Review of Math

MEB: Chapter 9

EECS3342 E: System Specification and Refinement
Fall 2025

CHEN-WEI WANG

Learning Outcomes of this Lecture

This module is designed to help you **review**:

- Propositional Logic
- Predicate Logic
- Sets, Relations, and Functions

Propositional Logic (1)

- A *proposition* is a statement of claim that must be of either *true* or *false*, but not both.
- Basic logical operands are of type Boolean: *true* and *false*.
- We use logical operators to construct compound statements.
  - Unary logical operator: negation ($\neg$)

p	$\neg p$
true	*false*
false	*true*

  - Binary logical operators: conjunction ($\wedge$), disjunction ($\vee$), implication ($\Rightarrow$), equivalence ($\equiv$), and if-and-only-if ($\iff$).

p	q	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \iff q$	$p \equiv q$
true	*true*	*true*	*true*	*true*	*true*	*true*
true	*false*	*false*	*true*	*false*	*false*	*false*
false	*true*	*false*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*true*	*true*	*true*

Propositional Logic: Implication (1)

- Written as $p \Rightarrow q$ [pronounced as "p implies q"]
  - We call p the antecedent, assumption, or premise.
  - We call q the consequence or conclusion.
- Compare the *truth* of $p \Rightarrow q$ to whether a contract is *honoured*:
  - antecedent/assumption/premise $p \approx$ promised terms [e.g., salary]
  - consequence/conclusion $q \approx$ obligations [e.g., duties]
- When the promised terms are met, then the contract is:
  - *honoured* if the obligations fulfilled. [$(true \Rightarrow true) \iff true$]
  - *breached* if the obligations violated. [$(true \Rightarrow false) \iff false$]
- When the promised terms are not met, then:
  - Fulfilling the obligation (q) or not ($\neg q$) does *not breach* the contract.

p	q	$p \Rightarrow q$
false	*true*	*true*
false	*false*	*true*

Propositional Logic: Implication (2)

There are alternative, equivalent ways to expressing $p \Rightarrow q$:

- q **if** p

  q is *true* if p is *true*
- p **only if** q

  If p is *true*, then for $p \Rightarrow q$ to be *true*, it can only be that q is also *true*.
  Otherwise, if p is *true* but q is *false*, then $(\textit{true} \Rightarrow \textit{false}) \equiv \textit{false}$.

Note. To prove $p \equiv q$, prove $p \iff q$ (pronounced: "p if and only if q"):

  - p **if** q [$q \Rightarrow p$]
  - p **only if** q [$p \Rightarrow q$]
- p is **sufficient** for q

  For q to be *true*, it is sufficient to have p being *true*.
- q is **necessary** for p [similar to p **only if** q]

  If p is *true*, then it is necessarily the case that q is also *true*.
  Otherwise, if p is *true* but q is *false*, then $(\textit{true} \Rightarrow \textit{false}) \equiv \textit{false}$.
- q **unless** $\neg p$ [When is $p \Rightarrow q$ *true*?]

  If q is *true*, then $p \Rightarrow q$ *true* regardless of p.
  If q is *false*, then $p \Rightarrow q$ cannot be *true* unless p is *false*.

Propositional Logic: Implication (3)

Given an implication $p \Rightarrow q$, we may construct its:

- **Inverse**: $\neg p \Rightarrow \neg q$ [negate antecedent and consequence]
- **Converse**: $q \Rightarrow p$ [swap antecedent and consequence]
- **Contrapositive**: $\neg q \Rightarrow \neg p$ [inverse of converse]

Propositional Logic (2)

- **Axiom**: Definition of $\Rightarrow$

$$p \Rightarrow q \equiv \neg p \vee q$$

- **Theorem**: Identity of $\Rightarrow$

$$\textit{true} \Rightarrow p \equiv p$$

- **Theorem**: Zero of $\Rightarrow$

$$\textit{false} \Rightarrow p \equiv \textit{true}$$

- **Axiom**: De Morgan

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

- **Axiom**: Double Negation

$$p \equiv \neg(\neg p)$$

- **Theorem**: Contrapositive

$$p \Rightarrow q \equiv \neg q \Rightarrow \neg p$$

Predicate Logic (1)

- A *predicate* is a ***universal*** or ***existential*** statement about objects in some universe of disclosure.
- Unlike propositions, predicates are typically specified using ***variables***, each of which declared with some ***range*** of values.
- We use the following symbols for common numerical ranges:
  - $\mathbb{Z}$: the set of integers [$-\infty, \dots, -1, 0, 1, \dots, +\infty$]
  - $\mathbb{N}$: the set of natural numbers [$0, 1, \dots, +\infty$]
- Variable(s) in a predicate may be ***quantified***:
  - *Universal quantification* :

    All values that a variable may take satisfy certain property.
    e.g., Given that i is a natural number, i is *always* non-negative.
  - *Existential quantification* :

    Some value that a variable may take satisfies certain property.
    e.g., Given that i is an integer, i *can be* negative.

