

```
def merge_sort(arr):  
    if len(arr) > 1:  
        mid = len(arr) // 2 # Find the middle of the array  
        left_half = arr[:mid] # Divide the array into two halves  
        right_half = arr[mid:]  
        merge_sort(left_half) # Sort the first half  
        merge_sort(right_half) # Sort the second half  
        i = j = k = 0  
        # Merge the sorted halves  
        while i < len(left_half) and j < len(right_half):  
            if left_half[i] < right_half[j]:  
                arr[k] = left_half[i]  
                i += 1  
            else:  
                arr[k] = right_half[j]  
                j += 1  
            k += 1  
        # Check if any element was left  
        while i < len(left_half):  
            arr[k] = left_half[i]  
            i += 1  
            k += 1  
        while j < len(right_half):  
            arr[k] = right_half[j]  
            j += 1  
            k += 1  
    # Example usage  
    arr = [38, 27, 43, 3, 9, 82, 10]  
    merge_sort(arr)  
    print("Sorted array:", arr)
```

☰

←

🔍 Ctrl + K

🗨️

📱

👤 TAMMISETTI AKHIL

main.py

+

🤖 AI

NEW

PYTHON ▾

RUN ▶

⋮

```
1 def merge_sort(arr):
2     if len(arr) > 1:
3         mid = len(arr) // 2 # Find the middle of the array
4         left_half = arr[:mid] # Divide the array into two halves
5         right_half = arr[mid:]
6
7         merge_sort(left_half) # Sort the first half
8         merge_sort(right_half) # Sort the second half
9
10        i = j = k = 0
11
12        # Merge the sorted halves
13        while i < len(left_half) and j < len(right_half):
14            if left_half[i] < right_half[j]:
15                arr[k] = left_half[i]
16                i += 1
17            else:
18                arr[k] = right_half[j]
19                j += 1
20                k += 1
21
22        # Check if any element was left
23        while i < len(left_half):
24            arr[k] = left_half[i]
25            i += 1
26            k += 1
27
28        while j < len(right_half):
29            arr[k] = right_half[j]
30            j += 1
31            k += 1
32
```

STDIN

Input for the program (Optional)

Output:

Sorted array: [3, 9, 10, 27, 38, 43, 82]

New

History

Save