【C++】Day ten

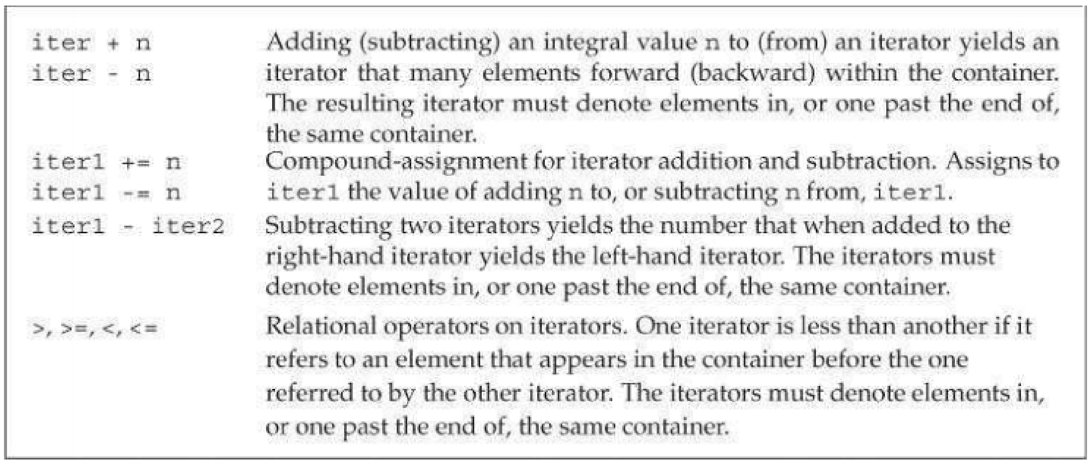
Class	C++
Date	@November 28, 2021
Material	
Series Number	
Summary	Iterator Arithmetic

【Ch3】Iterator Arithmetic

3.4.2 Iterator Arithmetic

Iterators for string and vector support additional operations that can move an iterator multiple lements at a time.They also support relational operators.

Table 3.7. Operations Supported by vector and string Iterators



`iter + n`
`iter - n`
	Adding (subtracting) an integral value n to (from) an iterator yields an iterator that many elements forward (backward) within the container. The resulting iterator must denote elements in, or one past the end of, the same container.

`iter1 += n`
`iter1 -= n`
	Compound-assignment for iterator addition and subtraction. Assigns to `iter1` the value of adding n to, or subtracting n from, `iter1`.

`iter1 - iter2`	Subtracting two iterators yields the number that when added to the right-hand iterator yields the left-hand iterator. The iterators must denote elements in, or one past the end of, the same container.
`>, >=, <, <=`	Relational operators on iterators. One iterator is less than another if it refers to an element that appears in the container before the one referred to by the other iterator. The iterators must denote elements in, or one past the end of, the same container.

We can add(or subtract) an integral value and an iterator. Doing so returns an iterator positioned forward(or backward) that many elements. When we add or subtract an integral value and an iterator, the result must denote an element in the same vector(or string) or denote one past the end of the associated vector(or string).

As an example, we can compute an iterator to the element nearest the middle of a vector:

```
//compute an iterator to the element closest to the midpoint of vi
auto mid = vi.begin() + vi.size()/2;
```

In addition to comparing two iterators for equality, we can compare vector and string iterators using the relational operators(<, <=, > ,>=). The iterators must be valid and must denote elements in(or one past the end of) the same vector or string.

For example, assuming it is an iterator into the same vector as mid, we can check whether it dentoes an element before or after mid as follows:

```
if(it < mid)
  //process elements in the first half of vi
```

We can also subtract two iterators so long as they refer to elements in, or one off the end of, the same vector or string. The result is the distance between the iterators. By distance we mean the amount by which we'd have to change one iterator to get the other. The result type is a signed integral type named `difference_type`. Both vector and string define `difference_type`. This type is signed, because subtraction might have a negative result.

Note: We cannot add two iterators together