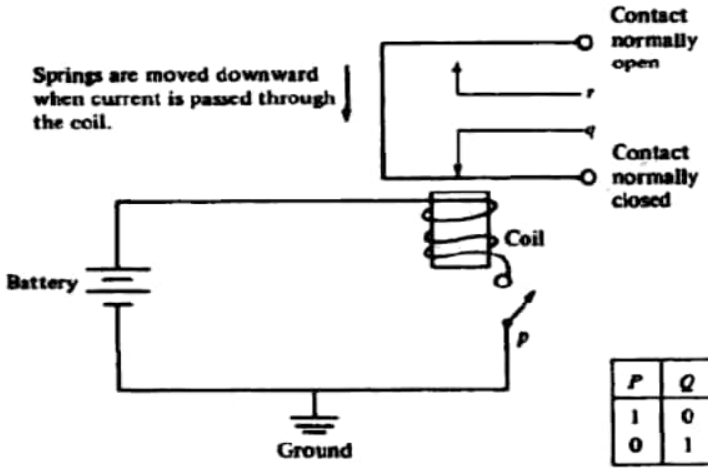
P	Q	S
1	1	1
1	0	1
0	1	1
0	0	0

FIGURE 1-2.5 A two-state device for OR logic.

Figure 1-2.5 contains a circuit and its associated table of combinations in which two switches are connected in parallel. From the table it is clear that $P \vee Q \Leftrightarrow S$.

We have just shown how the connectives $\wedge$ and $\vee$ correspond to switches connected in series and in parallel, respectively.

We now consider an example of a switch controlled by a relay. A simplified configuration and its associated table of combinations are given in Fig. 1-2.6. When the switch p is open (P is false, because we shall use P: The switch p is closed.), no current flows and the contact q which is normally closed remains closed and the contact r remains open. When p is closed, the current will flow from the battery through the coil which will cause the movement of a relay armature, which in turn causes the springs to move downward and the normally closed contact q to open while the normally open contact r closes. If p is opened, then the contact q closes and r opens because the spring moves upward to its original position. Thus with the statements P, Q, and R denoting the switches p, q, and r to be closed respectively, we can represent the operation of the device by the table of combinations in Fig. 1-2.6. In fact, the switches q and r are always in the opposite states, that is, $Q \Leftrightarrow \neg R$, $Q \Leftrightarrow \neg P$, and $R \Leftrightarrow P$. Note that the output Q is the negation of the input P.



P	Q	R
1	0	1
0	1	0

FIGURE 1-2.6 A relay-switching device.

Instead of representing the logical connectives $\neg$, $\vee$, and $\wedge$ by the circuits just given or by some other equivalent circuits (consisting of semiconductor devices, for example), they are generally represented by block diagrams or *gates*. Each gate has one or more input wires and one output wire.

The logical connectives $\neg$, $\vee$, and $\wedge$ will be denoted by the symbols $^-$, $+$, and $\cdot$ respectively in the remainder of this section. This denotation is in keeping with the terminology used in switching theory.

The block-diagram symbol for an *OR* gate is shown in Fig. 1-2.7 along with the table of combinations relating the inputs and output of the gate. Any such gate is also called a *module*.

Figure 1-2.8 shows the block-diagram symbol for an *AND* gate as well as its associated table of combinations.

The negation operator can be represented by the block diagram in Fig. 1-2.9.

The block diagrams not only replace switches and relays but can also be used to represent "gates" in a more general sense. We may use p to denote voltage potential of an input which is "high" or "low" to allow a transistor to be in a conducting or nonconducting state. It is therefore convenient to use these modules and interpret the symbols according to the context. The number of inputs to *OR* gates and *AND* gates can be extended to more than 2.

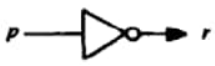
The above modules, or gates, can be interconnected to realize various logical expressions. These systems of modules are known as logic or combinational networks. Figure 1-2.10*a* shows a logic network with three inputs a, b, and c and an output expression $(a + b) \cdot c$. Networks which form expressions $(a \cdot \bar{b}) + (\bar{a} \cdot b)$ and $(a + b) \cdot (\bar{a} + \bar{b})$ are given in Fig. 1-2.10*b* and *c* respectively.

Input		Output
p	q	$r(p + q)$
1	1	1
1	0	1
0	1	1
0	0	0

FIGURE 1-2.7 An *OR* gate.

Input		Output
p	q	$r(p \cdot q)$
1	1	1
1	0	0
0	1	0
0	0	0

FIGURE 1-2.8 An *AND* gate.



Input	Output
p	$r(\bar{p})$
1	0
0	1

FIGURE 1-2.9 A *NOT* gate.

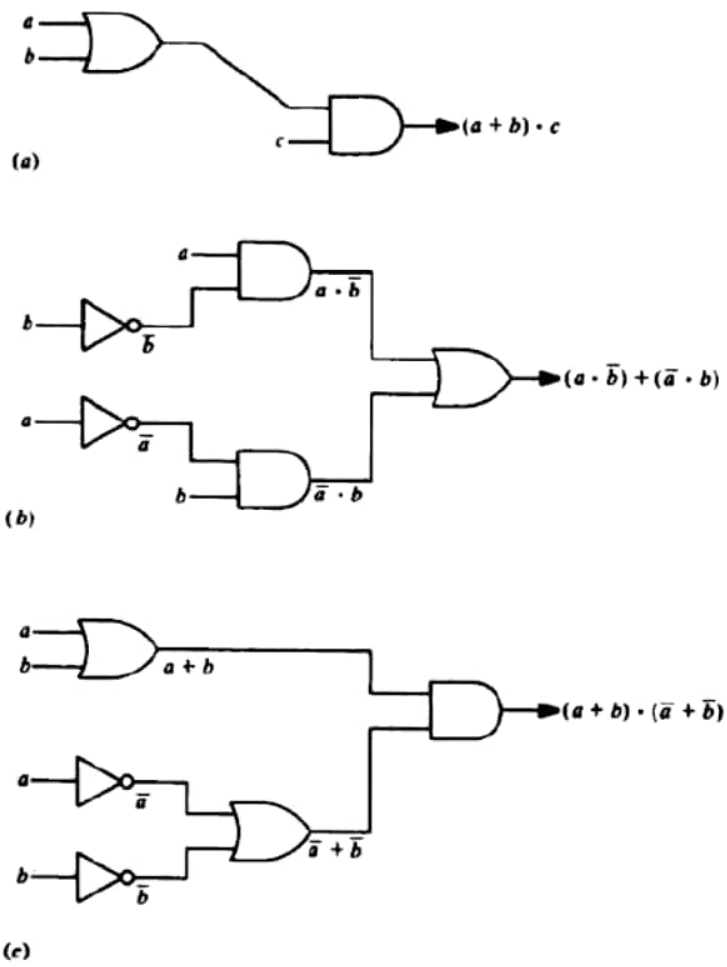


FIGURE 1-2.10 Logic networks. (a) Logic network for $(a + b)\cdot c$. (b) Logic network for $a\cdot\bar{b} + \bar{a}\cdot b$. (c) Logic network for $(a + b)\cdot(\bar{a} + \bar{b})$.

