

[Home](#)

[DAA](#)

[DS](#)

[DBMS](#)

[Aptitude](#)

[Selenium](#)

[Kotlin](#)

[C#](#)

[HTML](#)

[CSS](#)

[JavaScript](#)

Merge Sort

Merge sort is yet another sorting algorithm that falls under the category of **Divide and Conquer** technique. It is one of the best sorting techniques that successfully build a recursive algorithm.

Divide and Conquer Strategy

In this technique, we segment a problem into two halves and solve them individually. After finding the solution of each half, we merge them back to represent the solution of the main problem.

Suppose we have an array **A**, such that our main concern will be to sort the subsection, which starts at index **p** and ends at index **r**, represented by **A[p..r]**.

Divide

If assumed **q** to be the central point somewhere in between **p** and **r**, then we will fragment the subarray **A[p..r]** into two arrays **A[p..q]** and **A[q+1, r]**.

Conquer

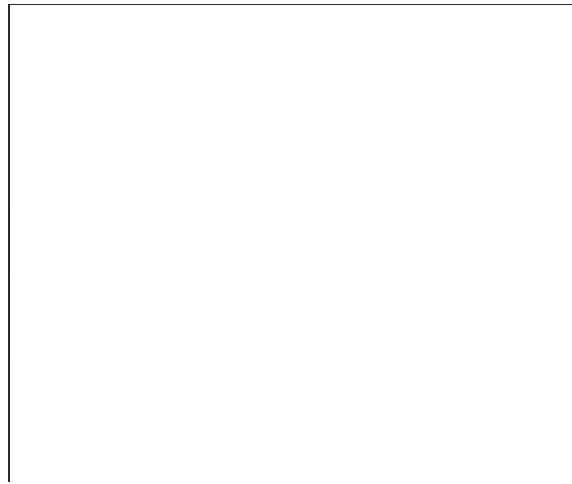
After splitting the arrays into two halves, the next step is to conquer. In this step, we individually sort both of the subarrays **A[p..q]** and **A[q+1, r]**. In case if we did not reach the base situation, then we again follow the same procedure, i.e., we further segment these subarrays followed by sorting them separately.

Combine

As when the base step is acquired by the conquer step, we successfully get our sorted subarrays **A[p..q]** and **A[q+1, r]**, after which we merge them back to form a new sorted array **[p..r]**.

Merge Sort algorithm

The MergeSort function keeps on splitting an array into two halves until a condition is met where we try to perform MergeSort on a subarray of size 1, i.e., **p == r**.



And then, it combines the individually sorted subarrays into larger arrays until the whole array is merged.

ALGORITHM-MERGE SORT

1. If $p < r$
2. Then $q \rightarrow (p + r) / 2$
3. MERGE-SORT (A, p, q)
4. MERGE-SORT (A, q+1, r)
5. MERGE (A, p, q, r)

Here we called **MergeSort(A, 0, length(A)-1)** to sort the complete array.

As you can see in the image given below, the merge sort algorithm recursively divides the array into halves until the base condition is met, where we are left with only 1 element in the array. And then, the merge function picks up the sorted sub-arrays and merge them back to sort the entire array.

The following figure illustrates the dividing (splitting) procedure.

