# CPE111
# Discrete Mathematics for Computer Engineers

Discrete Mathematics /= Calculus (Continuous Mathematics)

We consider sets, logic, Boolean algebra, counting number, Relations, graphs, trees, etc.

## Course Description

- Provide a foundation of discrete mathematics for computer engineers. See more detail in the course syllabus.

## Expected learning outcome

- Understand the fundamental of discrete mathematics.
- Able to apply the knowledge from this course to solve related problems
- Able to work in group and present in both oral and written forms.

# Tentative Class Schedule

| Weeks | Topics | Text References |
|---|---|---|
| 1, 2 | Basic of Logic, Sets and Functions | Ch. 1-2 |
| | Numbering System | Ch. 4 |
| 3 | Boolean Algebra | - |
| 4 | Introduction to Complexity of Algorithms | Ch. 3 |
| 5-6 | Mathematical Reasoning | Ch. 5 |
| | - Mathematical Induction | |
| | - Recursive Definition & Algorithms | |
| 6-7 | Counting | Ch. 6 |
| | - Basics of Counting | |
| | - Pigeonhole Principle | |
| | - Permutations & Combinations | |
| 8-9 | **Midterm Examination** | |

| Weeks | Topics | Text References |
|---|---|---|
| 10 | Advanced Counting | Ch. 8 |
| | - Recurrence Relations | |
| | - Divide & Conquer | |
| | - Inclusion & Exclusion | |
| 11 | Discrete Probability | Ch. 7 |
| 12 | Relations | Ch. 9 |
| | - Relations and Their Properties | |
| | - Closures of Relations | |
| | - Equivalence Relations | |
| 13-14 | Graphs and Trees | Ch. 10-11 |
| | - Graph Terminology & Connectivity | |
| | - Introduction to Trees | |
| 15 | Finite State Machine, context-free grammar, and Turing machine | |
| 16 | **Final Examination** | |

**Textbook:**

Kenneth H. Rosen, Discrete Mathematic and Its Applications, 2019, 8th Edition, McGraw-Hill.

**Course Grade:**

| | |
|---|---|
| Midterm Exam | 30% |
| Final Exam | 30% |
| Quizzes | 15% |
| Project | 10% |
| Homework and Assignment | 10% |
| Class Participation | 5% |

# Goals of this Course

1. Study and apply mathematical reasoning and logic
2. Understand basic principles of Boolean algebra, mathematical reasoning, set, counting, discrete probability, relations, graphs and trees.
3. Create foundation for other related topics:

   - Data structure
   - Database theory
   - Digital system design
   - Computer Network
   - Computer security

# Chapter 1: Logic and Proofs

**Topics:**

1.1 Propositional Logic
1.2 Applications of Propositional Logic
1.3 Propositional Equivalences
1.4 Predicates and Quantifiers
1.6 Rules of Interference
1.8 Introduction to Proofs

# Chapter 1: Logic and Proofs

- Logic is the foundation of all mathematical reasoning.
- Logic can be applied to
  - The design of computer systems
  - System specification
  - Programming languages
- Proofs are methods to verify
  - Mathematical statement/argument
  - Computer program if it produces correct outputs for all possible input cases

# Chapter 1: Logic and Proofs

| | | | |
|---|---|---|---|
| **1.1 Propositional Logic** | | | |
| **Proposition**: A proposition is a statement that is either true or false, but not both. | | | |
| **Ex :** | 1 | Bangkok is the capital of Thailand | |
| | 2 | $1+2 = 4$ | |
| | 3 | What time is it ? | |
| | 4 | $a + b = z$ | |
| | 5 | Wait a minute | |

# Chapter 1: Logic and Proofs

| | | | |
|---|---|---|---|
| **1.1 Propositional Logic** | | | |
| **Proposition**: A proposition is a statement that is either true or false, but not both. | | | |
| **Ex :** | 1 | Bangkok is the capital of Thailand | True |
| | 2 | $1+2 = 4$ | False |
| | 3 | What time is it ? | Not Proposition |
| | 4 | $a + b = z$ | Not Proposition |
| | 5 | Wait a minute | Not Proposition |

## DEFINITIONS 1-6:

1. Let $P$ be a proposition then $\sim P$ = ***negation of p***

Let $P$ and $Q$ be propositions:

2. "*P and Q*" $= P \wedge Q =$ *Conjunction* of P and Q
3. "*P or Q*" = P v Q = *Disjunction* of P and Q
4. "$P \oplus Q$" = *Exclusive Or of P and Q*
5. "$P \rightarrow Q$" = *Implication* of P and Q, where $P$ = hypothesis,

   $Q$ = conclusion
6. "$P \leftrightarrow Q$" = *Biconditional* P and Q

# The truth table for many propositions

| p | q | $p \wedge q$ | $p \vee q$ | $p \oplus q$ | $p \rightarrow q$ | $p \leftrightarrow q$ |
|---|---|---|---|---|---|---|
| T | T | | | | | |
| T | F | | | | | |
| F | T | | | | | |
| F | F | | | | | |

# Logic and Bit Operations

A ***bit*** has two possible values : 0 (zero) and 1(one)

A Boolean Variable : False and True

**<u>DEFINITION 7:</u>** A ***bit string*** is a sequence of zero or more bits.

The length of the string is the number of bits in the string.

**<u>Ex</u>:** 10101001101 is a bit string of length __________

**<u>Ex</u>:** What are the values of the corresponding propositions

when A = 1, B = 0? A and B, A or B, A xor B, $A \longrightarrow B$, $A \longleftrightarrow B$

**Ex:** What are the values of the corresponding propositions when A = 10110, B = 01110?

1) A and B

2) A or B

3) A xor B

# Precedence of Logical Operators

| Operator | Precedence |
|---|---|
| NOT ~ | 1 |
| AND ^<br>OR $\vee$ | 2<br>3 |
| IF…THEN $\rightarrow$<br>IF…AND ONLY IF $\leftrightarrow$ | 4<br>5 |

Express the following operations using parenthesis.

Ex1. $A \wedge B \vee C \rightarrow D \wedge E$

Ex2. $A \vee \sim B \wedge C \rightarrow D \vee E$

Ex3. $C \wedge \sim A \vee B \leftrightarrow \sim D$

$A \vee B \wedge C = A \vee (B \wedge C)$

# Precedence of Logical Operators

| Operator | Precedence |
|---|---|
| NOT $\sim$ | 1 |
| AND $\wedge$<br>OR $\vee$ | 2<br>3 |
| IF…THEN $\rightarrow$<br>IF…AND ONLY IF $\leftrightarrow$ | 4<br>5 |

Ex1. $A \wedge B \vee C \rightarrow D \wedge E = ((A \wedge B) \vee C) \rightarrow (D \wedge E)$

Ex2. $A \vee \sim B \wedge C \rightarrow D \vee E = (A \vee ((\sim B) \wedge C) \rightarrow (D \vee E)$

Ex3. $C \wedge \sim A \vee B \leftrightarrow \sim D = ((C \wedge (\sim A)) \vee B) \leftrightarrow (\sim D)$