In the remainder of this section it is assumed that a variable and its negation are available without having to use an inverter. The mechanical relay was an example of a device which supplied both. There are other basic transistor devices which also supply both.

In order to simplify expressions by reducing the number of parentheses used, we specify that $\cdot$ has precedence over $+$. For example, $(a\cdot b) + (c\cdot d)$ will be written as $a\cdot b + c\cdot d$.

Consider the logical expression $a\cdot b + c\cdot d$, which consists of a disjunction of two conjunctions. A logic network to realize this expression can be a two-level network as shown in Fig. 1-2.11*a*. A *two-level network* is one in which the longest path through which information must pass from input to output is two gates. A two-level network consisting of *AND* gates at the input stage followed by *OR* gates at the output stage is sometimes called a *sum-of-products* (*AND*-to-OR) *network*.

Another possibility in a two-level network is to have *OR* gates at the input stage followed by *AND* gates at the output of the network. Such a configuration

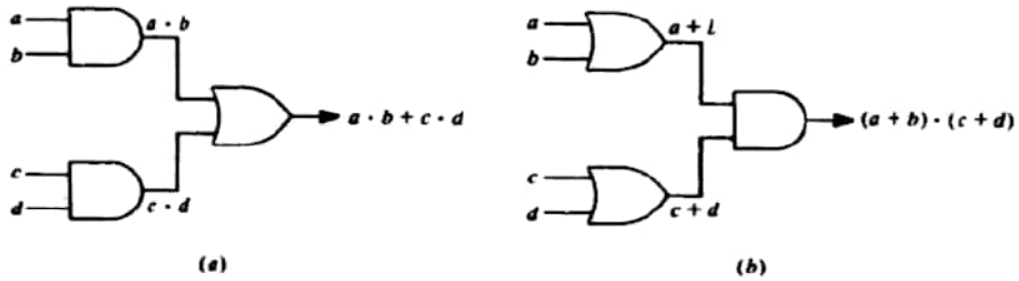


FIGURE 1-2.11 Two-level networks. (*a*) *AND-to-OR* logic network. (*b*) *OR-to-AND* logic network.

is shown for the expression $(a + b) \cdot (c + d)$ in Fig. 1-2.11*b* and is called a *product-of-sums* (*OR*-to-*AND*) *network*.

Other types of gates frequently used in computers are *NOR* gates and *NAND* gates. Figure 1-2.12*a* represents a *NOR* gate and its associated table. Figure 1-2.12*b* is an equivalent representation of a *NOR* gate consisting of an *OR* gate followed by an inverter.

Figure 1-2.13*a* shows a *NAND* gate with its table. The other part of the diagram is an equivalent *NAND* gate representation consisting of an *AND* gate followed by an inverter.

In Sec. 1-2.14 it was shown that each of the connectives *NAND* ($\uparrow$) and *NOR* ($\downarrow$) is functionally complete; either can be used to obtain the *AND*, *OR*, and *NOT* operations. The realizations of these operations in terms of *NAND* gates and *NOR* gates only are given in Figs. 1-2.14 and 1-2.15 respectively. The

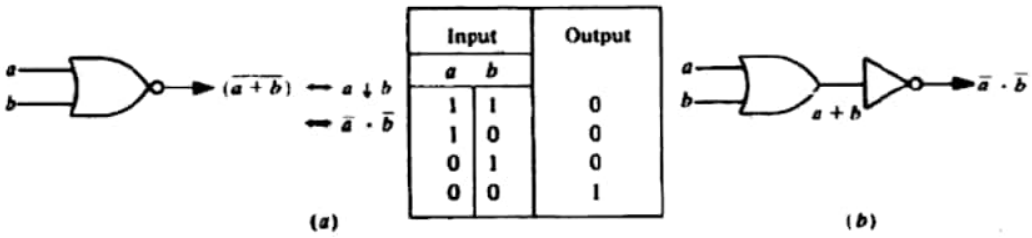


FIGURE 1-2.12 A *NOR* gate. (*a*) Block-diagram symbol and truth table for a *NOR* gate. (*b*) Equivalent *NOR* gate network.

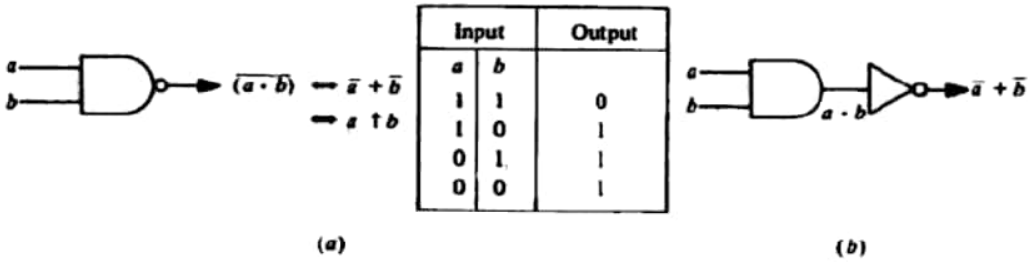


FIGURE 1-2.13 A *NAND* gate. (*a*) Block-diagram symbol and truth table for a *NAND* gate. (*b*) Equivalent *NAND* gate network.

use of a single type of gate such as *NAND* or *NOR* is often preferred to a variety of gate types because it is cheaper to produce and maintain just one type of gate.

One method used to produce a circuit containing only *NAND* or only *NOR* gates is to replace the *AND*, *OR*, and *NOT* gates by NAND or NOR gates, according to Figs. 1-2.14 and 1-2.15. This replacement often results in a circuit which contains more *NAND* or *NOR* gates than necessary. There are other techniques to generate equivalent circuits containing fewer *NAND* or *NOR* gates. As an example, each of the circuits in Fig. 1-2.11, if substituted according to Figs. 1-2.14 and 1-2.15, would require nine gates. But Fig. 1-2.16 contains different realizations of the same expressions, using only three gates.