Predicate Logic (2.1): Universal Q. ($\forall$)

- A *universal quantification* has the form $(\forall X \bullet R \Rightarrow P)$
  - X is a comma-separated list of variable names
  - R is a *constraint on types/ranges* of the listed variables
  - P is a *property* to be satisfied
- ***For all*** (combinations of) values of variables listed in X that satisfies R, it is the case that P is satisfied.
  - $\forall i \bullet i \in \mathbb{N} \Rightarrow i \geq 0$ [*true*]
  - $\forall i \bullet i \in \mathbb{Z} \Rightarrow i \geq 0$ [*false*]
  - $\forall i, j \bullet i \in \mathbb{Z} \land j \in \mathbb{Z} \Rightarrow i < j \lor i > j$ [*false*]
- *Proof Strategies*
  1. How to prove $(\forall X \bullet R \Rightarrow P)$ ***true***?
     - **Hint.** When is $R \Rightarrow P$ ***true***? [*true* $\Rightarrow$ *true*, *false* $\Rightarrow$ _]
     - Show that for all instances of $x \in X$ s.t. $R(x)$, $P(x)$ holds.
     - Show that for all instances of $x \in X$ it is the case $\neg R(x)$.
  2. How to prove $(\forall X \bullet R \Rightarrow P)$ ***false***?
     - **Hint.** When is $R \Rightarrow P$ ***false***? [*true* $\Rightarrow$ *false*]
     - Give a ***witness/counterexample*** of $x \in X$ s.t. $R(x)$, $\neg P(x)$ holds.

Predicate Logic (2.2): Existential Q. ($\exists$)

- An *existential quantification* has the form $(\exists X \bullet R \land P)$
  - X is a comma-separated list of variable names
  - R is a *constraint on types/ranges* of the listed variables
  - P is a *property* to be satisfied
- ***There exist*** (a combination of) values of variables listed in X that satisfy both R and P.
  - $\exists i \bullet i \in \mathbb{N} \land i \geq 0$ [*true*]
  - $\exists i \bullet i \in \mathbb{Z} \land i \geq 0$ [*true*]
  - $\exists i, j \bullet i \in \mathbb{Z} \land j \in \mathbb{Z} \land (i < j \lor i > j)$ [*true*]
- *Proof Strategies*
  1. How to prove $(\exists X \bullet R \land P)$ ***true***?
     - **Hint.** When is $R \land P$ ***true***? [*true* $\land$ *true*]
     - Give a ***witness*** of $x \in X$ s.t. $R(x)$, $P(x)$ holds.
  2. How to prove $(\exists X \bullet R \land P)$ ***false***?
     - **Hint.** When is $R \land P$ ***false***? [*true* $\land$ *false*, *false* $\land$ _]
     - Show that for all instances of $x \in X$ s.t. $R(x)$, $\neg P(x)$ holds.
     - Show that for all instances of $x \in X$ it is the case $\neg R(x)$.

Predicate Logic (3): Exercises

- Prove or disprove: $\forall x \bullet (x \in \mathbb{Z} \land 1 \leq x \leq 10) \Rightarrow x > 0$.
  All 10 integers between 1 and 10 are greater than 0.
- Prove or disprove: $\forall x \bullet (x \in \mathbb{Z} \land 1 \leq x \leq 10) \Rightarrow x > 1$.
  Integer 1 (a witness/counterexample) in the range between 1 and 10 is *not* greater than 1.
- Prove or disprove: $\exists x \bullet (x \in \mathbb{Z} \land 1 \leq x \leq 10) \land x > 1$.
  Integer 2 (a witness) in the range between 1 and 10 is greater than 1.
- Prove or disprove that $\exists x \bullet (x \in \mathbb{Z} \land 1 \leq x \leq 10) \land x > 10$?
  All integers in the range between 1 and 10 are *not* greater than 10.

