­­­­­

如果你想要一个具有排序后的数据的话，通常可以选择Map这种类型。或者想要打印具有一定顺序的元素。

如果你只想记录数据而不是想要将数据进行排序的话，那么就可以选择unordered\_map这种数据结构。

Unique函数

类属性算法unique的作用是从输入序列中“**删除”所有相邻的重复元素**。

该算法删除相邻的重复元素，然后重新排列输入范围内的元素，并且返回一个迭代器（容器的长度没变，只是元素顺序改变了），表示无重复的值范围得结束。

Unique\_copy函数

算法标准库定义了一个名为unique\_copy的函数，其操作类似于unique。

唯一的区别在于：**前者接受第三个迭代器实参，用于指定复制不重复元素的目标序列。**

**unique\_copy根据字面意思就是去除重复元素再执行copy运算。**

Back\_inserter

除了普通迭代器，C++标准模板库还定义了几种特殊的迭代器，分别是插入迭代器、流迭代器、反向迭代器和移动迭代器，定义在<iterator>头文件中，下面主要介绍三种插入迭代器（back\_inserter,inserter,front\_inserter）的区别。

首先，什么是插入迭代器？插入迭代器是指被绑定在一个容器上，可用来向容器插入元素的迭代器。

back\_inserter：创建一个使用push\_back的迭代器

inserter：此函数接受第二个参数，这个参数必须是一个指向给定容器的迭代器。元素将被插入到给定迭代器所表示的元素之前。

front\_inserter：创建一个使用push\_front的迭代器（元素总是插入到容器第一个元素之前）

由于list容器类型是双向链表，支持push\_front和push\_back操作，因此选择list类型来试验这三个迭代器。

e.g.

list<int> lst = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

list<int> lst2 ={10}, lst3={10},lst4={10};

copy(lst.cbegin(), lst.cend(), back\_inserter(lst2));

//lst2包含10,1,2,3,4,5,6,7,8,9

copy(lst.cbegin(), lst.cend(), inserter(lst3, lst3.begin()));

//lst3包含1,2,3,4,5,6,7,8,9,10

copy(lst.cbegin(), lst.cend(), front\_inserter(lst4));

//lst4包含9,8,7,6,5,4,3,2,1,10

由示例便可轻易看出这三者的区别了。

Find函数