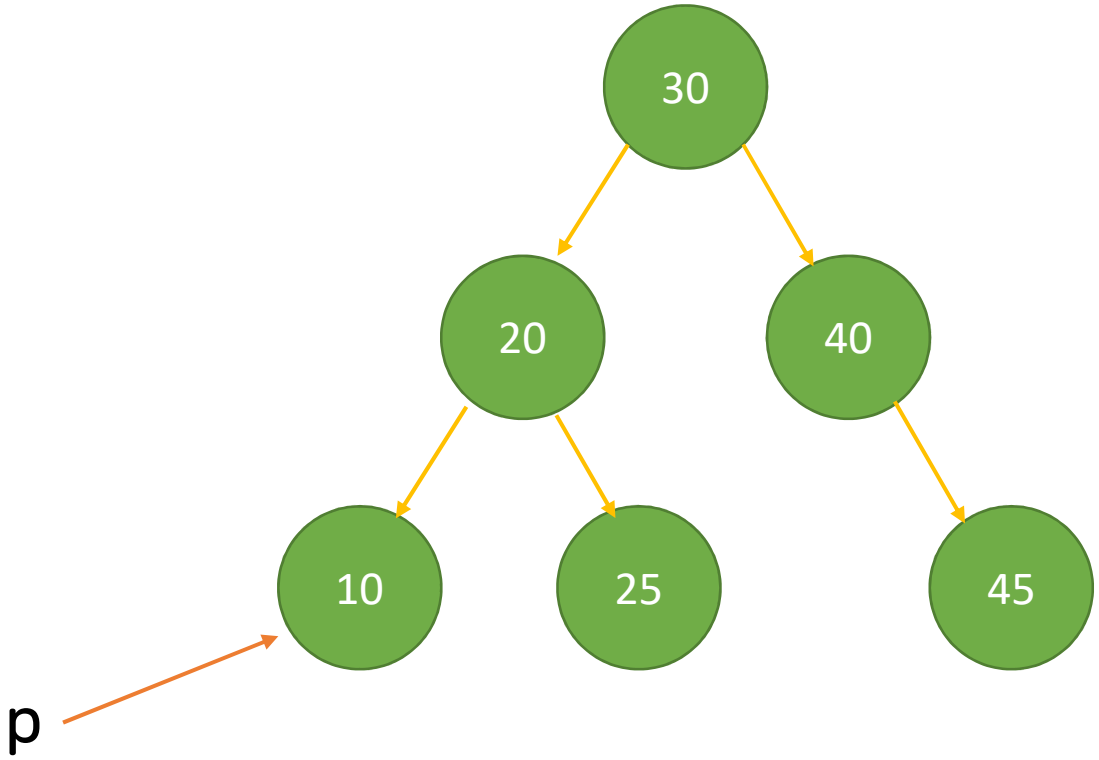


C++ 표준은 세부 구현을 정의 하지 않는다.

- set 를 구현하기 위해 어떤 자료 구조를 사용해도 상관없다. 하지만, 대부분의 STL 구현은 RB Tree 로 구현한다.

