The Elements of Z

- language proper:
  small number of basic elements
- mathematical toolkit:
  an extensive library of commonly used tools
- style and conventions:
  tailored for software specifications

There are only two kinds of basic elements in Z:

- sets
- predicates

Set

What is a set?

A collection of objects.

A set can be defined by

- enumeration
  - red, yellow, and blue colors.
- characterization
  - The set of prime colors.

Typed Sets

a.k.a. homogeneous sets.

All elements are of the same knid, called *type*.

- { 747 jet, lamp }

Ok, but of no practical use. Excluded in Z.

A *type* is a category of objects that all of its members exhibit some distinctive commonality and are different from objects of other types.

Consider, *Student* and *Faculty*.

Given Sets

- The primitive types are introduced as given sets.
- Each given set is a different type.
- A type is represented by the set containing all its members.

$$[STUDENT, FACULTY]$$

Declarations

$$\begin{array}{|l} Mark, Kevin : STUDENT \\ Einstein : FACULTY \end{array}$$

A declaration is in the following form:

$$x_1, x_2, \ldots, x_n : T$$

Where $x_1, x_2, \ldots, x_n$ are names, and T is a type, which must be a set expression.

Pre-Defined Set

There is only one:

$[\mathbb{Z}]$

The set of integers.
Arithmetic operations:

- $x + y$, $x - y$, $x * y$
- $x \operatorname{div} y$, $x \bmod y$
- $-x$

Comparisons: also defined:

- $=, \neq, <, \leq, >, \geq$.

Defining Sets

- Empty set: $\varnothing$
- enumeration

$$\{2, 3, 5, 7, 11\}$$

- sub-ranges of integers.

$$n \mathrel{..} m$$

Free Type Definition

$GRADE ::= pass \mid fail$

- $GRADE$ is a given set.
- $pass$ and $fail$ are both member of $GRADE$.
- $pass$ and $fail$ are different entities, i.e., $pass \neq fail$.
- $pass$ and $fail$ are the only members of $GRADE$.

Characterization

$\{x : S \mid p(x)\}$

Examples:

- $\{x : \mathbb{Z} \mid 1 \leq x \leq 100\}$
  All the integers from 1 to 100.
- $\{x : \mathbb{Z} \mid x \bmod 2 = 0\}$
  All the even numbers, positive and negative.

Characterization (cont'd)

A more general form:

$$\{x : S \mid p(x) \bullet t(x)\}$$

Examples:

- $\{x : \mathbb{Z} \mid 1 \leq x \leq 100 \bullet 2 * x\}$
  All the even numbers from 2 to 200, i.e., $\{2, 4, 6, \ldots, 198, 200\}$.
- $\{x : \mathbb{Z} \mid x \bmod 2 = 0 \bullet x + 1\}$
  All the odd numbers, positive and negative.

Characterization (cont'd)

Yet another form:

$\{x : S \bullet t(x)\}$

Same as:

$\{x : S \mid true \bullet t(x)\}$

- $\{x : 1 \mathrel{..} 5 \bullet x * x\}$
  The set $\{1, 4, 9, 16, 25\}$.

Natural Numbers

- $\mathbb{N}$: natural numbers
- $\mathbb{N}_1$: positive natural numbers

$$\mathbb{N} == \{x : \mathbb{Z} \mid x \geq 0\}$$
$$\mathbb{N}_1 == \{x : \mathbb{Z} \mid x \geq 1\}$$

Syntactical Equivalence

$==$ denotes *syntactical equivalence*

$$n == E$$

where n is a name, and E is an expression.

- n is an abbreviation of E.
- every occurance of n can be replaced by E without altering the meaning

Set Operations

- $\#S$ is the number of elements in set S, or the *cardinality* of S
- $S_1 \cup S_2$ is the *union* of S_1 and S_2, which contains all the elements in either S_1 or S_2 or both.
- $S_1 \cap S_2$ is the *intersection* of S_1 and S_2, which contains all the elements in both S_1 and S_2.
- $S_1 \setminus S_2$ is the *difference* of S_1 and S_2, which contains all the elements in S_1 but not in S_2.

Set Relations

- Membership. $x \in S$ if x is a member of set S. It reads x is in S.
- Non-membership. $x \notin S$ if x is not in S.
- Equality. $S_1 = S_2$ if S_1 and S_2 contain exactly the same elements.
- Inequality. $S_1 \neq S_2$ if S_1 and S_2 are not equal.
- Subset. $S_1 \subseteq S_2$ if every element in S_1 is also in S_2.
- Proper subset. $S_1 \subset S_2$ if every element in S_1 is also in S_2 and there are elements in S_2 that are not in S_1.

Power Sets

The *power set* of a set S consists of all the subsets of of S.

Let S be the set $\{1, 2\}$.

The power set of S, denoted $\mathbb{P}\,S$, is:

$$\{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$$

$s : STUDENT$
$ss : \mathbb{P}\,STUDENT$

- s is an individual student
- ss is a set of students

Pairs

A *pair* is consists of two elements, and is denoted (x, y).