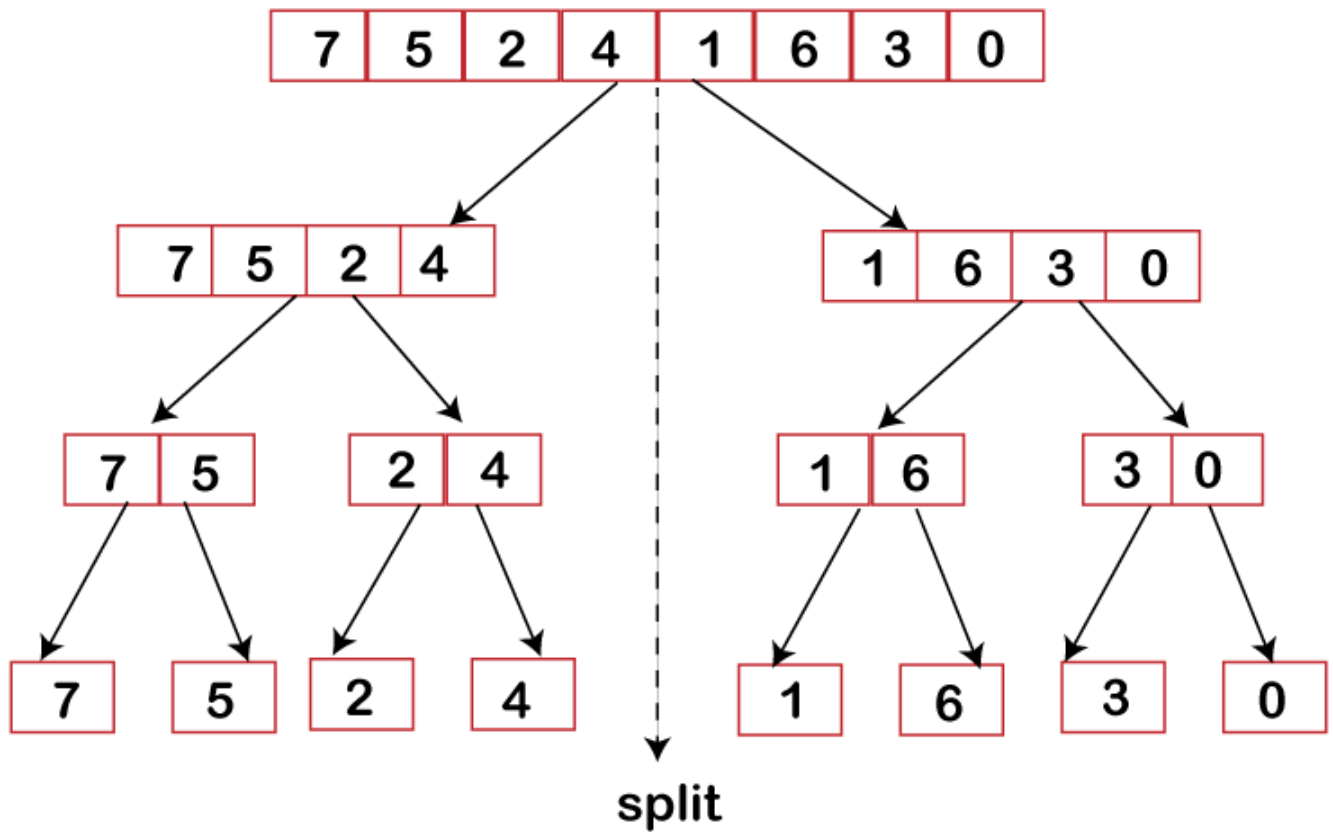


Figure 1: Merge Sort Divide Phase

FUNCTIONS: MERGE (A, p, q, r)

1. $n_1 = q - p + 1$
2. $n_2 = r - q$
3. create arrays $L[1 \dots n_1 + 1]$ and $R[1 \dots n_2 + 1]$
4. for $i \leftarrow 1$ to n_1
5. do $L[i] \leftarrow A[p + i - 1]$
6. for $j \leftarrow 1$ to n_2
7. do $R[j] \leftarrow A[q + j]$
8. $L[n_1 + 1] \leftarrow \infty$
9. $R[n_2 + 1] \leftarrow \infty$
10. $i \leftarrow 1$
11. $j \leftarrow 1$
12. For $k \leftarrow p$ to r
13. Do if $L[i] \leq R[j]$
14. then $A[k] \leftarrow L[i]$
15. $i \leftarrow i + 1$
16. else $A[k] \leftarrow R[j]$

