- A ***first-order language*** is a symbolic language used to express statements in first-order logic. ***Well-formed formulas*** are its constructed expressions, used to precisely express complex statements about objects and their relationships.
- **Definition 3.1:** The alphabet of a first-order language is defined as follows:

(1) Individual constants: $a,b,c,\ldots,a_i,b_i,c_i,\ldots$, where $i \geq 1$

(2) Individual variables: $x,y,z,\ldots,x_i,y_i,z_i,\ldots$, where $i \geq 1$

(3) Function symbols: $f,g,h,\ldots,f_i,g_i,h_i,\ldots$, where $i \geq 1$

(4) Predicate symbols: $F,G,H,\ldots,F_i,G_i,H_i,\ldots$, where $i \geq 1$

(5) Quantifier symbols: ∀,∃

(6) Connective symbols: ¬,∧,∨,→,↔

(7) Parentheses and commas: ( ) , ,

- **Definition 3.2：** The definition of ***terms*** in a first-order language is as follows:

  (1) ***Individual constants*** and ***individual variables*** are terms.

  (2) If $\boldsymbol{\varphi}(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n)$ is an arbitrary n-ary function symbol, and $\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_n$ are arbitrary terms, then $\boldsymbol{\varphi}(\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_n)$ is also a term.

  (3) All terms are obtained through a finite number of applications of (1) and (2).

- **Definition 3.3:** Let $\boldsymbol{R}(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n)$ be an arbitrary n-ary predicate symbol in a first-order language, and let $\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_n$ be arbitrary terms. Then, $\boldsymbol{R}(\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_n)$ is called an ***atomic formula***.

- **Definition 3.4**: ***Well-formed formulas*** in a first-order language are defined as follows:

(1) **Atomic formulas** are well-formed formulas.

(2) If *A* is a well-formed formula, then ¬*A* is also a well-formed formula.

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

(4) If *A* is a well-formed formula, then $\forall x A$ and $\exists x A$ are also well-formed formulas.

(5) Only those expressions formed by a finite application of rules (1) through (4) are considered well-formed formulas.

- Well-formed formulas are also referred to as ***predicate formulas*** or simply ***formulas***.

- **Definition 3.5:** In the formulas $\forall x A$ and $\exists A$, $\boldsymbol{x}$ is called the ***bound variable*** (or binder), and A is the scope of the respective quantifier. In the scope of $\forall \boldsymbol{x}$ and $\exists \boldsymbol{x}$, all occurrences of $\boldsymbol{x}$ are called ***bound occurrences***. Variables in A that are not bound occurrences are called ***free occurrences***.

e.g. »»**Example**: The formula $\forall x(F(x,y) \to \exists y G(x,y,z))$

- The scope of $\forall \boldsymbol{x}$ is $(F(x,y) \to \exists y G(x,y,z))$, with $\boldsymbol{x}$ as the **bound variable**. Both occurrences of $\boldsymbol{x}$ are **bound occurrences**.
- The scope of $\exists \boldsymbol{y}$ is $G(x,y,z)$, with $\boldsymbol{y}$ as the **bound variable**.
- The first occurrence of $\boldsymbol{y}$ is a **free occurrence**, and the second occurrence is **a bound occurrence**.
- $\boldsymbol{z}$ is a **free occurrence**.

↳Closed well-formed formula(closed formula)

e.g. **Example:** The formula $\forall x(F(x)\to\exists xG(x))$

- The scope of $\forall x$ is $(F(x)\to\exists xG(x))$, with x as the bound variable.
- The scope of $\exists x$ is $G(x)$, with x as the bound variable.
- Both occurrences of x are bound occurrences: the first in $\forall x$, and the second in $\exists x$.

- **Closed formula:** A formula that contains **no free occurrences** of individual variables is called a ***closed well-formed formula***, abbreviated as ***closed formula***.

