Input Code :

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// Function to merge two halves
void merge(int arr[], int temp[], int left, int mid, int right) {
    int i = left, j = mid + 1, k = left;

    // Merge the two halves into temp[]
    while (i <= mid && j <= right) {
        if (arr[i] <= arr[j]) {
            temp[k++] = arr[i++];
        } else {
            temp[k++] = arr[j++];
        }
    }

    // Copy the remaining elements of the left half (if any)
    while (i <= mid) {
        temp[k++] = arr[i++];
    }

    // Copy the remaining elements of the right half (if any)
    while (j <= right) {
        temp[k++] = arr[j++];
    }

    // Copy the sorted subarray into the original array
    for (i = left; i <= right; i++) {
        arr[i] = temp[i];
    }
}

// Function to implement merge sort
void mergeSort(int arr[], int temp[], int left, int right) {
    if (left < right) {
        int mid = (left + right) / 2;

        // Recursively divide the array into two halves
        mergeSort(arr, temp, left, mid);
        mergeSort(arr, temp, mid + 1, right);

        // Merge the sorted halves
        merge(arr, temp, left, mid, right);
    }
}

// Function to test merge sort with different dataset sizes
void testMergeSort(int sizes[], int numTests) {
    for (int i = 0; i < numTests; i++) {
        int size = sizes[i];

        // Allocate memory for the array and temporary array
        int *arr = (int*)malloc(size * sizeof(int));
        int *temp = (int*)malloc(size * sizeof(int));

        // Fill the array with random integers
        for (int j = 0; j < size; j++) {
            arr[j] = rand() % 100000; // Random integers between 0 and 99999
        }

        // Record start time
        clock_t start_time = clock();

        // Perform Merge Sort
        mergeSort(arr, temp, 0, size - 1);

        // Record end time
        clock_t end_time = clock();

        // Calculate time taken in seconds
        double time_taken = (double)(end_time - start_time) / CLOCKS_PER_SEC;

        // Output the results
        printf("Time taken to sort array of size %d: %.6f seconds\n", size, time_taken);

        // Free allocated memory
        free(arr);
        free(temp);
    }
}

int main() {
    // Sizes of datasets to test
    int dataset_sizes[] = {1000, 5000, 10000, 50000, 100000};
    int num_tests = sizeof(dataset_sizes) / sizeof(dataset_sizes[0]);

    // Seed the random number generator
    srand(time(NULL));

    // Run tests
    testMergeSort(dataset_sizes, num_tests);

    return 0;
}
```

Output :

```
Time taken to sort array of size 1000: 0.002134 seconds
Time taken to sort array of size 5000: 0.018547 seconds
Time taken to sort array of size 10000: 0.034289 seconds
Time taken to sort array of size 50000: 0.202395 seconds
Time taken to sort array of size 100000: 0.402938 seconds
```