17. $j \leftarrow j+1$

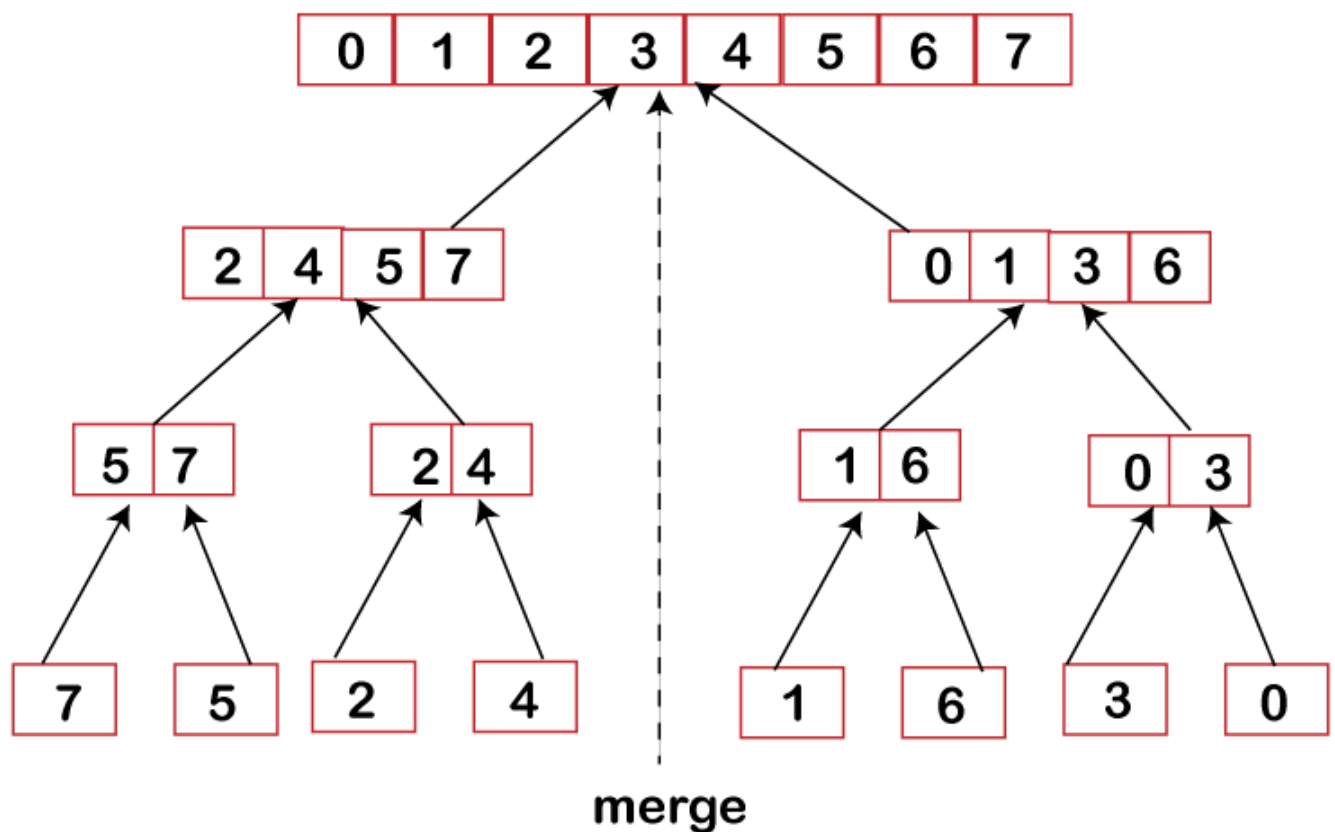


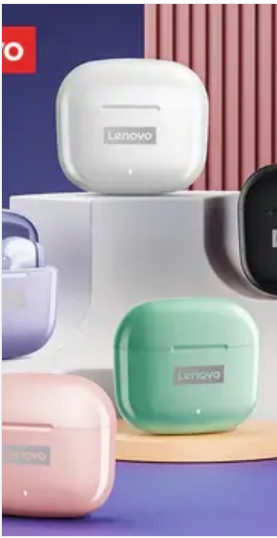
Figure 2: Merge Sort Combine Phase

The merge step of Merge Sort

Mainly the recursive algorithm depends on a base case as well as its ability to merge back the results derived from the base cases. Merge sort is no different algorithm, just the fact here the **merge** step possesses more importance.