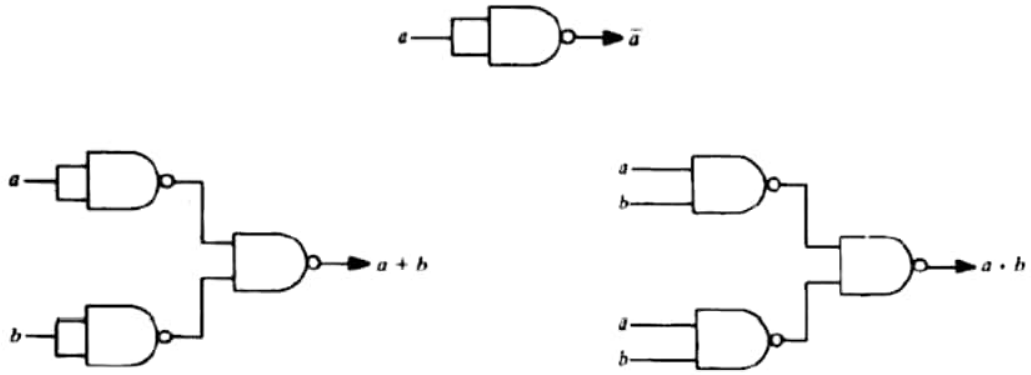


FIGURE 1-2.14. *NAND* gate generation of $+$, $\cdot$, and $^-$ operations.

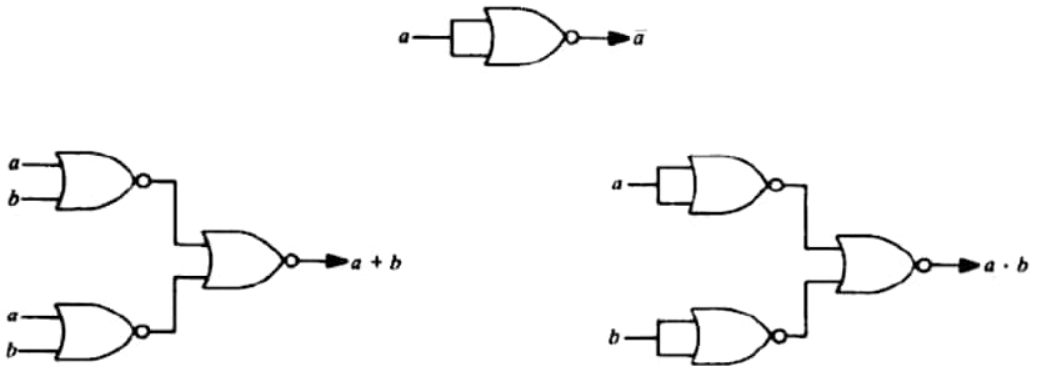


FIGURE 1-2.15. *NOR* gate generation of $+$, $\cdot$, and $^-$ operations.

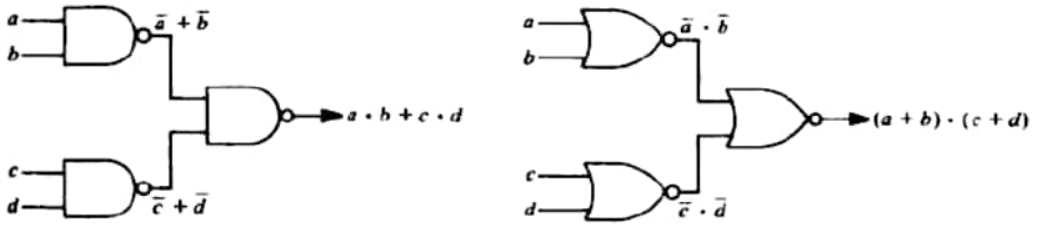


FIGURE 1-2.16. Two-level *NAND* and *NOR* gate networks.

So far we have discussed certain elements which correspond to some of the connectives of statement logic. Normally, if a formula containing these connectives is given, we can physically realize a circuit that corresponds to the formula by replacing the connectives by the appropriate gates and the variables by certain physical quantities such as voltage, current, etc. This method may, however, not yield the best design from the point of view of using a minimum number of gates, or the design it yields may not be a minimal design in some other sense. Sometimes even the formula may not be available and all we may have is its truth table. Even in this case, we may be required to physically realize the formula, not in any manner, but in some minimal way. The design procedure in both cases consists of the following steps:

1 Express the given information in terms of logical variables.
2 Get a formula for the required output in terms of the variables defined in step 1, using the logical connectives.
3 Obtain a formula which is logically equivalent to the one formed in step 2 and which will result in a least expensive (minimal) physical realization.
4 Replace the logical connectives in the formula in step 3 by proper logic blocks.

A discussion of step 3 is considered in some detail in Chap. 4. Steps 1 and 2 are explained by means of an example.

EXAMPLE 1 A certain government installation has an intruder alarm system which is to be operative only if a manual master switch situated at a security office is in the closed position. If this master switch is on (i.e., closed), an alarm will be sounded if a door to a restricted-access area within the installation is disturbed, or if the main gates to the installation are opened without the security officer first turning a special switch to the closed position. The restricted-access area door is equipped with a sensing device that causes a switch to set off the alarm if this door is disturbed in any way whenever the master switch is closed. However, the main gates are opened during daytime hours to allow the public to enter the installation grounds. Furthermore, at certain specified time intervals during a 24-hour period, the master switch is turned off to allow the authorized personnel to enter or leave the restricted-access area. During the period when the main gates are open and the master switch is turned off, it is required that some automatic recording instrument make an entry every time the door to the restricted-access area is opened.

SOLUTION *Step 1* Let us first designate each primary statement by a variable. This assignment will permit us to consider the assignment of all possible truth values to the variables. Thus let

A: The alarm will be given.
M: The manual master switch is closed.
G: The main gates to the installation are open.
R: The restricted-area door has been disturbed.
S: The special switch is closed.
E: The recording equipment is activated.

Step 2 The output variables are A and E. The conditions given in the problem require that

$$A \Leftrightarrow M \cdot (R + (G \cdot \bar{S}))$$
$$E \Leftrightarrow \bar{M} \cdot G \cdot R \qquad ////$$

EXERCISES 1-2.15

1 Construct a circuit diagram for a simple elevator control circuit which operates as follows. When a person pushes the button to summon the elevator to a floor, the elevator responds to this request unless it is already answering a previous call to another floor. In this latter case, the request is ignored. Assume that there are only three floors in the building.

2 For the formula $(P \wedge Q) \vee (\neg R \wedge \neg P)$, draw a corresponding circuit diagram using (*a*) *NOT*, *AND*, and *OR* gates; (*b*) *NAND* gates only.