Predicate Logic (4): Switching Quantifications

Conversions between $\forall$ and $\exists$:

$$(\forall X \bullet R \Rightarrow P) \iff \neg(\exists X \bullet R \land \neg P)$$
$$(\exists X \bullet R \land P) \iff \neg(\forall X \bullet R \Rightarrow \neg P)$$

Sets: Definitions and Membership

- A *set* is a collection of objects.
  - Objects in a set are called its *elements* or *members*.
  - *Order* in which elements are arranged does not matter.
  - An element can appear *at most once* in the set.
- We may define a set using:
  - ***Set Enumeration***: Explicitly list all members in a set.
    e.g., $\{1, 3, 5, 7, 9\}$
  - ***Set Comprehension***: Implicitly specify the condition that all members satisfy.
    e.g., $\{x \mid 1 \le x \le 10 \land x \texttt{ is an odd number}\}$
- An empty set (denoted as $\{\}$ or $\varnothing$) has no members.
- We may check if an element is a *member* of a set:
  e.g., $5 \in \{1, 3, 5, 7, 9\}$ [*true*]
  e.g., $4 \notin \{x \mid x \le 1 \le 10, x \texttt{ is an odd number}\}$ [*true*]
- The number of elements in a set is called its *cardinality*.
  e.g., $|\varnothing| = 0$, $|\{x \mid x \le 1 \le 10, x \texttt{ is an odd number}\}| = 5$

Set Relations

Given two sets S_1 and S_2:

- S_1 is a ***subset*** of S_2 if every member of S_1 is a member of S_2.

$$S_1 \subseteq S_2 \iff (\forall x \bullet x \in S1 \Rightarrow x \in S2)$$

- S_1 and S_2 are ***equal*** iff they are the subset of each other.

$$S_1 = S_2 \iff S_1 \subseteq S_2 \land S_2 \subseteq S_1$$

- S_1 is a ***proper subset*** of S_2 if it is a strictly smaller subset.

$$S_1 \subset S_2 \iff S_1 \subseteq S_2 \land |S1| < |S2|$$

Set Relations: Exercises

$? \subseteq S$ always holds	[$\varnothing$ and S]
$? \subset S$ always fails	[S]
$? \subset S$ holds for some S and fails for some S	[$\varnothing$]
$S_1 = S_2 \Rightarrow S_1 \subseteq S_2$?	[Yes]
$S_1 \subseteq S_2 \Rightarrow S_1 = S_2$?	[No]

Set Operations

Given two sets S_1 and S_2:

- ***Union*** of S_1 and S_2 is a set whose members are in either.

$$S_1 \cup S_2 = \{x \mid x \in S_1 \lor x \in S_2\}$$

- ***Intersection*** of S_1 and S_2 is a set whose members are in both.

$$S_1 \cap S_2 = \{x \mid x \in S_1 \land x \in S_2\}$$

- ***Difference*** of S_1 and S_2 is a set whose members are in S_1 but not S_2.

$$S_1 \setminus S_2 = \{x \mid x \in S_1 \land x \notin S_2\}$$

Power Sets

The *power set* of a set S is a *set* of all S's *subsets*.

$$\mathbb{P}(S) = \{s \mid s \subseteq S\}$$

The power set contains subsets of *cardinalities* 0, 1, 2, ..., $|S|$.
e.g., $\mathbb{P}(\{1,2,3\})$ is a set of sets, where each member set s has cardinality 0, 1, 2, or 3:

$$\left\{ \begin{array}{l} \varnothing, \\ \{1\},\ \{2\},\ \{3\}, \\ \{1,2\},\ \{2,3\},\ \{3,1\}, \\ \{1,2,3\} \end{array} \right\}$$

Exercise: What is $\mathbb{P}(\{1,2,3,4,5\}) \setminus \mathbb{P}(\{1,2,3\})$?

Set of Tuples

Given n sets S_1, S_2, ..., S_n, a ***cross/Cartesian product*** of theses sets is a set of n-tuples.
Each ***n-tuple*** $(e_1, e_2, \dots, e_n)$ contains n elements, each of which a member of the corresponding set.

$$S_1 \times S_2 \times \cdots \times S_n = \{(e_1, e_2, \dots, e_n) \mid e_i \in S_i \land 1 \le i \le n\}$$

e.g., $\{a,b\} \times \{2,4\} \times \{\$,\&\}$ is a set of triples:

$$\begin{array}{rl} & \{a,b\} \times \{2,4\} \times \{\$,\&\} \\ = & \{\ (e_1, e_2, e_3) \mid e_1 \in \{a,b\} \land e_2 \in \{2,4\} \land e_3 \in \{\$,\&\}\ \} \\ = & \left\{ \begin{array}{l} (a,2,\$),(a,2,\&),(a,4,\$),(a,4,\&), \\ (b,2,\$),(b,2,\&),(b,4,\$),(b,4,\&) \end{array} \right\} \end{array}$$

