**Name:** Purvi Singhvi

**Enrollement No.:** 180099

**PROGRAM-6**

**Aim:** Write an algorithm and program to sort n numbers using Merge sort technique.

**Algorithm:**

MERGE(A, p, q, r)

1. n1= q-p+1

2. n2= r-q

3. Let L[1….. n1+1] and R[1…...n2+1] are new arrays

4. for (i=1 to n1) do

5. L[i] = A[p+i-1]

6. END for

7. for(j=1 to n2)do

8. R[j]=A[q+j]

9. END for

10. L[n1+1]= infinity

11. R[n2+1]=inifinity

12. for(k=p to r)do

13. if(L[i] <= R[j])

14. A[k] = L[i]

15. i=i+1

16. Else

17. A[k]=R[j]

18. j=j+1

19. End if

MERGE SORT(A,p,r)

1. if(p<r)

2. q= floor((p+q)/2)

3. MERGE SORT(A, p, q)

4. MERGE SORT(A, q+1, r)

5. MERGE(A, p, q, r)

6. End if

7. STOP

**i)** Using normal approach i.e. recursion illustrating Devide and Conquer.

**Source Code:**

#include <stdio.h>

#include<conio.h>

#include<stdlib.h>

#define max 15

int a[11], b[10];

void merging(int low, int mid, int high) {

int l1, l2, i;

for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++) {

if(a[l1] <= a[l2])

b[i] = a[l1++];

else

b[i] = a[l2++];

}

while(l1 <= mid)

b[i++] = a[l1++];

while(l2 <= high)

b[i++] = a[l2++];

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

a[i] = b[i];

}

void sort(int low, int high) {

int mid;

if(low < high) {

mid = (low + high) / 2;

sort(low, mid);

sort(mid+1, high);

merging(low, mid, high);

} else {

return;

}

}

void main() {

int i,s;

clrscr();

printf("\n Enter size of array:");

scanf("%d", &s);

printf("Enter the elements:");

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

{

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

}

printf("Before sorting\n");

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

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

sort(0, s-1);

printf("\nAfter sorting\n");

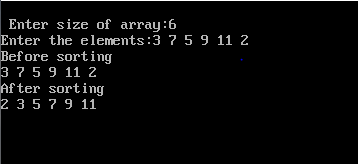
for(i = 0; i < s; i++)

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

getch();

}

**Output:**



**Complexity:**

Best case: nlog(n)

Worst case: nlog(n)

Average case: nlog(n)