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

Soln:

```
#include <stdio.h>

#include <stdlib.h>

#include <time.h>


void swap(int *xp, int *yp) {

    int temp = *xp;

    *xp = *yp;

    *yp = temp;

}


void genrandm(int arr[], int n) {

    int i;

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

        arr[i] = rand() % 10000;

    }

}


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 + j + 1];
}

i = 0;
j = 0;
k = l; // Initialize k to the start of the merge range

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

// Copy remaining elements of L[], if any
while (i < n1) {
    arr[k] = L[i];
    i++;
    k++;
}

// Copy remaining elements of R[], if any
while (j < n2) {
    arr[k] = R[j];
    j++;
    k++;
```

```
    }
}

void mergesort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r - l) / 2; // Correct calculation of midpoint
        mergesort(arr, l, m);
        mergesort(arr, m + 1, r);
        merge(arr, l, m, r);
    }
}

int main() {
    int n;
    double cpu_time_used, tottime = 0;
    srand(time(NULL)); // Seed for random number generation

    for (n = 5000; n <= 100000; n += 5000) {
        int* arr = (int*)malloc(n * sizeof(int));
        genrandm(arr, n);

        double start, end;
        start = clock();
        mergesort(arr, 0, n - 1);
        end = clock();

        cpu_time_used = ((double)(end - start)) / CLOCKS_PER_SEC;
        printf("Time taken to sort %d elements: %f seconds\n", n, cpu_time_used);
        tottime += cpu_time_used;
```

```
        free(arr);
    }


    printf("Total execution time = %f seconds\n", tottime);

//use this above line if you are running this line in vs code,else if you are running this code in codeblocks,it doesn’t required//

return 0;

}
```