NAME: SWETHA.J
REGISTER NO.:230701357

EX-15: Program to perform Sorting

Quick Sort

```
#include <stdio.h>

void QuickSort(int a[], int left, int right);

int main() {
    int i, n, a[10];
    printf("Enter the limit: ");
    scanf("%d", &n);
    printf("Enter the elements: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &a[i]);
    }
    QuickSort(a, 0, n - 1);
    printf("The sorted elements are: ");
    for (i = 0; i < n; i++) {
        printf("%d\t", a[i]);
    }
    return 0;
}

void QuickSort(int a[], int left, int right) {
    int i, j, temp, pivot;
    if (left < right) {
        pivot = left;
        i = left + 1;
        j = right;
        while (i <= j) { // Change here to i <= j instead of i < j
            while (i <= right && a[i] < a[pivot]) i++; // Add boundary
check
            while (j >= left && a[j] > a[pivot]) j--; // Add boundary
check
            if (i < j) {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
                i++; // Move pointers after swapping
                j--;
            } else if (i == j) {
                i++;
            }
        }
        temp = a[pivot];
        a[pivot] = a[j];
        a[j] = temp;
        QuickSort(a, left, j - 1);
        QuickSort(a, j + 1, right);
    }
}
```

MERG SORT

```
#include <stdio.h>

void MergeSort(int arr[], int left, int right);
void Merge(int arr[], int left, int center, int right);

int main() {
    int i, n, arr[20];
    printf("Enter the limit: ");
    scanf("%d", &n);
    printf("Enter the elements: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    MergeSort(arr, 0, n - 1);
    printf("The sorted elements are: ");
    for (i = 0; i < n; i++) {
        printf("%d\t", arr[i]);
    }
    return 0;
}

void MergeSort(int arr[], int left, int right) {
    int center;
    if (left < right) {
        center = (left + right) / 2;
        MergeSort(arr, left, center);
        MergeSort(arr, center + 1, right);
        Merge(arr, left, center, right);
    }
}

void Merge(int arr[], int left, int center, int right) {
    int a[20], b[20], n1, n2, aptr, bptr, cptr, i, j;
    n1 = center - left + 1;
    n2 = right - center;

    for (i = 0; i < n1; i++) {
        a[i] = arr[left + i];
    }
    for (j = 0; j < n2; j++) {
        b[j] = arr[center + 1 + j];
    }

    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++;
    }
}
```