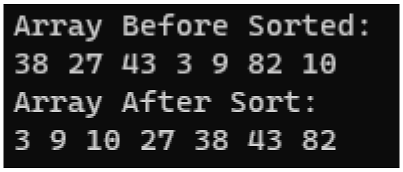
```c
// Merge Sort
#include <stdio.h>
#include <stdlib.h>

// arr[] = 38,27,43,,3,9,82,10
// Output will be = {3,9,10,27,38,43,82}


int Merge(int arr[], int p, int q, int r) {

  int n1 = q - p + 1;
  int n2 = r - q;

  int L[n1], M[n2];

  for (int i = 0; i < n1; i++)
    L[i] = arr[p + i];
  for (int j = 0; j < n2; j++)
    M[j] = arr[q + 1 + j];
```

```
int i, j, k;
i = 0;
j = 0;
k = p;

while (i < n1 && j < n2) {
  if (L[i] <= M[j]) {
    arr[k] = L[i];
    i++;
  } else {
    arr[k] = M[j];
    j++;
  }
  k++;
}

while (i < n1) {
  arr[k] = L[i];
  i++;
```

```
    k++;
  }

  while (j < n2) {
    arr[k] = M[j];
    j++;
    k++;
  }
}

int MergeSort(int arr[], int l, int r) {
  if (l < r) {

    int m = l + (r - l) / 2;
    MergeSort(arr, l, m);
    MergeSort(arr, m + 1, r);

    Merge(arr, l, m, r);
  }
}
```

```
// Array Printing Function:
void printArray(int arr[], int size) {
  for (int i = 0; i < size; i++)
    printf("%d ", arr[i]);
  printf("\n");
}


int main() {
  int arr[] = {38,27,43,3,9,82,10};
  int size = sizeof(arr) / sizeof(arr[0]);

  printf("Array Before Sorted: \n");
  printArray(arr,size);
  MergeSort(arr, 0, size - 1);
  printf("Array After Sort: \n");
  printArray(arr, size);
}
```

Output:



```
Array Before Sorted:
38 27 43 3 9 82 10
Array After Sort:
3 9 10 27 38 43 82
```

Time Complexity of MergeSort Function:

O (n*log n)