超快复习C++ —— STL By 软件工程11级李嘉良

废话：

STL组成 ：Iterator（迭代器） Container（容器）Algorithm（算法）

原理：容器储存的元素（类，结构体，变量）通过迭代器（接口的功能）使用算法处理。

必要提醒：

13个头文件：<algorithm>、<deque>、<functional>、<iterator>、<vector>、<list>、<map>、<memory>、<numeric>、<queue>、<set>、<stack>和<utility>

//算法部分主要由头文件<algorithm>，<numeric>和<functional>组成

红色：必选 蓝色：推荐 绿色：视情况

分类：

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<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>

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>

主要掌握vector，map，list

1. Vector

头文件：

#include <iostream>

#include <vector>

#include <utility>

#include <algorithm>

using namespace std;//or using std::vector

初始化：

vector<元素类型> 名字;//初始化一个空向量

vector<元素类型(如int,)> 名字(如arr)(长度(如10) );//初始化长度的vector

vector<元素类型> vector\_a(vector\_b);//拷贝初始化

vector<元素类型> 名字(长度,初始值);//初始化长度并每个赋值为初始值

vector< int > vecname( a, a+6 );

//初始化为已有数组的一部分,此例整型数组a[0]到a[5]共6个元素初始化为int型的vector

访问：

1. 下标访问（非所有容器均支持）：vector<int> a(10,-1); cout << a[6] << endl;//输出-1

2）迭代器访问（不知道是啥？当数组指针用）：

开头第一个元素的指针默认为（容器名.begin()如a.begin()）

最后一个元素的下一个位置的指针默认为（容器名.end()如a.end()）

vector<string>::iterator it = text.begin();

 //定义迭代器（容器类型<元素类型>::iterator 名字）并指向容器的第一个元素

++it，--it//迭代器可以如同指针一样移动

\*it; //it相当于指向向量容器text的一个指针，可以使用\*it这样的方式来取到元素

例子：

//用途：把容器text里面的内容逐个输出

//迭代器访问

for ( vector<string>::iterator it = text.begin();it != text.end(); ++it ) cout << \*it << ' ';

//习惯使用it != text.end()而不是>、<，因为有些迭代器不支持比较

//注意text.end()不是最后一个元素，是最后一个元素的下一个位置（不理解暂时死记）

//

//下标访问

for ( int i = 0; i < text.size(); ++i) cout << text[ i] << ' ';

//text.size返回容器的长度

功能(最最基本，其他自己找)：

1）修改

Vector.pop\_back()//删除最后一个数据

Vector.push\_back(element)//尾部加入一个数据

Vector.erase(position) //删除position位置的数据

Vector.erase(begin, end) //删除[begin, end)区间的数据

Vector.insert(position, element)//在position位置插入一个element拷贝

2）信息检索

Vector.size()//返回容器中实际数据的个数。

Vector.begin()//返回指向容器第一个元素的迭代器

Vector.end()//返回指向容器最后一个元素的下一个位置的迭代器

3）查找

vector <Item>::interator it = find(vi.begin(), vi.end(), targetItem) ;

//从开头找到末尾，返回找到的元素的指针给it，没有则返回vi.end()

cout << \*it << endl ; ///返回容器内找到值的位置。

4）排序

sort(vi.begin() , vi.end()); // 从小到大

//以上只能比较已有类型，需要比较自定义类的，需在类里面写上比较函数，写法自寻

// sort(vi.begin(), vi.end(),compare()) ;

注意：

//插入和赋值不要搞混！

vector<int> vec;

vec[0] = 1024; //错误！vec是空的，应该使用push\_back！

//插入在最后一个位置！

vector<int> vec(10);

vec.push\_back(2)//size变成11！

1. map（key-value，其中第一个称为关键字，每个关键字只能出现一次，第二个可称为该关键字的值）

头文件：

#include <iostream>

#include <vector>

#include <utility>

#include <algorithm>

using namespace std;//or using std::map

初始化：

map<type1,type2>name\_;

//例：map<string, int >string\_int; map<int ,string >int\_string;