EXERCISES 1-2

1 Show the equivalences (1) to (9) listed in Table 1-2.15 of Sec. 1-2.9.

2 Show the implications (1) to (12) listed in Table 1-2.17 of Sec. 1-2.11.

3 Show the following

$$\neg(P \uparrow Q) \Leftrightarrow \neg P \downarrow \neg Q \qquad \neg(P \downarrow Q) \Leftrightarrow \neg P \uparrow \neg Q$$

4 Write a formula which is equivalent to the formula

$$P \wedge (Q \rightleftarrows R)$$

and contains the connective *NAND* ($\uparrow$) only. Obtain an equivalent formula whicl contains the connective *NOR* ($\downarrow$) only.

5 Show that $P \wedge \neg P \wedge Q \Rightarrow R$ and $R \Rightarrow P \vee \neg P \vee Q$.

6 Show the equivalences (1) to (3) given in Sec. 1-2.14.

7 Show that the set $\{\neg, \rightarrow\}$ is functionally complete.

8 A connective denoted by $\dot{\rightarrow}$ is defined by Table 1-2.21. Show that $\{\neg, \dot{\rightarrow}\}$ is functionally complete.

9 Show the following equivalences.
(*a*) $A \rightarrow (P \vee C) \Leftrightarrow (A \wedge \neg P) \rightarrow C$
(*b*) $(P \rightarrow C) \wedge (Q \rightarrow C) \Leftrightarrow (P \vee Q) \rightarrow C$
(*c*) $((Q \wedge A) \rightarrow C) \wedge (A \rightarrow (P \vee C)) \Leftrightarrow (A \wedge (P \rightarrow Q)) \rightarrow C$
(*d*) $((P \wedge Q \wedge A) \rightarrow C) \wedge (A \rightarrow (P \vee Q \vee C)) \Leftrightarrow (A \wedge (P \rightleftarrows Q)) \rightarrow C$
(See Sec. 1-4.4 for application of these equivalences.)

Table 1-2.21

P	Q	$P \dot{\rightarrow} Q$
T	T	T
T	F	T
F	T	F
F	F	T

10 Obtain formulas having the simplest possible form which are equivalent to the formulas given here.
(*a*) $((P \rightarrow Q) \rightleftarrows (\neg Q \rightarrow \neg P)) \wedge R$
(*b*) $P \vee (\neg P \vee (Q \wedge \neg Q))$
(*c*) $(P \wedge (Q \wedge S)) \vee (\neg P \wedge (Q \wedge S))$

11 A chemical plant produces sulfuric acid with a pH value in the range 2.5 to 3.0. There is a main tank in which the acid is held while it is being diluted or heated (concentrated) to put it within this range. If an output valve which empties the tank is open, no heating or diluting can take place because the current batch is being moved out of the tank for final storage. Dilution will occur when the pH is less than 2.5 (as long as the tank is not full) and also whenever the level of the liquid remains below a specified level (the tank must be filled somehow after it has been emptied). Heating will take place when the pH is greater than 3.0 as long as the tank is not full. (A full tank would spill over during the agitation caused by vigorous heating.)

Express the control circuit as block diagrams for both the heating control and the dilution control.

1-3 NORMAL FORMS

Let $A(P_1, P_2, \ldots, P_n)$ be a statement formula where $P_1, P_2, \ldots, P_n$ are the atomic variables. If we consider all possible assignments of the truth values to $P_1, P_2, \ldots, P_n$ and obtain the resulting truth values of the formula A, then we get the truth table for A. Such a truth table contains 2^n rows. The formula may have the truth value T for all possible assignments of the truth values to the variables $P_1, P_2, \ldots, P_n$. In this case, A is said to be identically true, or a tautology. If A has the truth value F for all possible assignments of the truth values to $P_1, P_2, \ldots, P_n$, then A is said to be identically false, or a contradiction. Finally, if A has the truth value T for at least one combination of truth values assigned to $P_1, P_2, \ldots, P_n$, then A is said to be *satisfiable*.

The problem of determining, in a finite number of steps, whether a given statement formula is a tautology or a contradiction or at least satisfiable is known as a *decision problem*. Obviously, the construction of truth tables involves a finite number of steps, and, as such, a decision problem in the statement calculus always has a solution. Similarly, decision problems can be posed for other logical systems, particularly for the predicate calculus. However, in the latter case, the solution of the decision problem may not be simple.

As was mentioned earlier, the construction of truth tables may not be practical, even with the aid of a computer. We therefore consider other procedures known as reduction to normal forms.

1-3.1 Disjunctive Normal Forms

It will be convenient to use the word "product" in place of "conjunction" and "sum" in place of "disjunction" in our current discussion.

A product of the variables and their negations in a formula is called an *elementary product*. Similarly, a sum of the variables and their negations is called an *elementary sum*.

Let P and Q be any two atomic variables. Then P, $\neg P \wedge Q$, $\neg Q \wedge P \wedge$

$\neg P$, $P \wedge \neg P$, and $Q \wedge \neg P$ are some examples of elementary products. On the other hand, P, $\neg P \vee Q$, $\neg Q \vee P \vee \neg P$, $P \vee \neg P$, and $Q \vee \neg P$ are examples of elementary sums of the two variables. Any part of an elementary sum or product which is itself an elementary sum or product is called a *factor* of the original elementary sum or product. Thus $\neg Q$, $P \wedge \neg P$, and $\neg Q \wedge P$ are some of the factors of $\neg Q \wedge P \wedge \neg P$. The following statements hold for elementary sums and products.

> A necessary and sufficient condition for an elementary product to be identically false is that it contain at least one pair of factors in which one is the negation of the other.
>
> A necessary and sufficient condition for an elementary sum to be identically true is that it contain at least one pair of factors in which one is the negation of the other.

We shall now prove the first of these two statements. The proof of the second will follow along the same lines.

We know that for any variable P, $P \wedge \neg P$ is identically false. Hence if $P \wedge \neg P$ appears in the elementary product, then the product is identically false. On the other hand, if an elementary product is identically false and does not contain at least one factor of this type, then we can assign truth values T and F to variables and negated variables, respectively, that appear in the product. This assignment would mean that the elementary product has the truth value T. But that statement is contrary to our assumption. Hence the statement follows.

A formula which is equivalent to a given formula and which consists of a sum of elementary products is called a *disjunctive normal form* of the given formula.

We shall first discuss a procedure by which one can obtain a disjunctive normal form of a given formula. It has already been shown that if the connectives $\rightarrow$ and $\rightleftarrows$ appear in the given formula, then an equivalent formula can be obtained in which "$\rightarrow$" and "$\rightleftarrows$" are replaced by $\wedge$, $\vee$, and $\neg$. This statement would be true of any other connective yet undefined. The truth of this statement will become apparent after our discussion of principal disjunctive normal forms. Therefore, there is no loss of generality in assuming that the given formula contains the connectives $\wedge$, $\vee$, and $\neg$ only.

