From Mathematics to Generic Programming

Brooks Mershon

March 2017

11.12

Solution.

We are essentially checking in the `while()` loop that our iterators have not run into one another. This check involves first ensuring that (1) the iterators are not already converged onto the same element either at the beginning of the `rotate()` function, or after one or more iterations of the `while()` loop; this can occur if the number of elements in our range is odd. But consider what happens when the number of elements in the range is even; we must then check that (2) the increment of iterator `f` and decrement of iterator `l` will not cause these two to pass one another!

Here we see a general heuristic as to how one might quickly arrive at an answer to the question: *Explain why we need two tests per iteration in the preceding* `while` loop. One might think there is an isomorphism between the two possibilities of either an even or odd number of elements in the range to reverse and the two tests are performed on each iteration. If it turns out that there is only a coincidence, rather than an isomorphism, then surely *putting two and two together* cannot hurt when we are first attempting to answer the question.