

**Merge Sort Algorithm – part 2**

The *merge sort* algorithm is a recursive algorithm. As with all recursive algorithms, the problem is solved by breaking it down into a similar, smaller problem, then combining the solutions of those smaller problems into the solution for the larger problem. We already looked at an algorithm, *merge*, that merges two sorted arrays into a larger sorted array. This is used for the larger merge sort algorithm, the steps for which are:

- Break the list into two smaller lists (usually break it approximately in half)
- Call *merge\_sort* to sort each of the smaller lists.
- Call *merge* to merge the two sorted lists into the final sorted list.

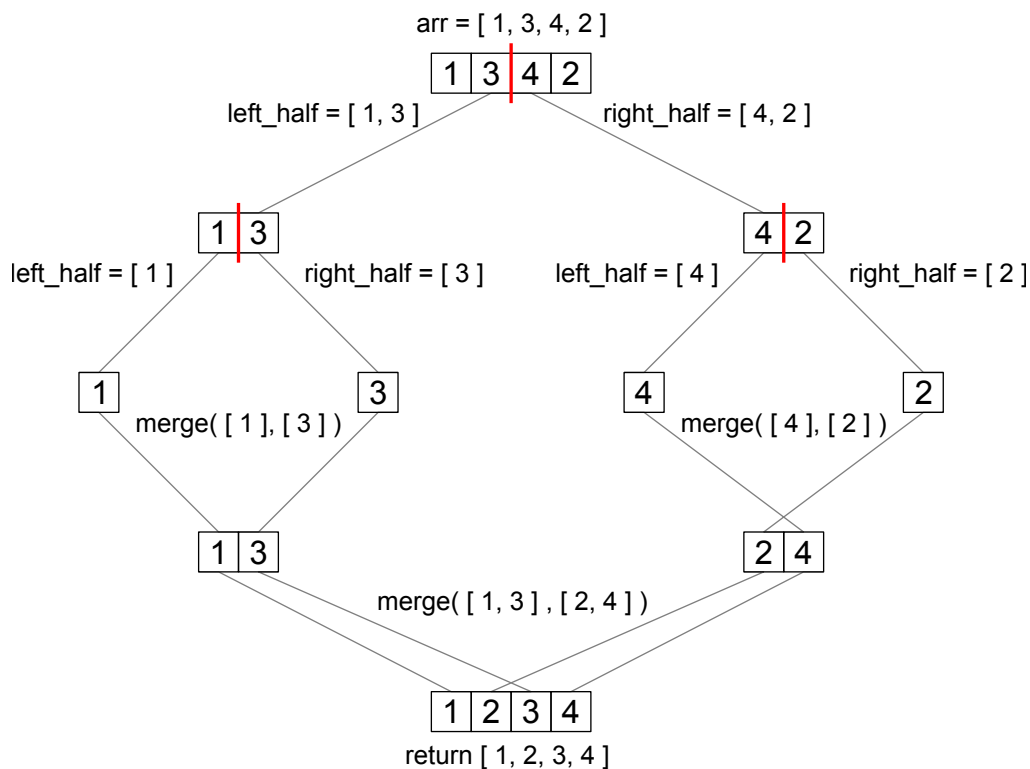
The base case (when to stop breaking down the problem) is when there is only a single element in the list – because a list with a single item is necessarily sorted. Examine the given code for *merge\_sort*.

```

1 def merge_sort(arr):
2     """Sort a list using the merge sort algorithm."""
3     # Base case: a list of 0 or 1 elements is already sorted
4     if len(arr) <= 1:
5         return arr
6
7     # Divide the list into two halves
8     mid = len(arr) // 2
9     left_half = arr[:mid]
10    right_half = arr[mid:]
11
12    # Recursively sort both halves
13    left_sorted = merge_sort(left_half)
14    right_sorted = merge_sort(right_half)
15
16    # Merge the sorted halves
17    return merge(left_sorted, right_sorted)

```

Examine the diagram showing the merge sort algorithm operating on an example list with four elements:



## Merge Sort Algorithm – part 2

1. Sort the lists using the merge sort algorithm.

a)

0	1	2	3
7	4	2	3

--	--	--	--

--	--	--	--

--	--	--	--

--	--	--	--

b)

0	1	2	3	4	5	6	7
9	1	7	6	3	5	2	8

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

c)

0	1	2	3	4	5	6	7	8	9	10
8	3	3	1	9	7	5	3	4	0	6

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--