Ex. No. 10e **Merge Sort**

Date:

**Aim**

To sort an array of N numbers using Merge sort.

**Algorithm**

1. Start

2. Read number of array elements n

3. Read array elements Ai

4. Divide the array into sub-arrays with a set of elements

5. Recursively sort the sub-arrays

6. Merge the sorted sub-arrays onto a single sorte

7. Stop

**Program**

/\* Merge sort \*/

#include <stdio.h>

#include <stdlib.h>

void merge(int [],int ,int ,int );

void part(int [],int ,int );

int size;

void main()

{

int i, arr[30];

printf("Enter total no. of elements : ");

scanf("%d", &size);

printf("Enter array elements : ");

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

scanf("%d", &arr[i]);

part(arr, 0, size-1);

printf("\n Merge sorted list : ");

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

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

}

void part(int arr[], int min, int max)

{

int i, mid;

if(min < max)

{

mid = (min + max) / 2;

part(arr, min, mid);

part(arr, mid+1, max);

merge(arr, min, mid, max);

}

if (max-min == (size/2)-1)

{

printf("\n Half sorted list : ");

for(i=min; i<=max; i++)

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

}

}

void merge(int arr[],int min,int mid,int max)

{

int tmp[30];

int i, j, k, m;

j = min;

m = mid + 1;

for(i=min; j<=mid && m<=max; i++)

{

if(arr[j] <= arr[m])

{

tmp[i] = arr[j];

j++;

}

else

{

tmp[i] = arr[m];

m++;

}

}

if(j > mid)

{

for(k=m; k<=max; k++)

{

tmp[i] = arr[k];

i++;

}

}

else

{

for(k=j; k<=mid; k++)

{

tmp[i] = arr[k];

i++;

}

}

for(k=min; k<=max; k++)

arr[k] = tmp[k];

}

**Output**

**Result**

Thus array elements was sorted using merge sort's divide and conquer method.