面向对象程序设计(OOP)

C++(第9章：泛型程序设计和C++标准模板库STL)

福州大学·软件学院·软件工程系
王灿辉(wangcanhui@fzu.edu.cn)

第9章：泛型程序设计和C++标准模板库

概述：使用标准库

- 不要像重新发明车轮那样企图做每件事，去使用C++标准库！！为什么不？
- 相信自己：自己编写正确性有保证、效率更高、容易控制
- 学习方法导致：学习计算机语言的流行方法是用到那学到那！

第9章：泛型程序设计和C++标准模板库

概述：使用标准库

- 我们使用**标准库就是想从别人的工作中获益**。使用库函数、类、算法等能使我们免去许多工作：不必再去发明、设计、书写某些东西，排除其中的错误，以及为它撰写文档，等等。使用标准库也能使产生出的代码更**便于其他人阅读**，只要他们熟悉这个库。否则的话，人们就不得不花许多时间和努力去理解那些“家酿”的代码。

第9章：泛型程序设计和C++标准模板库

概述：C++标准库

- 标准库的功能都定义在std命名空间里，用一组头文件的方式呈现。所有名字以字母c开头的标准头文件<cX>(在std命名空间里定义)等价于C标准库中的一个有文件<X.h>(在全局命名空间里定义)。以下的几张幻灯片分类介绍库的组织。

第9章：泛型程序设计和C++标准模板库

概述：C++标准库--容器

- <vector>:T的一维数组
- <list>:T的双向链表
- <deque>:T的双端队列
- <queue>和<priority_queue>:T的队列
- <stack>:T的堆栈
- <map>和<multimap>:T的关联数组
- <set>和<multiset>:T的集合
- <bitset>:布尔量的集合

第9章：泛型程序设计和C++标准模板库

C++标准库：通用功能和迭代器

- <utility>:运算符和对偶
- <functional>:函数对象
- <memory>:容器用的分配器和auto_ptr模板
- <ctime>:C风格的日期和时间
- <iterator>:迭代器和迭代器支持，使标准算法能通用于标准容器和类似类型

第9章：泛型程序设计和C++标准模板库

C++标准库：算法和诊断

- <algorithm>:通用算法
- <cstdlib>:bsearch()和qsort()
- <exception>:异常类
- <stdexcept>:标准异常
- <cassert>:assert宏
- <cerror>:C风格的错误处理

第9章：泛型程序设计和C++标准模板库

C++标准库：串

- <string>:T类型的串
- <ctype>:字符分类
- <cwtype>:宽字符分类
- <cstring>:C风格的字符串函数
- <cwchar>:C风格的宽字符串函数
- <cstdlib>:C风格的串函数(atoi等函数)

第9章：泛型程序设计和C++标准模板库

C++标准库：输入/输出

- <iosfwd>:I/O功能的前导声明
- <iostream><istream><ostream>:输入/输出流模板，<ios>:iostream基类
- <iomanip>:操控符，<streambuf>:流缓冲区
- <sstream>:以串作为I/O对象的流
- <fstream>:以文件作为I/O对象的流
- <cstdio>:printf()族I/O功能
- <cwchar>:宽字符的printf()风格I/O

第9章：泛型程序设计和C++标准模板库

C++标准库：语言支持

- <limits>:数值范围，<climits>:C风格的数值标量范围宏，<cfloat>: C风格的浮点数值范围宏，<new>:动态存储分配，<typeinfo>:运行时类型标识支持功能，<cstddef>:C库语言支持，<cstdarg>:变长函数参数表，<csetjmp>:C风格的堆栈回退，<csignal>:C风格的信号处理，<cstdlib>:程序终止。

第9章：泛型程序设计和C++标准模板库

C++标准库：数值和本地化

- <complex>:复数及其运算
- <valarray>:数值向量和运算
- <numeric>:通用数值运算
- <cmath>:标准数学函数
- <cstdlib>:C风格的随机数,abs函数等
- <locale>:表示文化差异
- <clocale>:表示文化差异，C风格

第9章：泛型程序设计和C++标准模板库

泛型程序设计

- 泛型程序设计(Generic Programming)的主要思想是：将算法从特定的数据结构中抽象出来，使算法成为通用的，可以作用于各种不同的数据结构。它与OO的思想相违背！

第9章：泛型程序设计和C++标准模板库

标准模板库STL

- 标准模板库(STL:Standard Template Library)是C++标准库的核心，它采用泛型程序设计思想，深刻影响了标准库的整体结构和组成。构建STL框架最关键的4个组件是：容器(container)、迭代器(iterator)、算法(algorithm)、和函数对象(function object)。

