Generating and Manipulating Collections

Kate Gregory

@gregcons www.gregcons.com/kateblog

Copying

operator=

You could write a loop

copy()

copy_if()

copy_n()

copy_backward()

Collection of Noncopyable Types

move()

move_backward()

There is no move_if()
or move_n()

Removing Elements

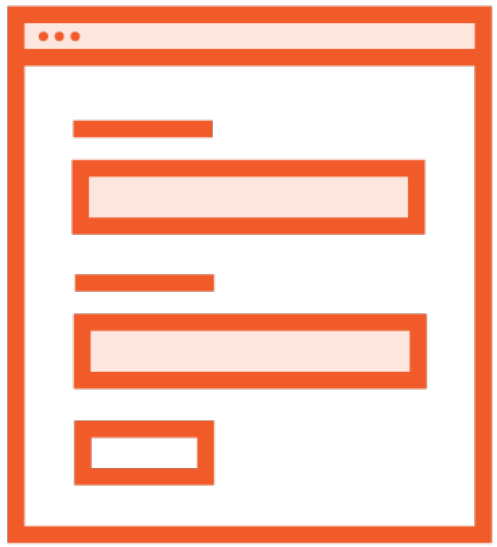
remove()

remove_if()

Does not shrink the container or change values past the new end

Use erase() when you're finished

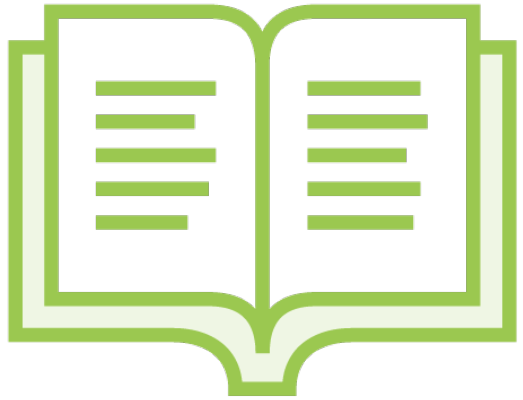
Creating and Filling Collections



Non empty collections default-initialize elements

fill(), fill_n()

iota()



generate(), generate_n()

Replacing Values

replace()

replace_if()

Transform

Do something to every element in a range

Put the result back into the same collection

Or into a different one

Or can work pairwise with elements

Eliminating Duplicates

sort(), adjacent_find(), remove()

sort(), unique()

sort(), copy_if() or transform(), cleanup

sort(), unique_copy()

Reversing and Swapping

reverse()

iter_swap()

reverse_copy()

Summary

You still shouldn't be writing raw loops

Whether copying, removing, or generating entire collections

- There's a function for that

Replace any value with another

Transform is incredibly powerful

Removing duplicates, reversing, swapping elements

- Having a name helps express your intent
- Edge cases (empty collections, overlapping ranges, and so on) are handled and tested already