

# Merge Sort

```
File Edit Selection View Go Run Terminal Help MergeSort.cpp - DAA-practical - Visual Studio Code
MergeSort.cpp x
Practical-3 > MergeSort.cpp > inputArray(int [], int)
1 // Author: Anish Tilloo
2 // Roll No. : 34
3 // Program: Merge Sort
4
5 #include <bits/stdc++.h>
6 using namespace std;
7
8 void merge(int arr[], int left, int mid, int right){
9     // size of array one and array two
10    int arrayOne = mid - left + 1;
11    int arrayTwo = right - mid;
12
13    // dynamicall allocating two sub array
14    auto *subArrayOne = new int[arrayOne];
15    auto *subArrayTwo = new int[arrayTwo];
16
17    // putting the elements of array one and two from main array
18    for (int i = 0; i < arrayOne; i++)
19    {
20        subArrayOne[i] = arr[left + i];
21    }
22    for (int j = 0; j < arrayTwo; j++)
```

```
File Edit Selection View Go Run Terminal Help MergeSort.cpp - DAA-practical - Visual Studio Code
MergeSort.cpp x
Practical-3 > MergeSort.cpp > inputArray(int [], int)
21 }
22 for (int j = 0; j < arrayTwo; j++)
23 {
24     subArrayTwo[j] = arr[mid + 1 + j];
25 }
26
27 int indexOfOne = 0;
28 int indexOfTwo = 0;
29 int indexOfMerged = left;
30
31 // merging the arrays
32 while (indexOfOne < arrayOne && indexOfTwo < arrayTwo)
33 {
34     if (subArrayOne[indexOfOne] <= subArrayTwo[indexOfTwo])
35     {
36         arr[indexOfMerged] = subArrayOne[indexOfOne];
37         indexOfOne++;
38     }
39     else
40     {
41         arr[indexOfMerged] = subArrayTwo[indexOfTwo];
42         indexOfTwo++;
```

```
File Edit Selection View Go Run Terminal Help MergeSort.cpp - DAA-practical - Visual Studio Code
MergeSort.cpp x
Practical-3 > MergeSort.cpp > inputArray(int [], int)
40     else
41     {
42         arr[indexOfMerged] = subArrayTwo[indexOfTwo];
43         indexOfTwo++;
44     }
45     indexOfMerged++;
46
47     // if one array end the copy all the remaining elements from the remaining array
48     while (indexOfOne < arrayOne)
49     {
50         arr[indexOfMerged] = subArrayOne[indexOfOne];
51         indexOfOne++;
52         indexOfMerged++;
53     }
54
55     while (indexOfTwo < arrayTwo)
56     {
57         arr[indexOfMerged] = subArrayTwo[indexOfTwo];
58         indexOfTwo++;
59         indexOfMerged++;
60     }
61 }
```

```
File Edit Selection View Go Run Terminal Help MergeSort.cpp - DAA-practical - Visual Studio Code
MergeSort.cpp x
Practical-3 > MergeSort.cpp > inputArray(int [], int)
59     indexOfMerged++;
60 }
61 }
62
63 void mergeSort(int arr[], int left, int right){
64     if (left < right)
65     {
66         int mid = (left + right) / 2;
67         mergeSort(arr, left, mid);
68         mergeSort(arr, mid + 1, right);
69
70         merge(arr, left, mid, right);
71     }
72
73 }
74
75 void inputArray( int arr[], int arraySize){
76     cout << "Enter the elements of the array one by one " << endl;
77     for (int i = 0; i < arraySize; i++)
78     {
79         cin >> arr[i];
80     }
```

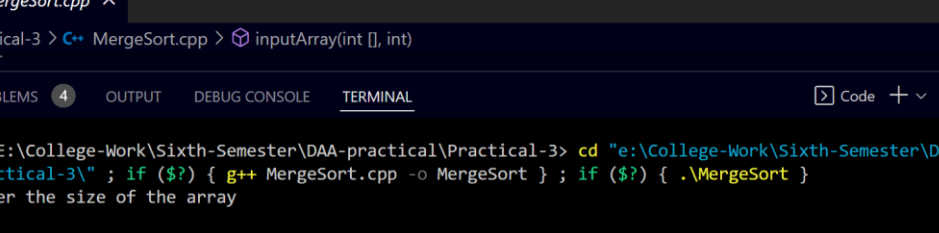
The image shows a Visual Studio Code editor window with a C++ file named 'MergeSort.cpp'. The code is as follows:

```

83 void printArray(int arr[], int arraySize){
84     for (int i = 0; i < arraySize; i++)
85     {
86         cout << arr[i] << " ";
87     }
88 }
89
90 int main(){
91     int arraySize;
92     cout << "Enter the size of the array " << endl;
93     cin >> arraySize;
94
95     int arr[arraySize];
96     inputArray(arr, arraySize);
97
98     mergeSort(arr, 0, arraySize - 1);
99
100    cout << "Sorted Array" << endl;
101    printArray(arr, arraySize);
102 }

