**Program – 6**

**Write an algorithm and program to sort n numbers using merge sort technique**

1. **Using normal approach i.e. recursion illustrating Divide and Conquer**

**Algorithm**

MergeSort(arr[], l, r), where l is the index of the first element & r is the index of the last element.  
If r > l  
1. Find the middle index of the array to divide it in two halves:  
m = (l+r)/2  
2. Call MergeSort for first half:  
mergeSort(array, l, m)  
3. Call mergeSort for second half:  
mergeSort(array, m+1, r)  
4. Recursively, merge the two halves in a sorted manner, so that only one sorted array is left:  
merge(array, l, m, r)

**Source Code:**

#include <iostream>

using namespace std;

void merge(intarr[], int l, int m, int r)

{

int n1 = m - l + 1;

int n2 = r - m;

int L[n1], R[n2];

for (inti = 0; i< n1; i++)

L[i] = arr[l + i];

for (int j = 0; j < n2; j++)

R[j] = arr[m + 1 + j];

inti = 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++;

}

}

voidmergeSort(intarr[],intl,int r){

if(l>=r){

Return;

}

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

mergeSort(arr,l,m);

mergeSort(arr,m+1,r);

merge(arr,l,m,r);

}

voidprintArray(int A[], int size)

{

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

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

}

int main()

{

intarr[] = { 12, 11, 13, 5, 6, 7 };

intarr\_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;

}

1. **Without using Recursion.**

**Source Code:**

#include<bits/stdc++.h>

using namespace std;

void merge(intarr[], int l, int m, int r);

voidmergeSort(intarr[], 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);

}

}

void merge(intarr[], int l, int m, int r)

{

int k;

int n1 = m - l + 1;

int n2 = r - m;

int L[n1], R[n2];

for(inti = 0; i< n1; i++)

L[i] = arr[l + i];

for(int j = 0; j < n2; j++)

R[j] = arr[m + 1+ j];

inti = 0;

int j = 0;

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++;

}

}

voidprintArray(int A[], int size)

{

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

printf("%d ", A[i]);

cout<< "\n";

}

int main()

{

intarr[] = { 12, 11, 13, 5, 6, 7 };

intarr\_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;

}