第9章：泛型程序设计和C++标准模板库

标准模板库STL

- 容器:就是能保存其他对象的对象。如:向量、队列、映射等。
- 迭代器:是面向对象版本的指针。它提供了访问容器和序列中每个元素的方法。
- 算法:包括排序、查找等70多个算法。
- 适配器:为已有的类提供新的接口
- 容器的接口:容器的方法和运算符

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector是定义在命名空间std中的一个模板，由<vector>给出。


```
template<class _Ty,class _A=allocator<_Ty> >
class vector {
public:
  //定义标准的类型名
  //一批构造函数、拷贝构造函数、析构函数
  //各种成员函数(assign,size,empty等)
}
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的4个构造函数：

```
vector<int> v1; //构造一个空的向量
vector<int> v2(3,11); //构造向量(3个元素，值均为11)
vector<int> v3(v2); //拷贝构造
vector<int> v4(v3.begin()+2,v3.end());
//由迭代器指定的范围构造新的向量
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数：

```
vector<int> v; //必须#include <vector>
//输出向量的大小(size)和容量(capacity)
cout<<"size:"<<v.size()<<"\t\t";
cout<<"capacity:"<<v.capacity()<<'\t';
//输出向量允许存储的最大数(max_size)
cout<<"max size:"<<v.max_size()<<'\t';
//empty()成员函数：判向量是否为空?
if(v.empty()) cout<<"Vector is empty!\n";
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

//将值添加到向量的末尾--**向量根据需要会自动增加大小**


```
for (i=0;i<11;i++) v.push_back(i+1);
//输出向量当前的内容
cout<<" Currect contents:";
for (i=0;i<v.size();i++)
    cout<<v[i]<<' '; //重载下标运算符
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//输出向量的第1个元素和最后1个元素
cout<<"front:"<<v.front()<<'\t';
cout<<"back:"<<v.back()<<'\t';
//cout<<v.at(v.size());//带检查的访问！
//上述语句将抛出out_of_range异常！
cout<<v[v.size()]<<endl;//不加检查的访问！
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//删除向量的最后一个元素(无返回值!)
v.pop_back();  //原型为:void pop_back();
//修改向量的内容
for (i=0;i<v.size();i++)
   v.at(i)=v.at(i)+v.at(i);
//由于已经在循环中进行了下标的检查,
//所以这里可不必使用带检查的下标访问at()
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//声明一个(常)迭代器(类似用指针访问数组)
vector<int>::const_iterator cp;
//输出向量当前的内容
//begin()返回向量对象中第1个元素的迭代器
//end()返回超出最后元素1个位置的迭代器
for (cp=v.begin();cp!=v.end();cp++)
    cout<<*cp<<' ';
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//将值插入到向量中--向量会自动增加大小
for (i=0;i<3;i++)
   v.insert(v.begin()+i+10,61+i);
//将22在指定位置插入2次！
v2.insert(v2.begin()+1,2,22);
//把指定范围的元素复制到指定位置
v3.insert(v3.begin()+2,v2.begin()+1,v2.begin()+3);
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//输出向量的内容
cout<<"Current contents:";
//创建输出迭代器
ostream_iterator<int> output(cout," ");
copy(v.begin(),v.end(),output);
//不需要写循环就可以输出向量中的所有元素！
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//删除向量中一些元素
vector<int>::iterator p; //普通迭代器
p=v.begin()+5;   v.erase(p,p+5);
//用常迭代器不能修改向量的值
//所以必须使用普通迭代器来访问向量
v3.erase(v3.begin()+1);//删指定位置的元素
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//改变向量对象的大小，如需要在末尾补元素
v.resize(18,0);
//设定容量，值必须大于现容量，否则无效！
v.reserve(35);
//删除向量对象中的所有元素
v.clear();
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//声明一个反向迭代器(类似用指针访问数组)
vector<int>::reverse_iterator rp;
//(反向)输出向量当前的内容
//rbegin()返回向量中最后1个元素的迭代器
//rend()返回超出第1个元素1个位置的迭代器
for (rp=v.rbegin();rp!=v.rend();rp++)
    cout<<*rp<<' ';
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：向量vector

- 标准vector的部分成员函数(续)：

```
//将迭代器指定范围的元素赋值给向量
v1.assign(v3.begin()+3,v3.end()-1);
```


- 其他成员函数的介绍详见(美)Bjarne Stroustrup著，裘宗燕 译，C++程序设计语言，北京：机械工业出版社，2002年7月，或参见其它书籍

第9章：泛型程序设计和C++标准模板库
标准模板库STL：算法和函数对象

- 标准vector的排序(必须#include <algorithm>和#include <functional>)：

```
//将向量中的所有元素以升序排列
stable_sort(v.begin(),v.end());
//sort和stable_sort为STL中的排序算法
//将向量中的所有元素以降序排列
sort(v.begin(),v.end(),greater<int>());
//greater<int>()为STL中的函数对象(模板)
```

