C, C++, DSA in depth

set

Saurabh Shukla (MySirG)

Agenda

1. set
2. Creating set object
3. Accessing set elements
4. Implicit and explicit iterators
5. set methods

set

The values of the elements in the set are unique and serve as the key values according to which data is automatically ordered.

The value of an element in a set may not be changed directly. Instead you must delete old value and insert an element with new value.

A set is an associative container, which a variable size container that supports the efficient retrieval of element values based on an associated Key value.

It provides a bidirectional iterator

A set is sorted because its elements are ordered by key values within the container in accordance with a specified comparison function

The header for the STL set library is

<set>

set is based on Binary Search Tree.

How to create a set object?

```
set <int> s1;
set <int> s2= {6, 10, 20, 15};
```

Accessing set elements

You can access set in the variety of ways.

1. implicit iterator
2. explicit iterator

Implicit Iterator

```
set <int> s1 = {10, 20, 30, 40};

for ( int x : s1 )
    cout<< x <<" ";
```

or

```
for (auto x : s1)
    cout << x <<" ";
```

Explicit iterator

You can get an iterator object from the following members

①	begin()	end()	iterator
②	cbegin()	cend()	const_iterator
③	rbegin()	rend()	reverse_iterator
④	crbegin()	crend()	const_reverse_iterator

Explicit Iterator

```
set <int> S1 = { 50, 40, 10, 70, 60 };
set <int> :: iterator it ;
for (it = S1.begin() ; it != S1.end() ; it++)
    cout << *it << " " ;

set <int> :: const_iterator it ;
for (it = S1.cbegin() ; it != S1.cend() ; it++)
    cout << *it << " " ;
```

Methods of set

count()
empty()

emplace()
insert()

clear()
erase()

swap()
size()