INTERMEDIATE LOGIC — Glossary of key terms

This glossary includes terms that are defined in the text in the lesson and on the page noted. It does not include the rules that are given in the appendices, but does include some key terms carried over from *Introductory Logic*.

Algebraic identity Lesson 36, page 291

A rule of digital logic that can be used to simplify a proposition (see Appendix D).

AND gate Lesson 32, page 266

A logic gate that performs the logical operation *conjunction*.

Antecedent Lesson 4, page 27

In a conditional *if p then q*, the antecedent is the proposition represented by the *p*.

Argument *Introductory Logic*, lesson 19, page 141

A set of statements, one of which appears to be implied or supported by the others.

Biconditional Lesson 5, page 35

The logical operator "if and only if" symbolized by ≡, that joins two propositions and is true when both propositions have the same truth value, and false when their truth values differ.

Binary number system Lesson 30, page 254

A base-two number system, using only the numerals 0 and 1 to represent any number (cf. the standard decimal number system, which is base ten, using the numerals 0 through 9).

Bit Lesson 29, page 249

The smallest amount of information that a computer stores, having a binary value of 1 or 0.

Bubble pushing Lesson 38, page 303

A technique used to simplify logic circuits by visually applying De Morgan's theorem.

Byte Lesson 29, page 249

A set of bits (usually eight) required to encode one character.

Circuit (see **logic circuit**)

Closed branch Lesson 22, page 179

A path of decomposed propositions on a truth tree which includes a contradiction.

Compound proposition Lesson 1, page 9

A proposition that has more than one component part, or is modified in some other way.

Conditional Lesson 4, page 27

The logical operator "if…then…" symbolize by the horseshoe ⊃, that (in standard form) is false if and only if the proposition following the "if" (the antecedent) is true and the proposition following the "then" (the consequent) is false.

Conditional Proof Lesson 18, page 137

A rule used in formal proofs which allows one to assume the antecedent of a conditional and, once the consequent is deduced, to conclude the entire conditional.

Conjunction Lesson 2, page 16

The logical operator "and" symbolized by the dot •, that joins two propositions and is true if and only if both propositions (conjuncts) are true (cf. AND gate).

Consequent Lesson 4, page 27

In a conditional *if p then q*, the consequent is the proposition represented by the *q*.

Consistent Lesson 10, page 65

A set of propositions is consistent if the propositions can be true at the same time.

Contradictory Lesson 6, page 40

Two propositions are contradictory if and only if they have opposite truth values in a truth table (i.e. the biconditional of contradictory propositions is a self-contradiction).

Decompose a proposition Lesson 22, page 177

To break down a compound proposition into its literals (see Appendix C).

Defining truth table Lesson 2, page 15

A listing of the truth values produced by a logical operator modifying a minimum number of variables (see Appendix A).

Digital display Lesson 29, page 249

A seven-segment display used to represent the numerals 0 through 9.

Digital logic Lesson 29, page 249.

A branch of formal logic that is applied to electronics.

Dilemma Lesson 12, page 73

A valid argument that presents a choice between two conditionals. A *constructive* dilemma follows the form $(p \supset q) \bullet (r \supset s)$, $p \vee q$, $\therefore r \vee s$, while a *destructive* dilemma follows the form $(p \supset q) \bullet (r \supset s)$, $\sim q \vee \sim s$, $\therefore \sim p \vee \sim r$.

Disjunction Lesson 2, page 16

The logical operator "or" symbolized by the $\vee$, that joins two propositions and is true if and only if one or both of the propositions (disjuncts) is true (cf. OR gate).

Equivalent (see **logically equivalent**)

Exclusive or Lesson 2, page 16

A disjunction that is true when either one or the other disjunct, but not both, is true (cf. XOR gate).

Formal proof of validity Lesson 13, page 101

A step-by-step deduction of a conclusion from a set of premises, each step being justified by an appropriate basic rule.

Gate (see **logic gate**)

Going between the horns Lesson 12, page 74

A method of refuting a dilemma that denies the disjunctive premise and provides another alternative (cf. grasping the horns, rebutting the horns).

Grasping the horns Lesson 12, page 75

A method of refuting a dilemma that rejects one of the conditionals in the conjunctive premise (cf. going between the horns, rebutting the horns).

Inclusive or Lesson 2, page 16

A disjunction that is true when the one disjunct or the other is true, or both are true (cf. OR gate). The standard disjunction is the inclusive or.

Inconsistent Lesson 10, page 66

A set of propositions is inconsistent if the propositions cannot be true at the same time.

Invalid Lesson 7, page 43

In an invalid argument, it is possible for the premises to be true and the conclusion false.

