**16 模板与泛型编程**

1，函数模版

template <typename T>

int compare(const T &v1, const T &v2){}

类型参数前必须使用关键字class或typename，几乎没什么区别。

除了类型参数，还可以在模版中定义非类型参数，一个非类型参数表示一个值。

第一个模版参数表示第一个数组的长度，第二个参数表示第二个数组的长度：

template<unsigned N, unsigned M>

int compare(const char (&p1)[N], const char (&p2)[M]){}

compare("hi","mom")

一个非类型参数可以是一个整型，或者是一个指向对象或函数类型的指针或引用。

编写泛型代码的两个重要原则：1，模版中的函数参数是const的引用；2，函数体中的条件判断仅使用<比较运算

函数模版和类模版成员函数的定义通常放在头文件中。

2，类模版

template <typename T>

class Blob{}

定义在类模版之外的成员函数就必须以关键字template开始，后接类模版参数列表。

在一个类模版的作用域内，我们可以直接使用模版名而不必指定模版实参。

3，模板参数

默认模版实参：

template <typename T,typename F = less<T>>

int compare(const T &v1, const T &v1, F f = F())

对于一个模版参数，只有当它右侧的所有参数都有默认实参时，它才可以有默认实参。

模版默认实参与类模版

template <class T = int> class Numbers{};

待续