**MERGE SORT**

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
  
void merge(int a[], int low, int mid, int high);  
void merge\_sort(int a[], int low, int high);  
  
int main() {  
    int i, n;  
    clock\_t start, end;  
    double time\_taken;  
     
    printf("Enter the number of elements: ");  
    scanf("%d", &n);  
     
    int \*a = (int \*)malloc(n \* sizeof(int));  
    if (a == NULL) {  
        printf("Memory allocation failed\n");  
        return 1;  
    }  
     
    printf("Enter the array elements: ");  
    for (i = 0; i < n; i++) {  
        scanf("%d", &a[i]);  
    }  
     
    start = clock();  
    merge\_sort(a, 0, n - 1);  
    end = clock();  
     
    time\_taken = (double)(end - start) / CLOCKS\_PER\_SEC;  
     
    printf("Sorted array: ");  
    for (i = 0; i < n; i++) {  
        printf("%d ", a[i]);  
    }  
    printf("\n");  
     
    printf("Time taken to sort: %f seconds\n", time\_taken);  
     
    free(a);  
    return 0;  
}  
  
void merge\_sort(int a[], int low, int high) {  
    if (low < high) {  
        int mid = (low + high) / 2;  
        merge\_sort(a, low, mid);  
        merge\_sort(a, mid + 1, high);  
        merge(a, low, mid, high);  
    }  
}

void merge(int a[], int low, int mid, int high) {  
    int i = low, j = mid + 1, k = 0;  
    int \*c = (int \*)malloc((high - low + 1) \* sizeof(int));  
    if (c == NULL) {  
        printf("Memory allocation failed\n");  
        exit(1);  
    }  
     
    while (i <= mid && j <= high) {  
        if (a[i] < a[j]) {  
            c[k++] = a[i++];  
        } else {  
            c[k++] = a[j++];  
        }  
    }  
     
    while (i <= mid) {  
        c[k++] = a[i++];  
    }  
     
    while (j <= high) {  
        c[k++] = a[j++];  
    }  
     
    for (i = 0; i < k; i++) {  
        a[low + i] = c[i];  
    }  
     
    free(c);  
}

**OUTPUT:**