Relations (1): Constructing a Relation

A *relation* is a set of mappings, each being an ***ordered pair*** that maps a member of set S to a member of set T.
e.g., Say $S = \{1,2,3\}$ and $T = \{a,b\}$

- $\varnothing$ is the ***minimum*** relation (i.e., an empty relation).
- $S \times T$ is the ***maximum*** relation (say r_1) between S and T, mapping from each member of S to each member in T:

$$\{(1,a),(1,b),(2,a),(2,b),(3,a),(3,b)\}$$

- $\{(x,y) \mid (x,y) \in S \times T \land x \neq 1\}$ is a relation (say r_2) that maps only some members in S to every member in T:

$$\{(2,a),(2,b),(3,a),(3,b)\}$$

Relations (2.1): Set of Possible Relations

- We use the ***power set*** operator to express the set of ***all possible relations*** on S and T:

$$\mathbb{P}(S \times T)$$

  Each member in $\mathbb{P}(S \times T)$ is a relation.
- To declare a relation variable r, we use the colon (`:`) symbol to mean ***set membership***:

$$r : \mathbb{P}(S \times T)$$

- Or alternatively, we write:

$$r : S \leftrightarrow T$$

  where the set $S \leftrightarrow T$ is synonymous to the set $\mathbb{P}(S \times T)$

Relations (2.2): Exercise

Enumerate $\{a, b\} \leftrightarrow \{1, 2, 3\}$.

- **Hints**:
  - You may enumerate all relations in $\mathbb{P}(\{a, b\} \times \{1, 2, 3\})$ via their *cardinalities*: 0, 1, ..., $|\{a, b\} \times \{1, 2, 3\}|$.
  - What's the *maximum* relation in $\mathbb{P}(\{a, b\} \times \{1, 2, 3\})$?

    $\{ (a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3) \}$
- The answer is a set containing ***all*** of the following relations:
  - Relation with cardinality 0: $\varnothing$
  - How many relations with cardinality 1? $[\ \binom{|\{a,b\}\times\{1,2,3\}|}{1} = 6\ ]$
  - How many relations with cardinality 2? $[\ \binom{|\{a,b\}\times\{1,2,3\}|}{2} = \frac{6\times5}{2!} = 15\ ]$

    ...
  - Relation with cardinality $|\{a, b\} \times \{1, 2, 3\}|$:

    $\{ (a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3) \}$

Relations (3.1): Domain, Range, Inverse

Given a relation

r = {(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)}

- ***domain*** **of** r: set of first-elements from r
  - Definition: $\text{dom}(r) = \{ d \mid (d, r') \in r \}$
  - e.g., $\text{dom}(r) = \{a, b, c, d, e, f\}$
  - ASCII syntax: `dom(r)`
- ***range*** **of** r: set of second-elements from r
  - Definition: $\text{ran}(r) = \{ r' \mid (d, r') \in r \}$
  - e.g., $\text{ran}(r) = \{1, 2, 3, 4, 5, 6\}$
  - ASCII syntax: `ran(r)`
- ***inverse*** **of** r: a relation like r with elements swapped
  - Definition: $r^{-1} = \{ (r', d) \mid (d, r') \in r \}$
  - e.g., $r^{-1} = \{(1, a), (2, b), (3, c), (4, a), (5, b), (6, c), (1, d), (2, e), (3, f)\}$
  - ASCII syntax: `r~`

Relations (3.2): Image

Given a relation

r = {(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)}

relational image **of** r **over set** s: sub-range of r mapped by s.

- Definition: $r[s] = \{ r' \mid (d, r') \in r \land d \in s \}$
- e.g., $r[\{a, b\}] = \{1, 2, 4, 5\}$
- ASCII syntax: `r[s]`

