Categories

Iterators can be categorized into three primary types, each with its own advantages.

Iterators can be put into three categories according to their capabilities. C++ has forward, bidirectional, and random access iterators. With the forward iterator, we can iterate the container forward, with the bidirectional iterator, in both directions. With the random access iterator, we can directly access an arbitrary element. In particular, this means for the last one, we can use iterator arithmetic and ordering comparisons (e.g.: `<`). The category of an iterator depends on the type of container used.

The table below is a representation of containers and their iterator categories. The bidirectional iterator includes forward iterator functionalities, and the random access iterator includes the forward and bidirectional iterator functionalities. `It` and `It2` are iterators, `n` is a natural number.

Iterator category	Properties	Container
Forward iterator	`++It, It++, *It`	unordered associative container
	`It == It2, It != It2`	`std::forward_list`
Bidirectional iterator	`--It, It--`	ordered associative container
		`std::list`
Random access iterator	`It[i]`	`std::array`
	`It+= n, It - n`	`std::vector`

	`It+= n, It-= n`	`std::vector`
	`It+n, It-n`	`std::deque`
	`n+It`	`std::string`
	`It-It2`	
	`It < It2, It <= It2, It > It2`	
	`It >= It2`	

The iterator categories of the container

The input iterator and the output iterator are special forward iterators: they can read and write their pointed element only once.

In the next lesson, we'll discuss how a map creates and handles its iterator.