访问：

Int T=string\_int[“target”];//直接使用key访问value

string V=int\_string[2];

it->first;it->second;//定义迭代器，first与second分别为key与value

功能：

1. 修改

Map.insert(pair<int,string>(102,"aclive"));//关于pair类型请自行查找相关信息

Map.insert(map<int,string>::value\_type(321,"hai"));//前两种只适用于key不存在！

Map[112]="April";//简单最常用的插入添加！当key已经存在时，会修改对应value

//用迭代器删除

map<int, string>::iterator iter;

iter = mapStudent.find(1);

mapStudent.erase(iter);

//用关键字删除

Int n = mapStudent.erase(1);//删除了返回1，否则返回0

//用迭代器，成片的删除

mapStudent.earse(mapStudent.begin(), mapStudent.end());

//成片删除要注意的是，删除区间是一个前闭后开的集合

1. 查找

it=string\_int.find(targetItem);//find函数可以如同之前vector一样定义范围=

if(it==string\_int.end())

       {/\*找不到数据\*/}

else {/\*找到数据\*/}

1. 信息检索

Map.size()//返回容器中实际数据的个数

Map.begin()//返回指向容器第一个元素的迭代器

Map.end()//返回指向容器最后一个元素的下一个位置的迭代器

4）排序

自动按key升序排列，key有固定的含义，一般是不需要排序的

1. List（双向链表，允许快速的插入和删除，但是随机访问却比较慢）

头文件：

#include <iostream>

#include <list>

#include <algorithm>

using namespace std;

初始化：

list<类型> 容器名;

访问：

不能通过下标，只能逐项遍历！

for (list <int>::iterator it = test.begin (); it != test.end (); it ++)  
 {

     if (\*it == target)

{ /\*已经得到\*it \*/ }

}

功能：

1. 修改

List.assign(长度,初始值);//整体赋值

List.push\_front (target);//前添加

List.push\_back (target);//后添加

List.pop\_front();//前移除

List.pop\_back();//后移除

List.insert(List.begin()+1,3,9);//从第二位开始插入3个9

List.erase(List.begin()+1);//擦除第二位元素

1. 查找

it=List.find(targetItem);//find函数可以如同之前vector一样定义范围=

if(it==List.end())

       {/\*找不到数据\*/}

else {/\*找到数据\*/}

1. 信息检索

List.size()//返回容器中实际数据的个数

List.front()//返回容器第一个元素

List.back()//返回容器最后一个元素

List.begin()//返回指向容器第一个元素的迭代器

List.end()//返回指向容器最后一个元素的下一个位置的迭代器

1. 排序

List.sort();//已有类型排序

List.reverse();//把list的元素倒转

实际应用可看附带的代码！祝好运！

附录：

标准 C++

#include <algorithm>　　　  通用算法  
#include <bitset>　　　　　 位集容器  
#include <cctype>  
#include <cerrno>  
#include <clocale>  
#include <cmath>  
#include <complex>　　　　 复数类  
#include <cstdio>  
#include <cstdlib>  
#include <cstring>  
#include <ctime>  
#include <deque>　　　　　 双端队列容器  
#include <exception>　　　 异常处理类  
#include <fstream>  
#include <functional>　　　 定义运算函数（代替运算符）  
#include <limits>  
#include <list>　　　　　　 线性列表容器  
#include <map>　　　　　　 映射容器  
#include <iomanip>  
#include <ios>　　　　　　基本输入／输出支持  
#include <iosfwd>　　　　输入／输出系统使用的前置声明  
#include <iostream>  
#include <istream>　　　　 基本输入流  
#include <ostream>　　　　 基本输出流  
#include <queue>　　　　　  队列容器  
#include <set>　　　　　　 集合容器  
#include <sstream>　　　　 基于字符串的流  
#include <stack>　　　　　 堆栈容器　　　　  
#include <stdexcept>　　　 标准异常类  
#include <streambuf>　　　底层输入／输出支持  
#include <string>　　　　　字符串类  
#include <utility>　　　　 通用模板类  
#include <vector>　　　　 动态数组容器  
#include <cwchar>  
#include <cwctype>