

1. 基本概念：

容器：可容纳各种数据类型的通用数据结构，是类模板。

迭代器：可用于依次存取容器中元素，类似于指针。

算法：用来操作容器中的元素的函数模板。

容器：

1) 常序容器

vector, deque, list

2) 关联容器

set, multiset, map, multimap

3) 容器适配器

stack, queue, priority-queue

放入容器的对象所属的类，往往还应该重载 == 和 < 运算符。

顺序容器简介

容器并非排序的，元素的插入位置同元素的值无关。

• vector

动态数组。元素在内存中连续存放，随机存取任何元素都能在常数时间完成。在尾端增加元素具有较佳的性能。

• deque

双向队列。元素在内存中连续存放，随机存取任何元素都

能在常数时间完成(但次于vector)。在两端增加元素具有较佳的性能。

- list

双向链表。元素在内存中不连续存放。在任何位置增加元素都能在常数时间完成。不支持随机存取。

关联容器简介：

- △ 元素是排序的。
- △ 插入任何元素，都按相应的排序规则来确定其位置。
- △ 在查找时具有非常好的性能。
- △ 通常以平衡二叉树方式实现，插入和检索时间都是 $O(\log N)$

- set/multiset

set 即集合。set 中不允许相同元素。multiset 中允许存在相同的元素。

- map/multimap

map 与 set 的不同在于 map 中存放的元素有且仅有两个成员变量，一个名为 first，另一个名为 second，map 根据 first 值对元素进行从小到大排序，并可快速地根据 first 来检索元素。

map 与 multimap 的不同在于是否允许相同 first 值的元素。

容器适配器简介：

- **stack**

栈，是项的有限序列，并且在序列中被删除、检索和修改时项只能是最近插入序列的项(栈顶项)。后进先出。

- **queue**

队列，插入只可以在尾部进行，删除、检索和修改只允许从头部进行。先进先出。

- **priority - queue**

优先级队列，最高优先级元素总是第一个输出。

迭代器：

- △ 用于指向顺序容器和关联容器中的元素
- △ 迭代器用法和指针类似
- △ 有 **const** 和非 **const** 两种
- △ 通过迭代器可以读取它指向的元素
- △ 通过非 **const** 迭代器还能修改其指向的元素

容器

vector

dequeue

list

set/multiset

map/multimap

stack

迭代器

随机访问

随机访问

双向

双向

双向

不支持

queue

不支持

priority-queue

不支持

有的算法需要通过随机访问迭代器来访问容器中的元素。