**NAM:** MIRUTHULA.B

**REG NO.** 231501100

**EX NO 15** PROGRAM TO PERFORM SORTING

Quick sort

#include <stdio.h>

#define size 8 // Define the correct size based on the number of elements

int a[size] = {40, 20, 70, 14, 60, 61, 97, 30}; // Initialize array correctly

void quick(int a[], int l, int r)

{

int i, j, t, p;

if (l < r)

{

p = l;

i = l;

j = r;

while (i < j)

{

while (a[i] <a[p])

i++;

while (a[j] > a[p])

j--;

if (i < j)

{

t = a[i];

a[i] = a[j];

a[j] = t;

}

}

t = a[p];

a[p] = a[j];

a[j] = t;

quick(a, l, j - 1);

quick(a, j + 1, r);

}

}

int main()

{

quick(a, 0, size - 1);

printf("Sorted array: ");

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

{

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

}

printf("\n");

return 0;

}

Merge sort

#include <stdio.h>

#define SIZE 7

int arr[SIZE] = {99, 0, 12, 58, 69, 77, 2};

void mer(int arr[], int left, int centre, int right) {

int n1 = centre - left + 1;

int n2 = right - centre;

int a[n1], b[n2];

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

a[i] = arr[left + i];

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

b[j] = arr[centre + 1 + j];

int aptr = 0, bptr = 0, cptr = left;

while (aptr < n1 && bptr < n2) {

if (a[aptr] <= b[bptr]) {

arr[cptr] = a[aptr];

aptr++;

} else {

arr[cptr] = b[bptr];

bptr++;

}

cptr++;

}

while (aptr < n1) {

arr[cptr] = a[aptr];

aptr++;

cptr++;

}

while (bptr < n2) {

arr[cptr] = b[bptr];

bptr++;

cptr++;

}

}

void merge(int arr[], int left, int right) {

if (left < right) {

int centre = (left+right) / 2;

merge(arr, left, centre);

merge(arr, centre + 1, right);

mer(arr, left, centre, right);

}

}

int main() {

merge(arr, 0, SIZE - 1);

for (int i = 0; i < SIZE; i++) {

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

}

return 0;

}

OUTPUT:

