

**Sort**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> Sort		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 26, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>Sort</b>	<b>1</b>
1.1	Sort V1.00 . . . . .	1
1.2	sortdown . . . . .	1
1.3	sortup . . . . .	2

# Chapter 1

## Sort

### 1.1 Sort V1.00

PureBasic Sort library V1.00

This library allows you to sort numerical arrays in ascending or descending order. The routine used is based on the QuickSort algorithm, rewritten in highly optimized assembler. So it's probably one of the fastest sorting routines ever available for the 680x0 processor family.

Commands summary:

SortDown  
SortUp

Example:

Sort demo

### 1.2 sortdown

SYNTAX

SortDown(array(), start.w, end.w)

STATEMENT

Sort the specfied array in descending order (high numbers first).

array(): name of the array to sort. The array must be an array of byte, word or long. String arrays aren't supported.

start.w: index for the start of the sort. You can sort only a part of the array if you need to.

end.w : index for the end of the sort.

NOTE: start.w and end.w must be valid values !

---

## 1.3 sortup

### SYNTAX

```
SortUp(array(), start.w, end.w)
```

### STATEMENT

Sort the array in ascending order (low numbers first).

`array()`: name of the array to sort. The array must be an array of byte, word or long. Strings arrays aren't supported.

`start.w`: index for the start of the sort. You can sort only a part of the array if you need to.

`end.w` : index for the end of the sort.

NOTE: `start.w` and `end.w` must be valid values !