

MERGE SORT ALGORITHM

DAA CSE 3004

19BCE7578

TARUN

CODE

```
import java.util.Scanner;

public class MergeSort
{
    void merge(int arr[], int l, int m, int r)
    {
        int n1 = m - l + 1;
        int n2 = r - m;

        int L[] = new int[n1];
        int R[] = new int[n2];

        for (int i = 0; i < n1; ++i)
            L[i] = arr[l + i];
        for (int j = 0; j < n2; ++j)
            R[j] = arr[m + 1 + j];

        int i = 0, j = 0;
        int k = l;
        while (i < n1 && j < n2) {
            if (L[i] <= R[j]) {
                arr[k] = L[i];
```

```

i++;

}

else {
arr[k] = R[j];

j++;

}

k++;

}

while (i < n1) {

arr[k] = L[i];

i++;

k++;

}

while (j < n2) {

arr[k] = R[j];

j++;

k++;

}

}

void sort(int arr[], int l, intr)

{

if (l < r) {

int m = l + (r - l) / 2;

sort(arr, l, m);

sort(arr, m + 1, r);

```

```

merge(arr, l, m, r);
}
}

static void printArray(int arr[])
{
int n = arr.length;
for (int i = 0; i < n; ++i)
System.out.print(arr[i] + " ");
System.out.println();
}

public static void main(String args[])
{
long start = System.nanoTime();

int arr[] = {75, 10, 58, 33, 50, 59, 64, 40, 97, 34, 58, 65};

System.out.println("Given Array");

printArray(arr);

MergeSort ob = new MergeSort();

ob.sort(arr, 0, arr.length - 1);

System.out.println("\nSorted array");

printArray(arr);

long end = System.nanoTime();

long time = (end - start)/1000000;

System.out.println("Running time in milli seconds: "+time);

```

}

}

```
1  import java.util.Scanner;
2  public class MergeSort
3  {
4  void merge(int arr[], int l, int m, int r)
5  {
6  int n1 = m - l + 1;
7  int n2 = r - m;
8
9  int L[] = new int[n1];
10 int R[] = new int[n2];
11
12 for (int i = 0; i < n1; ++i)
13 L[i] = arr[l + i];
14 for (int j = 0; j < n2; ++j)
15 R[j] = arr[m + 1 + j];
16 int i = 0, j = 0;
17 int k = l;
18 while (i < n1 && j < n2) {
19 if (L[i] <= R[j]) {
20 arr[k] = L[i];
21 i++;
22 }
23 else {
24 arr[k] = R[j];
25 j++;
26 }
27 k++;
28 }
29 while (i < n1) {
30 arr[k] = L[i];
31 i++;
32 k++;
33 }
34 while (j < n2) {
35 arr[k] = R[j];
36 j++;
37 k++;
38 }
39 }
40 void sort(int arr[], int l, int r)
41 {
42 if (l < r) {
43 int m = l + (r-l)/2;
44 sort(arr, l, m);
45 sort(arr, m + 1, r);
46 merge(arr, l, m, r);
47 }
48 }
49 static void printArray(int arr[])
```

```

57 public static void main(String args[])
58 {
59     long start = System.nanoTime();
60     int arr[] = {75, 10, 58, 33, 50, 59, 64, 40, 97, 34, 58, 65};
61
62     System.out.println("Given Array");
63     printArray(arr);
64     MergeSort ob = new MergeSort();
65     ob.sort(arr, 0, arr.length - 1);
66     System.out.println("\nSorted array");
67     printArray(arr);
68     long end = System.nanoTime();
69     long time = (end - start)/1000000;
70     System.out.println("Running time in milli seconds: "+time);
71 }
72 }

```

✓ Execute Mode, Version, Inputs & Arguments

JDK 11.0.4 ☐ Interactive Stdin Inputs

CommandLine Arguments

Result

Execution Time: 0.15 sec(s), Memory: 35872 kilobyte(s)

```

Given Array
75 10 58 33 50 59 64 40 97 34 58 65

Sorted array
10 33 34 40 50 58 59 64 65 75 97
Running time in milli seconds: 45

```

Time complexity:

$T(n)$ =total time

$T(n)=2T(n/2)+n-1$

$T(n)=2T(n/2)+n$ -----1

$n=n/2$

$T(n/2)=2T(n/4)+n/2$ -----2

Putting equ 2 in 1

$T(n)=4T(n/4)+2n$ -----3

Putting $n=n/4$ in equ 1

$T(n/4)=2T(n/4)+n/4$ -----4

Putting equ 4 in 3

$$T(n) = 8T(n/4) + 3n \text{-----} 5$$

$$\log n = i \log 2$$

$$\log n / \log 2 = i$$

$$= 2^0 + \log_2 n \cdot n$$

$$= T(n)$$

$$= n \log n$$