EX-1.10

Title :

Sort an array of integers in ascending order without using built-in functions, in O(n log n) time complexity.

Aim:

To design and implement a Python program to sort an array in ascending order using Merge Sort without built-in sort functions, achieving O(n log n) time complexity and minimal space complexity.

Procedure:

1. Read size n and array `nums`.
2. Implement the merge sort algorithm:
    - Recursively split the array into halves until single-element arrays.
    - Merge the sorted halves by comparing elements and building the sorted array.
3. Print the sorted array after sorting.

Algorithm:

1. Start
2. Read n and array `nums`.
3. If array length ≤ 1, return array.
4. Divide the array into two halves.
5. Recursively call merge sort on both halves.
6. Merge the two sorted halves:
    - Initialize empty array to hold merged results.
    - Compare front elements of both halves, append smaller to the merged array.
    - Continue until all elements are merged.
7. Return merged sorted array.
8. Print the sorted array.
9. Stop

Input:

7

12 11 13 5 6 7 1

Output:

1 5 6 7 11 12 13

Program :

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

    # Merge sorted left and right arrays
    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

    # Append remaining elements, if any
    while i < len(left):
        merged.append(left[i])
        i += 1
    while j < len(right):
        merged.append(right[j])
        j += 1

    return merged

def mergeSort(arr):
```

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

    mid = len(arr) // 2
    left = mergeSort(arr[:mid])
    right = mergeSort(arr[mid:])

    return merge(left, right)

# Taking user input
n = int(input("Enter size of array: "))
arr = list(map(int, input("Enter array elements: ").split()))

sorted_arr = mergeSort(arr)
print("Sorted array:", ' '.join(map(str, sorted_arr)))
```

Performance Analysis:

Time Complexity: O(n log n)

Space Complexity: O(n)

program output:

File Edit Format Run Options Window Help

```
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


def merge_sort(nums):
    if len(nums) <= 1:
        return nums

    mid = len(nums) // 2
    left = merge_sort(nums[:mid])
    right = merge_sort(nums[mid:])

    return merge(left, right)


nums = list(map(int, input("Enter elements of the array (space separated): "

sorted_nums = merge_sort(nums)
print("Sorted array:", sorted_nums)
```

IDLE Shell 3.13.5

File Edit Shell Debug Options Window Help

```
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 experiments files/exp1.10 ======
Enter elements of the array (space separated): 64 34 25 12 22 11 90
Sorted array: [11, 12, 22, 25, 34, 64, 90]
>>>
```

Ln: 7 Col: 0

Result :

Thus the given program Merge Sort is executed and got output successfully.