Course title : **CSE2001**

Course title : **Data Structures and Algorithms**

Module : **4**

Topic : **3**

Merge Sort

Objectives

This session will give the knowledge about

- Merge sort

Merge Sort

Merge-sort is based on the divide-and-conquer paradigm.

It involves the following three steps:

- Divide the array into two (or more) subarrays
- Sort each subarray (Conquer)
- Merge them into one (in a smart way!)

Merge Sort

Consider the array of numbers: 27 10 12 25 34 16 15 31

Step1: divide it into two parts

27 10 12 25 34 16 15 31

Step2: divide each part into two parts

27 10 12 25 34 16 15 31

Step3: compare each parts swap it, it not in order

10 27 12 25 16 34 15 31

Step4: merge parts

10 27 12 25 16 34 15 31

Step5: merge parts

10 12 25 27 15 16 31 34

Step6: merge parts into one

10 12 15 16 25 27 31 34

Merge Sort

```
public static void mergesort(int a[], int n) {
        int p, L[], R[];
        if (n > 1) {
                p = n / 2;
                L = new int[p];
                R = new int[n - p];
                for (int i = 0; i < p; i++)
                        L[i] = a[i];
                for (int j = p; j < n; j++)
                        R[j - p] = a[j];
                mergesort(L, p);
                mergesort(R, n - p);
                merge(a, L, p, R, n - p);
        }
}
```

Merge Sort

```
public static void merge(int a[], int L[], int l, int R[], int r) {
        int i = 0, j = 0, k = 0;
        while (i < l && j < r) {
                if (L[i] < R[j]) {
                        a[k] = L[i];
                        i++;
                } else {
                        a[k] = R[j];
                        j++;
                }
                k++;
        }
        while (i < l) {
                a[k] = L[i];
                i++;
                k++;
        }
        while (j < r) {
                a[k] = R[j];
                j++;
                k++;
        }
}
```

Complexity Analysis

Algorithm	Time Complexity		
	Best	Average	Worst
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)
Heap Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Quick Sort	Ω(n log(n))	θ(n log(n))	O(n^2)
Merge Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)
Radix Sort	Ω(nk)	θ(nk)	O(nk)

Summary

At the of this session we learned about

- Merge Sort