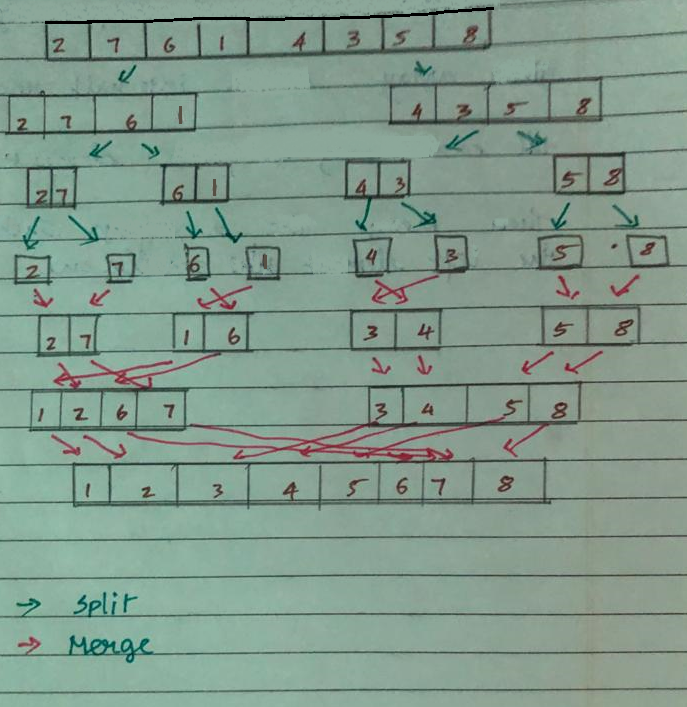
**Merge Sort**

Merge Sort is a sorting algorithm which uses divide and conquer algorithm.

The key idea in this method is we split the array into halves and sort the halves which makes the sorting of entire array faster

Let's consider an example here with an array of 8 elements.



We split the array into two parts with elements [ 2 , 7 , 6 , 1 ] on one side and [ 4 , 3 , 5 , 8 ] on the other side .

we keep splitting it on till we get one element (which is sorted) then we start merging so we get a sorted array

**Implementation (python)**

def merge\_sort(arr):

if len(arr) > 1:

mid = len(arr) // 2

left\_half = arr[:mid]

right\_half = arr[mid:]

left\_half = merge\_sort(left\_half)

right\_half = merge\_sort(right\_half)

arr = []

while len(left\_half) > 0 and len(right\_half) > 0:

if left\_half[0] < right\_half[0]:

arr.append(left\_half.pop(0))

else:

arr.append(right\_half.pop(0))

arr.extend(left\_half)

arr.extend(right\_half)

return arr

# sample test

a = [2,7,6,1,4,3,5,8]

print(merge\_sort(a))

# sample usage with user input

#n = int(input())

#arr = [int(x) for x in input().strip().split()]

**Time Complexity :**

Time complexity of Merge Sort is θ(nLogn) in all 3 cases (worst, average and best) as merge sort always divides the array into two halves and take linear time to merge two halves.

Contribution :

Code – Divakar Rajesh S

Content – Mohamed Niyaz M