

핵심 정리

- iterator_traits

1. 반복자타입::value_type

- 반복자가 가리키는 요소의 타입.

핵심 정리

- iterator_traits

1. 반복자의 2가지 형태

① User Define Type으로 만들어진 반복자 - value_type 이 있다.

② Raw Pointer - value_type 이 없다.

Raw Pointer 안에는 "Member Type" 이 없기 때문에 알고리즘 함수를 만들 때 문제가 생기게 된다.

🔨 핵심 정리

- iterator_traits

1. iterator_traits

- <iterator>
- Raw Pointer 에는 "Member Type" 이 없다는 문제를 해결하기 위한 도구.
- 반복자의 value_type이 필요할 때, iterator_traits 를 통해서 value_type 을 사용한다.

2. value_type을 사용하는 2가지 방식

- ① T::value_type : T 가 Raw Pointer 라면 error.
- ② iterator_traits<T>::value_type : T 가 Raw Pointer 라도 문제 없음.

🔨 핵심 정리

• iterator_traits

반복자가 가리키는
타입이 필요.

`T::value_type`

OK

객체형 반복자

using `value_type` = T;

ERROR

Raw Pointer

반복자가 가리키는
타입이 필요.

`iterator_traits<T*>::
value_type`

OK

iterator_traits

`value_type` = T::value_type;

OK

iterator_traits<T*>

`value_type` = T;

객체형 반복자

using `value_type` = T;

Raw Pointer

Member type	Definition
value_type	T
difference_type	Signed integer type (usually std::ptrdiff_t)
reference	value_type&
pointer	value_type*
iterator_category	컨테이너 마다 다른 정의

```
template<class T> struct iterator_traits
{
    using iterator_category = typename T::iterator_category;
    using value_type = typename T::value_type;
    using difference_type = typename T::difference_type;
    using pointer = typename T::pointer;
    using reference = typename T::reference;
};
```

```
template<class T> struct iterator_traits<T*>
{
    using iterator_category = random_access_iterator_tag;
    using value_type = T;
    using difference_type = std::ptrdiff_t;
    using pointer = T*;
    using reference = T&;
};
```

핵심 정리

- `iterator_traits`

1. `value_type` 대신에 `auto/decltype` 을 사용할 경우

- `type deduction` 규칙을 정확히 알고 사용해야 한다. - “C++ 중급 과정”

참고

핵심 정리

- C++17 과 iterator_traits

1. C++17 class template type deduction guide