Karnaugh map Lesson 40, page 313

A rectangular grid with cells that can contain the output of a given truth table of two, three, or four variables, used to quickly determine the proposition corresponding to that output (also called a K-map).

Literal Lesson 22, page 177

A simple proposition represented by a single letter, or the negation of the same.

Logic Introduction, page 5

The science and art of correct reasoning.

Logic circuit Lesson 32, page 267

A combination of logic gates, in which the outputs of some gates are joined to the input of other gates, used to perform complex operations.

Logic gate Lesson 32, page 265

A logical operator represented by a symbol in digital logic, with one or two inputs and a single output, which can be joined together in logic circuits.

Logical operator Lesson 1, page 10

Words (representable by symbols) that combine or modify simple propositions, making them compound (such as AND, OR, NOT) (see Appendix A).

Logically equivalent Lesson 6, page 39

Two propositions are logically equivalent if and only if they have identical truth values in a truth table (i.e. the biconditional of logically equivalent propositions is a tautology) (cf. XNOR gate).

Negation Lesson 2, page 15

The logical operator “not” symbolized by the tilde ~, that contradicts or denies a proposition. The negation of a proposition has the opposite truth value of that proposition (cf. NOT gate).

NAND gate Lesson 35, page 283

A logic gate that performs the logical operation *negated conjunction*.

NOR gate Lesson 35, page 283

A logic gate that performs the logical operation *negated disjunction.*

NOT gate Lesson 32, page 265

A logic gate that performs the logical operation *negation* (also called an inverter).

Open branch Lesson 22, page 178

A path of decomposed propositions on a truth tree which includes no contradictions.

OR gate Lesson 32, page 266

A logic gate that performs the logical operation *disjunction* (i.e. inclusive or)

Proof (see **formal proof of validity**)

Proposition Lesson 1, page 9

A statement, a sentence that is true or false.

Propositional constant Lesson 1, page 10

An uppercase letter that represents a single, given proposition, usually the first letter of a key word in the proposition.

Propositional logic Lesson 1, page 9

A branch of formal, deductive logic in which the basic unit of thought is the proposition (cf. *categorical* logic, in which the basic unit of thought is the category or term).

Propositional variable Lesson 1, page 10

A lowercase letter that represents any proposition, usually *p*, *q*, *r*, etc. usually used when the form of a compound proposition or argument is being emphasized.

Reason Introduction, page 5

To draw conclusions from premises.

Rebutting the horns Lesson 12, page 75

Refuting a dilemma by means of a counter-dilemma (cf. going between the horns, grasping the horns).

Recover truth values Lesson 22, page 178

We recover the truth values after creating a truth tree when we determine the truth values of the component simple propositions for which the propositions in the given set are consistent.

Reductio ad absurdum Lesson 19, page 143

A rule used in formal proofs which allows one to assume the negation of a proposition and, once a self-contradiction is deduced, to conclude the original proposition.

Rule of inference Lesson 13, page 103

A valid argument form which can be used to justify steps in a proof (see Appendix B).

Rule of replacement Lesson 16, page 123

Form of equivalent propositions which can be used to justify steps in a proof (see Appendix B). In digital logic, rules of replacement can be used to simplify logic circuits (see Appendix D).

Self-contradiction Lesson 6, page 39

A proposition that is false by logical structure (false for every row in a truth table).

Simple proposition Lesson 1, page 9

A proposition that has only one component part.

Tautology Lesson 6, page 39

A proposition that is true by logical structure (true for every row in a truth table).

Truth-functional Lesson 1, page 9

A proposition is truth-functional when the truth value of the proposition depends upon the truth value of its component parts (cf. self-reports, tautologies, self-contradictions, which are *not* truth functional).

Truth-functionally complete Lesson 21, page 154

A set of logical operators is truth-functionally complete if and only if all possible combinations of true and false are derivable using only those logical operators, i.e. any truth-functional proposition can be written using only those logical operators. In digital logic, a truth-functionally complete gate is a logic gate that by itself can be used to design any basic logic circuit.

Truth table Lesson 2, page 15

A listing of the possible truth values for a set of one or more propositions.

Truth tree Lesson 22, page 177

A diagram that shows a set of propositions being decomposed into their literals to determine their consistency.

Valid Lesson 7, page 43

In a valid argument, the premises imply the conclusion (if the premises are true, the conclusion must be true). Validity is the key concept in formal logic.

XNOR gate Lesson 39, page 308

A logic gate that performs the logical operation *negated exclusive or* (i.e. biconditional).

XOR gate Lesson 39, page 307

A logic gate that performs the logical operation *exclusive or* (symbolized by $\oplus$).