NAME: M.SHESHANK
ROLL NO: CH.SC.U4CSE24125

MERGE SORT:
CODE:

```c
#include <stdio.h>
void merge(int arr[], int left, int mid, int right) {
    int temp[right - left + 1];
    int i = left, j = mid + 1, k = 0;
    while (i <= mid && j <= right) {
        if (arr[i] <= arr[j])
            temp[k++] = arr[i++];
        else
            temp[k++] = arr[j++];
    }
    while (i <= mid)
        temp[k++] = arr[i++];
    while (j <= right)
        temp[k++] = arr[j++];
        for (i = left, k = 0; i <= right; i++, k++)
        arr[i] = temp[k];
}
void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}
void displayArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
```

```c
void displayArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
int main() {
    int n, i;
    printf("=== MERGE SORT PROGRAM ===\n\n");
    printf("Enter the number of elements: ");
    scanf("%d", &n);
    int arr[n];
    printf("\nEnter %d elements:\n", n);
    for (i = 0; i < n; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
    printf("\nOriginal array: ");
    displayArray(arr, n);
    mergeSort(arr, 0, n - 1);
    printf("Sorted array:   ");
    displayArray(arr, n);
    printf("\n=== Array has been sorted successfully! ===\n");
    return 0;
}
```

OUTPUT:

```
=== MERGE SORT PROGRAM ===

Enter the number of elements: 4

Enter 4 elements:
Element 1: 1
Element 2: 2
Element 3: 1
Element 4: 3

Original array: 1 2 1 3
Sorted array:   1 1 2 3

=== Array has been sorted successfully! ===
```

QUICK SORT:

CODE:

```c
#include <stdio.h>
void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++) {
        if (arr[j] <= pivot) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return i + 1;
}
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
int main() {
    int n;
```

```
27      int n;
28      printf("Enter number of elements: ");
29      scanf("%d", &n);
30      int arr[n];
31      printf("Enter elements:\n");
32      for (int i = 0; i < n; i++)
33          scanf("%d", &arr[i]);
34      quickSort(arr, 0, n - 1);
35      printf("Sorted array:\n");
36      for (int i = 0; i < n; i++)
37          printf("%d ", arr[i]);
38      return 0;
39  }
```

OUTPUT:

```
Enter number of elements:
5
Enter elements:
3 2 7 6 9
Sorted array:
2 3 6 7 9
```