To any given problem, the merge step is one such solution that combines the two individually sorted lists(arrays) to build one large sorted list(array).

The merge sort algorithm upholds three pointers, i.e., one for both of the two arrays and the other one to preserve the final sorted array's current index.



Lenovo LP40 Pro TWS

Lenovo LP40 Pro TWS Earphones Wireless Bluetooth 5.1
AilExpress

Did you reach the end of the array?

No:

Firstly, start with comparing the current elements of both the arrays.

Next, copy the smaller element into the sorted array.

Lastly, move the pointer of the element containing a smaller element.

Yes:

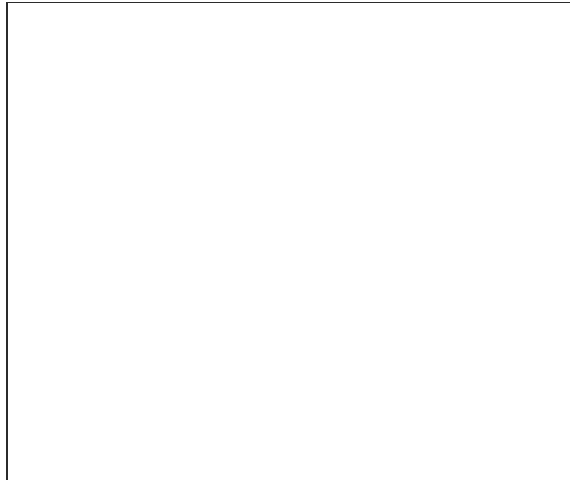
Simply copy the rest of the elements of the non-empty array

Merge() Function Explained Step-By-Step

Consider the following example of an unsorted array, which we are going to sort with the help of the Merge Sort algorithm.

A= (36,25,40,2,7,80,15)

Step1: The merge sort algorithm iteratively divides an array into equal halves until we achieve an atomic value. In case if there are an odd number of elements in an array, then one of the halves will have more elements than the other half.



Step2: After dividing an array into two subarrays, we will notice that it did not hamper the order of elements as they were in the original array. After now, we will further divide these two arrays into other halves.

