

# DESIGN ANALYSIS AND ALGORITHMS

## LAB EXPERIMENT 2

NAME: ARYAN MAAN

SAPID: 590015512

BATCH: 34

SUBMITTED TO : MR. ARYAN GUPTA

## CODE:

```
public class MergeSortTest {  
    public static void merge(int[] arr, int left, int mid, int right) {  
        int n1 = mid - left + 1;  
        int n2 = right - mid;  
        int[] L = new int[n1];  
        int[] R = new int[n2];  
  
        for (int i = 0; i < n1; ++i)  
            L[i] = arr[left + i];  
        for (int j = 0; j < n2; ++j)  
            R[j] = arr[mid + 1 + j];  
  
        int i = 0, j = 0;  
        int k = left;  
        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++;  
    }  
}
```

```
public static void mergeSort(int[] arr, int left, int right) {  
    if (left < right) {  
        int mid = (left + right) / 2;  
  
        mergeSort(arr, left, mid);  
        mergeSort(arr, mid + 1, right);  
  
        merge(arr, left, mid, right);  
    }  
}
```

```

public static void printArray(int[] arr) {
    for (int value : arr) {
        System.out.print(value + " ");
    }
    System.out.println();
}

public static void main(String[] args) {
    int[][] testCases = {
        {12, 11, 13, 5, 6, 7},
        {1, 2, 3, 4, 5, 6},
        {6, 5, 4, 3, 2, 1},
        {10, 10, 10, 10},
        {5},
        {},
        {8, -3, 7, -1, 2, 0},
        {100, 50, 200, 150, 25},
        {3, 3, 2, 1, 2, 1},
        {9, 1, 8, 2, 7, 3, 6, 4, 5}
    };

    for (int i = 0; i < testCases.length; i++) {
        System.out.println("Test Case " + (i + 1) + ":");
        int[] arr = testCases[i];
        System.out.print("Original: ");
        printArray(arr);

        mergeSort(arr, 0, arr.length - 1);

        System.out.print("Sorted: ");
        printArray(arr);
        System.out.println("-----");
    }
}

```

OUTPUT:  
Test Case 1:  
Original: 12 11 13 5 6 7  
Sorted: 5 6 7 11 12 13

-----  
Test Case 2:  
Original: 1 2 3 4 5 6  
Sorted: 1 2 3 4 5 6

-----  
Test Case 3:  
Original: 6 5 4 3 2 1  
Sorted: 1 2 3 4 5 6

-----  
Test Case 4:  
Original: 10 10 10 10  
Sorted: 10 10 10 10

-----  
Test Case 5:  
Original: 5  
Sorted: 5

-----  
Test Case 6:  
Original:  
Sorted:

-----  
Test Case 7:  
Original: 8 -3 7 -1 2 0  
Sorted: -3 -1 0 2 7 8

-----  
Test Case 8:  
Original: 100 50 200 150 25  
Sorted: 25 50 100 150 200

-----  
Test Case 9:  
Original: 3 3 2 1 2 1  
Sorted: 1 1 2 2 3 3

-----  
Test Case 10:  
Original: 9 1 8 2 7 3 6 4 5  
Sorted: 1 2 3 4 5 6 7 8 9

-----



```

1 public class MergeSortTest {
2
3
4     public static void merge(int[] arr, int left, int mid, int right) {
5         int n1 = mid - left + 1;
6         int n2 = right - mid;
7
8
9         int[] L = new int[n1];
10        int[] R = new int[n2];
11
12
13        for (int i = 0; i < n1; ++i)
14            L[i] = arr[left + i];
15        for (int j = 0; j < n2; ++j)
16            R[j] = arr[mid + 1 + j];
17
18
19        int i = 0, j = 0;
20        int k = left;
21        while (i < n1 && j < n2) {
22            if (L[i] <= R[j]) {
23                arr[k] = L[i];
24                i++;
25            } else {
26                arr[k] = R[j];
27                j++;
28            }
29            k++;
30        }

```

```

Test Case 1:
Original: 12 11 13 5 6 7
Sorted:   5 6 7 11 12 13
-----

Test Case 2:
Original: 1 2 3 4 5 6
Sorted:   1 2 3 4 5 6
-----

Test Case 3:
Original: 6 5 4 3 2 1
Sorted:   1 2 3 4 5 6
-----

Test Case 4:
Original: 10 10 10 10
Sorted:   10 10 10 10
-----

Test Case 5:
Original: 5
Sorted:   5
-----

Test Case 6:
Original:
Sorted:
-----

Test Case 7:
Original: 8 -3 7 -1 2 0
Sorted:   -3 -1 0 2 7 8
-----

Test Case 8:
Original: 100 50 200 150 25

```



```

33     while (i < n1) {
34         arr[k] = L[i];
35         i++;
36         k++;
37     }
38     while (j < n2) {
39         arr[k] = R[j];
40         j++;
41         k++;
42     }
43 }
44 public static void mergeSort(int[] arr, int left, int right) {
45     if (left < right) {
46         int mid = (left + right) / 2;
47         mergeSort(arr, left, mid);
48         mergeSort(arr, mid + 1, right);
49         merge(arr, left, mid, right);
50     }
51 }
52
53
54 public static void printArray(int[] arr) {
55     for (int value : arr) {
56         System.out.print(value + " ");
57     }
58     System.out.println();
59 }
60
61
62 public static void main(String[] args) {