Relations (3.3): Restrictions

Given a relation

r = {(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)}

- ***domain restriction*** **of** r **over set** ds: sub-relation of r with domain ds.
  - Definition: $ds \lhd r = \{ (d, r') \mid (d, r') \in r \land d \in ds \}$
  - e.g., $\{a, b\} \lhd r = \{(\mathbf{a}, 1), (\mathbf{b}, 2), (\mathbf{a}, 4), (\mathbf{b}, 5)\}$
  - ASCII syntax: `ds <| r`
- ***range restriction*** **of** r **over set** rs: sub-relation of r with range rs.
  - Definition: $r \rhd rs = \{ (d, r') \mid (d, r') \in r \land r' \in rs \}$
  - e.g., $r \rhd \{1, 2\} = \{(a, \mathbf{1}), (b, \mathbf{2}), (d, \mathbf{1}), (e, \mathbf{2})\}$
  - ASCII syntax: `r |> rs`

Relations (3.4): Subtractions

Given a relation

$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$

- ***domain subtraction*** of r over set ds : sub-relation of r with domain not ds.
  - Definition: $ds \lhd\!\!\!- r = \{\, (d, r') \mid (d, r') \in r \wedge d \notin ds \,\}$
  - e.g., $\{a, b\} \lhd\!\!\!- r = \{(\mathbf{c}, 3), (\mathbf{c}, 6), (\mathbf{d}, 1), (\mathbf{e}, 2), (\mathbf{f}, 3)\}$
  - ASCII syntax: `ds <<| r`
- ***range subtraction*** of r over set rs : sub-relation of r with range not rs.
  - Definition: $r \rhd\!\!\!- rs = \{\, (d, r') \mid (d, r') \in r \wedge r' \notin rs \,\}$
  - e.g., $r \rhd\!\!\!- \{1, 2\} = \{(c, \mathbf{3}), (a, \mathbf{4}), (b, \mathbf{5}), (c, \mathbf{6}), (f, \mathbf{3})\}$
  - ASCII syntax: `r |>> rs`

Relations (3.5): Overriding

Given a relation

$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$

overriding of r with relation t : a relation which agrees with t within $\mathrm{dom}(t)$, and agrees with r outside $\mathrm{dom}(t)$

- Definition: $r \mathbin{\lhd\!\!+} t = \{\, (d, r') \mid (d, r') \in t \vee ((d, r') \in r \wedge d \notin \mathrm{dom}(t)) \,\}$
- e.g.,

$$
\begin{aligned}
& r \mathbin{\lhd\!\!+} \{(a, 3), (c, 4)\} \\
= \; & \underbrace{\{(a, 3), (c, 4)\}}_{\{(d,r') \mid (d,r') \in t\}} \cup \underbrace{\{(b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}}_{\{(d,r') \mid (d,r') \in r \wedge d \notin \mathrm{dom}(t)\}} \\
= \; & \{(a, 3), (c, 4), (b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}
\end{aligned}
$$

- ASCII syntax: `r <+ t`

Relations (4): Exercises

1. *Define $r[s]$ in terms of other relational operations.*

   Answer: $r[s] = \mathrm{ran}(s \lhd r)$

   e.g.,

$$
r[\underbrace{\{a, b\}}_{s}] = \mathrm{ran}(\underbrace{\{(\mathbf{a}, 1), (\mathbf{b}, 2), (\mathbf{a}, 4), (\mathbf{b}, 5)\}}_{\{a,b\} \lhd r}) = \{1, 2, 4, 5\}
$$

2. *Define $r \mathbin{\lhd\!\!+} t$ in terms of other relational operators.*

   Answer: $r \mathbin{\lhd\!\!+} t = t \cup (\mathrm{dom}(t) \lhd\!\!\!- r)$

   e.g.,

$$
\begin{aligned}
& r \mathbin{\lhd\!\!+} \underbrace{\{(a, 3), (c, 4)\}}_{t} \\
= \; & \underbrace{\{(a, 3), (c, 4)\}}_{t} \cup \underbrace{\{(b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}}_{\underbrace{\mathrm{dom}(t)}_{\{a,c\}} \lhd\!\!\!- r} \\
= \; & \{(a, 3), (c, 4), (b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}
\end{aligned}
$$

Functions (1): Functional Property

- A ***relation*** r on sets S and T (i.e., $r \in S \leftrightarrow T$) is also a *function* if it satisfies the ***functional property***:

  `isFunctional(r)`

  $\Longleftrightarrow$

  $\forall s, t_1, t_2 \bullet (s \in S \wedge t_1 \in T \wedge t_2 \in T) \Rightarrow ((s, t_1) \in r \wedge (s, t_2) \in r \Rightarrow t_1 = t_2)$

  - That is, in a ***function***, it is forbidden for a member of S to map to more than one members of T.
  - Equivalently, in a ***function***, two distinct members of T cannot be mapped by the same member of S.
- e.g., Say $S = \{1, 2, 3\}$ and $T = \{a, b\}$, which of the following ***relations*** satisfy the above ***functional property***?
  - $S \times T$ [No]