# 1.2 Applications of Propositional Logic

# Logic Circuits

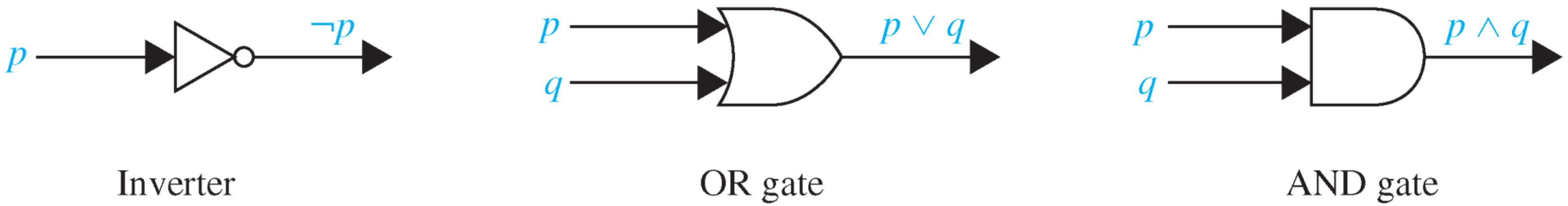


FIGURE 1 **Basic logic gates.**

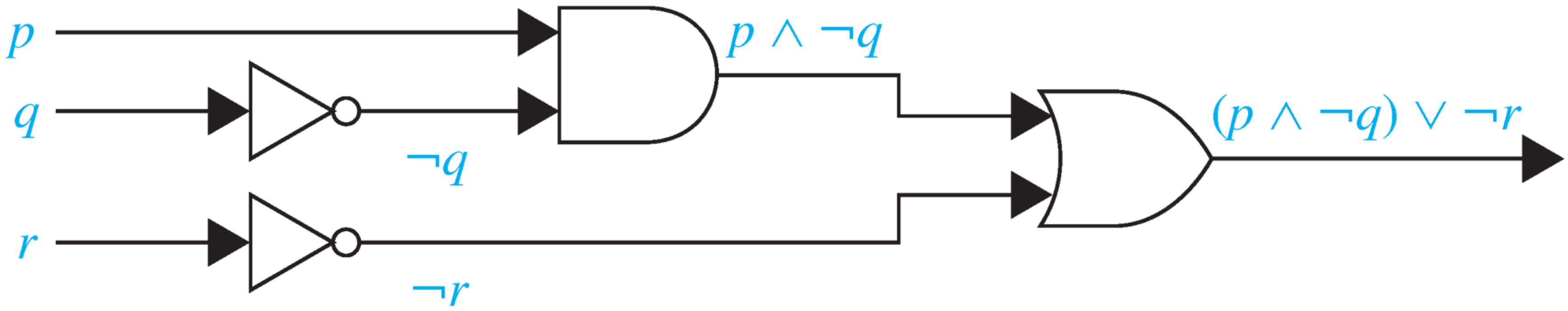


FIGURE 2 **A combinatorial circuit.**

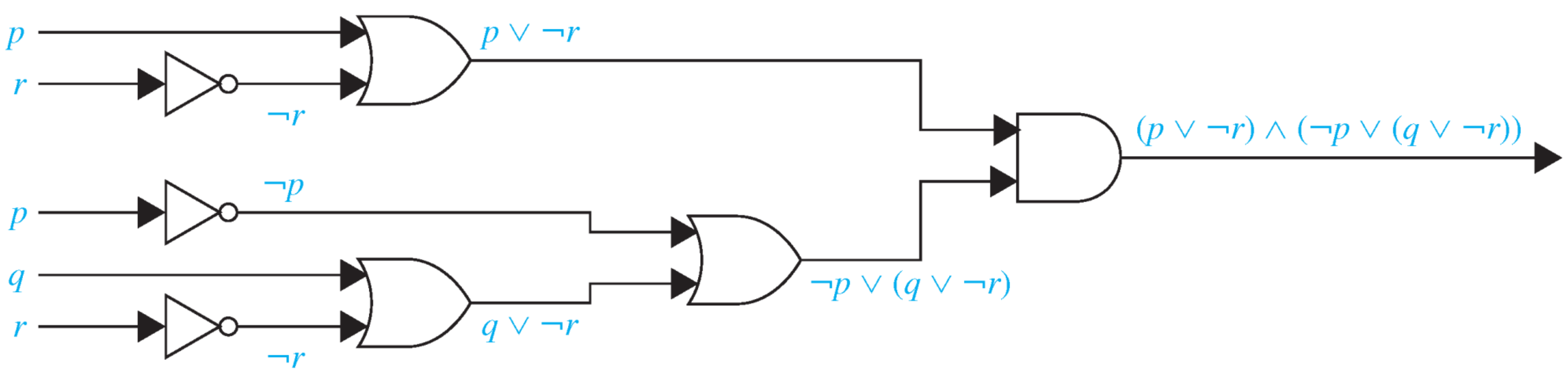


FIGURE 3 **The circuit for $(p \vee \neg r) \wedge (\neg p \vee (q \vee \neg r))$.**

