॥वसुधैव कुटुम्बकम्॥

Symbiosis Institute of Technology

Faculty of Engineering
CSE - Academic Year 2023-24
Data Structures Lab Batch 2022-26

Lab Assignment 2

Name	Ayushi Kapgate
PRN no	22070122093
Batch	2022-26
Class	CS B1
Academic year & semester	2023-24
Date of submission	31/08/2023
Title of Assignment	Implement following sorting techniques and find the time complexity for merge sort.

Theory

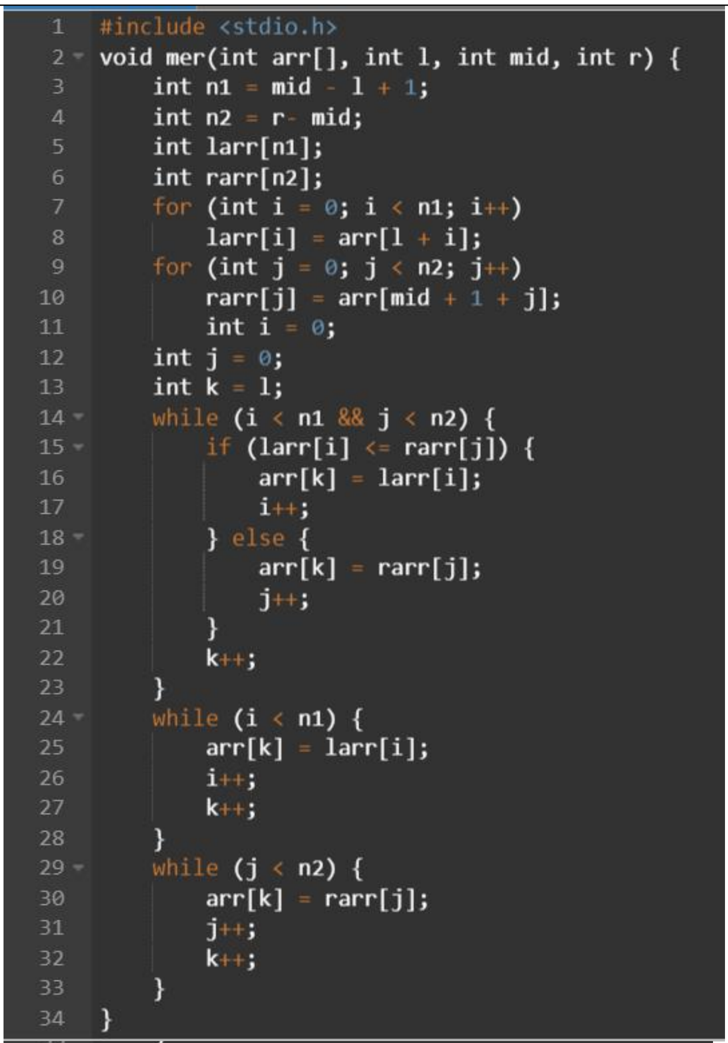
A table comparing the best case, average case, and worst case time complexities of Merge Sort.

Algorithm	**Best Case**	**Avg Case**	**Worst Case**
Merge sort	O(n log n)	O(n log n)	O(n log n)

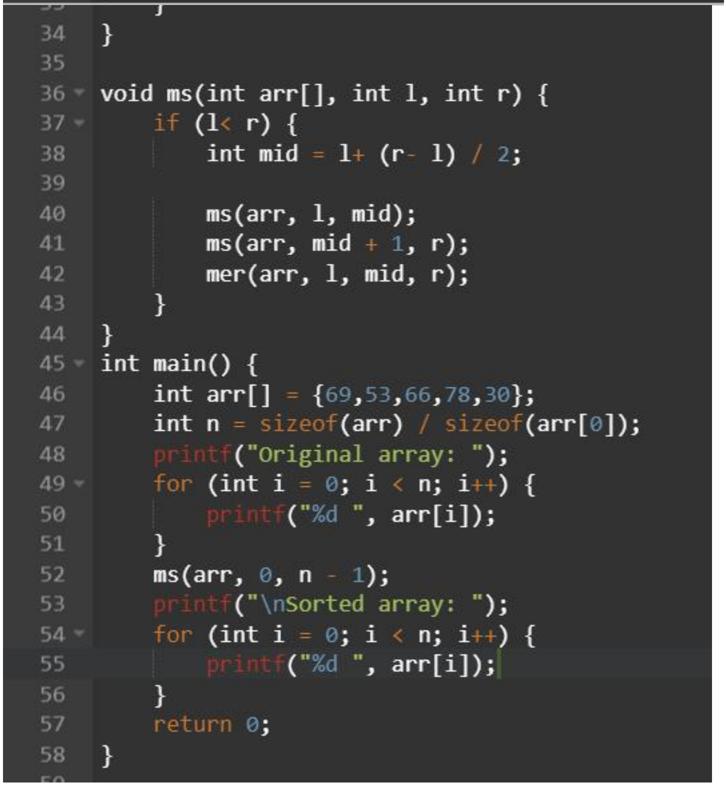
Best case and Worst case time complexities of merge sort.
Merge Sort:
Best Case: O(n log n) - The array is divided evenly at each step, leading to balanced merging.
Worst Case: O(n log n) - The array is divided unevenly at each step, still leading to efficient merging due to divide-and-conquer.

Source Code:

Merge Sort:



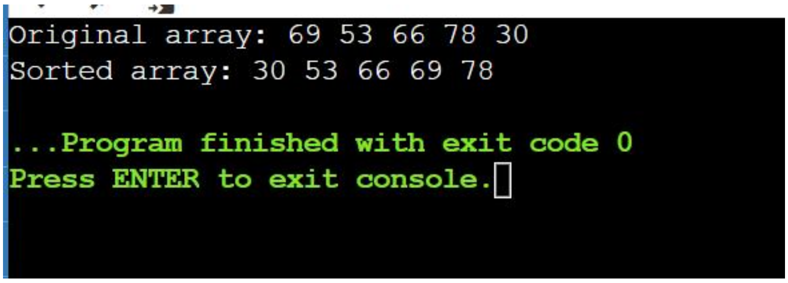
```c
#include <stdio.h>
void mer(int arr[], int l, int mid, int r) {
    int n1 = mid - l + 1;
    int n2 = r- mid;
    int larr[n1];
    int rarr[n2];
    for (int i = 0; i < n1; i++)
        larr[i] = arr[l + i];
    for (int j = 0; j < n2; j++)
        rarr[j] = arr[mid + 1 + j];
        int i = 0;
    int j = 0;
    int k = l;
    while (i < n1 && j < n2) {
        if (larr[i] <= rarr[j]) {
            arr[k] = larr[i];
            i++;
        } else {
            arr[k] = rarr[j];
            j++;
        }
        k++;
    }
    while (i < n1) {
        arr[k] = larr[i];
        i++;
        k++;
    }
    while (j < n2) {
        arr[k] = rarr[j];
        j++;
        k++;
    }
}
```



```c
}

void ms(int arr[], int l, int r) {
    if (l< r) {
        int mid = l+ (r- l) / 2;

        ms(arr, l, mid);
        ms(arr, mid + 1, r);
        mer(arr, l, mid, r);
    }
}
int main() {
    int arr[] = {69,53,66,78,30};
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Original array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    ms(arr, 0, n - 1);
    printf("\nSorted array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Output :	 Original array: 69 53 66 78 30 Sorted array: 30 53 66 69 78 ...Program finished with exit code 0 Press ENTER to exit console.
Conclusion:	Thus, we have studied merge sort algorithm and its time complexity.