## 全称为 Standard Template Library ，即标准模板库

要使用STL，要了解以下几个基本概念：

容器：可以把它理解为存放数据的地方，常用的一些容器有 链表(list) 栈(stack) 动态数组 (vector) 双端队列(deque) 队列(queue) 映射(map)

游标(iterator)：可以把它理解为指针类型，STL中的许多函数需要用到它们作为参数

算法：它们通常需要与容器和游标配合使用，使用它们，你可以方便地对容器中的数据进行各种常见的操作，如排序操作，寻找最大元素的操作等

一、容器模板的使用

大致有下面6个步骤：

1.添加相应的头文件(如 #include <list> )( 注意，没有 .h )

2.添加std命名空间(用 using namespace std; )

3.赋予模板具体的使用类型(如 typedef list<string> LISTSTR; )

4.实例化模板(如 LISTSTR test; )

5.实例化游标(如 LISTSTR::iterator i; )

6.通过迭代器对象访问模板对象，例如

// 逐个输出链表test中的元素

for ( i = test.begin(); i != test.end(); ++i )

cout << \*i << " ";

类型：

向量(vector) 连续存储的元素<vector>

列表(list) 由节点组成的双向链表，每个结点包含着一个元素<list>

双队列(deque) 连续存储的指向不同元素的指针所组成的数组<deque>

集合(set) 由节点组成的红黑树，每个节点都包含着一个元素，节点之间以某种作用于元素对的谓词排列，没有两个不同的元素能够拥有相同的次序 <set>

多重集合(multiset) 允许存在两个次序相等的元素的集合 <set>

栈(stack) 后进先出的值的排列 <stack>

队列(queue) 先进先出的执的排列 <queue>

优先队列(priority\_queue) 元素的次序是由作用于所存储的值对上的某种谓词决定的的一种队列 <queue>

映射(map) 由{键，值}对组成的集合，以某种作用于键对上的谓词排列 <map>

多重映射(multimap) 允许键对有相等的次序的映射 <map>

二、容器模板中的常用函数

assign() 赋值

empty() 容器为空则返回非0值

erase() 删除指定位置或指定范围内的元素

push\_front() 从容器头部插入元素

push\_back() 从容器尾部插入元素

pop\_front() 删除第一个元素

pop\_back() 删除最后一个元素

back() 返回最后一个元素的引用

front() 返回第一个元素的引用

begin() 返回指向第一个元素的游标 (与迭代器配合使用)

end() 返回指向最后一个元素的后一个位置的游标 (最后1个元素再加1) (与迭代器配合使用)

迭代器在STL中用来将算法和容器联系起来，起着一种黏和剂的作用。几乎STL提供的所有算法都是通过迭代器存取元素序列进行工作的，每一个容器都定义了其本身所专有的迭代器，用以存取容器中的元素。

1. map的使用

map.insert(make\_pair(key,value))和map[key] = value的区别在于insert的只是用于新增，而后者是在不存在的时候新增，在存在的时候更新，使用时应注意场景。

而在函数调用的时候，使用迭代器find(key)之后返回it->second和不使用迭代器直接返回map[key]的区别是：后者会造成当该key的索引不存在的时候，新增一个map的索引，造成数据的错误。

1. STL中的指针的使用

例：map<key,int\*> a和map<key,int\*> b

在这个例子中，a和b的value如果在赋值的时候是指向同一块地址，那么a和b中的value是一样的，当改变其中一个的时候，另一个也会发生改变。

应用场景：当一个值分别可以用不同的key指向的时候，并且这两个key使用的场景有区别时，该使用的方法效率较高。