set template 모양

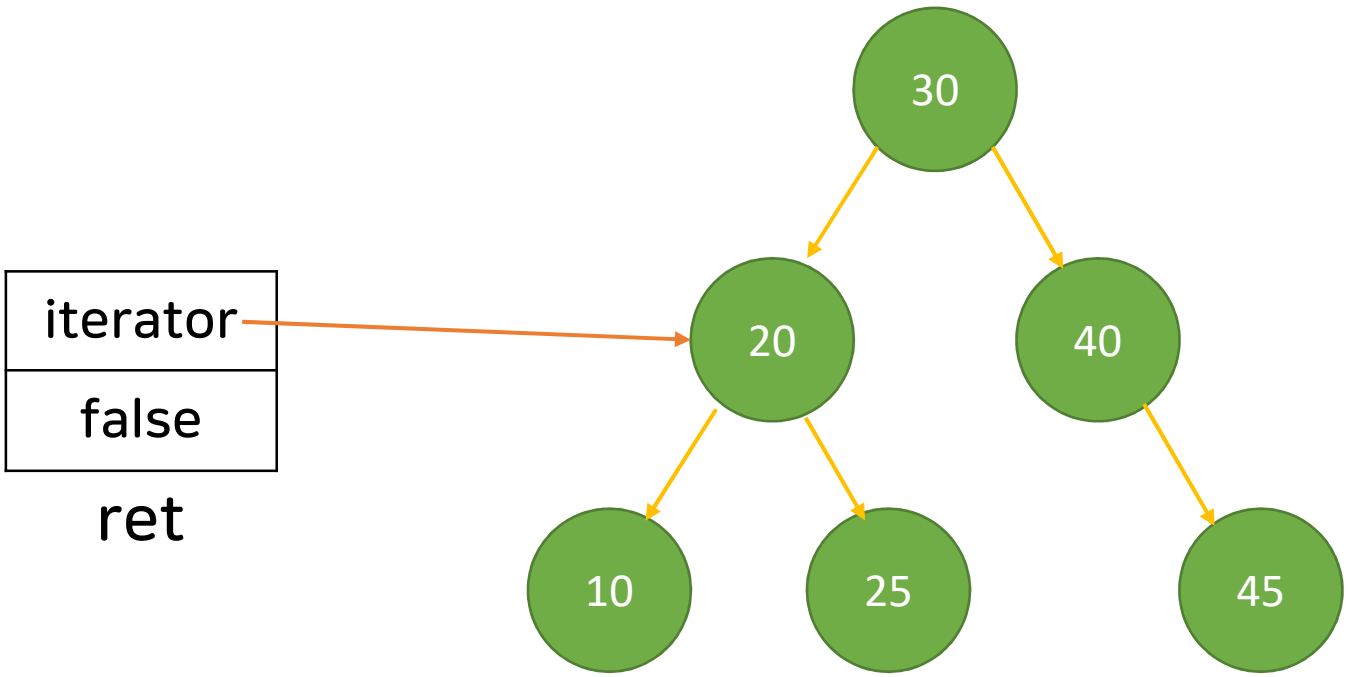
```
template<
    class Key,
    class Compare = std::less<Key>,
    class Allocator = std::allocator<Key>
> class set;
```

요소를 삽입하는 방법

- push_xxx 함수는 사용할 수 없다. insert 또는 emplace 로 만 사용할 수 있다.
- 반복자를 통해서 값을 변경할 수 없다.

요소를 검색하는 방법.

- find 알고리즘을 사용할 수 있지만 멤버 함수를 find를 사용하는 것이 좋다.


iterator
false
ret
30
20
40
10
25
45

set에 사용자 정의 타입을 넣으려면

- 사용자 정의 타입 안에 < 연산을 제공하거나
- 사용자 정의 타입에 대해 < 연산을 수행하는 함수 객체를 set의 2번째 템플릿 인자로 전달한다.

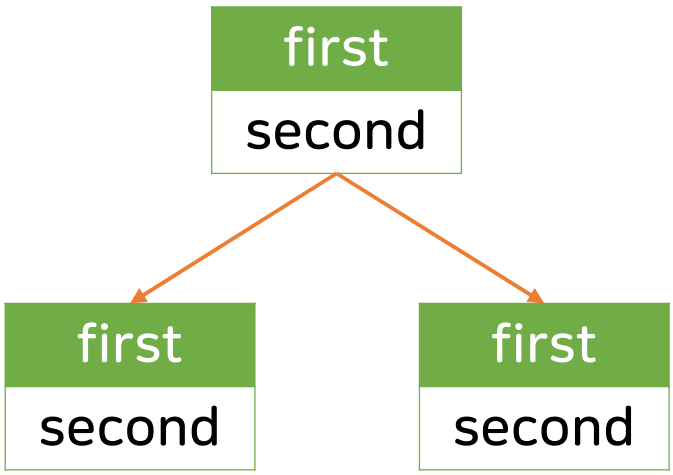
set 에서 a와 b의 상등(equality)을 조사하는 방법

- !(a<b) && !(b<a)
- == 연산이 제공되어도 사용되지 않는다.

insert 보다는 emplace 를 사용하는 것이 좋다.