↳Interpretation, assignment, and quantification of formulas

- A formula is merely a framework of logical expressions. **To evaluate its truth value**, the following tasks must be completed:

(1) **Interpretation**: Assign specific (semantic) meaning to this framework.

(2) **Assignment**: Under a given interpretation, assign values to the free variables in the formula.

(3) **Quantification**: Eliminate the free variables so that the truth value of the formula no longer depends on specific assignments but is determined by the overall situation of all possible assignments.

e.g. Example: Formula $\forall x(F(x) \rightarrow G(x))$

- **Specification1: Domain:** All individuals. ***F*(*x*): *x*** is a person. ***G*(*x*): *x*** is Asian. This formula translates to "For all x, if x is a person, then x is Asian." This is a **false statement** because not all people are Asian.
- **Specification 2: Domain:** The set of real numbers. ***F*(*x*): *x* > 10**. ***G*(*x*): *x*** > 0. This formula translates to "For all x, if x > 10, then x > 0." This is a **true statement** because any number greater than 10 is also greater than 0.

e.g. Example: Formula $\exists x F(x,y)$

- **Specification:Domain:** The set of natural numbers. ***F*(*x*,*y*)**: *x* = ***y***. ***y* = 0.** Since ***x***,***y*** both belong to the set of natural numbers N, for any ***y*∈*N***, there exists an ***x***=**y** such that the equation holds. Therefore, the statement is **always true.**

e.g. **Example:** Given the interpretation *I* and assignment *σ* as follows :

① **Definition** *D*=N ; ② $\bar{a} = 0$; ③ $\bar{f}(x, y) = x + y, \bar{g}(x, y) = xy$) ;

④ $\bar{F}(x, y): x = y$. **Value σ:** σ(x)=0, σ(y)=1, σ(z)=2.

Explain the meaning of the following formula under interpretation *I* and assignment *σ*, and discuss its truth value.

(1) ∀xF(g(x,a),y): ∀x(0x=1) False

(2) ∀x∀y(F(f(x,a),y)→F(f(y,a),x)) Truth

f(x,a)=x,f(y,a)=y, ∀x∀y(F(x,y)→F(y,x)), (x=y)→(y=x)

(3) ∀x∀y∃zF(f(x,y),z) x∀y∃z (x+y=z) Truth

(4) ∃xF(f(x,y),g(x,z)) ∃x(x+1=2x) Truth

(5) F(f(x,a), g(y,a)) x+0=1×0 Truth

(6) ∀x(F(x,y)→∃yF(f(x,a), g(y,a))) ∀x(x=1→∃y(x+2=2y)) False

↳Classification of first-order Logic formulas

- ***Tautology*** (logically valid formula): no false interpretation and assignment.
- ***Contradiction*** (contravalid formula): no true interpretation and assignment.
- ***Satisfiable formula***: at least one true interpretation and assignment.

i

- A tautology is a satisfiable formula, but the converse is not true.
- In first-order logic, the satisfiability (tautology, contradiction) of a formula is undecidable, meaning there is no algorithm that can determine in finite steps whether a given formula is satisfiable (a tautology, a contradiction).

↳Substitution instance of a propositional formula

- **Definition 3.6:** Let A_0 be a propositional formula containing propositional variables $p_1,p_2,\ldots,p_n$, and let $A_1,A_2,\ldots,A_n$ be predicate formulas. The formula A, obtained by uniformly replacing each p_i (for $1\le i\le n$) in A_0 with A_i, is called a ***substitution instance*** of A_0.
  - **Such as:** $F(x)\rightarrow G(x)$ and $\forall xF(x)\rightarrow\exists yG(y)$ are substitution instances of $p\rightarrow q$.
- **Theorem 3.2:** All substitution instances of a tautology are **logically valid**, and all substitution instances of a contradiction are **contradictions.**

↳Classification of first-order Logic formulas(e.g.)

e.g.⋙**Example:** Determine the type of the following formula:

(1) $\forall x(F(x)\to G(x))$

I_1: D_1=R, $\bar{F}(x)$: x **integer**, $\bar{G}(x)$: x is **rational**. (1) **is a true proposition.**

I_2: D_2=R, $\bar{F}(x)$: x **integer**, $\bar{G}(x)$: x is **natural number**. (1) **is a false proposition.** (1) is a **satisfiable formula** (not logically valid formula).

