Garbage Value Fix

bubble sort algorithm in cpp

Here's the **Bubble Sort** algorithm implemented in C++:

🔹 Bubble Sort Explanation:

- Bubble Sort repeatedly **swaps adjacent elements** if they are in the wrong order.
- The largest elements "bubble" to the right with each pass.
- It has **$O(n^2)$** worst-case complexity, making it inefficient for large datasets.

✅ C++ Code for Bubble Sort

cpp

```cpp
`#include <bits/stdc++.h>
using namespace std;

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        bool swapped = false; // Optimization: If no swaps, array is sorted
        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                swap(arr[j], arr[j + 1]);
                swapped = true;
            }
        }
        if (!swapped) break; // Exit if no swaps occurred
    }
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main() {
    int n;
    cout << "Enter array size:\n> ";
    cin >> n;
    int arr[n];

    cout << "Enter array elements:\n> ";
    for (int i = 0; i < n; i++)
        cin >> arr[i];

    cout << "\nOriginal Array: ";
    printArray(arr, n);

    bubbleSort(arr, n);

    cout << "\nSorted Array: ";
    printArray(arr, n);

    return 0;
}
`
```

📌 Optimizations Used:

1. **Swapped Flag**: If no swaps occur in a pass, the array is already sorted, reducing unnecessary iterations.
2. `` `n - i - 1` ``: Avoids checking already sorted elements in the last positions.

🔥 Time Complexity:

- **Best Case (Sorted Input):** `` `O(n)` `` (due to the swapped flag)
- **Worst/Average Case:** `` `O(n²)` ``

Let me know if you need modifications! 🚀