If the negation is applied to the formula or to a part of the formula and not to the variables appearing in it, then by using De Morgan's laws an equivalent formula can be obtained in which the negation is applied to the variables only. After this step, the formula obtained may still fail to be in disjunctive normal form because there may be some parts of it which are products of sums. By repeated application of the distributive laws we obtain the required normal form.

EXAMPLE 1 Obtain disjunctive normal forms of (*a*) $P \wedge (P \rightarrow Q)$; (*b*) $\neg(P \vee Q) \rightleftarrows (P \wedge Q)$.

SOLUTION

(*a*) $P \wedge (P \rightarrow Q) \Leftrightarrow P \wedge (\neg P \vee Q) \Leftrightarrow (P \wedge \neg P) \vee (P \wedge Q)$

(*b*) $\neg(P \vee Q) \rightleftarrows (P \wedge Q)$

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

$$[\text{using } R \rightleftarrows S \Leftrightarrow (R \wedge S) \vee (\neg R \wedge \neg S)]$$

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

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

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

which is the required disjunctive normal form. ////

The disjunctive normal form of a given formula is not unique. In fact, different disjunctive normal forms can be obtained for a given formula if the distributive laws are applied in different ways. Apart from this fact, the factors in each elementary product, as well as the factors in the sum, can be commuted. However, we shall not consider as distinct the various disjunctive normal forms obtained by reordering the factors either in the elementary products or in the sums.

Consider the formula $F \vee (Q \wedge R)$. Here the formula is already in the disjunctive normal form. However, we may write

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R) \Leftrightarrow (P \wedge P) \vee (P \wedge Q) \vee (P \wedge R) \vee (Q \wedge R)$$

the last equivalent formula being another equivalent disjunctive normal form. Of course, different disjunctive normal forms of the same formula are equivalent. In order to arrive at a unique normal form of a given formula, we introduce the principal disjunctive normal form in Sec. 1-3.3.

Finally, we remark that a given formula is identically false if every elementary product appearing in its disjunctive normal form is identically false. For the assumption to be true, every elementary product would have to have at least two factors, of which one is the negation of the other.

1-3.2 Conjunctive Normal Forms

A formula which is equivalent to a given formula and which consists of a product of elementary sums is called a *conjunctive normal form* of the given formula.

The method for obtaining a conjunctive normal form of a given formula is similar to the one given for disjunctive normal forms and will be demonstrated by examples. Again, the conjunctive normal form is not unique. Furthermore, a given formula is identically true if every elementary sum in its conjunctive normal form is identically true. This would be the case if every elementary sum appearing in the formula had at least two factors, of which one is the negation of the other.

EXAMPLE 1 Obtain a conjunctive normal form of each of the formulas given in Example 1 of Sec. 1-3.1.

SOLUTION

(a) $P \wedge (P \rightarrow Q) \Leftrightarrow P \wedge (\neg P \vee Q)$. Hence $P \wedge (\neg P \vee Q)$ is a required form.

(b) $$\begin{aligned}\neg(P \vee Q) \rightleftarrows (P \wedge Q) &\Leftrightarrow (\neg(P \vee Q) \rightarrow (P \wedge Q)) \wedge ((P \wedge Q) \\ &\quad \rightarrow \neg(P \vee Q)) \\ [\text{using } R \rightleftarrows S &\Leftrightarrow (R \rightarrow S) \wedge (S \rightarrow R)] \\ &\Leftrightarrow ((P \vee Q) \vee (P \wedge Q)) \wedge (\neg(P \wedge Q) \\ &\quad \vee (\neg P \wedge \neg Q)) \\ &\Leftrightarrow ((P \vee Q \vee P) \wedge (P \vee Q \vee Q)) \\ &\quad \wedge ((\neg P \vee \neg Q) \vee (\neg P \wedge \neg Q)) \\ &\Leftrightarrow (P \vee Q \vee P) \wedge (P \vee Q \vee Q) \\ &\quad \wedge (\neg P \vee \neg Q \vee \neg P) \wedge (\neg P \vee \neg Q \vee \neg Q)\end{aligned}$$

////

EXAMPLE 2 Show that the formula $Q \vee (P \wedge \neg Q) \vee (\neg P \wedge \neg Q)$ is a tautology.

SOLUTION First we obtain a conjunctive normal form of the given formula.

$$\begin{aligned}Q \vee (P \wedge \neg Q) \vee (\neg P \wedge \neg Q) &\Leftrightarrow Q \vee ((P \vee \neg P) \wedge \neg Q) \\ &\Leftrightarrow (Q \vee (P \vee \neg P)) \wedge (Q \vee \neg Q) \\ &\Leftrightarrow (Q \vee P \vee \neg P) \wedge (Q \vee \neg Q)\end{aligned}$$

Since each of the elementary sums is a tautology, the given formula is a tautology. ////

1-3.3 Principal Disjunctive Normal Forms

Let P and Q be two statement variables. Let us construct all possible formulas which consist of conjunctions of P or its negation and conjunctions of Q or its negation. None of the formulas should contain both a variable and its negation. Furthermore, any formula which is obtained by commuting the formulas in the conjunction is not included in the list because such a formula will be equivalent to one included in the list. For example, either $P \wedge Q$ or $Q \wedge P$ is included, but not both. For two variables P and Q, there are 2^2 such formulas given by

$$P \wedge Q \qquad P \wedge \neg Q \qquad \neg P \wedge Q \qquad \text{and} \qquad \neg P \wedge \neg Q$$

These formulas are called *minterms* or Boolean conjunctions of P and Q. From the truth tables of these minterms, it is clear that no two minterms are equivalent. Each minterm has the truth value T for exactly one combination of the truth values of the variables P and Q. This fact is shown in Table 1-3.1.

We assert that if the truth table of any formula containing only the variables P and Q is known, then one can easily obtain an equivalent formula which consists of a disjunction of some of the minterms. This statement is demonstrated as follows.

For every truth value T in the truth table of the given formula, select the

Table 1-3.1

P	Q	$P \wedge Q$	$P \wedge \neg Q$	$\neg P \wedge Q$	$\neg P \wedge \neg Q$
T	T	T	F	F	F
T	F	F	T	F	F
F	T	F	F	T	F
F	F	F	F	F	T

minterm which also has the value T for the same combination of the truth values of P and Q. The disjunction of these minterms will then be equivalent to the given formula.

This discussion provides the basis for a proof that a formula containing any connective is equivalent to a formula containing $\wedge$, $\vee$, and $\neg$.

