

Exp-3.4

Title:

Merge Sort Algorithm for Sorting an Unsorted Array

Aim:

To implement the Merge Sort algorithm and modify it to count the number of comparisons made during the sorting process, and print the count along with the sorted array.

Algorithm

1. Start.
2. If the array length is 1 or 0, return the array and zero comparisons.
3. Split the array into two halves.
4. Recursively apply Merge Sort on both halves, accumulating comparison counts.
5. Merge the two sorted halves while counting comparisons during merging.
6. Return the merged sorted array along with the total comparison count.
7. Stop.

Input:

Sorted array: 1,4,12,23,45,67,78,89

Number of comparisons: 15

Output:

Sorted array: 1,4,12,23,45,67,78,89

Number of comparisons: 15

Program:

```
def merge_sort_count(arr):
    if len(arr) <= 1:
        return arr, 0

    mid = len(arr) // 2
    left, left_count = merge_sort_count(arr[:mid])
    right, right_count = merge_sort_count(arr[mid:])

    merged, merge_comparisons = merge_count(left, right) # Renamed to avoid
    conflict
    total_count = left_count + right_count + merge_comparisons

    return merged, total_count

def merge_count(left, right):
    merged = []
    i = j = 0
    comparisons = 0

    while i < len(left) and j < len(right):
        comparisons += 1
        if left[i] < right[j]:
            merged.append(left[i])
            i += 1
        else:
            merged.append(right[j])
            j += 1
```

```
merged.extend(left[i:])  
merged.extend(right[j:])
```

```
return merged, comparisons
```

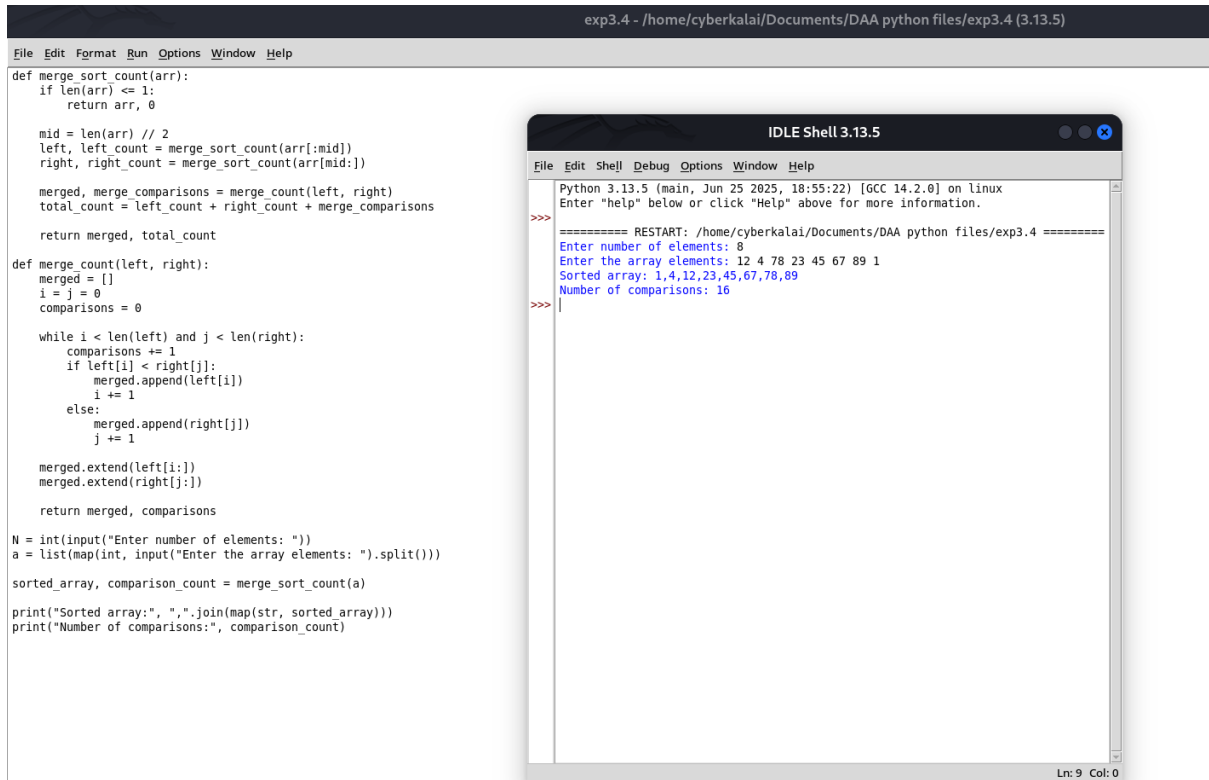
```
N = int(input("Enter number of elements: "))  
a = list(map(int, input("Enter the array elements: ").split()))  
  
sorted_array, comparison_count = merge_sort_count(a)  
  
print("Sorted array:", ", ".join(map(str, sorted_array)))  
print("Number of comparisons:", comparison_count)
```

Performance Analysis:

Time Complexity: $O(n \log n)$

Space Complexity: $O(n)$

Program Output:



The screenshot displays the Python IDLE Shell 3.13.5 interface. The main window shows the source code for a Merge Sort algorithm that counts comparisons. The code defines two functions: `merge_sort_count` and `merge_count`. `merge_sort_count` recursively splits the array and calls `merge_count` to merge the sub-arrays while counting comparisons. `merge_count` uses a while loop to compare elements from two sub-arrays and merge them into a new array. The main execution block takes user input for the number of elements and the array elements, sorts the array using the defined functions, and prints the sorted array and the total number of comparisons.

```
exp3.4 - /home/cyberkalai/Documents/DAA python files/exp3.4 (3.13.5)
File Edit Format Run Options Window Help
def merge_sort_count(arr):
    if len(arr) <= 1:
        return arr, 0

    mid = len(arr) // 2
    left, left_count = merge_sort_count(arr[:mid])
    right, right_count = merge_sort_count(arr[mid:])

    merged, merge_comparisons = merge_count(left, right)
    total_count = left_count + right_count + merge_comparisons

    return merged, total_count

def merge_count(left, right):
    merged = []
    i = j = 0
    comparisons = 0

    while i < len(left) and j < len(right):
        comparisons += 1
        if left[i] < right[j]:
            merged.append(left[i])
            i += 1
        else:
            merged.append(right[j])
            j += 1

    merged.extend(left[i:])
    merged.extend(right[j:])

    return merged, comparisons

N = int(input("Enter number of elements: "))
a = list(map(int, input("Enter the array elements: ").split()))

sorted_array, comparison_count = merge_sort_count(a)

print("Sorted array:", ", ".join(map(str, sorted_array)))
print("Number of comparisons:", comparison_count)
```

The IDLE Shell window shows the following output:

```
Python 3.13.5 (main, Jun 25 2025, 18:55:22) [GCC 14.2.0] on linux
Enter "help" below or click "Help" above for more information.
>>>
===== RESTART: /home/cyberkalai/Documents/DAA python files/exp3.4 =====
Enter number of elements: 8
Enter the array elements: 12 4 78 23 45 67 89 1
Sorted array: 1,4,12,23,45,67,78,89
Number of comparisons: 16
>>>
```

Ln: 9 Col: 0

Result:

Thus, the modified Merge Sort program executed successfully with comparison counting.