binary search was discussed earlier. And it was found that the complexity for an ordered List was log N, and for an unordered List was N. However we can order it in Nlog N time. If we do this, now we can search in log N, so the total complecsity is N\*logN + logN but it is more than common linear search

However if we need to search it more than once method with sorting becomes better and better

So how could we get N\*logN sorting complexity ?

Answer is – divide and conquer algorithm:

-split the problem into several subproblems of the same type

- solve independently

-combine solutions

Merge sort is an application of this method. Conceptually, a merge sort works as follows:

Divide the unsorted list into n sublists, each containing one element (a list of one element is considered sorted).

Repeatedly merge sublists to produce new sorted sublists until there is only one sublist remaining. This will be the sorted list.