    Witness 1: $(1, a), (1, b)$; ***Witness*** 2: $(2, a), (2, b)$; ***Witness*** 3: $(3, a), (3, b)$.
  - $(S \times T) \setminus \{(x, y) \mid (x, y) \in S \times T \wedge x = 1\}$ [No]

    Witness 1: $(2, a), (2, b)$; ***Witness*** 2: $(3, a), (3, b)$
  - $\{(1, a), (2, b), (3, a)\}$ [Yes]
  - $\{(1, a), (2, b)\}$ [Yes]

Functions (2.1): Total vs. Partial

Given a **relation** $r \in S \leftrightarrow T$

- r is a ***partial function*** if it satisfies the ***functional property***:

  $$\boxed{r \in S \nrightarrow T} \iff (\texttt{isFunctional(r)} \land \mathrm{dom}(r) \subseteq S)$$

  Remark. $r \in S \nrightarrow T$ means there **may (or may not) be** $s \in S$ s.t. $r(s)$ is ***undefined*** (i.e., $r[\{s\}] = \varnothing$).
  - e.g., $\{\ \{(\mathbf{2}, a), (\mathbf{1}, b)\}, \{(\mathbf{2}, a), (\mathbf{3}, a), (\mathbf{1}, b)\}\ \} \subseteq \{1, 2, 3\} \nrightarrow \{a, b\}$
  - ASCII syntax: `r : +->`
- r is a ***total function*** if there is a mapping for each $s \in S$:

  $$\boxed{r \in S \rightarrow T} \iff (\texttt{isFunctional(r)} \land \mathrm{dom}(r) = S)$$

  Remark. $r \in S \rightarrow T$ implies $r \in S \nrightarrow T$, but not vice versa. Why?
  - e.g., $\{(\mathbf{2}, a), (\mathbf{3}, a), (\mathbf{1}, b)\} \in \{1, 2, 3\} \rightarrow \{a, b\}$
  - e.g., $\{(\mathbf{2}, a), (\mathbf{1}, b)\} \notin \{1, 2, 3\} \rightarrow \{a, b\}$
  - ASCII syntax: `r : -->`

Functions (2.2): Relation Image vs. Function Application

- Recall: A ***function*** is a ***relation***, but a ***relation*** is not necessarily a ***function***.
- Say we have a ***partial function*** $f \in \{1, 2, 3\} \nrightarrow \{a, b\}$:

  $$f = \{(\mathbf{3}, a), (\mathbf{1}, b)\}$$

  - With f wearing the ***relation*** hat, we can invoke *relational images* :

    $$\begin{array}{lcl} f[\{3\}] & = & \{a\} \\ f[\{1\}] & = & \{b\} \\ f[\{2\}] & = & \varnothing \end{array}$$

    Remark. $\Rightarrow |f[\{v\}]| \leq 1$ ∵:
    - each member in $\mathrm{dom}(f)$ is mapped to at most one member in $\mathrm{ran}(f)$
    - each input set $\{v\}$ is a **singleton** set
  - With f wearing the ***function*** hat, we can invoke *functional applications* :

    $$\begin{array}{lcl} f(3) & = & a \\ f(1) & = & b \\ f(2) & \texttt{is} & \textbf{\texttt{undefined}} \end{array}$$

Functions (2.3): Modelling Decision

An organization has a system for keeping **track** of its employees as to where they are on the premises (e.g., ``Zone A, Floor 23''). To achieve this, each employee is issued with an active badge which, when scanned, synchronizes their current positions to a central database.

Assume the following two sets:

- *Employee* denotes the **set** of all employees working for the organization.
- *Location* denotes the **set** of all valid locations in the organization.

1. Is it appropriate to *model/formalize* such a **track** functionality as a ***relation*** (i.e., $\textit{where\_is} \in \textit{Employee} \leftrightarrow \textit{Location}$)?
   Answer. No – an employee cannot be at distinct locations simultaneously.
   e.g., $\textit{where\_is}[\textit{Alan}] = \{$ ``Zone A, Floor 23'', ``Zone C, Floor 46'' $\}$
2. How about a ***total function*** (i.e., $\textit{where\_is} \in \textit{Employee} \rightarrow \textit{Location}$)?
   Answer. No – in reality, not necessarily all employees show up.
   e.g., $\textit{where\_is}(\textit{Mark})$ should be ***undefined*** if Mark happens to be on vacation.
3. How about a ***partial function*** (i.e., $\textit{where\_is} \in \textit{Employee} \nrightarrow \textit{Location}$)?
   Answer. Yes – this addresses the inflexibility of the total function.