```

```

Test Case 1:
Original: 12 11 13 5 6 7
Sorted:   5 6 7 11 12 13
-----

```

```

Test Case 2:
Original: 1 2 3 4 5 6
Sorted:   1 2 3 4 5 6
-----

```

```

Test Case 3:
Original: 6 5 4 3 2 1
Sorted:   1 2 3 4 5 6
-----

```

```

Test Case 4:
Original: 10 10 10 10
Sorted:   10 10 10 10
-----

```

```

Test Case 5:
Original: 5
Sorted:   5
-----

```

```

Test Case 6:
Original:
Sorted:
-----

```

```

Test Case 7:
Original: 8 -3 7 -1 2 0
Sorted:   -3 -1 0 2 7 8
-----

```

```

Test Case 8:
Original: 100 50 200 150 25

```



```

public static void main(String[] args) {
    int[][] testCases = {
        {12, 11, 13, 5, 6, 7},
        {1, 2, 3, 4, 5, 6},
        {6, 5, 4, 3, 2, 1},
        {10, 10, 10, 10},
        {5},
        {},
        {8, -3, 7, -1, 2, 0},
        {100, 50, 200, 150, 25},
        {3, 3, 2, 1, 2, 1},
        {9, 1, 8, 2, 7, 3, 6, 4, 5}
    };

    for (int i = 0; i < testCases.length; i++) {
        System.out.println("Test Case " + (i + 1) + ":");
        int[] arr = testCases[i];
        System.out.print("Original: ");
        printArray(arr);

        mergeSort(arr, 0, arr.length - 1);

        System.out.print("Sorted: ");
        printArray(arr);
        System.out.println("-----");
    }
}

```

Test Case 4:

Original: 10 10 10 10

Sorted: 10 10 10 10

-----

Test Case 5:

Original: 5

Sorted: 5

-----

Test Case 6:

Original:

Sorted:

-----

Test Case 7:

Original: 8 -3 7 -1 2 0

Sorted: -3 -1 0 2 7 8

-----

Test Case 8:

Original: 100 50 200 150 25

Sorted: 25 50 100 150 200

-----

Test Case 9:

Original: 3 3 2 1 2 1

Sorted: 1 1 2 2 3 3

-----

Test Case 10:

Original: 9 1 8 2 7 3 6 4 5

Sorted: 1 2 3 4 5 6 7 8 9

-----



50.8k  
Shares



```
public class mergesorttest {  
    public static void merge(int arr, int left, int mid, int right, int n1, int mid, int left, int n2,  
        int right, int l, new int n1, int r, new int n2) {  
        for (int i = 0; i < n1; i++) {  
            arr[i] = li[i];  
        }  
        for (int j = 0; j < n2; j++) {  
            arr[j + mid] = rj[j];  
        }  
        int k = left; while (i < n1 && j < n2) {  
            if (li[i] < rj[j]) {  
                arr[k] = li[i];  
                i++;  
            } else {  
                arr[k] = rj[j];  
                j++;  
            }  
            k++;  
        }  
        while (i < n1) {  
            arr[k] = li[i];  
            i++;  
            k++;  
        }  
        while (j < n2) {  
            arr[k] = rj[j];  
            j++;  
            k++;  
        }  
    }  
    public static void mergesort(int arr, int left, int right) {  
        if (left < right) {  
            int mid = (left + right) / 2;  
            mergesort(arr, left, mid);  
            mergesort(arr, mid + 1, right);  
            merge(arr, left, mid, right);  
        }  
    }  
    public static void printArray(int arr) {  
        for (int i = 0; i < arr.length; i++) {  
            System.out.print(arr[i] + " ");  
        }  
        System.out.println();  
    }  
    public static void main(String args[]) {  
        int testcases = 12; // 12 11 13 5 6 7 1 2 3 4 5 6 6 5 4 3 2 1 10  
        // 10 10 10 5 8 -3 7 -12 0 100 50 200 150 25 3 3 2 12 19 18 2 7 3 6 4 5  
        for (int i = 0; i < testcases; i++) {  
            System.out.print("Test case " + (i + 1) + ": ");  
            int arr[] = new int[testcases];  
            printArray(arr);  
            mergesort(arr, 0, arr.length - 1);  
            printArray(arr);  
            System.out.println("-----");  
        }  
    }  
}
```

SIMILAR

19.8%

ORIGINAL

80.2%

MAKE IT UNIQUE

Text matches these sources

Sources:

Similarity:

1. <https://studyx.ai/homework/1119156...> 18.4%

Exclude source

View source

2. <https://skillapp.co/blog/mastering-...> 12.8%

3. <https://prepinsta.com/java-program...> 12.8%

4. <https://www.mygreatlearning.com/...> 12.5%



I'm not a robot



reCAPTCHA  
Privacy - Terms