For a given formula, an equivalent formula consisting of disjunctions of minterms only is known as its *principal disjunctive normal form*. Such a normal form is also called the *sum-of-products canonical form*.

EXAMPLE 1 The truth tables for $P \rightarrow Q$, $P \vee Q$, and $\neg(P \wedge Q)$ are given in Table 1-3.2. Obtain the principal disjunctive normal forms of these formulas.

SOLUTION

$$P \rightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q)$$

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

$$\neg(P \wedge Q) \Leftrightarrow (P \wedge \neg Q) \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q) \qquad ////$$

Note that the number of minterms appearing in the normal form is the same as the number of entries with the truth value T in the truth table of the given formula. Thus every formula which is not a contradiction has an equivalent principal disjunctive normal form. Further, such a normal form is unique, except for the rearrangements of the factors in the disjunctions as well as in each of the minterms. One can get a unique normal form by imposing a certain order in which the variables appear in the minterms as well as a definite order in which the minterms appear in the disjunction. In that case, if two given formulas are equivalent, then both of them must have identical principal disjunctive normal forms. Therefore, if we can devise a method other than the construction of truth tables to obtain the principal disjunctive normal form of a given formula, then

Table 1-3.2

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

the same method can be used to determine whether two given formulas are equivalent.

Although our discussion of the principal disjunctive normal form was restricted to formulas containing only two variables, it is possible to define the minterms for three or more variables. Minterms for the three variables P, Q, and R are

$$P \wedge Q \wedge R \qquad P \wedge Q \wedge \neg R \qquad P \wedge \neg Q \wedge R \qquad P \wedge \neg Q \wedge \neg R$$
$$\neg P \wedge Q \wedge R \qquad \neg P \wedge Q \wedge \neg R \qquad \neg P \wedge \neg Q \wedge R \qquad \neg P \wedge \neg Q \wedge \neg R$$

These minterms satisfy properties similar to those given for two variables. An equivalent principal disjunctive normal form of any formula which depends upon the variables P, Q, and R can be obtained. Note that there are 2^3 minterms for three variables or, more generally, 2^n minterms for n variables. For any formula containing n variables which are denoted by $P_1, P_2, \ldots, P_n$, an equivalent disjunctive normal form can be obtained by selecting appropriate minterms out of the set of 2^n possible minterms.

If a formula is a tautology, then obviously all the minterms appear in its principal disjunctive normal form; it is also possible to determine whether a given formula is a tautology by obtaining its principal disjunctive normal form.

In order to obtain the principal disjunctive normal form of a given formula without constructing its truth table, one may first replace the conditionals and biconditionals by their equivalent formulas containing only $\wedge$, $\vee$, and $\neg$. Next, the negations are applied to the variables by using De Morgan's laws followed by the application of distributive laws, as was done earlier in obtaining the disjunctive or conjunctive normal forms. Any elementary product which is a contradiction is dropped. Minterms are obtained in the disjunctions by introducing the missing factors. Identical minterms appearing in the disjunctions are deleted. This procedure is demonstrated by means of examples.

EXAMPLE 2 Obtain the principal disjunctive normal forms of (*a*) $\neg P \vee Q$; (*b*) $(P \wedge Q) \vee (\neg P \wedge R) \vee (Q \wedge R)$.

SOLUTION

(*a*) $\neg P \vee Q \Leftrightarrow (\neg P \wedge (Q \vee \neg Q)) \vee (Q \wedge (P \vee \neg P))$

$(A \wedge \mathbf{T} \Leftrightarrow A)$

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

(distributive laws)

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

(commutative law and $P \vee P \Leftrightarrow P$)

(See Example 1.)

(*b*) $(P \wedge Q) \vee (\neg P \wedge R) \vee (Q \wedge R)$

$\Leftrightarrow (P \wedge Q \wedge (R \vee \neg R)) \vee (\neg P \wedge R \wedge (Q \vee \neg Q))$
$\vee (Q \wedge R \wedge (P \vee \neg P))$

$\Leftrightarrow (P \wedge Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (\neg P \wedge Q \wedge R)$
$\vee (\neg P \wedge \neg Q \wedge R)$ ////

EXAMPLE 3 Show that the following are equivalent formulas.

(*a*) $P \vee (P \wedge Q) \Leftrightarrow P$
(*b*) $P \vee (\neg P \wedge Q) \Leftrightarrow P \vee Q$

SOLUTION We write the principal disjunctive normal form of each formula and compare these normal forms.

$$(a)\quad P \vee (P \wedge Q) \Leftrightarrow (P \wedge (Q \vee \neg Q)) \vee (P \wedge Q) \Leftrightarrow (P \wedge Q) \vee (P \wedge \neg Q)$$
$$P \Leftrightarrow P \wedge (Q \vee \neg Q) \Leftrightarrow (P \wedge Q) \vee (P \wedge \neg Q)$$
$$(b)\quad P \vee (\neg P \wedge Q) \Leftrightarrow (P \wedge (Q \vee \neg Q)) \vee (\neg P \wedge Q)$$
$$\Leftrightarrow (P \wedge Q) \vee (P \wedge \neg Q) \vee (\neg P \wedge Q)$$
$$P \vee Q \Leftrightarrow (P \wedge (Q \vee \neg Q)) \vee (Q \wedge (P \vee \neg P))$$
$$\Leftrightarrow (P \wedge Q) \vee (P \wedge \neg Q) \vee (\neg P \wedge Q) \qquad ////$$

EXAMPLE 4 Obtain the principal disjunctive normal form of

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

SOLUTION Using $P \rightarrow Q \Leftrightarrow \neg P \vee Q$ and De Morgan's law, we obtain

$$P \rightarrow ((P \rightarrow Q) \wedge \neg(\neg Q \vee \neg P))$$
$$\Leftrightarrow \neg P \vee ((\neg P \vee Q) \wedge (Q \wedge P))$$
$$\Leftrightarrow \neg P \vee (\neg P \wedge (Q \wedge P)) \vee (Q \wedge (Q \wedge P))$$
$$\Leftrightarrow \neg P \vee (Q \wedge P)$$
$$\Leftrightarrow (\neg P \wedge (Q \vee \neg Q)) \vee (Q \wedge P)$$
$$\Leftrightarrow (\neg P \wedge Q) \vee (\neg P \wedge \neg Q) \vee (P \wedge Q) \qquad ////$$

The procedure described above becomes tedious if the given formula is complicated and contains more than two or three variables. When the number of variables is large, even a comparison of two principal disjunctive normal forms becomes cumbersome. In Sec. 1-3.5, we describe an ordering procedure for the variables and a notation which make such a comparison easy. We also discuss in Chap. 2 a computer program to obtain the sum-of-products canonical form for a given formula.

