**ASSIGNMENT NUMBER 2**

**Name: Abhishek Santosh Gaikwad**

**Class: TY(A) Roll no.: 371017**

**PRN: 22210975 Subject: DAA**

**SOURCE CODE:**

########## Quick Sort ##########

def partition(arr, s, e):

    # Suppose the pivot element is arr[s]

    pivot = arr[s]

    cnt = 0  # Count to store the number of elements less than the pivot element

    # Count elements less than or equal to the pivot

    for i in range(s + 1, e + 1):

        if arr[i] <= pivot:

            cnt += 1

    # Calculating the correct pivot index

    pivotIndex = s + cnt

    # Swapping pivot element with element at pivotIndex

    arr[s], arr[pivotIndex] = arr[pivotIndex], arr[s]

    # Rearrange elements on either side of the pivot

    i, j = s, e

    while i < pivotIndex and j > pivotIndex:

        while i < pivotIndex and arr[i] <= pivot:

            i += 1

        while j > pivotIndex and arr[j] >= pivot:

            j -= 1

        if i < pivotIndex and j > pivotIndex:

            arr[i], arr[j] = arr[j], arr[i]

            i += 1

            j -= 1

    return pivotIndex

def quickSort(arr, s, e):

    # Base condition

    if s < e:

        # Index of pivot element

        p = partition(arr, s, e)

        # Recursively sorting left part

        quickSort(arr, s, p - 1)

        # Recursively sorting right part

        quickSort(arr, p + 1, e)

########## Merge Sort ##########

def mergeSort(arr, s, e):

    # Base case for recursion

    if s < e:

        # Finding the middle index

        mid = (s + e) // 2

        # Recursively dividing the array

        mergeSort(arr, s, mid)

        mergeSort(arr, mid + 1, e)

        # Create two subparts of the main array

        left = arr[s:mid + 1]

        right = arr[mid + 1:e + 1]

        i, j, k = 0, 0, s

        # Merging the sorted subarrays

        while i < len(left) and j < len(right):

            if left[i] < right[j]:

                arr[k] = left[i]

                i += 1

            else:

                arr[k] = right[j]

                j += 1

            k += 1

        # Copy any remaining elements of left subarray

        while i < len(left):

            arr[k] = left[i]

            i += 1

            k += 1

        # Copy any remaining elements of right subarray

        while j < len(right):

            arr[k] = right[j]

            j += 1

            k += 1

# Take array as input

size = int(input("Enter the size of array: "))

arr = []  # Initializing an empty list for array input

for i in range(size):

    element = int(input("Enter the element: "))

    arr.append(element)  # Using append to add elements to the list

# Perform Quick Sort

quick\_sorted\_arr = arr.copy()

quickSort(quick\_sorted\_arr, 0, size - 1)

print("Array after Quick Sort:", quick\_sorted\_arr)

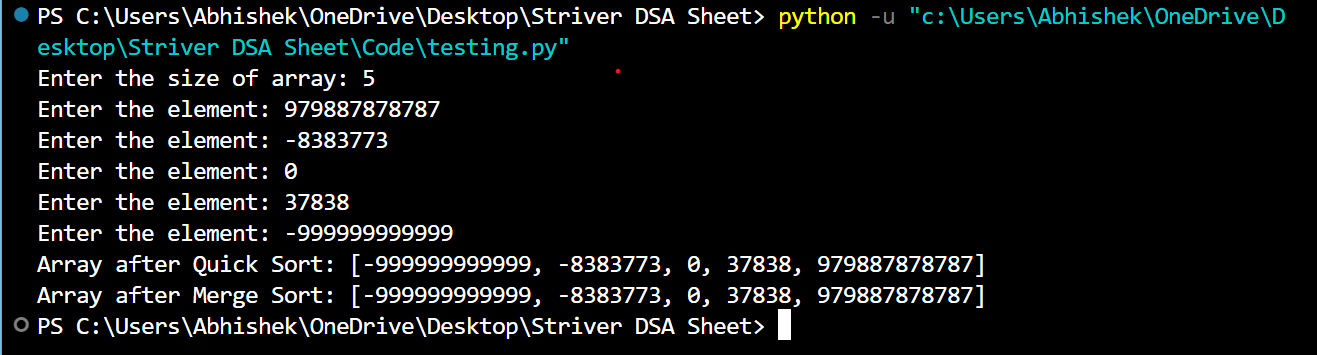
# Perform Merge Sort

merge\_sorted\_arr = arr.copy()

mergeSort(merge\_sorted\_arr, 0, size - 1)

print("Array after Merge Sort:", merge\_sorted\_arr)

**OUTPUT**

****