std::map

- <map>
- pair를 저장하는 set
- key(first) 값으로 data(second)를 저장



map 에 항목을 넣는 방법

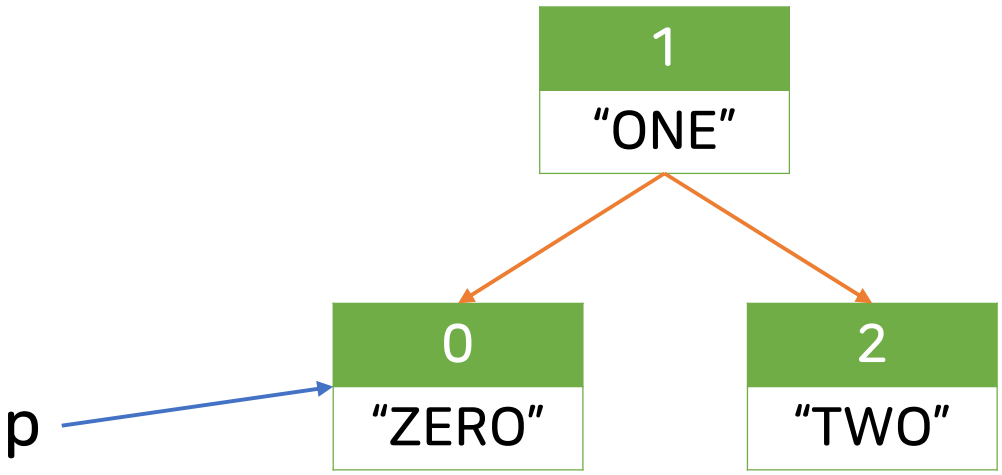
- pair 객체를 만들어서 insert
- make_pair 함수 사용
- 배열 연산자[] 사용

map 의 []연산

- 키값이 없을 경우, 새롭게 생성하게 된다.
- 키값이 존재하는 지를 확인하려면 []가 아니라 find 멤버 함수를 사용한다.

map의 반복자

- 반복자 : 요소를 가리키는 포인터 역할의 객체
- pair 를 가리키는 포인터



C++ 표준 stream 의 3가지 종류

basic_istream<> basic_ostream<>	<iostream>	표준 입출력
basic_ifstream<> basic_ofstream<>	<fstream>	파일 입출력
basic_istringstream<> basic_ostringstream<>	<sstream>	메모리(string) 입출력

파일에서 단어 분석하기

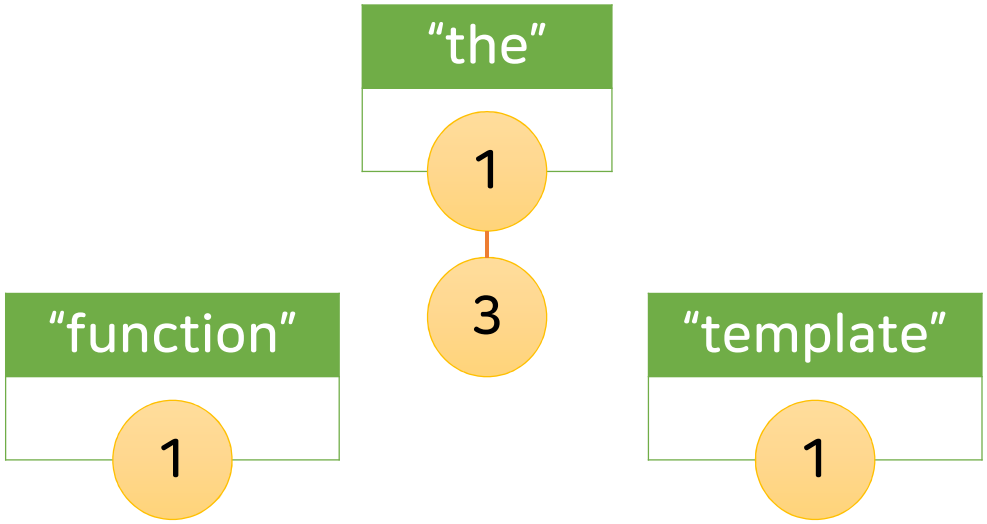
The template function async runs the function asynchronously and returns a std::future that will eventually hold the result of that function call