1-3.4 Principal Conjunctive Normal Forms

In order to define the principal conjunctive normal form, we first define formulas which are called maxterms. For a given number of variables, the *maxterm* consists of disjunctions in which each variable or its negation, but not both, appears only once. Thus the maxterms are the duals of minterms. Either from the duality principle or directly from the truth tables, it can be ascertained that each of the maxterms has the truth value F for exactly one combination of the truth values of the variables. Also different maxterms have the truth value F for different combinations of the truth values of the variables.

For a given formula, an equivalent formula consisting of conjunctions of the maxterms only is known as its *principal conjunctive normal form*. This normal

form is also called the *product-of-sums canonical form*. Every formula which is not a tautology has an equivalent principal conjunctive normal form which is unique except for the rearrangement of the factors in the maxterms as well as in their conjunctions. The method for obtaining the principal conjunctive normal form for a given formula is similar to the one described previously for the principal disjunctive normal form. In fact, all the assertions made for the principal disjunctive normal forms can also be made for the principal conjunctive normal forms by the duality principle.

If the principal disjunctive (conjunctive) normal form of a given formula A containing n variables is known, then the principal disjunctive (conjunctive) normal form of $\neg A$ will consist of the disjunction (conjunction) of the remaining minterms (maxterms) which do not appear in the principal disjunctive (conjunctive) normal form of A. From $A \Leftrightarrow \neg\neg A$ one can obtain the principal conjunctive (disjunctive) normal form of A by repeated applications of De Morgan's laws to the principal disjunctive (conjunctive) normal form of $\neg A$. This procedure will be illustrated by an example.

In order to determine whether two given formulas A and B are equivalent, one can obtain any of the principal normal forms of the two formulas and compare them. It is not necessary to assume that both formulas have the same variables. In fact, each formula can be assumed to depend upon all the variables that appear in both formulas, by introducing the missing variables and then reducing them to their principal normal forms.

EXAMPLE 1 Obtain the principal conjunctive normal form of the formula S given by $(\neg P \rightarrow R) \wedge (Q \rightleftarrows P)$.

SOLUTION

$$
\begin{aligned}
&(\neg P \rightarrow R) \wedge (Q \rightleftarrows P) \\
&\quad \Leftrightarrow (P \vee R) \wedge ((Q \rightarrow P) \wedge (P \rightarrow Q)) \\
&\quad \Leftrightarrow (P \vee R) \wedge (\neg Q \vee P) \wedge (\neg P \vee Q) \\
&\quad \Leftrightarrow (P \vee R \vee (Q \wedge \neg Q)) \wedge (\neg Q \vee P \vee (R \wedge \neg R)) \\
&\qquad \wedge (\neg P \vee Q \vee (R \wedge \neg R)) \\
&\quad \Leftrightarrow (P \vee Q \vee R) \wedge (P \vee \neg Q \vee R) \wedge (P \vee \neg Q \vee \neg R) \\
&\qquad \wedge (\neg P \vee Q \vee R) \wedge (\neg P \vee Q \vee \neg R)
\end{aligned}
$$

Now the conjunctive normal form of $\neg S$ can easily be obtained by writing the conjunction of the remaining maxterms; thus, $\neg S$ has the principal conjunctive normal form

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

By considering $\neg\neg S$, we obtain

$$
\begin{aligned}
&\neg(P \vee Q \vee \neg R) \vee \neg(\neg P \vee \neg Q \vee R) \vee \neg(\neg P \vee \neg Q \vee \neg R) \\
&\qquad \Leftrightarrow (\neg P \wedge \neg Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (P \wedge Q \wedge R)
\end{aligned}
$$

which is the principal disjunctive normal form of S. ////

EXAMPLE 2 The truth table for a formula A is given in Table 1-3.3. Determine its disjunctive and conjunctive normal forms.

SOLUTION By choosing the minterms corresponding to each T value of A, we obtain

$$A \Leftrightarrow (P \wedge \neg Q \wedge R) \vee (\neg P \wedge Q \wedge R) \vee (\neg P \wedge Q \wedge \neg R) \vee (\neg P \wedge \neg Q \wedge \neg R)$$

Similarly

$$A \Leftrightarrow (\neg P \vee \neg Q \vee \neg R) \wedge (\neg P \vee \neg Q \vee R) \wedge (\neg P \vee Q \vee R) \wedge (P \vee Q \vee \neg R)$$

Here the maxterms appearing in the normal form correspond to the F values of A. The maxterms are written down by including the variable if its truth value is F and its negation if the value is T. ////

1-3.5 Ordering and Uniqueness of Normal Forms

Given any n statement variables, let us first arrange them in some fixed order. If capital letters are used to denote the variables, then they may be arranged in alphabetical order. If subscripted letters are also used, then the following is an illustration of the order that may be used:

$$A, B, \ldots, Z, A_1, B_1, \ldots, Z_1, A_2, B_2, \ldots$$

As an example, if the variables are P_1, Q, R_2, S_1, T_2, and Q_3, then they may be arranged as

$$Q, P_1, S_1, T_2, Q_3, R_2$$

Once the variables have been arranged in a particular order, it is possible to designate them as the first variable, second variable, and so on.

Let us assume that n variables are given and are arranged in a particular order. The 2^n minterms corresponding to the n variables can be designated by $m_0, m_1, \ldots, m_{2^n-1}$. If we write the subscript of any particular minterm in binary and add an appropriate number of zeros on the left (if necessary) so that the number of digits in the subscript is exactly n, then we can obtain the corresponding minterm in the following manner. If in the ith location from the left there

Table 1-3.3

P	Q	R	A
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	F
F	F	F	T

appears 1, then the ith variable appears in the conjunction. If 0 appears in the ith location from the left, then the negation of the ith variable appears in the conjunction forming the minterm. Thus each of $m_0, m_1, \ldots, m_{2^n-1}$ corresponds to a unique minterm, which can be determined from the binary representation of the subscript. Conversely, given any minterm, one can find which of $m_0, m_1, \ldots, m_{2^n-1}$ designates it.

Let P, Q, and R be three variables arranged in that order. The corresponding minterms are denoted by $m_0, m_1, \ldots, m_7$. We can write the subscript 5 in binary as 101, and the minterm m_5 is $P \wedge \neg Q \wedge R$. Similarly m_0 corresponds to $\neg P \wedge \neg Q \wedge \neg R$. To obtain the minterm m_3, we write 3 as 11 and append a zero on the left to get 011, and m_3 is $\neg P \wedge Q \wedge R$.

