Amtiri

AMTIRI SCRIPT

Collections

Generated on November 25, 2026

Collections

Append

`arrayAdd` · `arrayAdd(array: [], values: any...)`

Gives a new list with values appended to the end, leaving the original list untouched.

Parameters

Name	Type	Description
`array`	`[]`	The list to append to.
`values`	`any` (1 or more)	A value to append. Add as many as needed.

Returns `[]`: A new list holding the original items followed by the values.

Examples

```
arrayAdd([1, 2], 3, 4) # [1, 2, 3, 4]
[TAG].arrayAdd("P-102") # ["P-101", "P-102"]
```

Average

`arrayAverage` · `arrayAverage(array: [])`

The arithmetic mean of a list's items.

Parameters

Name	Type	Description
`array`	`[]`	The list of numbers to average.

Returns `double`: The mean.

Examples

```
arrayAverage([40.0, 42.5, 45.0]) # 42.5
arrayAverage([1, 2]) # 1.5
```

Concatenate

`arrayConcat` · `arrayConcat(array: [], values: [])`

Joins two lists into a new one, leaving both originals untouched.

Parameters

Name	Type	Description
`array`	`[]`	The list whose items come first.
`values`	`[]`	The list whose items come after.

Returns `[]`: A new list holding both.

Examples

```
arrayConcat([1, 2], [3]) # [1, 2, 3]
```

Count

`count` · `count(array: [])`

How many items a list holds. Written as a method on the list, array.count().

Parameters

Name	Type	Description
`array`	`[]`	The list to count.

Returns `int`: The item count.

Examples

```
REGISTERS.count() # 4
count([]) # 0
```

Filter

`arrayFilter` · `arrayFilter(array: [], paramName: name, filterFunction: expression)`

Keeps only the items a condition accepts. Each item is bound to paramName in turn and the condition is evaluated against it.

Parameters

Name	Type	Description
`array`	`[]`	The list to filter.
`paramName`	`name`	The name each item is bound to while the condition runs, written bare rather than quoted.
`filterFunction`	`expression`	The condition, evaluated once per item. Refer to the item by paramName.

Returns `[]`: A new list holding the accepted items.

Examples

```
arrayFilter(REGISTERS, r, r > 0) # [16256, 65535, 1]
REGISTERS.arrayFilter(r, r > 60000) # [65535]
```

For each

`forEach` · `forEach(array: [], paramName: name, converter: expression)`

Converts every item of a list. Each item is bound to paramName in turn and the converter's result takes its place.

Parameters

Name	Type	Description
`array`	`[]`	The list to convert.
`paramName`	`name`	The name each item is bound to while the converter runs, written bare rather than quoted.
`converter`	`expression`	The expression producing the replacement for each item. Refer to the item by paramName.

Returns `[]`: A new list holding the converted items.

Examples

```
forEach([1, 2, 3], x, x * 10) # [10.0, 20.0, 30.0]
REGISTERS.forEach(r, bit(r, 0)) # [false, false, true, true]
```

List maximum

`arrayMax` · `arrayMax(array: [])`

The largest item of a list, or null when the list is empty.

Parameters

Name	Type	Description
`array`	`[]`	The list to scan.

Returns `any`: The largest item.

Examples

```
arrayMax(REGISTERS) # 65535
arrayMax([4.5, 2, 7]) # 7
arrayMax([]) # null
```

List minimum

`arrayMin` · `arrayMin(array: [])`

The smallest item of a list, or null when the list is empty.

Parameters

Name	Type	Description
`array`	`[]`	The list to scan.

Returns `any`: The smallest item.

Examples

```
arrayMin([4.5, 2, 7]) # 2
```

Number range

`serie` · `serie(from: int, size: int)`

Builds a list of consecutive whole numbers.

Parameters

Name	Type	Description
`from`	`int`	The first number in the list.
`size`	`int`	How many numbers to produce.

Returns `[]`: The list of numbers.

Examples

```
serie(1, 5) # [1, 2, 3, 4, 5]
serie(0, 3) # [0, 1, 2]
```

Sum

`arraySum` · `arraySum(array: [])`

Adds up every item of a list.

Parameters

Name	Type	Description
`array`	`[]`	The list of numbers to add up.

Returns `double`: The total.

Examples

```
arraySum([1.5, 2.5, 3]) # 7.0
REGISTERS.arraySum() # 81792.0
```

This page describes Amtiri Script 5.5.0 as shipped with Data Orchester engine 6.22.0 and Site 1.27.0.