(2) $\neg(\forall xF(x,y))\vee(\forall xF(x,y))$,

$\neg p\vee p$ substitution instances , $\neg p\vee p$ tautology, (2) is a **tautology.**

(3) $\neg(\forall xF(x)\to\exists yG(y))\wedge\exists yG(y)$,

$\neg(p\to q)\wedge q$ substitution instances, $\neg(p\to q)\wedge q$ contradiction, (3) is a **contradiction.**

↳Classification of first-order Logic formulas(e.g.)

e.g. **Example:** Determine the type of the following formula:

(4) $\forall x F(x,y)$

I_1: D_1=N, $\overline{F}(x,y)$: $x \ge y$, Assign $\sigma(y)=0$.

(4) is a true proposition.

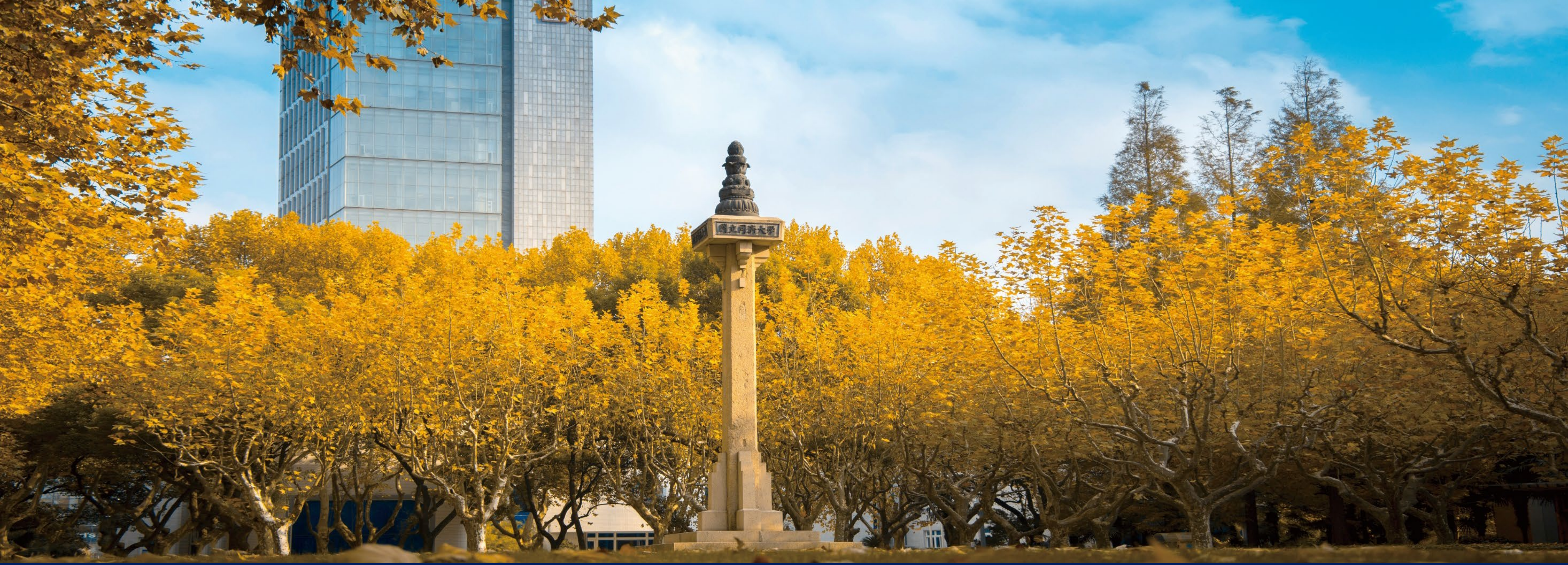
I_2: D_2=N, $\overline{F}(x,y)$: $x \ge y$, Assign $\sigma(y)=1$.