function : 1

template : 1

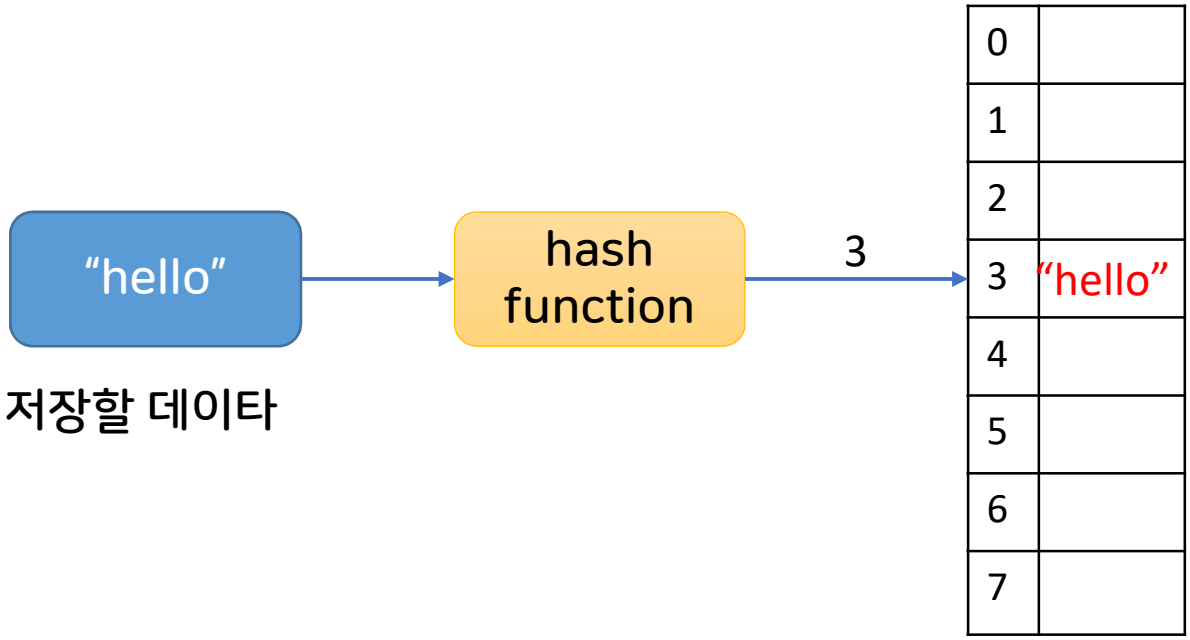
the : 1, 3

……



unordered_set, unordered_map

- hash table 기반의 자료 구조
- C++11부터 지원



hash function

- hash 라는 이름의 functor 로 제공.
- <functional>

```
template<> struct hash<bool>;
template<> struct hash<char>;
template<> struct hash<signed char>;
template<> struct hash<unsigned char>;
template<> struct hash<char16_t>;
template<> struct hash<char32_t>;
template<> struct hash<wchar_t>;
template<> struct hash<short>;
template<> struct hash<unsigned short>;
template<> struct hash<int>;
template<> struct hash<unsigned int>;
template<> struct hash<long>;
template<> struct hash<long long>;
template<> struct hash<unsigned long>;
template<> struct hash<unsigned long long>;
template<> struct hash<float>;
template<> struct hash<double>;
template<> struct hash<long double>;
template<> struct hash<string>;
template<> struct hash<wstring>;

template< class T > struct hash<T*>;
```

unordered_set

- hash table을 사용하는 set
- 정렬 상태를 유지 하지는 않는다.

unordered 컨테이너에 사용자 정의 타입을 넣으려면

```
template<
    class Key,
    class Hash = std::hash<Key>,
    class KeyEqual = std::equal_to<Key>,
    class Allocator = std::allocator<Key>

> class unordered_set;
```

- 사용자 타입에 대한 hash 함수(Functor)가 필요하다.
- 사용자 타입에 대한 equality 를 조사하는 Functor가 필요