📘 Erasing Multiple Items from a `std::vector` in a Loop

In C++, `std::vector` provides the `erase()` method to remove elements. However, **erasing multiple elements in a loop** requires careful handling to avoid **iterator invalidation** and **skipped elements**.

🔹 The Problem: Iterator Invalidation

When you erase an element from a vector using `erase()`, all iterators **after the erased element** become invalid. If you're iterating through the vector and erasing elements, this can lead to:

- **Accessing invalid memory**
- **Skipping elements unintentionally**
- **Undefined behavior**

🔸 Safe Pattern for Erasing in a Loop

To safely erase multiple elements, use a `for` loop with an iterator, *only increment the iterator if no erasure occurs*.

✅ Example: Remove All Even Numbers

```cpp
#include <iostream>
#include <vector>

int main() {
    std::vector<int> nums = {1, 2, 3, 4, 5, 6, 7, 8};

    for (auto it = nums.begin(); it != nums.end(); ) {
        if (*it % 2 == 0) {
            it = nums.erase(it);  // erase returns the next valid iterator
        } else {
            ++it;  // only increment if no erase
        }
    }

    // Output the result
    for (int n : nums) {
        std::cout << n << " ";
    }
    return 0;
}
```

🧠 Key Insight:

- `erase(it)` removes the element and returns a **valid iterator** pointing to the next element.
- If you increment the iterator after `erase()`, you skip the next element.

🔹 Alternative: Reverse Iteration

If you're removing elements based on **index** or **position**, iterating **backward** can be safer:

```cpp
for (int i = nums.size() - 1; i >= 0; --i) {
    if (nums[i] % 2 == 0) {
        nums.erase(nums.begin() + i);
    }
}
```

✅ Why This Works:

- Erasing from the end avoids invalidating earlier indices.
- No iterator invalidation occurs for elements before the erased one.

🧠 Best Practices

- Prefer **iterator-based loops** when working with `erase()`.
- Avoid `range-based for` loops when modifying the container.

🎯 Quiz!

Here's a short quiz on the topic: quiz

🔭 Markdown Viewer

How to view the markdown files in a browser...

- Markdown Viewer

🧠 Lecture Practices

Here are the lecture Practices...

- Day 1
- Day 2
- Day 3

🔍 Lecture Quizzes

Here are the lecture quizzes...

- Day 1
- Day 2
- Day 3

⚙️ Weekly Topics

Here are the topics for the week...

- Calling Methods
- Calling Methods 2
- Creating Methods
- Iterators
- Vectors
- References
- Const
- Erasing in a Loop
- Default Parameters