

SEPTEMBER 1,2020

DESIGN AND ANALYSIS OF ALGORITHMS

(QUESTION: 2, MERGE SORT)
EXERCISE 2

SUBMITTED BY-

NAME - RAJ KRISHNA

ROLL NO - 1805233

BATCH - CSE-G1

GROUP - B

1. The objective of the Experiment

The objective of the experiment is to sort the numbers present in the given array using **Merge Sort**.

2. Solution Code

```
#include <bits/stdc++.h>
using namespace std;

int count=0;
void mergeSort(vector<int>& left, vector<int>& right, vector<int>& bars)
{
    int nL = left.size();
    int nR = right.size();
    int i = 0, j = 0, k = 0;

    while (j < nL && k < nR)
    {
        if (left[j] < right[k]) {
            bars[i] = left[j];
            j++;
            count++;
        }
        else {
            bars[i] = right[k];
            k++;
            count++;
        }
        i++;
    }
    while (j < nL) {
```

```

        bars[i] = left[j];
        j++; i++;
    }
    while (k < nR) {
        bars[i] = right[k];
        k++; i++;
    }
}

```

```

void sort(vector<int> & bar) {

    if (bar.size() <= 1) return;

    int mid = bar.size() / 2;
    vector<int> left;
    vector<int> right;

    for (size_t j = 0; j < mid; ++j){
        left.push_back(bar[j]);
    }

    for (size_t j = 0; j < (bar.size()) - mid; ++j){
        right.push_back(bar[mid + j]);
    }

    sort(left);
    sort(right);
    mergeSort(left, right, bar);
}

```

```

int main() {

    vector<int> nums { 7,9,2,11,19,17,12,5,7,12 };

```

```

    cout<<"Array initialisation "<<endl;

    for (size_t i = 0; i < nums.size(); ++i){
        cout<<" "<<nums[i];
    }

    cout<<endl;
    sort(nums);
    cout<<"After MergeSort- "<<endl;

    for (size_t i = 0; i < nums.size(); ++i){
        cout<<" "<<nums[i];
    }
    cout<<endl;
    cout<<"Number of comparison "<<count<<endl;
    return 0;
}

```

3. Summary of the program

Merge sort is a sorting technique based on **Divide and Conquer** technique. Using the **Divide and Conquer** technique, we divide a **problem** into **subproblems**. When the solution to each subproblem is ready, we 'combine' the results from the subproblems to solve the main problem. Merge sort first divides the array into equal halves and then combines them in a sorted manner.

Suppose we had to sort an array **nums**. A subproblem would be to sort a sub-section of this array starting at index **left** and ending at index **right**, denoted as **nums[left..right]**.

Divide

If **mid** is the half-way point between **left** and **right**, then we can split the subarray **nums**[left..right] into two arrays **nums**[left..mid] and **nums**[mid+1, right].

Conquer

In the conquer step, we try to sort both the subarrays **nums**[left..mid] and **nums**[mid+1, right]. If we haven't yet reached the base case, we again divide both these subarrays and try to sort them.

Combine

When the conquer step reaches the base step and we get two sorted subarrays **nums**[left..mid] and **nums**[mid+1, right] for array **nums**[left..right], we combine the results by creating a sorted array **nums**[left..right] from two sorted subarrays **nums**[left..mid] and **nums**[mid+1, right].

Merge sort is one of the efficient & fastest sorting algorithms with the following time complexity.

Best Case - $O(n \cdot \log_2 n)$

Average Case - $O(n \cdot \log_2 n)$

Worst Case - $O(n \cdot \log_2 n)$

4. Sample Output

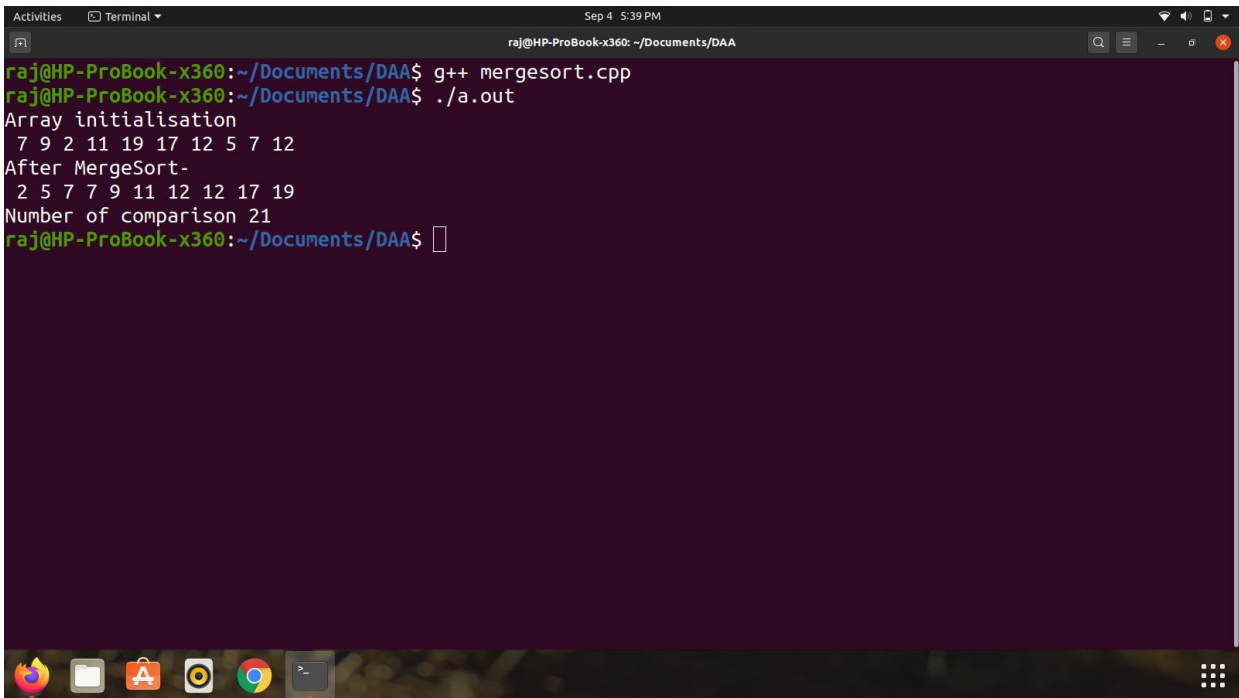
Array initialisation

7 9 2 11 19 17 12 5 7 12

After MergeSort-

2 5 7 7 9 11 12 12 17 19

Number of comparison 21



```
raj@HP-ProBook-x360: ~/Documents/DAA$ g++ mergesort.cpp
raj@HP-ProBook-x360: ~/Documents/DAA$ ./a.out
Array initialisation
7 9 2 11 19 17 12 5 7 12
After MergeSort-
2 5 7 7 9 11 12 12 17 19
Number of comparison 21
raj@HP-ProBook-x360: ~/Documents/DAA$
```

The screenshot shows a terminal window with a dark purple background. The user has compiled a C++ program named 'mergesort.cpp' using 'g++' and then executed it with './a.out'. The program's output is displayed in the terminal, showing the initial array, the array after MergeSort, and the total number of comparisons. The terminal window's title bar indicates the user is 'raj' on an 'HP-ProBook-x360' in the directory '~/Documents/DAA'. The system clock shows 'Sep 4 5:39 PM'. The bottom of the screen shows a standard Linux desktop environment with various application icons.