第9章：泛型程序设计和C++标准模板库
标准模板库STL：算法和函数对象

- 标准vector的统计：

```
cout<<"当前向量中等于25的元素个数为：";
cout<<count(v3.begin(),v3.end(),25);
cout<<"\t\t小于20的元素个数为：";
cout<<count_if(v3.begin(),v3.end(),bind2nd(less<double>(),20));
cout<<"当前向量中所有元素的累计值为：";
cout<<accumulate(v3.begin(),v3.end(),0);
//必须#include <numeric>
```

第9章：泛型程序设计和C++标准模板库
标准模板库STL：算法和函数对象

- 标准vector的统计：

```
double multi(double x,double y)
   {return x*y;} //定义一个普通函数
cout<<"\n所有元素的乘积为：";
cout<<accumulate(v3.begin(),v3.end(),1,
   multi)<<endl;
cout<<"当前向量中所有元素的乘积为：";
cout<<accumulate(v1.begin(),v1.end(),1,
   multiplies<double>())<<endl;
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：双端队列deque

- 双端队列deque及部分成员函数：

```
//定义双端队列(8个元素，值均为33)
deque<double> d(8,33);//#include <deque>
d.pop_front();   //删除前端元素
d.push_front(4);//添加在双端队列的前端
d.at(2)=32;      //带检查赋值
d[1]=31;         //无检查赋值
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：双端队列deque

- 双端队列deque的输出(其他方法参见向量的输出)：

```
//输出向量和双端队列的内容
template <class T> void show(T &x) {
  ostream_iterator<double> output(cout," ");
  cout<<"Currect contents("<<x.size()<<"): ";
  copy(x.begin(),x.end(),output);
  cout<<endl;
}
```

第9章：泛型程序设计和C++标准模板库
标准模板库STL：列表list

- 列表list及部分成员函数：

```
//定义列表(4个元素，值均为日期:2005.10.27)
list<Mydate> l1(4,Mydate(2005,10,27));
//输出列表内容，必须#include <list>
list<Mydate>::const_iterator cp;
cout<<"列表内容("<<l1.size()<<"): ";
for (cp=l1.begin();cp!=l1.end();cp++)
   cout<<*cp<<‘ ’; //列表合并等函数：略
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：栈stack

- 栈stack及部分成员函数：

```
stack<int> s1;//定义栈,#include <stack>
for (i=0;i<15;++i) s1.push(i+10); //压栈
//输出栈s1的元素
cout<<"Stack("<<s1.size()<<"): ";
while (!s1.empty()) {
  cout<<s1.top()<<' '; //读栈顶元素
  s1.pop();            //弹栈
}
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL：队列queue

- 队列queue及部分成员函数：

```
queue<int> q1;//定义队列,#include <queue>
for (i=0;i<8;++i) q1.push(i+20);//入队列
//输出队列q1的元素
cout<<"Queue("<<q1.size()<<"): ";
while (!q1.empty()) {
  cout<<q1.front()<<' '; //读队列元素
  q1.pop();              //删除已出列的元素
}
```

第9章：泛型程序设计和C++标准模板库
标准模板库STL：map类(关联容器)

- map类(关联容器)及相关函数：

```
//定义map类(关联容器)
map<char,int> m1; //必须#include <map>
//将10个关键字/值对存储到map对象中
for (i=0;i<26;++i)
  m1.insert(pair<char,int>('A'+i,i+81));
```

第9章：泛型程序设计和C++标准模板库
标准模板库STL：map类(关联容器)

- map类(关联容器)及相关函数：

```
//根据关键字查找对应的值
map<char,int>::const_iterator mp;
mp=m1.find(key); //key为待查找的关键字
//输出查找结果
if (mp!=m1.end())
    cout<<"关键字["<<key<<"]对应的值为："<<mp->second;
else cout<<"Key not in map.";
```

第9章：泛型程序设计和C++标准模板库

标准模板库STL

- 标准模板库STL的向量vector类、迭代器、算法、函数对象等使用实例
- 标准模板库STL的其他类(双端队列、列表、栈、队列等)的使用实例
- 更详细的介绍参见(美)Bjarne Stroustrup著，裘宗燕译，C++程序设计语言，机械工业出版社，2002年7月，或读相应的头文件

本章内容讲授到此结束！

福州大学·软件学院·软件工程系
王灿辉(wangcanhui@fzu.edu.cn)