```

The status bar at the bottom indicates the current position is 'Ln 80, Col 6'. Other status bar items include 'Run Testcases', 'Anish', 'Live Share', 'Spaces: 4', 'UTF-8', 'CRLF', 'C++', 'Go Live', 'Spell', 'Win32', 'Prettier', and a search icon.



```
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> cd "e:\College-Work\Sixth-Semester\DAA-practical\Practical-3\" ; if ($?) { g++ MergeSort.cpp -o MergeSort } ; if ($?) { .\MergeSort }
Enter the size of the array
6
Enter the elements of the array one by one
90 5 12 7 34 2
Sorted Array
2 5 7 12 34 90

PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> cd "e:\College-Work\Sixth-Semester\DAA-practical\Practical-3\" ; if ($?) { g++ MergeSort.cpp -o MergeSort } ; if ($?) { .\MergeSort }
Enter the size of the array
5
Enter the elements of the array one by one
8 5 1 9 10
Sorted Array
1 5 8 9 10
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> |
```

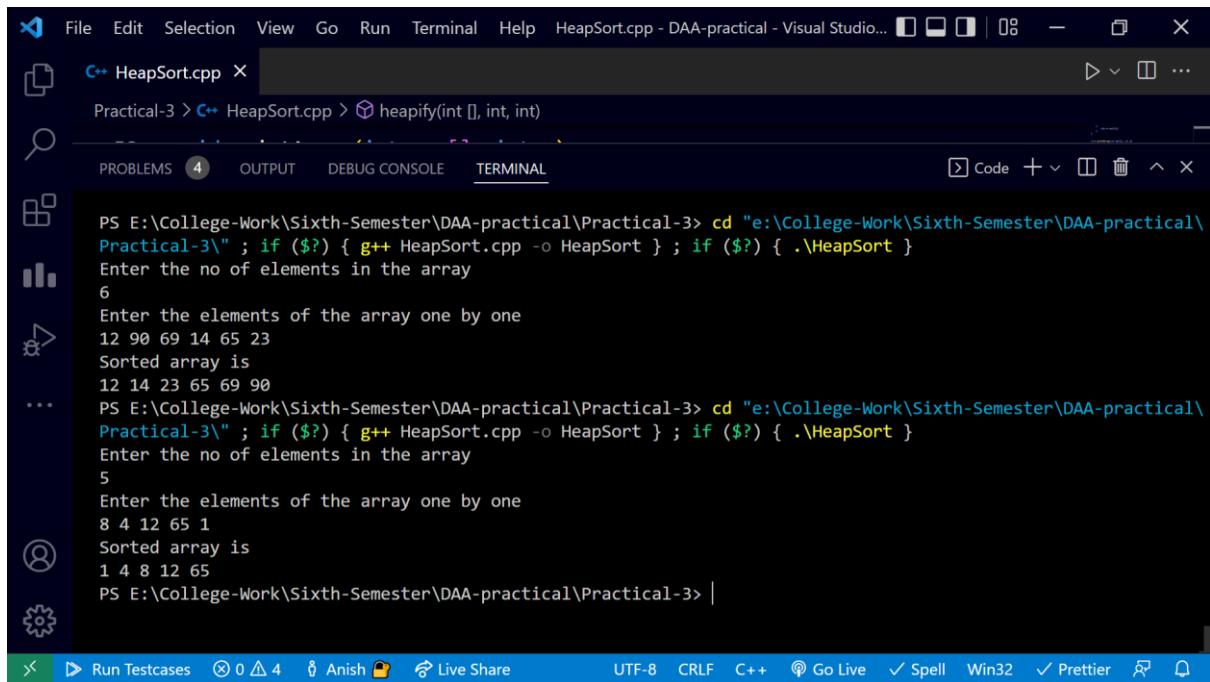
# Heap Sort

```
File Edit Selection View Go Run Terminal Help HeapSort.cpp - DAA-practical - Visual Studio Code
HeapSort.cpp x
Practical-3 > C++ HeapSort.cpp > heapify(int [], int, int)
1 // Author: Anish Tilloo
2 // Roll No. : 34
3 // Program: Heap Sort
4
5 #include <bits/stdc++.h>
6 using namespace std;
7
8 // Heap Sort
9 void heapify(int arr[], int n, int i)
10 {
11     int largest = i; // Initialize largest as root
12     int l = 2 * i + 1; // left = 2*i + 1
13     int r = 2 * i + 2; // right = 2*i + 2
14
15     // here we are checking if the left child index is greater than n or not
16     // and the left child is greater than the current element
17     if (l < n && arr[l] > arr[largest])
18         largest = l;
19
20     // here we are checking if the right child index is greater than n or not
21     // and the right child is greater than the current element
22     if (r < n && arr[r] > arr[largest])
```

```
File Edit Selection View Go Run Terminal Help HeapSort.cpp - DAA-practical - Visual Studio Code
HeapSort.cpp x
Practical-3 > C++ HeapSort.cpp > heapify(int [], int, int)
16 // and the left child is greater than the current element
17 if (l < n && arr[l] > arr[largest])
18     largest = l;
19
20 // here we are checking if the right child index is greater than n or not
21 // and the right child is greater than the current element
22 if (r < n && arr[r] > arr[largest])
23     largest = r;
24
25 // If largest is not root
26 if (largest != i) {
27     swap(arr[i], arr[largest]);
28     heapify(arr, n, largest);
