

MERGE SORT

```
#include <bits/stdc++.h>

using namespace std;

void merge(int arr[], int l, int m, int r)
{
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[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;
    int 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 mergeSort(int arr[],int l,int r){

    if(l>=r){

        return;

    }

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

    mergeSort(arr,l,m);

    mergeSort(arr,m+1,r);

    merge(arr,l,m,r);

}

void printArray(int A[], int size)

{

    for (int i = 0; i < size; i++)

        cout << A[i] << " ";

}

int main()

{

    int arr[] = {13,10,12,89,1,3,5,1,56};

    int arr_size = sizeof(arr) / sizeof(arr[0]);

    cout << "Given array is \n";

    printArray(arr, arr_size);

    mergeSort(arr, 0, arr_size - 1);

```

```

        cout << "\nSorted array is \n";

        printArray(arr, arr_size);

        return 0;
    }

```

Time Complexity:

In sorting n objects, merge sort has an average and worst-case performance of $O(n \log n)$.

```

1.  #include <bits/stdc++.h>
2.  using namespace std;
3.  void merge(int arr[], int l, int m, int r)
4.  {
5.      int n1 = m - l + 1;
6.      int n2 = r - m;
7.      int L[n1], R[n2];
8.      for (int i = 0; i < n1; i++)
9.          L[i] = arr[l + i];
10.     for (int j = 0; j < n2; j++)
11.         R[j] = arr[m + 1 + j];
12.     int i = 0;
13.     int j = 0;
14.     int k = l;
15.     while (i < n1 && j < n2) {
16.         if (L[i] <= R[j]) {
17.             arr[k] = L[i];
18.             i++;
19.         }
20.         else {
21.             arr[k] = R[j];
22.             j++;
23.         }
24.         k++;
25.     }
26.     while (i < n1)
27.     {
28.         arr[k] = L[i];
29.         i++;
30.         k++;
31.     }
32.     while (j < n2) {
33.         arr[k] = R[j];
34.         j++;
35.         k++;
36.     }
37. }
38. void mergeSort(int arr[],int l,int r){
39.     if(l>=r){
40.         return;
41.     }
42.     int m =l+ (r-l)/2;
43.     mergeSort(arr,l,m);
44.     mergeSort(arr,m+1,r);
45.     merge(arr,l,m,r);
46. }

```

```
47. void printArray(int A[], int size)
48. {
49.     for (int i = 0; i < size; i++)
50.         cout << A[i] << " ";
51. }
52. int main()
53. {
54.     int arr[] = {13,10,12,89,1,3,5,1,56};
55.     int arr_size = sizeof(arr) / sizeof(arr[0]);
56.     cout << "Given array is \n";
57.     printArray(arr, arr_size);
58.     mergeSort(arr, 0, arr_size - 1);
59.     cout << "\nSorted array is \n";
60.     printArray(arr, arr_size);
61.     return 0;
62. }
63.
```


Success #stdin #stdout 0s 5504KB

 comments (0)

 stdin

 copy

Standard input is empty

 stdout

 copy

Given array is
13 10 12 89 1 3 5 1 56
Sorted array is
1 1 3 5 10 12 13 56 89