If we have six variables $P_1, P_2, \ldots, P_6$, then there are $2^6 = 64$ minterms denoted by $m_0, m_1, \ldots, m_{63}$. To get a minterm, m_{38} say, we write 38 in binary as 100110; then the minterm m_{38} is $P_1 \wedge \neg P_2 \wedge \neg P_3 \wedge P_4 \wedge P_5 \wedge \neg P_6$.

Having developed a notation for the representation of the minterms, which can be further simplified by writing only the subscripts of $m_0, m_1, \ldots, m_{2^n-1}$, we designate the disjunction (sum) of minterms by the compact notation $\sum$. Using such a notation, the sum-of-products canonical form representing the disjunction of m_i, m_j, and m_k can be written down as $\sum i,j,k$. As an example, it is known that

$$(P \wedge Q) \vee (\neg P \wedge R) \vee (Q \wedge R) \Leftrightarrow (\neg P \wedge \neg Q \wedge R) \vee (\neg P \wedge Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (P \wedge Q \wedge R)$$

Thus we denote the principal disjunctive normal form of

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

as $\sum 1,3,6,7$. With this type of representation and simplification of notation, it is easy to compare two principal disjunctive normal forms.

We now proceed to obtain a similar representation for the product-of-sums (principal conjunctive normal) forms. We denote the maxterms of n variables by $M_0, M_1, \ldots, M_{2^n-1}$. Again, the maxterm corresponding to M_j, say, is obtained by writing j in binary and appending the required number of zeros to the left in order to get n digits. If 0 appears in the ith location from the left of this binary number, then the ith variable appears in the disjunction, while if 1 appears in the ith location, then the negation of the ith variable appears. Thus the binary representation of the subscript uniquely determines the maxterm, and, conversely, every binary representation of numbers between 0 and $2^n - 1$ determines a maxterm. Note that the convention regarding 1 and 0 here is the opposite of what was used for minterms. This convention is adopted with a view to connect the two principal normal forms of any given formula.

The maxterms, $M_0, M_1, \ldots, M_7$, corresponding to three variables P, Q, and R, are

$$\begin{array}{cccc} P \vee Q \vee R & P \vee Q \vee \neg R & P \vee \neg Q \vee R & P \vee \neg Q \vee \neg R \\ \neg P \vee Q \vee R & \neg P \vee Q \vee \neg R & \neg P \vee \neg Q \vee R & \neg P \vee \neg Q \vee \neg R \end{array}$$

As before, further simplification is introduced by using $\prod$ to denote the conjunction (product) of maxterms. Thus $\prod i,j,k$ represents the conjunction of maxterms M_i, M_j, M_k.

To illustrate this discussion, we consider $(P \wedge Q) \vee (\neg P \wedge R)$. We obtain its principal conjunctive normal form as follows.

$$
\begin{aligned}
&(P \wedge Q) \vee (\neg P \wedge R) \\
&\quad \Leftrightarrow ((P \wedge Q) \vee \neg P) \wedge ((P \wedge Q) \vee R) \\
&\quad \Leftrightarrow (P \vee \neg P) \wedge (Q \vee \neg P) \wedge (P \vee R) \wedge (Q \vee R) \\
&\quad \Leftrightarrow (Q \vee \neg P \vee (R \wedge \neg R)) \wedge (P \vee R \vee (Q \wedge \neg Q)) \\
&\qquad \wedge (Q \vee R \vee (P \wedge \neg P)) \\
&\quad \Leftrightarrow (Q \vee \neg P \vee R) \wedge (Q \vee \neg P \vee \neg R) \wedge (P \vee R \vee Q) \\
&\qquad \wedge (P \vee R \vee \neg Q) \wedge (Q \vee R \vee P) \wedge (Q \vee R \vee \neg P) \\
&\quad \Leftrightarrow (\neg P \vee Q \vee R) \wedge (\neg P \vee Q \vee \neg R) \wedge (P \vee Q \vee R) \\
&\qquad \wedge (P \vee \neg Q \vee R)
\end{aligned}
$$

Thus the product-of-sums canonical form of $(P \wedge Q) \vee (\neg P \wedge R)$ can be represented as $\prod 0,2,4,5$. Note that its disjunctive normal form is

$$
\begin{aligned}
&(P \wedge Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (\neg P \wedge Q \wedge R) \\
&\qquad \vee (\neg P \wedge \neg Q \wedge R) \Leftrightarrow \sum 1,3,6,7
\end{aligned}
$$

More generally, given any formula containing n variables and using the above notations to represent the equivalent principal disjunctive and conjunctive normal forms, we see clearly that all numbers lying between 0 and $2^n - 1$ which do not appear in one normal form will appear in the other. This statement follows from the principle of duality and the discussion given earlier regarding the relation between these two principal normal forms.

EXERCISES 1-3.5

1 Write equivalent forms for the following formulas in which negations are applied to the variables only.

(a) $\neg(P \vee Q)$ (d) $\neg(P \rightleftarrows Q)$
(b) $\neg(P \wedge Q)$ (e) $\neg(P \uparrow Q)$
(c) $\neg(P \rightarrow Q)$ (f) $\neg(P \downarrow Q)$

Obtain the principal conjunctive normal forms of (a), (c), and (d).

2 Obtain the product-of-sums canonical forms of the following formulas.

(a) $(P \wedge Q \wedge R) \vee (\neg P \wedge R \wedge Q) \vee (\neg P \wedge \neg Q \wedge \neg R)$
(b) $(\neg S \wedge \neg P \wedge R \wedge Q) \vee (S \wedge P \wedge \neg R \wedge \neg Q) \vee (\neg S \wedge P \wedge R \wedge \neg Q) \vee (Q \wedge \neg P \wedge \neg R \wedge S) \vee (P \wedge \neg S \wedge \neg R \wedge Q)$
(c) $(P \wedge Q) \vee (\neg P \wedge Q) \vee (P \wedge \neg Q)$
(d) $(P \wedge Q) \vee (\neg P \wedge Q \wedge R)$

3 Obtain the principal disjunctive and conjunctive normal forms of the following formulas.

(a) $(\neg P \vee \neg Q) \rightarrow (P \rightleftarrows \neg Q)$ (d) $(P \rightarrow (Q \wedge R)) \wedge (\neg P \rightarrow (\neg Q \wedge \neg R))$
(b) $Q \wedge (P \vee \neg Q)$ (e) $P \rightarrow (P \wedge (Q \rightarrow P))$
(c) $P \vee (\neg P \rightarrow (Q \vee (\neg Q \rightarrow R)))$ (f) $(Q \rightarrow P) \wedge (\neg P \wedge Q)$

Which of the above formulas are tautologies?