(4) is a false proposition.

(4) is a **satisfiable formula** (not logically valid formula).

Objective：

Key Concepts：



Discrete Mathematics 2025 Spring

魏可佶　kejiwei@tongji.edu.cn

- 3.1 Basic Concepts of First-Order Logic
- 3.2 Equivalence Calculus of First-Order Logic

- **3.2.1 First-Order Logic Equivalences and Substitution Rules**

  Basic Equivalences Substitution Rules, Renaming Rules

- 3.2.2 Prenex normal form of first-order logic

2.2.1 Equivalence Expressions and Equivalence Calculus
↳ Basic Equivalence Expressions

- Double Negation Law: $\neg\neg A \Leftrightarrow A$
- Idempotent Law: $A \vee A \Leftrightarrow A$, $A \wedge A \Leftrightarrow A$
- Commutative Law: $A \vee B \Leftrightarrow B \vee A$, $A \wedge B \Leftrightarrow B \wedge A$
- Associative Law: $(A \vee B) \vee C \Leftrightarrow A \vee (B \vee C)$
  $(A \wedge B) \wedge C \Leftrightarrow A \wedge (B \wedge C)$
- Distributive Law: $A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$
  $A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$
- De Morgan‘s Laws: $\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$
  $\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$
- Absorption Law: $A \vee (A \wedge B) \Leftrightarrow A$
  $A \wedge (A \vee B) \Leftrightarrow A$

同济经管 TONGJI SEM

2.2.1 Equivalence Expressions and Equivalence Calculus

↳ Basic Equivalence Expressions (cont.)

- Zero Law: $A \vee 1 \Leftrightarrow 1, \quad A \wedge 0 \Leftrightarrow 0$
- Identity Law: $A \vee 0 \Leftrightarrow A, \quad A \wedge 1 \Leftrightarrow A$
- Law of the Excluded Middle: $A \vee \neg A \Leftrightarrow 1$
- Law of Contradiction: $A \wedge \neg A \Leftrightarrow 0$
- Implication Equivalence: $A \rightarrow B \Leftrightarrow \neg A \vee B$
- Biconditional Equivalence: $A \leftrightarrow B \Leftrightarrow (A \rightarrow B) \wedge (B \rightarrow A)$
- Contraposition: $A \rightarrow B \Leftrightarrow \neg B \rightarrow \neg A$
- Negation of Equivalence: $A \leftrightarrow B \Leftrightarrow \neg A \leftrightarrow \neg B$
- Reductio ad Absurdum (Proof by Contradiction):

$$(A \rightarrow B) \wedge (A \rightarrow \neg B) \Leftrightarrow \neg A$$

↳ Tautology $A \leftrightarrow B$ vs. Equivalence $A \Leftrightarrow B$

- **Definition 3.7:** If $A \leftrightarrow B$ is a **tautology** (a valid formula), then A and B are called equivalent, denoted by $A \Leftrightarrow B$, and $A \Leftrightarrow B$ is referred to as an ***equivalence***.
- There are 24 propositional basic equivalences and their substitution examples, all of which are equivalences in first-order logic.
- For example:

$$\forall x F(x) \rightarrow \exists y G(y) \Leftrightarrow \neg \forall x F(x) \vee \exists y G(y)$$

$$\neg(\forall x F(x) \vee \exists y G(y)) \Leftrightarrow \neg \forall x F(x) \wedge \neg \exists y G(y)$$ **and so on**