Functions (3.1): Injective Functions

Given a ***function*** f (either partial or total):

- f is ***injective/one-to-one/an injection*** if f does **not** map more than one members of S to a single member of T.

  $$\textit{isInjective}(f) \iff \forall s_1, s_2, t \bullet (s_1 \in S \land s_2 \in S \land t \in T) \Rightarrow ((s_1, t) \in f \land (s_2, t) \in f \Rightarrow s_1 = s_2)$$

- If f is a ***partial injection***, we write: $\boxed{f \in S \rightarrowtail\!\!\!\!\!\nrightarrow T}$
  - e.g., $\{\ \varnothing, \{(1, \mathbf{a})\}, \{(2, \mathbf{a}), (3, \mathbf{b})\}\ \} \subseteq \{1, 2, 3\} \rightarrowtail\!\!\!\!\!\nrightarrow \{a, b\}$
  - e.g., $\{(1, \mathbf{b}), (2, a), (3, \mathbf{b})\} \notin \{1, 2, 3\} \rightarrowtail\!\!\!\!\!\nrightarrow \{a, b\}$ [total, not inj.]
  - e.g., $\{(1, \mathbf{b}), (3, \mathbf{b})\} \notin \{1, 2, 3\} \rightarrowtail\!\!\!\!\!\nrightarrow \{a, b\}$ [partial, not inj.]
  - ASCII syntax: `f : >+>`
- If f is a ***total injection***, we write: $\boxed{f \in S \rightarrowtail T}$
  - e.g., $\{1, 2, 3\} \rightarrowtail \{a, b\} = \varnothing$
  - e.g., $\{(2, d), (1, a), (3, c)\} \in \{1, 2, 3\} \rightarrowtail \{a, b, c, d\}$
  - e.g., $\{(\mathbf{2}, d), (\mathbf{1}, c)\} \notin \{1, 2, 3\} \rightarrowtail \{a, b, c, d\}$ [not total, inj.]
  - e.g., $\{(2, \mathbf{d}), (1, c), (3, \mathbf{d})\} \notin \{1, 2, 3\} \rightarrowtail \{a, b, c, d\}$ [total, not inj.]
  - ASCII syntax: `f : >->`

Functions (3.2): Surjective Functions

Given a ***function*** f (either partial or total):

- f is ***surjective***/*onto*/***a surjection*** if f maps to all members of T.

$$\texttt{isSurjective}(f) \iff \mathrm{ran}(f) = T$$

- If f is a ***partial surjection***, we write: $f \in S \mathrel{+\!\!\!\twoheadrightarrow} T$
  - e.g., $\{\ \{(1,\mathbf{b}),(2,\mathbf{a})\},\{(1,\mathbf{b}),(2,\mathbf{a}),(3,\mathbf{b})\}\ \} \subseteq \{1,2,3\} \mathrel{+\!\!\!\twoheadrightarrow} \{a,b\}$
  - e.g., $\{(2,\mathbf{a}),(1,\mathbf{a}),(3,\mathbf{a})\ \} \notin \{1,2,3\} \mathrel{+\!\!\!\twoheadrightarrow} \{a,b\}$ [total, not sur.]
  - e.g., $\{(2,\mathbf{b}),(1,\mathbf{b})\} \notin \{1,2,3\} \mathrel{+\!\!\!\twoheadrightarrow} \{a,b\}$ [partial, not sur.]
  - ASCII syntax: `f : +->>`
- If f is a ***total surjection***, we write: $f \in S \twoheadrightarrow T$
  - e.g., $\{\ \{(2,a),(1,b),(3,a)\},\{(2,b),(1,a),(3,b)\}\ \} \subseteq \{1,2,3\} \twoheadrightarrow \{a,b\}$
  - e.g., $\{(\mathbf{2},a),(\mathbf{3},b)\} \notin \{1,2,3\} \twoheadrightarrow \{a,b\}$ [not total, sur.]
  - e.g., $\{(2,\mathbf{a}),(3,\mathbf{a}),(1,\mathbf{a})\} \notin \{1,2,3\} \twoheadrightarrow \{a,b\}$ [total., not sur]
  - ASCII syntax: `f : -->>`

