STL的相关概念：

STL( Standard Template Library )

·是一个具有工业强度的，高效的C++程序库

·被容纳于C++标准程序库(C++ Standard Library)中，是ANSI / ISO C++ 标准中极具革命性的一部分；

·包含了诸多在计算机科学领域里所常用的基本数据结构和基本算法。

【例如排序算法，链表、栈等线性结构】

·提供了一个可扩展的应用框架，高度体现了软件的可复用性。

泛型程序设计：

·将算法从特定的数据结构中抽象出来，变得更为通用【程序越抽象，重用性越高】

【例如开不同类型的数组，数组里的元素关于加减是一样的，就抽象出来？】

·C++的模板为泛型程序设计奠定了关键的基础

STL是泛型程序设计的一个范例

·容器(container)【e.g.数组，链表 作为载体，可以存储任何类型】

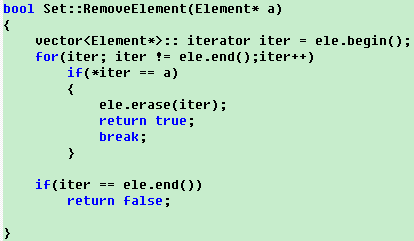
·迭代器(iterator)（作用为遍历容器，提供对容器内数据的访问）

·算法（algorithms）（对迭代器进行操作）

·函数对象（function object）【实例化后的】

迭代器

使用实例：



P.S. Element为之前定义好的一个类

实例化迭代器对象时必须指定iterator类型，

即为格式

迭代器可以看做是泛指针

·本质上是类，但可以当做指针

·有跟指针相关或者类似的操作，如

·迭代器类型根据容器类型定义【就是定义时的格式】

·算法可以对所有类型的迭代器进行操作

迭代器的基本操作：

1. 自加 ++ 
2. == 或者 ！=
3. \* 
4. c.begin() 【返回第一个】
5. c.end() 【返回最后一个的下一个】
6. – 自减 （自减操作符**--** 可用于动态可扩展数组，不可用于单向链表）
7. \*p 可读写

注意

·双向迭代器可以自增和自减，无法使用某些容器（如单向链表）

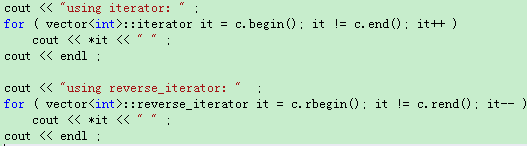
·可以像访问数组一样访问容器

Constant and Mutable Iterators

·默认的迭代器是mutable（可变的） 可以进行这样的读写操作

·这样定义了一个constant iterator，只能访问容器里的元素，但不能改变它们的值

迭代器可以回溯（从尾再到头，是环状的）



注意：回溯时迭代器的定义是rbegin()而不是begin()

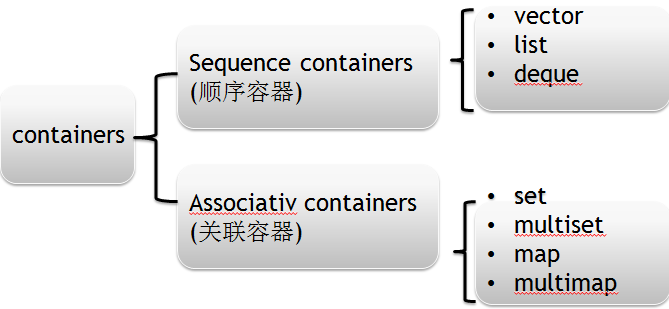
容器（三种）

·顺序容器（所有元素位置取决于插入顺序，例如数组等线性结构）

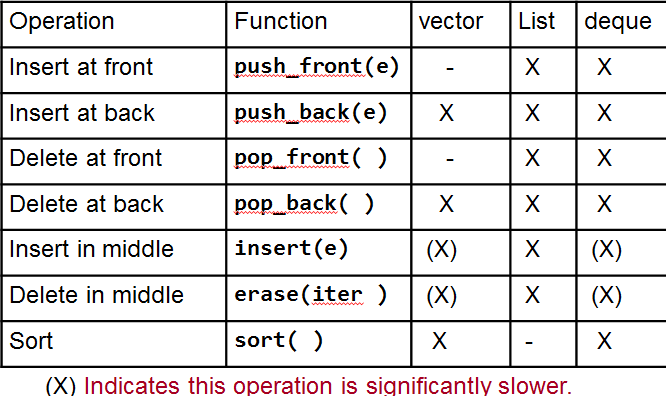
·容器适配器（顺序容器存储标，准库提供了三种顺序容器适配器：queue、priority\_queue、stack. 本质上，适配器是使一事物的行为类似于另一类事物的行为的一种机制。容器适配器让一种已存在的容器类型采用另一种不同的抽象类型的工作方式实现）

