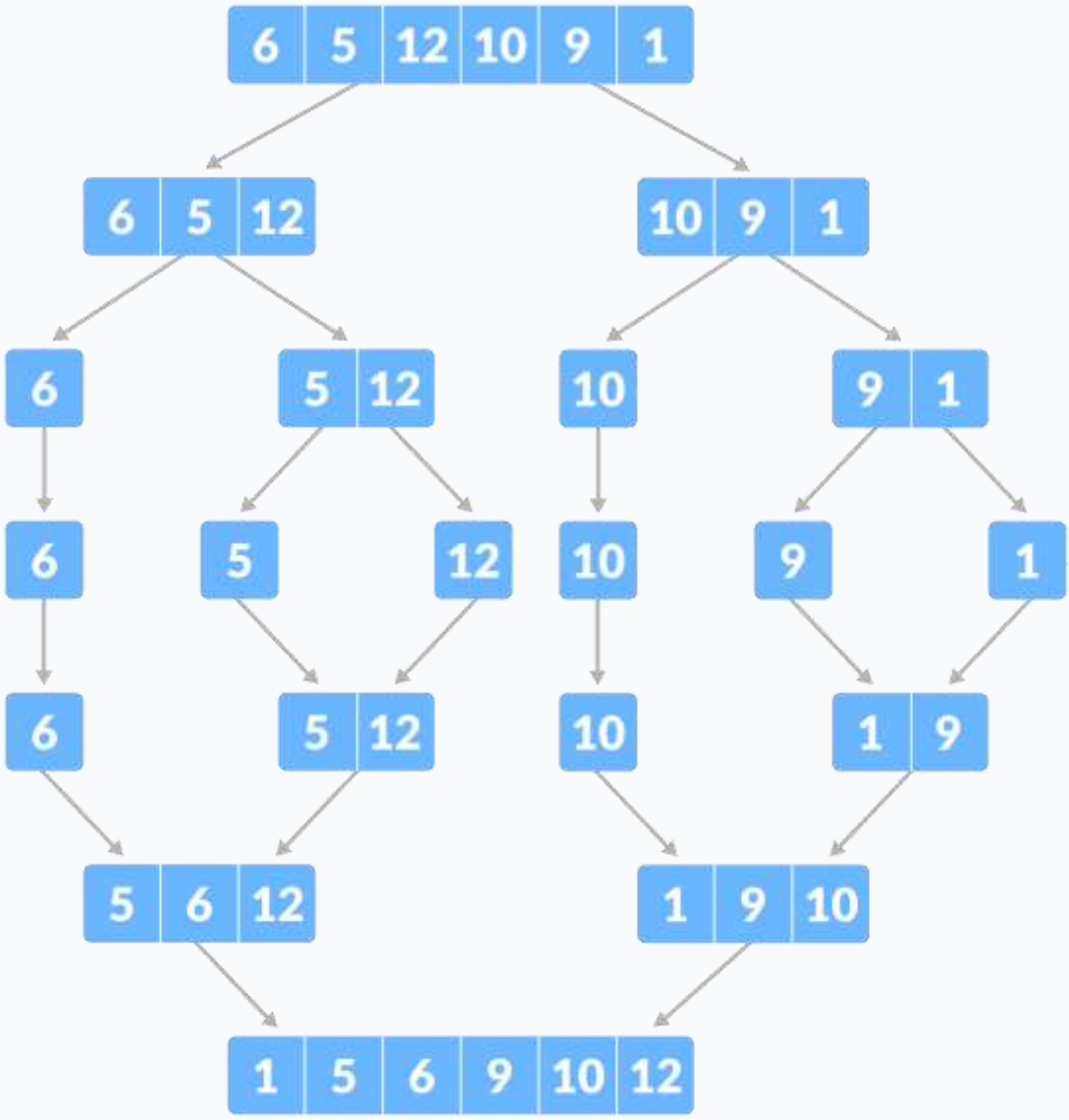
Merge Sort Algorithm

Merge Sort is one of the most popular sorting algorithms that is based on the principle of Divide and Conquer Algorithm.

Here, a problem is divided into multiple sub-problems. Each sub-problem is solved individually. Finally, sub-problems are combined to form the final solution.



Merge Sort example

Divide and Conquer Strategy

Using the **Divide and Conquer** technique, we divide a problem into subproblems. When the solution to each subproblem is ready, we 'combine' the results from the subproblems to solve the main problem.
Suppose we had to sort an array arr. A subproblem would be to sort a sub-section of this array starting at index start and ending at index end, denoted as arr[start..end]

Divide

If mid is the half-way point between start and end, then we can split the subarray arr[start..end] into two arrays arr[start..mid] and arr[mid+1,end]

Conquer

In the conquer step, we try to sort both the subarrays arr[start..mid] and arr[mid+1,end]. If we haven't yet reached the base case, we again divide both these subarrays and try to sort them.

Combine

When the conquer step reaches the base step and we get two sorted subarrays arr[start..mid] and arr[mid+1,end] for array arr[start..end], we combine the results by creating a sorted array arr[start..end] from two sorted subarrays arr[start..mid] and arr[mid+1,end].

MergeSort Algorithm

The MergeSort function repeatedly divides the array into two halves until we reach a stage where we try to perform MergeSort on a subarray of size 1 i.e. start==end.
After that, the merge function comes into play and combines the sorted arrays into larger arrays until the whole array is merged.

```
MergeSort(A, start, end):

    if start > end

        return

    mid = (start+end)/2

    mergeSort(A, start,mid)

    mergeSort(A, mid+1,end)

    merge(A, start,mid,end)
```

To sort an entire array, we need to call MergeSort(arr, 0, length(arr)-1).
As shown in the image below, the merge sort algorithm recursively divides the array into halves until we reach the base case of array with 1 element. After that, the merge function picks up the sorted sub-arrays and merges them to gradually sort the entire array.

Merge Sort

Merge Sort is a divide-and-conquer algorithm. In each iteration, merge sort divides the input array into two equal subarrays, calls itself recursively for the two subarrays, and finally merges the two sorted halves.

Time Complexity:

- *Worst Case:* O(n*logn)
- *Average Case:* O(n*logn)
- *Best case:* O(n*logn)

Space Complexity: O(n)

Use Cases

- Merge sort is quite fast in the case of linked lists.
- It is widely used for external sorting where random access can be quite expensive compared to sequential access.