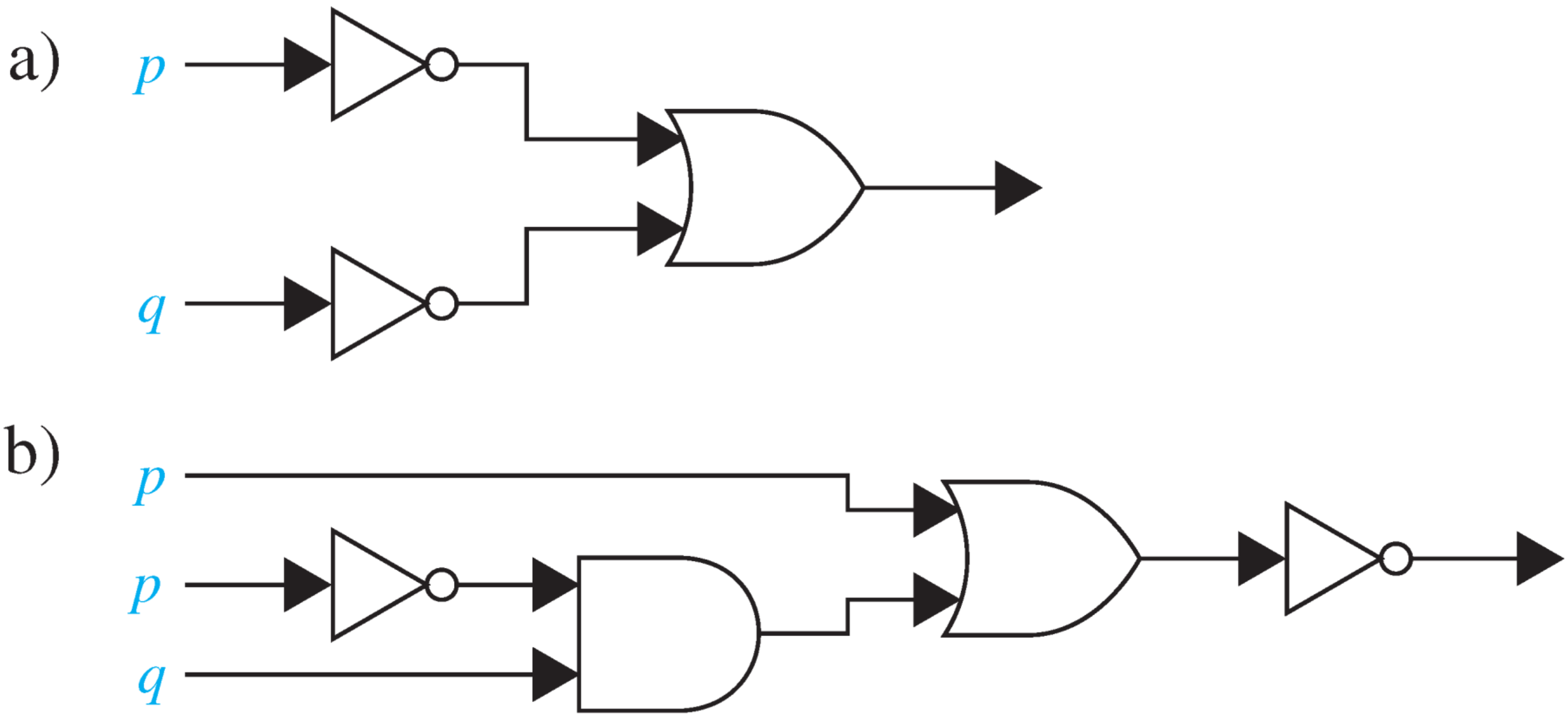


**Figure for Exercise 40.**

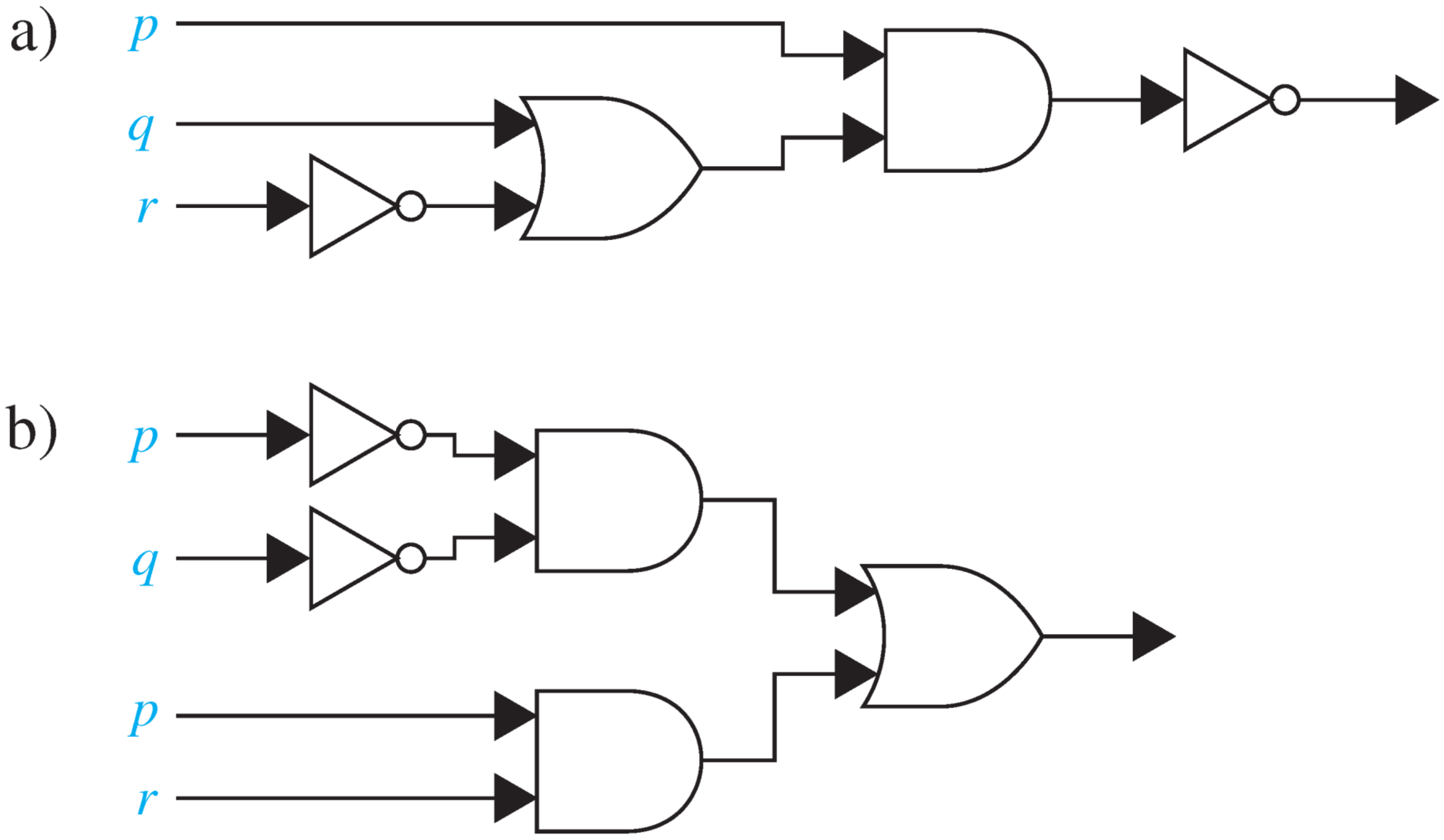


**Figure for Exercise 41.**

# 1.3 Propositional Equivalences

## Tautology vs Contradiction vs Contingency

**DEFINITION 1:** A compound proposition that is always true is called a ***Tautology***.

A compound proposition that is always false is called a ***Contradiction***.

A proposition that is neither a ***Tautology*** nor a ***Contradiction*** is called a ***Contingency***.

**Ex1:**

| P | ~P | P v ~ P | P ^ ~ P |
|---|---|---|---|
| T | F | T | F |
| F | T | T | F |

| | | (Tautology) | (Contradiction) |
|---|---|---|---|

# Logical Equivalences

**DEFINITION 2:** The propositions $P$ and $Q$ are called **logically equivalent**, **if $P \leftrightarrow Q$ is a tautology**. The notation $P \equiv Q$ denotes that $P$ and $Q$ are logically equivalent.

**<u>Ex2</u> : Show that** $\sim (P \vee Q) \equiv (\sim P \wedge \sim Q)$

| $P$ | $Q$ | $P \vee Q$ | $\sim(P \vee Q)$ | $\sim P$ | $\sim Q$ | $\sim P \wedge \sim Q$ |
|---|---|---|---|---|---|---|
| T | T | T | F | F | F | F |
| T | F | T | F | F | T | F |
| F | T | T | F | T | F | F |
| F | F | F | T | T | T | T |

