

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	$a + b$
	subtraction	$a - b$
	multiplication	$a * b$
	division	a / b
		$a \text{ div } b$
	modulus	$a \text{ mod } b$
	extrema	$a \text{ min } b$ -or- $\text{min}(a, b, c, \dots)$ $a \text{ max } b$ -or- $\text{max}(a, b, c, \dots)$
Relational	equals	$a = b$
	not equals	$a \lt\!> b$
	less than	$a < b$
	greater than	$a > b$
	at least	$a \geq b$
	at most	$a \leq b$
Logical	conjunction	$a \text{ and } b$
	disjunction	$a \text{ or } b$
	equivalence	$a \text{ eqv } b$
	implication	$a \text{ imp } b$
	negation	$\text{not } a$
Bitwise	and	$a \& b$
	inclusive or	$a \setminus b$
	exclusive or	$a ! b$
	not	$\sim a$
	left shift	$a \ll b$
	right shift	$a \gg b$

The following are the assignment operators:

Category	Syntax
Simple	$a := b$

Category	Syntax
Compound	A[i] :<operator>= v
	{l:_} :<operator>= v

Kozmos provides the following built-in constants:

Category	Constant
Arithmetic	+Inf
	-Inf
	NaN
Boolean	True
	False
Reference	Nil

Kozmos has the following built-in data structures:

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

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.)	a ++ b
	Array with single item	a ++ [b]
Slicing		a[i .. j]
Range		[a .. b]
Indexing		a[i]
List access	Head	{l:_}

Category	Subcategory	Operator
	Tail	{_:l}
	Append	{a} ++ b
	Prepend	b ++ {a}

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 one of the built-in uses.