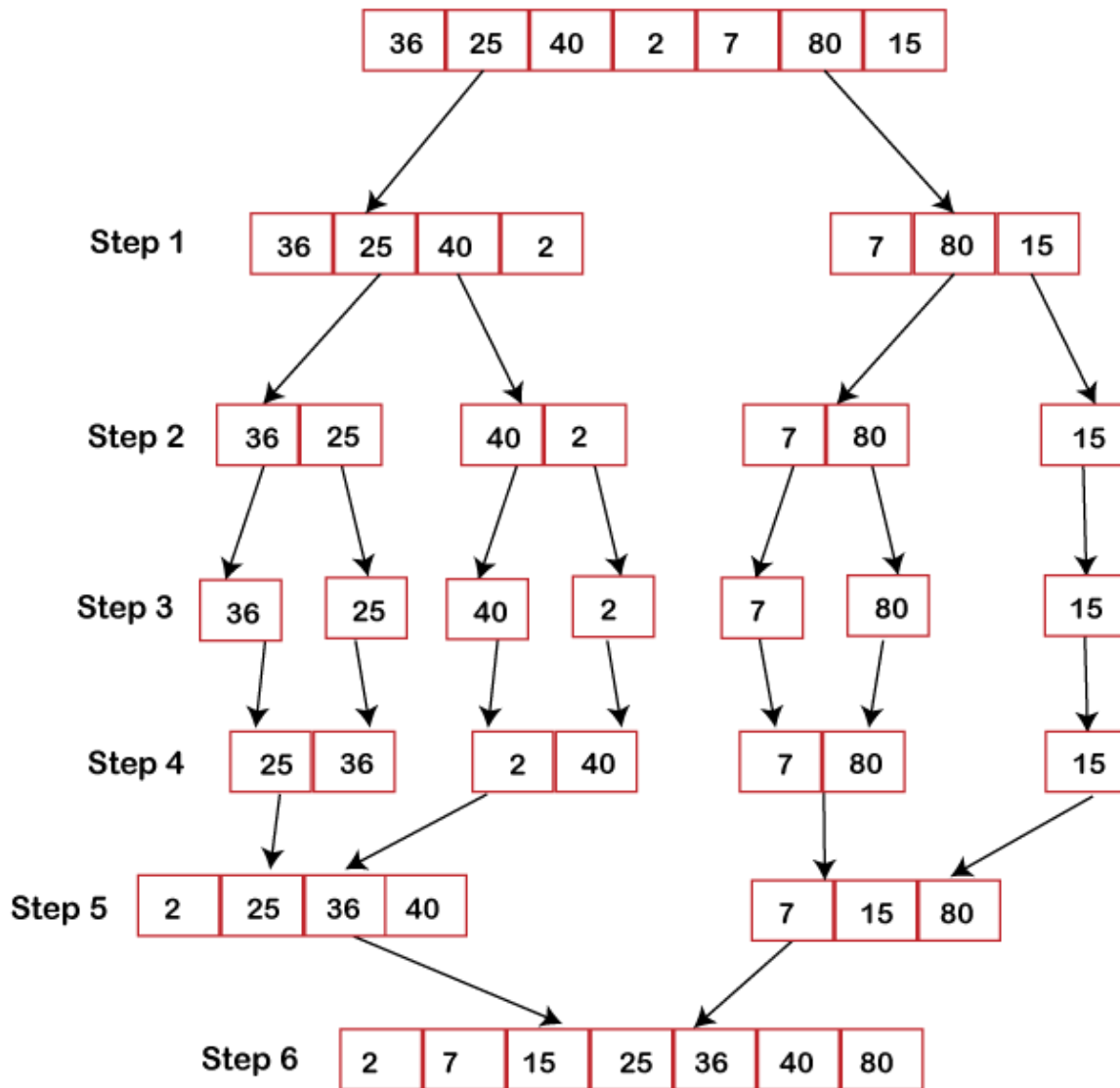
Step3: Again, we will divide these arrays until we achieve an atomic value, i.e., a value that cannot be further divided.

Step4: Next, we will merge them back in the same way as they were broken down.



Step5: For each list, we will first compare the element and then combine them to form a new sorted list.

Step6: In the next iteration, we will compare the lists of two data values and merge them back into a list of found data values, all placed in a sorted manner.



Hence the array is sorted.

Analysis of Merge Sort:

Let $T(n)$ be the total time taken by the Merge Sort algorithm.

- Sorting two halves will take at the most $2T \frac{n}{2}$ time.
- When we merge the sorted lists, we come up with a total $n-1$ comparison because the last element which is left will need to be copied down in the combined list, and there will be no comparison.

Thus, the relational formula will be

$$T(n) = 2T\left(\frac{n}{2}\right) + n - 1$$

But we ignore '-1' because the element will take some time to be copied in merge lists.

So $T(n) = 2T\left(\frac{n}{2}\right) + n$...equation 1

Note: Stopping Condition $T(1) = 0$ because at last, there will be only 1 element left that need to be copied, and there will be no comparison.

Putting $n = \frac{n}{2}$ in place of n inequation 1

$$T\left(\frac{n}{2}\right) = 2T\left(\frac{n}{2^2}\right) + \frac{n}{2}$$
.....equation2

Put 2 equation in 1 equation

$$T(n) = 2 \left[2T\left(\frac{n}{2^2}\right) + \frac{n}{2} \right] + n$$

$$= 2^2 T\left(\frac{n}{2^2}\right) + \frac{2n}{2} + n$$

$$T(n) = 2^2 T\left(\frac{n}{2^2}\right) + 2n$$
.....equation 3

Putting $n = \frac{n}{2^2}$ in equation 1

$$T\left(\frac{n}{2^2}\right) = 2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}$$
.....equation4

Putting 4 equation in 3 equation

$$T(n) = 2^2 \left[2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2} \right] + 2n$$

$$T(n) = 2^3 T\left(\frac{n}{2^3}\right) + n + 2n$$

$$T(n) = 2^3 T\left(\frac{n}{2^3}\right) + 3n \dots \dots \dots \text{equation 5}$$