【就是容器适配器可以强制性让不是栈的顺序容器以栈的方式工作。】

·关联容器（位置与key关联）



容器的操作支持：



通用算法

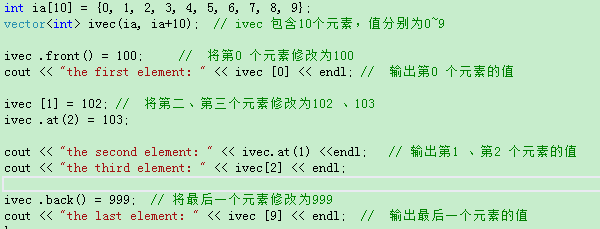
·STL提供标准算法，这些算法提供基本的服务：排序，搜索，拷贝，重排序，修改和数字处理

·算法不是容器类的成员函数，而是**全局函数**，通过iterator进行操作

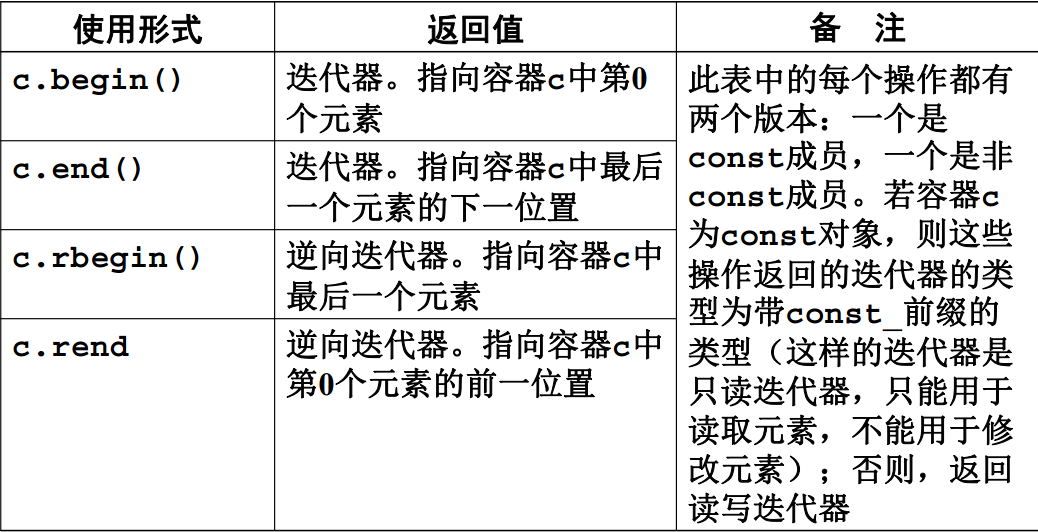
【可以理解为iterator关联容器内的元素，算法传入的参数为iterator，从而算法可以操作容器内的元素？？】