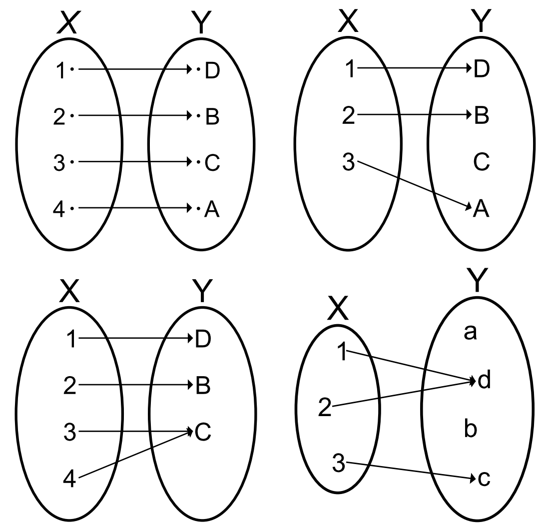
Functions (3.3): Bijective Functions

Given a function f:

f is ***bijective***/***a bijection***/*one-to-one correspondence* if f is ***total***, ***injective***, and ***surjective***.

- e.g., $\{1,2,3\} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \{a,b\} = \varnothing$
- e.g., $\{\ \{(1,a),(2,b),(3,c)\},\{(2,a),(3,b),(1,c)\}\ \} \subseteq \{1,2,3\} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \{a,b,c\}$
- e.g., $\{(\mathbf{2},b),(\mathbf{3},c),(\mathbf{4},a)\} \notin \{1,2,3,4\} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \{a,b,c\}$
  [not total, inj., sur.]
- e.g., $\{(1,\mathbf{a}),(2,b),(3,c),(4,\mathbf{a})\} \notin \{1,2,3,4\} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \{a,b,c\}$
  [total, not inj., sur.]
- e.g., $\{(1,\mathbf{a}),(2,\mathbf{c})\} \notin \{1,2\} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \{a,b,c\}$
  [total, inj., not sur.]
- ASCII syntax: `f : >->>`

Functions (4.1): Exercises



Functions (4.2): Modelling Decisions

1. Should an array `a` declared as "`String[] a`" be *modelled/formalized* as a *partial* function (i.e., $a \in \mathbb{Z} \mathrel{+\!\!\!\rightarrow} String$) or a *total* function (i.e., $a \in \mathbb{Z} \rightarrow String$)?
   Answer. $a \in \mathbb{Z} \rightarrow String$ is not appropriate as:
   - Indices are non-negative (i.e., $a(i)$, where $i < 0$, is ***undefined***).
   - Each array size is finite: not all positive integers are valid indices.
2. What does it mean if an **array** is *modelled/formalized* as a partial ***injection*** (i.e., $a \in \mathbb{Z} \mathrel{+\!\!\!\rightarrowtail} String$)?
   Answer. It means that the array does **not** contain any duplicates.
3. Can an integer array "`int[] a`" be *modelled/formalized* as a partial ***surjection*** (i.e., $a \in \mathbb{Z} \mathrel{+\!\!\!\twoheadrightarrow} \mathbb{Z}$)?
   Answer. Yes, if `a` stores all 2^{32} integers (i.e., $[-2^{31}, 2^{31}-1]$).
4. Can a string array "`String[] a`" be *modelled/formalized* as a partial ***surjection*** (i.e., $a \in \mathbb{Z} \mathrel{+\!\!\!\twoheadrightarrow} String$)?
   Answer. No $\because$ # possible strings is ∞.
5. Can an integer array "`int[]`" storing all 2^{32} values be *modelled/formalized* as a ***bijection*** (i.e., $a \in \mathbb{Z} \mathrel{\rightarrowtail\!\!\!\!\twoheadrightarrow} \mathbb{Z}$)?
   Answer. No, because it cannot be ***total*** (as discussed earlier).

Beyond this lecture ...

- For the *where_is* $\in$ *Employee* $\nrightarrow$ *Location* model, what does it mean when it is:
  - ***Injective*** [*where_is* $\in$ *Employee* $\rightarrowtail\!\!\!\!\!+$ *Location*]
  - ***Surjective*** [*where_is* $\in$ *Employee* $\twoheadrightarrow\!\!\!\!\!+$ *Location*]
  - ***Bijective*** [*where_is* $\in$ *Employee* $\rightarrowtail\!\!\!\!\twoheadrightarrow$ *Location*]
- Review examples discussed in your earlier math courses on ***logic*** and ***set theory***.

Index (1)

Index (2)

Index (3)

Index (4)