MERGE SORT

NAZIA AFSAR CHAND M19CS03

Merge sort is a divide-and-conquer algorithm based on the idea of breaking down a list into several sub-lists until each sublist consists of a single element and merging those sublists in a manner that results into a sorted list.

In [1]:
```
def mergeSort(myList):
    if len(myList) > 1:
        mid = len(myList) // 2
        left = myList[:mid]
        right = myList[mid:]

        # Recursive call on each half
        mergeSort(left)
        mergeSort(right)

        # Two iterators for traversing the two halves
        i = 0
        j = 0

        # Iterator for the main list
        k = 0

        while i < len(left) and j < len(right):
            if left[i] < right[j]:
              # The value from the left half has been used
              myList[k] = left[i]
              # Move the iterator forward
              i += 1
            else:
                myList[k] = right[j]
                j += 1
            # Move to the next slot
            k += 1

        # For all the remaining values
        while i < len(left):
            myList[k] = left[i]
            i += 1
            k += 1

        while j < len(right):
            myList[k]=right[j]
            j += 1
            k += 1

myList = [54,26,93,17,77,31,44,55,20]
mergeSort(myList)
print(myList)
```

```
[17, 20, 26, 31, 44, 54, 55, 77, 93]
```

In real life, we tend to break things up along useful lines. If we're sorting change, we first divide the coins up by denominations, then total up each denomination before adding them together.