29 }
30 }
31
32
33 void heapSort(int arr[], int n)
34 {
35     for (int i = n / 2 - 1; i >= 0; i--){
36         heapify(arr, n, i);
37     }
```

```
File Edit Selection View Go Run Terminal Help HeapSort.cpp - DAA-practical - Visual Studio Code
HeapSort.cpp x
Practical-3 > C++ HeapSort.cpp > heapify(int [], int, int)
33 void heapSort(int arr[], int n)
34 {
35     for (int i = n / 2 - 1; i >= 0; i--){
36         heapify(arr, n, i);
37     }
38
39     for (int i = n - 1; i >= 0; i--){
40         // Move current root to end
41         swap(arr[0], arr[i]);
42
43         // call heapify on the reduced heap
44         heapify(arr, i, 0);
45     }
46 }
47
48 void inputArray( int arr[], int arraySize){
49     cout << "Enter the elements of the array one by one " << endl;
50     for (int i = 0; i < arraySize; i++)
51     {
52         cin >> arr[i];
53     }
54 }
```

```
File Edit Selection View Go Run Terminal Help HeapSort.cpp - DAA-practical - Visual Studio Code
HeapSort.cpp x
Practical-3 > C++ HeapSort.cpp > heapify(int [], int, int)
58 void printArray(int arr[], int n)
59 {
60     for (int i = 0; i < n; ++i)
61         cout << arr[i] << " ";
62     cout << "\n";
63 }
64
65 int main()
66 {
67     int n;
68     cout << "Enter the no of elements in the array" << endl;
69     cin >> n;
70     int arr[n];
71     inputArray(arr, n);
72
73     heapSort(arr, n);
74
75     cout << "Sorted array is \n";
76     printArray(arr, n);
77 }
78
```



The image shows a Visual Studio Code window with a terminal running a C++ program. The terminal output shows two test cases for a heap sort algorithm. In the first case, an array of 6 elements [12, 90, 69, 14, 65, 23] is sorted to [12, 14, 23, 65, 69, 90]. In the second case, an array of 5 elements [8, 4, 12, 65, 1] is sorted to [1, 4, 8, 12, 65].

```
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> cd "e:\College-Work\Sixth-Semester\DAA-practical\Practical-3\" ; if ($?) { g++ HeapSort.cpp -o HeapSort } ; if ($?) { .\HeapSort }
Enter the no of elements in the array
6
Enter the elements of the array one by one
12 90 69 14 65 23
Sorted array is
12 14 23 65 69 90
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> cd "e:\College-Work\Sixth-Semester\DAA-practical\Practical-3\" ; if ($?) { g++ HeapSort.cpp -o HeapSort } ; if ($?) { .\HeapSort }
Enter the no of elements in the array
5
Enter the elements of the array one by one
8 4 12 65 1
Sorted array is
1 4 8 12 65
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-3> |
```