# 序列式容器

## vector

vector一般是相对于静态的array而言，array一旦配置了空间就不可以改变（如改变则需要重新配置新的空间，然后逐一将旧元素搬到新址），vector是动态空间，随着元素的加入，它的内部机制会自行扩充空间以容纳新元素。

vector维护的是一个连续线性空间，因此普通指针可以作为vector的迭代器，自然vector也是支持随机存取的。

为了降低空间配置时的速度成本，vector实际配置的大小可能比客户端需求量更大一些，以备将来可能的扩充，因此一个vector的容量永远大于或者等于其大小。一旦容量等于大小，便是满载，下次有新增元素时，整个vector另外寻找存储空间。

## list

相对于vector提前多分配存储空间的方案，list对于空间的运用绝对的精准，一点不浪费，每次插入或者删除一个元素，就配置或释放一个元素空间。而且，对于任何位置的元素插入或移除，list永远都是常数时间。

什么时候使用vector和list，要根据元素的多少、构造复杂度、元素存取行为而定。

list不像vector一样以普通指针作为迭代器，其节点不保证存储空间的连续性。List的迭代器必须有能力指向list的节点，并且有能力进行正确的递增、递减、取值、成员存取等操作。STL list是一个双向链表，因此list提供的是Bidirectional Iterators。

list相对于vector还有一个好处就是不会出现原有迭代器整体失效的情况。

## deque

与vector单向开口的连续空间不同，deque是一种双向开口的连续线性空间。二者最大的区别，一是deque允许在常数时间内对起头端进行元素的插入或移除操作，二是deque没有容量的概念，因为它是动态以分段连续空间组合而成，随时都可以增加一段新的空间并链接起来（首尾都可以扩充）。可以认为，deque综合了vector和list的优点。

# 关联式容器