

### Introduction

Best used for merging 2 or more sorted data lists - arrays, ArrayLists or LinkedLists.

Steps:

1. Divide the array/arrayList/LinkedList into 2 halves
2. Recursively divide the 2 sublists into 2 halves, and so on till each sublist contains only 1 element
3. Start merging the sublists i.e n sublists take n-merges.

Time Complexity:  $O(n \log n)$

Space Complexity:  $O(n)$  because it uses extra buffer to hold the elements during merge.

### Arrays

```
public class MergeSort {
    static int[] n = {5, 2, 4, 3, 1, 8, 7, 6};
    private static void sort(int[] n) {
        if (n.length <= 1)
            return;
        int[] left = new int[n.length/2];
        int[] right = new int[n.length-left.length];

        /*
        public static void arraycopy (Object src,
        int srcPos, Object dest, int destPos, int length)
        src - Source array (Object type)
        srcPos - Starting position in Source
        array (Integer type)
        dest - Destination array (Object Type)
        destpos - Starting position in destination
        array (Integer type)
        length - Number of elements to be
        copied (Integer type)
        */
        System.arraycopy(n, 0, left, 0, left.length);
        System.arraycopy(n, left.length, right, 0,
        right.length);
        sort(left);
        sort(right);
        merge(n, left, right);
    }
}
```

### Arrays (cont)

```
private static void merge(int[] n,int[] left,
int[] right){
    int index_left=0, index_n=0,index_right=0;
    while(index_left < left.length && index_
right < right.length){
        if(left[index_left] <= right[index_
right]){
            n[index_n++]=left[index_left++];
        }
        else
            n[index_n++] = right[index_rig-
ht++];
    }
    System.arraycopy(left,index_left,n,i-
ndex_n, left.length-index_left);
    System.arraycopy(right,index_right,n,i-
dex_n,right.length-index_right);
}
}
```

### ArrayList

```
public class MergeSortArrayList {
    static ArrayList<Integer> list = new ArrayL-
ist(Arrays.asList(5,3,4,1,2,8,7,9,11,9,12));
    private static void sort(ArrayList<Integer>
list){
        if(list.size()==1) return;
        ArrayList<Integer> left = new ArrayList<>
(list.size()/2);
        ArrayList<Integer> right = new ArrayList<>
(list.size()-left.size());
        int mid = list.size()/2;
        for(int i=0;i<mid;i++)
            left.add(list.get(i));

        for(int i=mid;i< list.size();i++)
            right.add(list.get(i));
        sort(left);
        sort(right);
        merge(list,left,right);
    }
}
```



By [evanescen09](#)

Not published yet.

Last updated 18th August, 2019.

Page 1 of 2.

Sponsored by [ApolloPad.com](#)

Everyone has a novel in them. Finish

Yours!

<https://apollopod.com>

### ArrayList (cont)

```
private static void merge(ArrayList<Integer>
list,ArrayList<Integer> left, ArrayList<Integer>
right){
    int index=0, index_left=0,index_right=0;
    while(index_left < left.size() && index_
right < right.size()){
        if(left.get(index_left).compareTo(ri-
ght.get(index_right)) < 0){
            list.set(index,left.get(index_
left));
            index_left++;
        }
        else{
            list.set(index,right.get(index_ri-
ght));
            index_right++;
        }
        index++;
    }
    for(int i=index_left;i<left.size();i++)
        list.set(index++,left.get(i));
    for(int i=index_right;i<right.size();i++)
        list.set(index++,right.get(i));
    }
}
```

### LinkedList (cont)

```
mid.next=null;
Node left=sort(head);
Node right=sort(midNext);
Node result=merge(left,right);
return result;
}

private static Node merge(Node left, Node
right){
    Node result = null;
    if(left==null)
        return right;
    if(right==null)
        return left;
    if(left.data <= right.data){
        result = left;
        result.next=merge(left.next,right);
    }
    else{
        result=right;
        result.next=merge(left,right.next);
    }
    return result;
}
```

### LinkedList

```
private static Node sort(Node head) {
    if(head==null || head.next==null) return
head;
    Node slow=head,fast=head;
    while(fast.next!=null && fast.next.next!=
=null){
        slow=slow.next;
        fast=fast.next.next;
    }
    Node mid = slow;
    Node midNext = mid.next;
```



By [evanescen09](#)

Not published yet.

Last updated 18th August, 2019.

Page 2 of 2.

Sponsored by [ApolloPad.com](#)

Everyone has a novel in them. Finish Yours!

<https://apollopapad.com>