- x and y may be of different types.
- $(x, y) \neq (y, x)$ unless $x = y$.
- $(x_1, y_1) = (x_2, y_2)$ if and only if $x_1 = x_2$ and $y_1 = y_2$.

Cartesian Product

The Cartesian product of sets X and Y, denoted $X \times Y$, is the set containing all such pairs (x, y) that $x \in X$ and $y \in Y$.

For example, $\{1, 2\} \times \{1, 2, 3\}$ yields

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

More generally

- n-tuple: $(x_1, x_2, \ldots, x_n)$
- $T_1 \times T_2 \times \cdots \times T_n$.

A Class Manager's Assistant

$[Student]$

A given set of students.

$$| \; size : \mathbb{N}$$

The maximum size of a class.

$$
\begin{aligned}
Response ::= \; & success \\
| \; & notenrolled \\
| \; & nocert \\
| \; & cert \\
| \; & alreadyenrolled \\
| \; & alreadytested \\
| \; & noroom
\end{aligned}
$$

Possible responses of the program.

The State

$$
\begin{array}{l}
\textit{Class} \\
\hline
enrolled, tested : \mathbb{P}\, Student \\
\hline
\#enrolled \leq size \\
tested \subseteq enrolled
\end{array}
$$

enrolled is the set of all students enrolled in the class.

tested is the set of those students who have passes the test.

The initial state

$$ClassInit == [Class' \mid enrolled' = \varnothing]$$

Enrolling a student

$$
\begin{array}{l}
\textit{Enrolok} \\
\hline
\Delta \textit{Class} \\
s? : \textit{Student} \\
r! : \textit{Response} \\
\hline
s? \notin \textit{enrolled} \\
\# \textit{enrolled} < \textit{size} \\
\textit{enrolled}' = \textit{enrolled} \cup \{s?\} \\
\textit{tested}' = \textit{tested} \\
r! = \textit{success}
\end{array}
$$

Enroll a new student into the class.

Testing a Student

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{Testok}} \\
\hline
\Delta \textit{Class} \\
s? : \textit{Student} \\
r! : \textit{Response} \\
\hline
s? \in \textit{enrolled} \\
s? \notin \textit{tested} \\
\textit{tested}' = \textit{tested} \cup \{s?\} \\
\textit{enrolled}' = \textit{enrolled} \\
r! = \textit{success} \\
\hline
\end{array}
$$

A student passes the test.

Discharging a Student

$$
\begin{array}{l}
\textit{Leaveok} \\
\hline
\Delta \textit{Class} \\
s? : \textit{Student} \\
r! : \textit{Response} \\
\hline
s? \in \textit{enrolled} \\
\textit{enrolled}' = \textit{enrolled} \setminus \{s?\} \\
((s? \in \textit{tested} \land \textit{tested}' = \textit{tested} \setminus \{s?\} \land r! = \textit{cert}) \\
\lor (s? \notin \textit{tested} \land \textit{tested}' = \textit{tested} \land r! = \textit{nocert}))
\end{array}
$$

A student leaves the class, with or without a certificate.

Enquiries

$$
\begin{array}{l}
\textit{Enquire} \\
\hline
\Xi \textit{Class} \\
s? : \textit{Student} \\
r! : \textit{Response} \\
\hline
((s? \notin \textit{enrolled} \land r! = \textit{notenrolled}) \\
\lor (s? \in \textit{enrolled} \setminus \textit{tested} \land r! = \textit{alreadyenrolled}) \\
\lor (s? \in \textit{tested} \land r! = \textit{alreadytested}))
\end{array}
$$

Inquire the status of a student.

Preconditions

Schema	Input	Precondition
Enrollok	$s?$	$s? \notin enrolled$ $\#enrolled < size$
Testok	$s?$	$s? \in enrolled$ $s? \notin tested$
Leaveok	$s?$	$s? \in enrolled$
Enquire	$s?$	true

Errors

$$
\begin{array}{|l}
\hline
\multicolumn{1}{l}{\textit{AlreadyEnrolled}} \\
\Xi Class \\
s? : Student \\
r! : Response \\
\hline
s? \in enrolled \\
r! = alreadyenrolled \\
\hline
\end{array}
$$

To enroll a student who is already enrolled in the class.

$$
\begin{array}{|l}
\hline
\multicolumn{1}{l}{\textit{NoRoom}} \\
\Xi Class \\
s? : Student \\
r! : Response \\
\hline
\# enrolled = size \\
r! = noroom \\
\hline
\end{array}
$$

To enroll a student when the class is full.

Errors (cont'd)

$AlreadyTested$

$\Xi Class$
$s? : Student$
$r! : Response$

$s? \in tested$
$r! = alreadytested$

To test a student who has already passed the test.

$NotEnrolled$

$\Xi Class$
$s? : Student$
$r! : Response$

$s? \notin enrolled$
$r! = notenrolled$

Actions on a student who is not enrolled in the class.

The User Interface

$$Enrol == Enrolok \lor NoRoom \lor AlreadyEnrolled$$
$$Test == Testok \lor NotEnrolled \lor AlreadyTested$$
$$Leave == Leaveok \lor NotEnrolled$$

The complete operations.