Bangladesh Army International University of Science & Technology

Department of Computer Science and Engineering

Lab Report

Lab Report No	**01**						
Lab Report Name	**Sorting an Array Using Merge Sort**						
Course Title	**Computer Algorithms & Complexity Sessional**						
Course Code	**CSE 222**						
Name	Rayhan Alam Towhid						
ID	1119007						
Level	2	**Term**	II	**Section**	A	**Group**	G1
Date of Submission	12-10-2025			**Session**	Fall-2025		

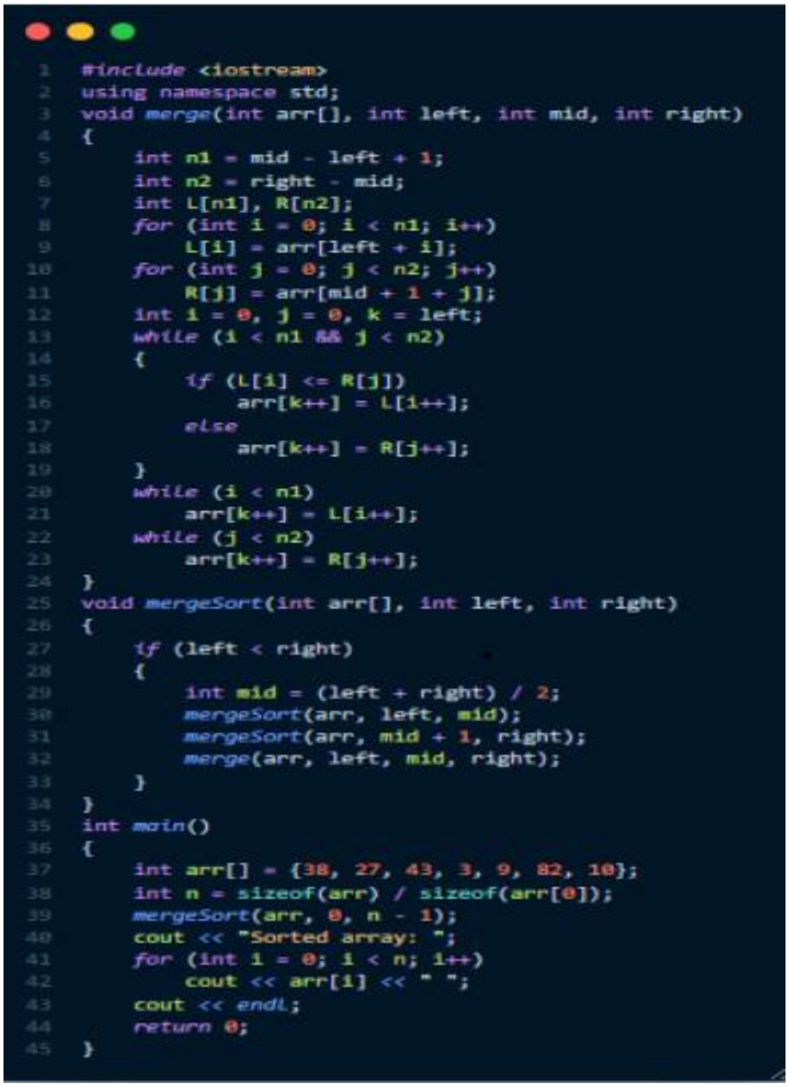
Marking Rubric:

Problem Understanding & Report Clarity (3)	**Implementation (5)**	**Results & Analysis (2)**	**Total (10)**

Key Learnings:

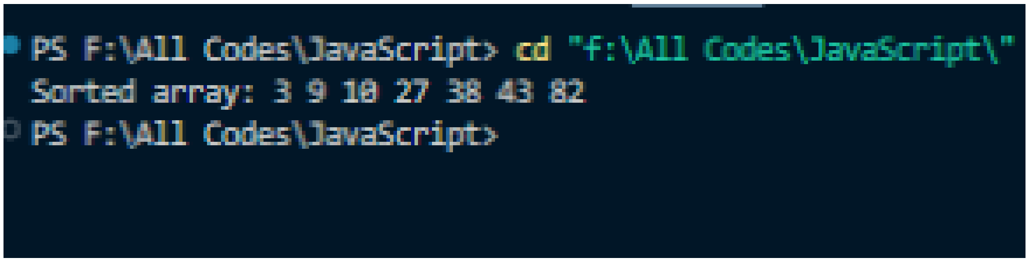
Merge Sort teaches divide and conquer by splitting an array into halves, sorting each half, and then **merging** them. You learn about stable sorting and handling recursion efficiently.

Code Implementation:



```cpp
#include <iostream>
using namespace std;
void merge(int arr[], int left, int mid, int right)
{
    int n1 = mid - left + 1;
    int n2 = right - mid;
    int L[n1], R[n2];
    for (int i = 0; i < n1; i++)
        L[i] = arr[left + i];
    for (int j = 0; j < n2; j++)
        R[j] = arr[mid + 1 + j];
    int 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++];
    }
    while (i < n1)
        arr[k++] = L[i++];
    while (j < n2)
        arr[k++] = R[j++];
}
void mergeSort(int arr[], int left, int right)
{
    if (left < right)
    {
        int mid = (left + right) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}
int main()
{
    int arr[] = {38, 27, 43, 3, 9, 82, 10};
    int n = sizeof(arr) / sizeof(arr[0]);
    mergeSort(arr, 0, n - 1);
    cout << "Sorted array: ";
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
    return 0;
}
```

Sample Input - Output:



```
PS F:\All Codes\JavaScript> cd "f:\All Codes\JavaScript\"
Sorted array: 3 9 10 27 38 43 82
PS F:\All Codes\JavaScript>
```

Result Analysis / Discussion:

Merge Sort works by recursively dividing the array into two halves until each part has one element, then merging them in sorted order. It is very reliable, stable, and has a consistent time complexity of O(n log n), making it efficient for large datasets.