例子：

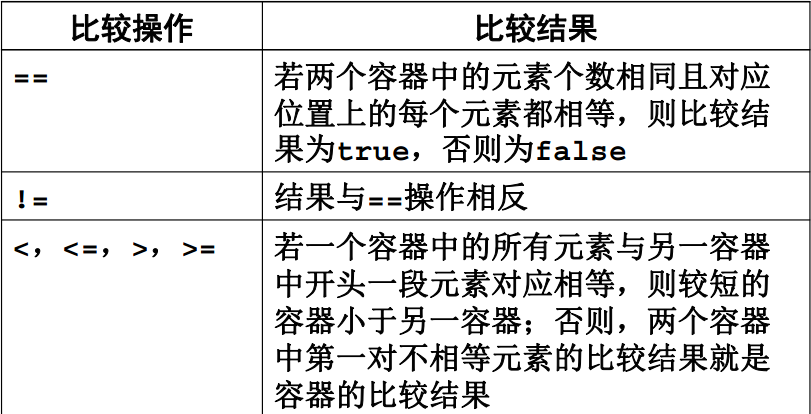


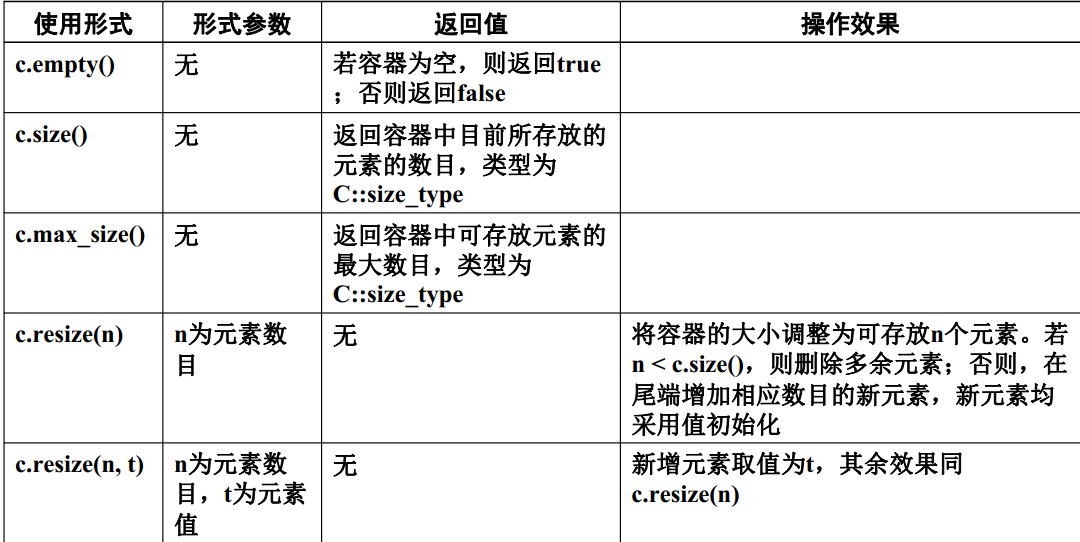
容器中的各种操作：

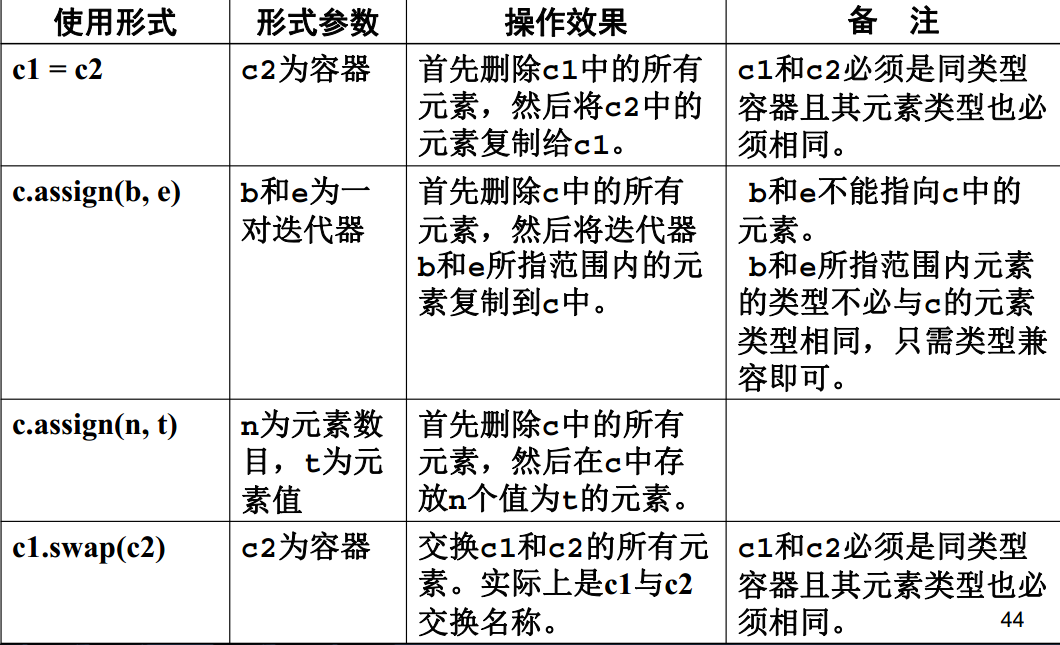












map

