11. Design and implement C/C++ Program to sort a given set of n integer elements using Merge Sort method and compute its time complexity. Run the program for varied values of n> 5000, and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

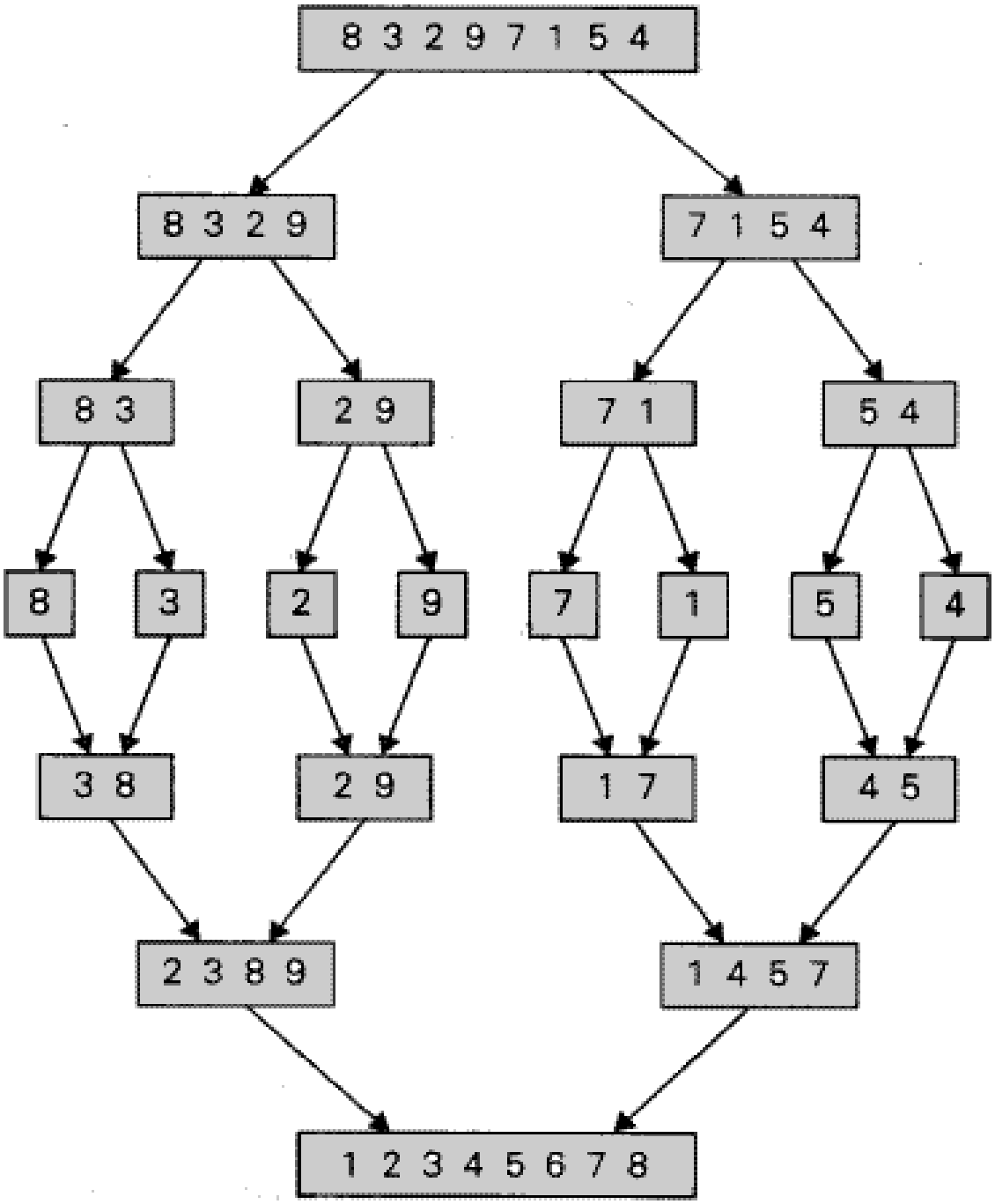
Algorithm :

```
Mergesort(A[0….n-1])
  // Input : An array A[0…n-1] of orderable elements
 // output : Array A[0…..n-1] sorted in non decreasing order

    if n>1
    copy A[0…n/2-1] to B[0……n/2-1]
    copy A[n/2…..n-1] to C[0……n/2-1]
    Mergesort(B[0….n/2-1])
    Mergesort(C[0….n/2-1])
    Merge(B,C,A)

Merge(B[0…..p-1],C[0…..q-1],A[0…p+q-1])
  // Input : Arrays B[0……p-1] and C[0…..q-1] both sorted
 // Output : Sorted array A[0….p+q-1] of the elements of B and C

    i←0; j←0; k←0
    while i<p and j<q do
    if B[i] ←C[j]
    A[k] ←B[i]; i<-i+1
    else
    A[k] ←C[j]; j<-j+1
    k←k+1
    if i=p
    copy C[j…q-1] to A[k……p+q-1]
    else
    copy B[i…p-1] to A[k……p+q-1]
```


