Deleting in STL Functions — erase, remove, remove_if Explained

1. erase()

- Belongs to containers (vector, list, map, etc.).
- Physically removes elements and changes container size.

Example:
vector v = {1, 2, 3, 4, 5};
v.erase(v.begin() + 2); // removes the 3rd element
Result → v = {1, 2, 4, 5}

2. remove()

- From header (not a container function).
- Moves unwanted elements to the end, does NOT change size.

Example:
vector v = {1, 2, 3, 2, 4, 2, 5};
auto it = remove(v.begin(), v.end(), 2);
v.erase(it, v.end()); // Erase–Remove Idiom
Result → v = {1, 3, 4, 5}

3. remove_if()

- Removes elements matching a condition (predicate).

Example:
vector v = {1, 2, 3, 4, 5, 6};
auto it = remove_if(v.begin(), v.end(), [](int x){ return x % 2 == 0; });
v.erase(it, v.end());
Result → v = {1, 3, 5}

4. erase() with range

Removes a range of elements.
Example:
vector v = {1, 2, 3, 4, 5};
v.erase(v.begin() + 1, v.begin() + 4);
Result → v = {1, 5}

5. erase() in list, set, map

- list has its own remove() that actually erases.
list l = {1, 2, 3, 2, 4, 2};
l.remove(2); // removes all 2’s

- set/map erase elements directly.
set s = {1, 2, 3, 4};
s.erase(2);

Summary Table

Function	Header	Affects Size	Works On	What It Does
erase(pos)	Container	Yes	vector, list, map	Physically removes elements
erase(first,last)	Container	Yes	vector, list	Removes range
remove()	<algorithm>	No	vector, array	Moves unwanted elements to end
remove_if()	<algorithm>	No	vector, array	Moves elements matching condition
erase(it,end)	Container	Yes	vector, list	Used with remove() to delete elements
list::remove()	Container	Yes	list	Actually deletes matching elements

■■ *Common Pitfall*

This will NOT delete elements (vector has no remove()):

```
v.remove(2); // ■ error
```

Correct way:

```
v.erase(remove(v.begin(), v.end(), 2), v.end());
```