Type	Operation	Description
OclAny	=	True if self and the argument are the same
	<>	True if self and the argument are not the same
	oclIsNew()	True if self was created during the operation
	oclIsUndefined()	True if self is undefined self as of the given type, type
	oclAsType(**t**)	True if self is an instance of the given type, type
	oclIsTypeOf(**t**)	True if self conforms to the given type, type
	oclIsKindOf(**t**)	True if self is in the given state, state
	oclType()	Evaluates to the type of which self is an instance.
	oclIsInvalid()	Evaluates to true if the self is equal to OclInvalid.
	oclAsSet()	Converse object to one element collection
Classifier	allInstances()	Set of all instances of the type T
Boolean	or, and, xor, not, implies, toString()	
Integer	+, -, *, /, mod(), div(), abs(), max(), min(), toString() **[Integer is a subtype of Real, so we can use '<']**	
Real	<, >, <=, >=, +, -, *, /, abs(), max(), min(), round(), floor(), toString()	
String	+**s**,concat(**s**), size(), toLowerCase(), toUpperCase(), substring(**i**,**i**), toInteger(), toReal(), indexOf(**s**), equalsIgnoreCase(**s**), at(**i**), characters():Sequence(string), toBoolean()	
Collection	count(**o**)	Number of occurrences of o in the collection
	excludes(**o**)	True if o is not an element of the collection
	excludesAll(**c**)	True if all elements of c are not present in the collection
	includes(**o**)	True if o is an element of the collection
	includesAll(**c**)	True if all the elements of c contained in the collection
	isEmpty() / notEmpty()	True if collection contain no element/ has elements
	including(o) / excluding(o)	Returns new collection with o element added/removed
	size()	Number of elements in the collection
	sum()	Addition of all elements in the collection
	min()/max()	The element with the min/max value of all elements
	product(**c**)	The cartesian product operation of self and c2.
	selectByKind(**t**)/ selectByType(**t**)	Returns the sub-Collection whose type is type but which are/aren't a subtype of type. The returned Collection element type T is the type specified as type.
	flatten()	Recursively flatten collections inside
	asBag()/ asOrderedSet()/ asSet()/ asSequence()	Change collection type

Operation	Set	OrderedSet	Bag	Sequence
=	O	O	O	O
<>	O	O	O	O
-	O	O		
append(o)		O		O
at(i)*		O		O
first()		O		O
indexOf(o)		O		O
insertAt(i, o)		O		O
intersection(c)	O		O	
last()		O		O
reverse()		O		O
prepend(o)		O		O
subOrderedSet(l, u)		O		
subsequence(l, u)				O
symmetricDifference(c)	O			
union(c)	O	O	O	O

*OCL uses 1-based index for ordered sets and sequences.

Operation	Description
any(expr)	Returns any element for which *expr* is true
collect(expr)	Returns a collection that results from evaluating *expr* for each element of *self*
collectNested(expr)	Returns a collection of collections that result from evaluating *expr* for each element of *self*
exists(expr)	Has at least one element for which *expr* is true?
forAll(expr)	Is *expr* true for all elements?
isUnique(expr)	Does *expr* has unique value for all elements?
iterate(x: S; y: T| expr)	Iterates over all elements
one(expr)	Has only one element for which *expr* is true?
reject(expr)	Returns a collection containing all elements for which *expr* is false
select(expr)	Returns a collection containing all elements for which *expr* is true
sortedBy(expr)	Returns a collection containing all elements ordered by *expr*