From eq 1, eq3, eq 5.....we get

$$T(n) = 2^i T\left(\frac{n}{2^i}\right) + in \dots \dots \dots \text{equation 6}$$

From Stopping Condition:

$$\frac{n}{2^i} = 1 \text{ And } T\left(\frac{n}{2^i}\right) = 0$$

$$n = 2^i$$

Apply log both sides:

$$\log n = \log_2 i$$

$$\log n = i \log 2$$

$$\frac{\log n}{\log 2} = i$$

$$\log_2 n = i$$

From 6 equation

$$T(n) = 2^i T\left(\frac{n}{2^i}\right) + in$$

$$= 2^i \times 0 + \log_2 n \cdot n$$

$$= T(n) = n \cdot \log n$$

Best Case Complexity: The merge sort algorithm has a best-case time complexity of $O(n \log n)$ for the already sorted array.

Average Case Complexity: The average-case time complexity for the merge sort algorithm is $O(n \log n)$, which happens when 2 or more elements are jumbled, i.e., neither in the ascending order nor in the descending order.

Worst Case Complexity: The worst-case time complexity is also $O(n \log n)$, which occurs when we sort the descending order of an array into the ascending order.

Space Complexity: The space complexity of merge sort is $O(n)$.

Merge Sort Applications

The concept of merge sort is applicable in the following areas:

- Inversion count problem
- External sorting
- E-commerce applications

← Prev

Next →

 [For Videos Join Our Youtube Channel: Join Now](#)


Feedback

- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share





Learn Latest Tutorials


 [Splunk tutorial](#)
Splunk


 [SPSS tutorial](#)
SPSS


 [Swagger tutorial](#)
Swagger

 [T-SQL tutorial](#)
Transact-SQL


 [Tumblr tutorial](#)
Tumblr


 [React tutorial](#)
ReactJS

 [Regex tutorial](#)
Regex

 [Reinforcement learning tutorial](#)
Reinforcement Learning

 [R Programming tutorial](#)
R Programming

 [RxJS tutorial](#)
RxJS

 [React Native tutorial](#)
React Native

 [Python Design Patterns](#)
Python Design Patterns



Python Pillow
tutorial

Python Pillow



Python Turtle
tutorial

Python Turtle



Keras tutorial

Keras

Preparation



Aptitude

Aptitude



Logical
Reasoning

Reasoning



Verbal Ability

Verbal Ability



Interview
Questions

Interview Questions



Company
Interview
Questions

Company Questions

Trending Technologies



Artificial
Intelligence

Artificial
Intelligence



AWS Tutorial

AWS



Selenium
tutorial

Selenium



Cloud
Computing

Cloud Computing



Hadoop tutorial

Hadoop



ReactJS
Tutorial

ReactJS



Data Science
Tutorial

Data Science



Angular 7
Tutorial

Angular 7



Blockchain
Tutorial

Blockchain



Git Tutorial

Git



Machine
Learning Tutorial

Machine Learning





DevOps
Tutorial


DevOps


B.Tech / MCA

 DBMS tutorial
DBMS


 Data Structures
tutorial
Data Structures


 DAA tutorial
DAA


 Operating
System
Operating System


 Computer
Network tutorial
Computer Network


 Compiler
Design tutorial
Compiler Design


 Computer
Organization and
Architecture
Computer
Organization

 Discrete
Mathematics
Tutorial
Discrete
Mathematics

 Ethical Hacking
Ethical Hacking


 Computer
Graphics Tutorial
Computer Graphics


 Software
Engineering
Software
Engineering


 html tutorial
Web Technology


 Cyber Security
tutorial
Cyber Security


 Automata
Tutorial
Automata


 C Language
tutorial
C Programming


 C++ tutorial
C++

 Java tutorial
Java

 .Net
Framework
tutorial
.Net

 Python tutorial
Python

 List of
Programs
Programs

 Control
Systems tutorial
Control System

 Data Mining
Tutorial
Data Mining

 Data
Warehouse
Tutorial
Data Warehouse

