**Exp-3.3**

**Title:**

Merge Sort Algorithm for Sorting an Unsorted Array

**Aim:**

To design and implement a program using Merge Sort algorithm to sort an unsorted array in ascending order.

**Algorithm**

1. Start.
2. If the array has 1 or 0 elements, it is already sorted; return it.
3. Otherwise, divide the array into two halves.
4. Recursively apply Merge Sort on the two halves.
5. Merge the two sorted halves into a single sorted array.
6. Return the merged sorted array.
7. Stop.

**Algorithm**

1. Start.
2. If the array has 1 or 0 elements, it is already sorted; return it.
3. Otherwise, divide the array into two halves.
4. Recursively apply Merge Sort on the two halves.
5. Merge the two sorted halves into a single sorted array.
6. Return the merged sorted array.
7. Stop.

**Input:**

```
Enter number of elements: 8
Enter the array elements: 31 23 35 27 11 21 15 28
```

**Output:**

```
11,15,21,23,27,28,31,35
```

**Program:**

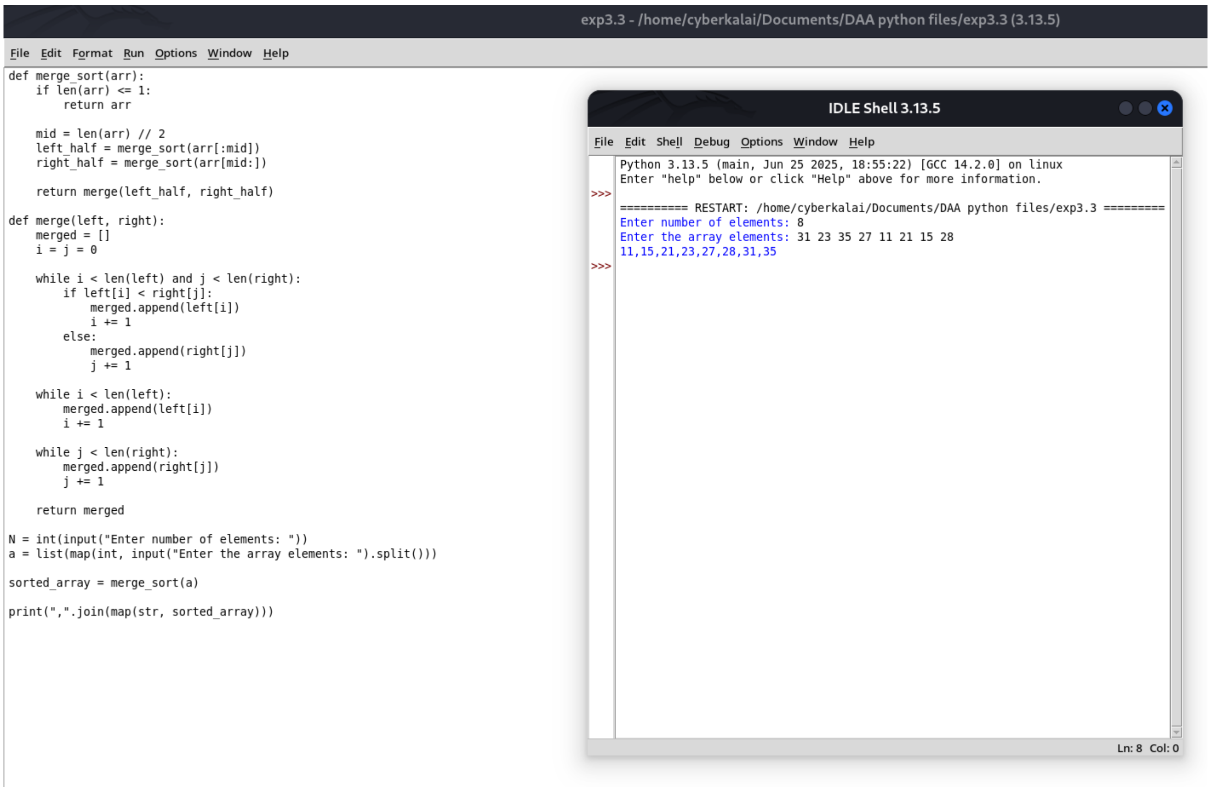
```
def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left_half = merge_sort(arr[:mid])
    right_half = merge_sort(arr[mid:])

    return merge(left_half, right_half)

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

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

    while i < len(left):
        merged.append(left[i])
        i += 1

    while j < len(right):
        merged.append(right[j])
        j += 1

    return merged

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

sorted_array = merge_sort(a)

print(",".join(map(str, sorted_array)))
```

**Performance Analysis:**

**Time Complexity: O(n log n)**

**Space Complexity: O(1)**

## Program Output:


exp3.3 - /home/cyberkalai/Documents/DAA python files/exp3.3 (3.13.5)
File Edit Format Run Options Window Help
IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Ln: 8 Col: 0

```python
def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left_half = merge_sort(arr[:mid])
    right_half = merge_sort(arr[mid:])

    return merge(left_half, right_half)

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

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

    while i < len(left):
        merged.append(left[i])
        i += 1

    while j < len(right):
        merged.append(right[j])
        j += 1

    return merged

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

sorted_array = merge_sort(a)

print(",".join(map(str, sorted_array)))
```

```
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/exp3.3 =========
Enter number of elements: 8
Enter the array elements: 31 23 35 27 11 21 15 28
11,15,21,23,27,28,31,35
>>>
```

## Result:

Thus, the Merge Sort program executed successfully and produced correct sorted outputs.