- **Quantifier Elimination Equivalences:** To transform logical expressions to remove quantifiers (∃, ∀), yielding an equivalent **quantifier-free** form.
  - Let $D=\{a_1,a_2,\ldots,a_n\}$

$$\forall x A(x) \Leftrightarrow A(a_1) \wedge A(a_2) \wedge \ldots \wedge A(a_n)$$

$$\exists x A(x) \Leftrightarrow A(a_1) \vee A(a_2) \vee \ldots \vee A(a_n)$$

- **Quantifier Negation Equivalences: To** convert quantifiers (**∀ and ∃**) and negation (**¬**), changing the scope of the negation without altering the logical meaning.
  - Let $A(x)$ be a formula in which x appears freely.

$$\neg \forall x A(x) \Leftrightarrow \exists x \neg A(x)$$

$$\neg \exists x A(x) \Leftrightarrow \forall x \neg A(x)$$

同济经管 TONGJI SEM

- **Quantifier Distribution Equivalences:** To allocate or restructure quantifiers (∀, ∃) to interact correctly with logical operations (∧, ∨, →) while preserving logical equivalence.
  - $\forall x(A(x) \land B(x)) \Leftrightarrow \forall x A(x) \land \forall x B(x)$

    $\exists x(A(x) \lor B(x)) \Leftrightarrow \exists x A(x) \lor \exists x B(x)$

Attention：∀ to ∨， ∃ to ∧ no Quantifier Distribution Equivalences.

CAMEA 中国高质量MBA教育认证 AACSB ACCREDITED EQUIS ACCREDITED

同济经管 TONGJI SEM

- **Quantifier scope reduction and expansion equivalences**: To adjust the scope of quantifiers (∀, ∃) while preserving logical equivalence.
  - Let $A(x)$ be a formula in which x appears freely, and let B be a formula in which x does not appear.

Universal quantifier	Existential quantifier
$\forall x(A(x)\vee B)\Leftrightarrow\forall xA(x)\vee B$	$\exists x(A(x)\vee B)\Leftrightarrow\exists xA(x)\vee B$
$\forall x(A(x)\wedge B)\Leftrightarrow\forall xA(x)\wedge B$	$\exists x(A(x)\wedge B)\Leftrightarrow\exists xA(x)\wedge B$
$\forall x(A(x)\rightarrow B)\Leftrightarrow\exists xA(x)\rightarrow B$	$\exists x(A(x)\rightarrow B)\Leftrightarrow\forall xA(x)\rightarrow B$
$\forall x(B\rightarrow A(x))\Leftrightarrow B\rightarrow\forall xA(x)$	$\exists x(B\rightarrow A(x))\Leftrightarrow B\rightarrow\exists xA(x)$

- **Substitution Rule:**

  Let $\Phi(A)$ be a formula containing formula A, and $\Phi(B)$ be the formula obtained by replacing all occurrences of A in $\Phi(A)$ with formula B. Then, $\Phi(A) \Leftrightarrow \Phi(B)$.

- **Renaming Rule:**

  In a formula A, change the bound variables (and their occurrences within the scope of the quantifier) of a quantifier to an individual term that has not appeared within the scope of that quantifier. The rest of the formula remains unchanged, and the resulting formula is denoted as A'. Then, $A' \Leftrightarrow A$.

Note:

(1) ***Substitution*** can be used to transform expressions and find equivalent forms of expression.

(2) ***Renaming*** can eliminate variable name conflicts and clarify the scope of quantifiers.

(3) When substituting, the replaced terms should not become variables within the scope of some quantifier.

(4) When renaming, only the variable names bound by quantifiers are changed, and the rest of the formula structure remains unchanged.

e.g. **Example:** Eliminate the individual variables that appear both in the constraints and as free variables in the equation.

(1) $\forall xF(x,y,z) \rightarrow \exists yG(x,y,z)$

$\Leftrightarrow \forall uF(u,y,z) \rightarrow \exists yG(x,y,z)$

$\Leftrightarrow \forall uF(u,y,z) \rightarrow \exists vG(x,v,z)$ （Renaming Rule Equivalence)

Avoid variable confusion and improve expression readability and consistency.

