Basic Generic Definitions

Example 1 : Generic Identity Function

The identityGB function works for any type:

$$[X]$$
$$identityGB : X \longrightarrow X$$
$$\forall x : X \bullet identityGB(x) = x$$

This defines identityGB polymorphically. It can be used with any type: identityGB applied to a natural number returns that number, identityGB applied to a set returns that set, etc.

Example 2 : Generic Constant

A generic empty set:

$$[X]$$
$$empty : \mathbb{P}\, X$$
$$empty = \{\}$$

The empty set can be instantiated for any type X.

Example 3 : Generic Pair Functions

Extract the first element of a pair:

$$[X, Y]$$
$$fst : X \times Y \longrightarrow X$$
$$\forall x : X \bullet \forall y : Y \bullet fst(x, y) = x$$

Extract the second element:

$$[X, Y]$$
$$snd : X \times Y \longrightarrow Y$$
$$\forall x : X \bullet \forall y : Y \bullet snd(x, y) = y$$

These work for pairs of any types.

Example 4 : Generic Singleton Set

Create a set containing a single element:

$$[X]$$
$$singleton : X \longrightarrow \mathbb{P}\, X$$
$$\forall x : X \bullet singleton(x) = \{x\}$$

Given any value, singleton produces a set containing just that value.

Example 5 : Generic Sequence Operations

The head of a sequence (generic version):

$$[X]$$
$$headOf : \text{seq } X \longrightarrow X$$
$$\forall s : \text{seq } X \mid \#s > 0 \bullet headOf(s) = s(1)$$

This is defined only for non-empty sequences.

Example 6 : Generic Set Membership (Conceptual)

Set membership can be tested using the built-in 'in' operator in Z notation. In other contexts, you might define a membership test function, but Z notation provides this as a primitive predicate: x in S is true exactly when x is an element of S.

Example 7 : Generic Optional Type (Conceptual)

Optional values can be modeled with free types in Z notation. For a specific type, you would *define* : *Option* ::= none | some⟨T⟩. Generic optional types require more complex setup beyond basic gendef blocks, as free types in Z are typically defined for specific types rather than being polymorphic.

Example 8 : Generic Swap Function

Swap the components of a pair:

$$[X, Y]$$
$$swap : X \times Y \longrightarrow Y \times X$$
$$\forall x : X \bullet \forall y : Y \bullet swap(x, y) = (y, x)$$

Converts (x, y) to (y, x).

Example 9 : Generic Cardinality

Size of a set (for finite sets):

$$[X]$$
$$size : \mathbb{P}\, X \longrightarrow \mathbb{N}$$
$$\forall S : \mathbb{P}\, X \bullet size(S) = \#S$$

Returns the number of elements in a finite set.

Example 10 : Generic List Construction

Construct a singleton sequence:

$$[X]$$
$$single : X \longrightarrow \text{seq } X$$
$$\forall x : X \bullet single(x) = \langle x \rangle$$

Wraps a value in a single-element sequence.

Example 11 : Using Generic Definitions

Once defined, generic functions can be instantiated for specific types:

$$id\_nat : \mathbb{N} \longrightarrow \mathbb{N}$$
$$id\_set : \mathbb{P}\,\mathbb{N} \longrightarrow \mathbb{P}\,\mathbb{N}$$
$$id\_nat = identityGB[\mathbb{N}]$$
$$id\_set = identityGB[\mathbb{P}\,\mathbb{N}]$$

Here identityGB[N] is the identityGB function instantiated for natural numbers, and identityGB[P N] is instantiated for sets of natural numbers.

Example 12 : Generic Constants

Define a generic empty sequence:

$$[X]$$
$$emptySeq : \text{seq } X$$
$$emptySeq = \langle\rangle$$

This can be instantiated as emptySeq[N] for an empty sequence of naturals, emptySeq[Person] for an empty sequence of persons, etc.