**Ex 3: Proof that ( $\sim$ P v Q ) and ( P $\rightarrow$ Q ) are logically equivalent**

**Using a truth table**

**<u>Ex 4</u>: Proof that (P v ( Q ^ R )) and (( P v Q ) ^ ( P v R )) are logically equivalent**

| P | Q | R | P v Q | P v R | Q v R | P v ( Q ^ R ) | **( P v Q ) ^ ( P v R )** |
|---|---|---|---|---|---|---|---|
| T | T | T | T | T | T | T | T |
| T | T | F | T | T | | | |
| T | F | T | T | T | | | |
| T | F | F | T | T | | | |
| F | T | T | T | T | | | |
| F | T | F | T | | | | |
| F | F | T | F | | | | |
| F | F | F | F | | | | |

# Table of Logical Equivalences

| Equivalence | Name |
|---|---|
| $P \wedge T \equiv P$ | Identity laws |
| $P \vee F \equiv P$ | |
| $P \vee T \equiv T$ | Domination laws |
| $P \wedge F \equiv F$ | |
| $P \vee P \equiv P$ | Idempotent laws |
| $P \wedge P \equiv P$ | |
| $\sim(\sim)P \equiv P$ | Double Negative laws |
| $P \vee Q \equiv Q \vee P$ | Commutative laws |
| $P \wedge Q \equiv Q \wedge P$ | |

| Equivalence | Name |
|---|---|
| $(P \vee Q) \vee R \equiv P \vee (Q \vee R)$ | Associative laws |
| $(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$ | |
| $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$ | Distributive laws |
| $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$ | |
| $\sim(P \wedge Q) \equiv \sim P \vee \sim Q$ | De Morgan's laws |
| $\sim(P \vee Q) \equiv \sim P \wedge \sim Q$ | |

# Other Equivalence Forms

$$P \vee \sim P \equiv T$$

$$P \wedge \sim P \equiv F$$

$$(P \rightarrow Q) \equiv (\sim P \vee Q)$$

$$\sim (P_1 \wedge P_2 \wedge P_3 \wedge \ldots \wedge P_n) \equiv (\sim P_1 \vee \sim P_2 \vee \ldots \vee \sim P_n)$$

$$\sim (P_1 \vee P_2 \vee P_3 \vee \ldots \vee P_n) \equiv (\sim P_1 \wedge \sim P_2 \wedge \ldots \wedge \sim P_n)$$

Examples: $\sim(A \wedge B \wedge C) = (\sim A) \vee (\sim B) \vee (\sim C)$

$$\sim(A \vee B \vee C) = \sim A \vee \sim B \vee \sim C$$

$$A \rightarrow (\sim B) = ((\sim A) \vee (\sim B))$$

# 1.4 Predicates and Quantifiers

**Predicate: A property that the subject of the statement can have**

**<u>EX</u>** $X > 3$

$X$ = **Subject** " > 3 " = **Predicate**

**IF** $P(X) = X > 3$

$P$ = **Predicate** " >3 ", $x$ = **Variable**

$P(x)$ = value of the propositional function $P$ at $x$

**Ex 1** Let P(x) denote the statement "$x > 3$". What are the truth values of P(2) and P(3)?

Both are FALSE

**Ex 2** Let Q(x,y) denote the statement "$x = y + 3$"

What are the truth values of Q(1, 2) and Q(3, 0)?

Q(1, 2) is FALSE and Q(3, 0) is TRUE

$P(x_1,x_2,x_3,\ldots,x_n)$ is the value of the propositional functional *P* on the n-tuple $(x_1,x_2,x_3,\ldots,x_n)$ and *P* is also called a ***predicate***.

# QUANTIFIERS

**DEFINITION 1:** The *universal quantification of P(x)* is the proposition.

"P(x) is true for all values of x in the universe of discourse"

$\forall x\, P(x)$ = " for all x P(x)"

**Ex**: Let P(x) : " $x < 2$"

What is the truth value of the quantification $\forall x P(x)$, when the universe of discourse is the set of real numbers?

False

**DEFINITION 2:** The *existential quantification* of P(x) is the proposition.

"There exists an element x in the universe of discourse such that P(x) is true"

$\exists$ x = there is at least one x, such that P(x) is true

**Ex :** Let P(x) : x > 3

$\exists x P(x)$ = ? : when x is a set of real numbers.

**Ex :** Let Q(x) : x = x+1

$\exists x Q(x)$ = ? : when x is a set of real numbers.

# Precedence of Quantifiers

- The quantifiers $\forall$ and $\exists$ have higher precedence than all the logical operators.
- For example, $\forall x\, P(x) \vee Q(x)$ means $(\forall x\, P(x)) \vee Q(x)$
- $\forall x\, (P(x) \vee Q(x))$ means something different.
- Unfortunately, often people write $\forall x\, P(x) \vee Q(x)$ when they mean $\forall x\, (P(x) \vee Q(x))$.