(2) $\forall x(F(x,y) \rightarrow \exists yG(x,y,z))$

$\Leftrightarrow \forall x(F(x,y) \rightarrow \exists tG(x,t,z))$ （Renaming Rule Equivalence）

Only changed the bound variable name of the existential quantifier y.

e.g. **Example:** Let the domain of individuals be $D=\{a,b,c\}$, eliminate the quantifiers in the following formula:

(1) $\forall x(F(x)\rightarrow G(x))$

$\Leftrightarrow (F(a)\rightarrow G(a))\wedge(F(b)\rightarrow G(b))\wedge(F(c)\rightarrow G(c))$

(2) $\forall x(F(x)\vee\exists yG(y))$

$\Leftrightarrow \forall xF(x)\vee\exists yG(y)$ （ Quantifier scope reduction ）

$\Leftrightarrow(F(a)\wedge F(b)\wedge F(c))\vee(G(a)\vee G(b)\vee G(c))$

(3) $\exists x\forall yF(x,y)$

$\Leftrightarrow \exists x(F(x,a)\wedge F(x,b)\wedge F(x,c))$

$\Leftrightarrow (F(a,a)\wedge F(a,b)\wedge F(a,c))\vee(F(b,a)\wedge F(b,b)\wedge F(b,c))$
$\vee(F(c,a)\wedge F(c,b)\wedge F(c,c))$

e.g. Example: Given *I*: (a) *D*={2,3}, (b) $\bar{f}$: $\bar{f}(2)=3, \bar{f}(3)=2$,

(c) $\bar{F}(x)$: x is even, $\bar{G}(x,y)$: x=2 ∨ y=2, $\bar{L}(x,y)$: x=y.

Solve the true value under *I*:

(1) $\exists x(F(f(x)) \wedge G(x, f(x)))$

Solve: $(F(f(2)) \wedge G(2, f(2))) \vee (F(f(3)) \wedge G(3, f(3)))$

$\Leftrightarrow (1 \wedge 1) \vee (0 \wedge 1) \Leftrightarrow 1$

(2) $\exists x \forall y L(x,y)$

Solve: $\forall y L(2,y) \vee \forall y L(3,y)$

$\Leftrightarrow (L(2,2) \wedge L(2,3)) \vee (L(3,2) \wedge L(3,3))$

$\Leftrightarrow (1 \wedge 0) \vee (0 \wedge 1) \Leftrightarrow 0$

e.g. **Example:** Prove the following equivalence:

$$\neg \exists x(M(x) \wedge F(x)) \Leftrightarrow \forall x(M(x) \to \neg F(x))$$

Prove:

Left $\Leftrightarrow \forall x\, \neg(M(x) \wedge F(x))$ （De Morgan's laws for quantifiers）

$\Leftrightarrow \forall x(\neg M(x) \vee \neg F(x))$

$\Leftrightarrow \forall x(M(x) \to \neg F(x))$

The proof applies three key logical transformation rules: **quantifier negation**, **De Morgan's laws**, and **implication equivalence**.

- 3.2.1 First-Order Logic Equivalences and Substitution Rules

  Basic Equivalences Substitution Rules, Renaming Rules

- 3.2.2 Prenex normal form of first-order logic

- **Definition 3.8:** Let *A* be a first-order logic formula. If *A* has the form

$$Q_1x_1Q_2x_2\ldots Q_kx_k\ B$$

where each Q_i is either ∀ or ∃ (for $1 \le i \le k$), and *B* is a formula without quantifiers, then *A* is called a ***prenex normal form***.

e.g. **Examples:**

$\forall x \exists y (F(x) \to (G(y) \land H(x,y)))$ (prenex normal form)

$\forall x \neg (F(x) \land G(x))$ (prenex normal form)

$\forall x (F(x) \to \exists y (G(y) \land H(x,y)))$ (not prenex normal form)

$\neg \exists x (F(x) \land G(x))$ (not prenex normal form)

Objective :

Key Concepts :

Objective :

Key Concepts ：