8 3 2 9 7 1 5 4
8 3 2 9
7 1 5 4
8 3
2 9
7 1
5 4
8
3
2
9
7
1
5
4
3 8
2 9
1 7
4 5
2 3 8 9
1 4 5 7
1 2 3 4 5 6 7 8

Program:

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

void merge(int arr[], int l, int m, int r) {
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;

    int L[n1], R[n2];

    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];

    i = 0;
    j = 0;
    k = l;

    while (i < n1 && j < n2) {
        if (L[i] <= R[j]) {
            arr[k] = L[i];
            i++;
        } else {
            arr[k] = R[j];
            j++;
        }
        k++;
    }

    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = (l + (r - 1)) / 2;
```

```
        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);

        merge(arr, l, m, r);
    }
}

int main() {
    srand(time(NULL));
    int n = 10000;
    int elements[n];


    for (int i = 0; i < n; i++) {
        elements[i] = rand() % 1000;
    }

    printf("Enter the number of elements to sort: ");
    int num;
    scanf("%d", &num);

    if (num > n || num <= 0) {
        printf("Invalid input size. Please enter a number between 1 and %d\n", n);
        return 1;
    }

    int sort_arr[num];

    printf("Enter %d numbers:\n", num);
    for (int i = 0; i < num; i++) {
        scanf("%d", &sort_arr[i]);
    }


    clock_t start = clock();
    mergeSort(sort_arr, 0, num - 1);
    clock_t end = clock();


    printf("Sorted Array:\n");
    for (int i = 0; i < num; i++) {
        printf("%d ", sort_arr[i]);
    }
    printf("\nTotal time taken to sort the Input Array is: %1f seconds\n", ((double)(end - start))
/ CLOCKS_PER_SEC);


    for (int size = 500; size <= n; size += 500) {
        int arr_temp[size];
```

```
        for (int i = 0; i < size; i++) {
            arr_temp[i] = elements[i];
        }

        clock_t start = clock();
        mergeSort(arr_temp, 0, size - 1);
        clock_t end = clock();

        printf("Total time taken to sort %d elements is %1f seconds\n", size, ((double)(end -
start)) / CLOCKS_PER_SEC);
    }

    return 0;
}
```

OUTPUT:

```
Total time taken to sort 5000 elements  is 0.001586
Total time taken to sort 10000 elements  is 0.003362
Total time taken to sort 15000 elements  is 0.005369
Total time taken to sort 20000 elements  is 0.006922
Total time taken to sort 25000 elements  is 0.009068
Total time taken to sort 30000 elements  is 0.011076
Total time taken to sort 35000 elements  is 0.012611
Total time taken to sort 40000 elements  is 0.014396
Total time taken to sort 45000 elements  is 0.016468
Total time taken to sort 50000 elements  is 0.018744
Total time taken to sort 55000 elements  is 0.020832
Total time taken to sort 60000 elements  is 0.022949
Total time taken to sort 65000 elements  is 0.024718
Total time taken to sort 70000 elements  is 0.026224
Total time taken to sort 75000 elements  is 0.028027
Total time taken to sort 80000 elements  is 0.029840
Total time taken to sort 85000 elements  is 0.031654
Total time taken to sort 90000 elements  is 0.034210
Total time taken to sort 95000 elements  is 0.036619
```