#include <stdio.h>

// Merge two sorted subarrays

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

int i, j, k;

int n1 = mid - left + 1;

int n2 = right - mid;

// Temporary arrays

int L[n1], R[n2];

// Copy data to temp arrays

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

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

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

R[j] = arr[mid + 1 + j];

// Merge the temp arrays back into arr[left..right]

i = 0; j = 0; k = left;

while (i < n1 && j < n2) {

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

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

else

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

}

// Copy remaining elements

while (i < n1)

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

while (j < n2)

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

}

// Recursive Merge Sort function

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

if (left < right) {

int mid = (left + right) / 2;

// Sort first and second halves

mergeSort(arr, left, mid);

mergeSort(arr, mid + 1, right);

// Merge the sorted halves

merge(arr, left, mid, right);

}

}

int main() {

int arr[100], n;

// Input: number of elements

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

scanf("%d", &n);

// Input: array elements

printf("Enter %d elements:\n", n);

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

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

// Perform Merge Sort

mergeSort(arr, 0, n - 1);

// Output: sorted array

printf("Sorted array using Merge Sort:\n");

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

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

printf("\n");

return 0;

}