

Kozmos Operators, Constants, and Data Structures

The following chart lists the built-in operators, constants, and data structures of the **Kozmos** programming language:

Category	Purpose	Operator
Arithmetic	addition	<code>a + b</code>
	subtraction	<code>a - b</code>
	multiplication	<code>a * b</code>
	division	<code>a / b</code>
		<code>a div b</code>
	modulus	<code>a mod b</code>
	extrema	<code>a min b</code> -or- <code>min(a, b, c, ...)</code>
		<code>a max b</code> -or- <code>max(a, b, c, ...)</code>
Relational	equals	<code>a = b</code>
	not equals	<code>a <> b</code>
	less than	<code>a < b</code>
	greater than	<code>a > b</code>
	at least	<code>a >= b</code>
	at most	<code>a <= b</code>
Logical	conjunction	<code>a and b</code>
	disjunction	<code>a or b</code>
	equivalence	<code>a eqv b</code>
	implication	<code>a imp b</code>
	negation	<code>not a</code>
Bitwise	and	<code>a & b</code>
	inclusive or	<code>a \ b</code>
	exclusive or	<code>a ! b</code>
	not	<code>~a</code>
	left shift	<code>a << b</code>
	right shift	<code>a >> b</code>

The following are the assignment operators:

Category	Syntax
Simple	<code>a := b</code>
Compound	<code>A[i] :<operator>= v</code> <code>{l:_} :<operator>= v</code>

Kozmos provides the following built-in constants:

Category	Constant
Arithmetic	<code>+Inf</code>
	<code>-Inf</code>
	<code>NaN</code>
Boolean	<code>True</code>
	<code>False</code>
Reference	<code>Nil</code>

Kozmos has the following built-in data structures:

Category	Declaration
Array	<code>Array<T></code>
Linked list	<code>List<T></code>
Hash Map	<code>Map<K -> V></code>
Hash Set	<code>Set<T></code>
Priority Queue	<code>Queue<T></code>
Minima Heap	<code>Heap<T></code>

NOTE: **Kozmos** does not support general-purpose *generics*. Instead, it has *trait compliance* where the type parameter has to be with one of the built-in root traits such as `Eq`, `Ord`, `Sync`, etc., or a user designed trait that implements those root traits.

In a data structure declaration, the type parameter has to be a known trait, e.g. `<Ord>`; a descendant of a trait, e.g. `<T: Ord>`; or a descendant of multiple traits (i.e. a *union* of them), e.g. `<T: Ord | Sync>`.

The following operations are available for built-in data structures

Category	Subcategory	Operator
Catenation	Two arrays (Strings incl.)	<code>a ++ b</code>
	Array with single item	<code>a ++ [b]</code>

Category	Subcategory	Operator
Slicing		<code>a[i .. j]</code>
Range		<code>[a .. b]</code>
Indexing		<code>a[i]</code>
List access	Head	<code>{l: _}</code>
	Tail	<code>{_: l}</code>
	Append	<code>{a} ++ b</code>
	Prepend	<code>b ++ {a}</code>

Kozmos does **not** support operator *overloading*. Instead, it supports operator *extension*: operators can be extended for a certain type (i.e. a `record`, a `trait`, or a `class`) only on the condition that the expression reduces to the built-in use.