# Negating Quantified Expressions

- Consider $\forall x\, J(x)$

  "Every student in your class has taken a course in Java."

  Here $J(x)$ is "x has taken a course in Java" and

  the domain is students in your class.

- Negating the original statement gives "It is not the case that every student in your class has taken Java." This implies that "There is a student in your class who has not taken Java."

  Symbolically $\neg \forall x\, J(x)$ and $\exists x\, \neg J(x)$ are equivalent.

# Negating Quantified Expressions (*continued*)

- Now Consider $\exists x\, J(x)$

  "There is a student in this class who has taken a course in Java."

  Where $J(x)$ is "x has taken a course in Java."

- Negating the original statement gives "It is not the case that there is a student in this class who has taken Java." This implies that "Every student in this class has not taken Java"

  Symbolically $\neg\exists x\, J(x)$ and $\forall x\, \neg J(x)$ are equivalent.

# De Morgan's Laws for Quantifiers

- The rules for negating quantifiers are:

**TABLE 2** De Morgan's Laws for Quantifiers.

| *Negation* | *Equivalent Statement* | *When Is Negation True?* | *When False?* |
|---|---|---|---|
| $\neg\exists x P(x)$ | $\forall x \neg P(x)$ | For every $x$, $P(x)$ is false. | There is an $x$ for which $P(x)$ is true. |
| $\neg\forall x P(x)$ | $\exists x \neg P(x)$ | There is an $x$ for which $P(x)$ is false. | $P(x)$ is true for every $x$. |

- The reasoning in the table shows that:

$$\neg\forall x P(x) \equiv \exists x \neg P(x)$$
$$\neg\exists x P(x) \equiv \forall x \neg P(x)$$

- These are important. You will use these.

# Translation from English to Logic

**Examples**:

1. "Some student in this class has visited Mexico."

   **Solution**: Let $M(x)$ denote "$x$ has visited Mexico" and $S(x)$ denote "$x$ is a student in this class," and $U$ be all people.

   $$\exists x \ (S(x) \wedge M(x))$$

2. "Every student in this class has visited Canada or Mexico."

   **Solution**: Add $C(x)$ denoting "$x$ has visited Canada."

   $$\forall x \ (S(x) \rightarrow (M(x) \vee C(x)))$$

# System Specification Example

- Predicate logic is used for specifying properties that systems must satisfy.
- For example, translate into predicate logic:
  - "Every mail message larger than one megabyte will be compressed."
  - "If a user is active, at least one network link will be available."
- Decide on predicates and domains (left implicit here) for the variables:
  - Let $L(m, y)$ be "Mail message $m$ is larger than $y$ megabytes."
  - Let $C(m)$ denote "Mail message $m$ will be compressed."
  - Let $A(u)$ represent "User $u$ is active."
  - Let $S(n, x)$ represent "Network link $n$ is in state $x$.
- Now we have:

$$\forall m(L(m,1) \to C(m))$$

$$\exists u\, A(u) \to \exists n\, S(n, available)$$

# Lewis Carroll Example

Charles Lutwidge Dodgson
(AKA Lewis Caroll) (1832-1898)

- The first two are called *premises* and the third is called the *conclusion*.
  1. "All lions are fierce."
  2. "Some lions do not drink coffee."
  3. "Some fierce creatures do not drink coffee."

- Here is one way to translate these statements to predicate logic. Let P(x), Q(x), and R(x) be the propositional functions "x is a lion," "x is fierce," and "x drinks coffee," respectively.
  1. $\forall x\, (P(x) \rightarrow Q(x))$
  2. $\exists x\, (P(x) \land \neg R(x))$
  3. $\exists x\, (Q(x) \land \neg R(x))$
- Later we will see how to prove that the conclusion follows from the premises.

# 1.5 Nested Quantifiers

- Nested quantifiers are often necessary to express the meaning of sentences in English as well as important concepts in computer science and mathematics.

  **Example**: "Every real number has an inverse" is

  $$\forall x\, \exists y(x + y = 0)$$

  where the domains of x and y are the real numbers.

