C++ Assignment: Implementing Merge Sort

This assignment focuses on understanding and implementing the **Merge Sort algorithm** in C++. You will learn how to divide an array into smaller subarrays, sort them, and merge the sorted subarrays into a single sorted array.

Objective

1. Understand the **divide-and-conquer** technique used in Merge Sort.
2. Implement recursive functions to split and merge arrays.
3. Practice sorting arrays using efficient algorithms.

Problem Statement

Write a C++ program to sort an array of integers using the **Merge Sort algorithm**. The program should:

1. Take input for the size of the array and its elements from the user.
2. Sort the array using Merge Sort.
3. Display the sorted array.

Key Concepts to Apply

1. **Divide-and-Conquer**:
   - Divide the array into two halves.
   - Recursively sort each half.
   - Merge the two sorted halves.
2. **Merge Function**:
   - Combine two sorted subarrays into a single sorted array.
3. **Recursion**:
   - Use recursion to break the array into smaller parts until each part has one element.

Program Requirements

1. Use a **recursive approach** to implement Merge Sort.
2. Implement a **merge function** to combine sorted subarrays.

3. The program must work for an array of **positive and negative integers**.
4. Do **not use STL containers or functions (e.g., std::vector)**.
5. Use **only arrays** for storing and processing data.

Instructions

1. **Define the Merge Sort Algorithm**:
   - Create a recursive function mergeSort that divides the array into two halves.
   - Call the merge function to combine the sorted halves.
2. **Merge Function**:
   - Write a merge function to combine two sorted subarrays.
   - Ensure the function handles duplicate values correctly.
3. **Input and Output**:
   - Input:
     - An integer for the size of the array.
     - Array elements entered by the user.
   - Output:
     - Display the sorted array.
4. **Constraints**:
   - The array size should be between 1 and 100.
   - The program should handle invalid input gracefully (e.g., non-integer values).

Steps to Implement

1. **Write the Input Function**:
   - Prompt the user to input the array size and elements.
2. **Implement Merge Sort**:
   - Write the recursive mergeSort function that splits the array into smaller subarrays.
   - Use recursion to sort the subarrays.

3. **Write the Merge Function**:
    - Combine two sorted subarrays into a single sorted array.
4. **Display the Results**:
    - Output the sorted array.

Grading Criteria

- **Code Functionality (50%)**:
    - Correct implementation of Merge Sort and its components.
- **Code Clarity (20%)**:
    - Use of meaningful variable names and clear function definitions.
- **Input Handling (10%)**:
    - Properly handles edge cases (e.g., empty array, invalid input).
- **Output (10%)**:
    - Accurate and neatly formatted output.
- **Documentation (10%)**:
    - Use comments to explain the logic and steps in the code.

Submission Instructions

1. Save your program in a file named .cpp.
2. Provide the following in your submission:
    - The complete code with comments.
    - A sample input-output screenshot of the program execution.
3. Submit your assignment before the deadline through the designated platform.

Resources

- Refer to your course material or textbook for Merge Sort algorithms.

- For any doubts, reach out during office hours.
- Good luck with your assignment! 🚀

BONUS:

Implement merge sort using linked list.