**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 <stdio.h>

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

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

int main()

{

int list[50];

int i, size;

printf("Enter total number of elements:");

scanf("%d", &size);

printf("Enter the elements:\n");

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

{

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

}

partition(list, 0, size - 1);

printf("After merge sort:\n");

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

{

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

}

return 0;

}

void partition(int list[],int low,int high)

{

int mid;

if(low < high)

{

mid = (low + high) / 2;

partition(list, low, mid);

partition(list, mid + 1, high);

mergeSort(list, low, mid, high);

}

}

void mergeSort(int list[],int low,int mid,int high)

{

int i, mi, k, lo, temp[50];

lo = low;

i = low;

mi = mid + 1;

while ((lo <= mid) && (mi <= high))

{

if (list[lo] <= list[mi])

{

temp[i] = list[lo];

lo++;

}

else

{

temp[i] = list[mi];

mi++;

}

i++;

}

if (lo > mid)

{

for (k = mi; k <= high; k++)

{

temp[i] = list[k];

i++;

}

}

else

{

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

{

temp[i] = list[k];

i++;

}

}

for (k = low; k <= high; k++)

{

list[k] = temp[k];

}

}

1. **Without using Recursion.**

**Source Code:**

#include <stdio.h>

#define MAX 30

int main()

{

int arr[MAX],temp[MAX],i,j,k,n,size,l1,h1,l2,h2;

printf("Enter the number of elements : ");

scanf("%d",&n);

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

{

printf("Enter element %d : ",i+1);

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

}

printf("Unsorted list is : ");

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

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

for(size=1; size < n; size=size\*2 )

{

l1=0;

k=0;

while( l1+size < n)

{

h1=l1+size-1;

l2=h1+1;

h2=l2+size-1;

if( h2>=n )

h2=n-1;

i=l1;

j=l2;

while(i<=h1 && j<=h2 )

{

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

temp[k++]=arr[i++];

else

temp[k++]=arr[j++];

}

while(i<=h1)

temp[k++]=arr[i++];

while(j<=h2)

temp[k++]=arr[j++];

l1=h2+1;

}

for(i=l1; k<n; i++)

temp[k++]=arr[i];

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

arr[i]=temp[i];

printf("\nSize=%d \nElements are : ",size);

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

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

}/\*End of for loop \*/

printf("Sorted list is :\n");

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

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

printf("\n");

return 0;

}