- We can also think of nested propositional functions:

  $\forall x\, \exists y(x + y = 0)$ can be viewed as $\forall x\, Q(x)$ where $Q(x)$ is $\exists y\, P(x, y)$ where $P(x, y)$ is $(x + y = 0)$

# Order of Quantifiers

**Examples**:

1. Let $P(x, y)$ be the statement "$x + y = y + x$." Assume that $U$ is the real numbers. Then $\forall x\ \forall y\ P(x, y)$ and $\forall y\ \forall x\ P(x, y)$ have the same truth value.

2. Let $Q(x, y)$ be the statement "$x + y = 0$." Assume that $U$ is the real numbers. Then $\forall x\ \exists y\ P(x, y)$ is true, but $\exists y\ \forall x\ P(x, y)$ is false.

# Questions on Order of Quantifiers

**Example 2**: Let $U$ be the real numbers,
Define $P(x, y) : x / y = 1$

What is the truth value of the following:

1. $\forall x \forall y P(x,y)$
   **Answer:** False
2. $\forall x \exists y P(x,y)$
   **Answer:** True
3. $\exists x \forall y\ P(x,y)$
   **Answer:** False
4. $\exists x\ \exists\ y\ P(x,y)$
   **Answer:** True

# Quantifications of Two Variables

| Statement | When True? | When False |
|---|---|---|
| $\forall x \forall y P(x, y)$<br>$\forall y \forall x P(x, y)$ | *P*(*x*, *y*) is true for every pair *x*, *y*. | There is a pair *x*, *y* for which *P*(*x*, *y*) is false. |
| $\forall x \exists y P(x, y)$ | For every *x* there is a *y* for which *P*(*x*, *y*) is true. | There is an x such that *P*(*x*, *y*) is false for every *y*. |
| $\exists x \forall y P(x, y)$ | There is an *x* for which *P*(*x*, *y*) is true for every *y*. | For every *x* there is a y for which *P*(x, y) is false. |
| $\exists x \exists y P(x, y)$<br>$\exists y \exists x P(x, y)$ | There is a pair *x*, *y* for which *P*(*x*, *y*) is true. | *P*(x, y) is false for every pair *x*, *y* |

# Translating Nested Quantifiers into English

**Example 1**: Translate the statement

$$\forall x \ (C(x) \vee \exists y \, (C(y) \wedge F(x, y)))$$

where C(x) is "*x* has a computer," and *F*(*x*, *y*) is "*x* and *y* are friends," and the domain for both *x* and *y* consists of all students in your school.

**Solution**: Every student in your school has a computer or has a friend who has a computer.

**Example 2**: Translate the statement

$$\exists x \, \forall y \ \forall z \, ((F(x, y) \wedge F(x, z) \wedge (y \neq z)) \rightarrow \neg F(y, z))$$

**Solution**: Some students have friends, and none of those friends are also friends with each other.

# Translating Mathematical Statements into Predicate Logic

**Example** : Translate "The sum of two positive integers is always positive" into a logical expression.

**Solution**:

1. Rewrite the statement to make the implied quantifiers and domains explicit:

   "For every two integers, if these integers are both positive, then the sum of these integers is positive."

2. Introduce the variables *x* and *y*, and specify the domain, to obtain:

   "For all positive integers *x* and *y*, *x* + *y* is positive."

3. The result is:

$$\forall x \, \forall y \, ((x > 0) \land (y > 0) \rightarrow (x + y > 0))$$

where the domain of both variables consists of all integers

# Translating English into Logical Expressions Example

**Example**: Use quantifiers to express the statement "There is a woman who has taken a flight on every airline in the world."

**Solution**:

1. Let $P(w, f)$ be "$w$ has taken $f$" and $Q(f, a)$ be "$f$ is a flight on $a$".
2. The domain of $w$ is all women, the domain of $f$ is all flights, and the domain of $a$ is all airlines.
3. Then the statement can be expressed as:

$$\exists w \, \forall a \, \exists f \; (P(w, f) \land Q(f, a))$$

# PROPERTY OF QUANTIFIERS

$\forall x \forall y ( x + y = y + x )$ : when x is a set of real numbers.

$\forall x \exists y ( x + y = 0 )$ : when x is a set of real numbers.

$\forall x \forall y \forall z [ x + (y + z) ] = [ (x + y) + z ]$ : Associative law, when x,y,z are sets of real numbers.