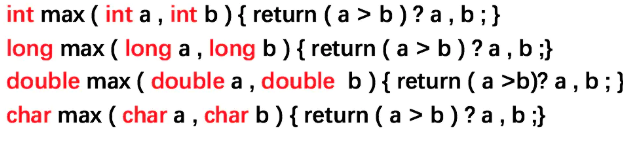
## 模板知识点总结

## 一、 为什么要使用模板？

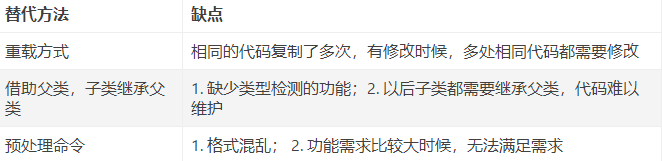
1. 计一个求两参数最大值的函数，在实践中我们可能需要定义四个函数：



2.函数几乎相同，唯一的区别就是形参类型不同。

3.事先知道有哪些类型会使用这些函数，对于未知类型这些函数不起作用。

4.其他替代方法对比

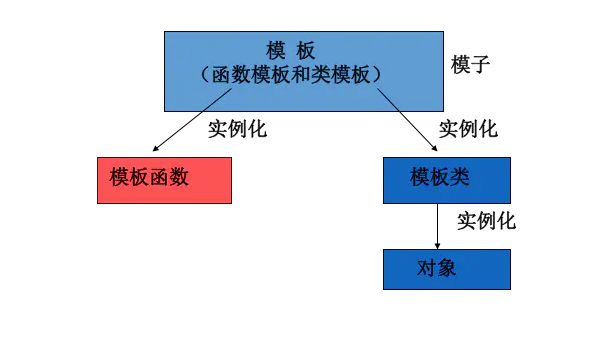


**二、模板的概念**

1. 所谓模板是一种使用无类型参数来产生一系列函数或类的机制。
2. 若一个程序的功能是对某种特定的数据类型进行处理，则可以将所处理的数据类型说明为参数，以便在其他数据类型的情况下使用，这就是模板的由来。
3. 模板是以一种完全通用的方法来设计函数或类而不必预先说明将被使用的每个对象的类型。
4. 通过模板可以产生类或函数的集合，使它们操作不同的数据类型，从而避免需要为每一种数据类型产生一个单独的类或函数。

# 三、模板的分类

1. 模板分为函数模板（模子）和类模板（模子），允许用户分别用它们构造（套印）出（模板）函数和（模板）类。
2. 图显示了模板（函数模板和类模板），模板函数，模板类和对象之间的关系。



三.1

## 函数模板

### 定义格式

template <class 形参名,class 形参名，...>

返回类型 函数名(参数列表)

{

... //函数体

}

<>括号中的参数叫模板形参，模板形参和函数形参很相像，**模板形参不能为空**。一但声明了模板函数就**可以在该函数中使用内置类型的地方都可以使用模板形参名**。模板形参需要调用该模板函数时提供的模板实参来初始化模板形参，一旦编译器确定了实际的模板实参类型就称他实例化了函数模板的一个实例。

例：

template <class T> void swap(T& a, T& b){}

当调用这样的模板函数时类型T就会被**被调用时的类型**所代替，比如swap(a,b)其中a和b是int 型，这时模板函数swap中的形参T就会被int 所代替，模板函数就变为swap(int &a, int &b)。而当swap(c,d)其中c和d是double类型时，模板函数会被替换为swap(double &a, double &b)，这样就实现了函数的实现与类型无关的代码。

**注意：**

对于函数模板而言不存在 h(int,int) 这样的调用，不能在函数调用的参数中指定模板形参的类型，对函数模板的调用应使用实参推演来进行，即只能进行 h(2,3) 这样的调用，或者int a, b; h(a,b)。

## 3.2. 类模板

### 定义格式

template<class 形参名，class 形参名，…>

class 类名

｛

// 类定义．．．

};

类模板和函数模板都是以template开始后接模板形参列表组成，**模板形参不能为空**，一但声明了类模板就可以用类模板的形参名声明类中的成员变量和成员函数，即可以在类中使用内置类型的地方都可以使用模板形参名来声明。

比如:

template<class T>

class A

{

public: T a; T b;

T hy(T c, T &d);

};

在类A中声明了两个类型为T的成员变量a和b，还声明了一个返回类型为T带两个参数类型为T的函数hy。

类模板的使用：

对于类模板，模板形参的类型必须在类名后的尖括号中明确指定。比如A<2> m;用这种方法把模板形参设置为int是错误的（编译错误：error C2079: 'a' uses undefined class 'A<int>'），类模板形参不存在实参推演的问题。也就是说不能把整型值2推演为int 型传递给模板形参。要把类模板形参调置为int 型必须这样指定A<int> m。

**在类模板外部定义成员函数的方法为：**

template<模板形参列表> 函数返回类型 类名<模板形参名>::函数名(参数列表){函数体}

比如有两个模板形参T1，T2的类A中含有一个void h()函数，则定义该函数的语法为：

template<class T1,class T2> void A<T1,T2>::h(){}

注意：

1.当在类外面定义类的成员时template后面的模板形参应与要定义的类的模板形参一致。

2.模板的声明或定义只能在全局，命名空间或类范围内进行。即不能在局部范围，函数内进行，比如不能在main函数中声明或定义一个模板。

3.其中，template 是声明类模板的关键字，表示声明一个模板，模板参数可以是一个，也可以是多个，可以是类型参数 ，也可以是非类型参数。类型参数由关键字 class或typename 及其后面的标识符构成。非类型参数由一个普通参数构成，代表模板定义中的一个常量。（局部特化）

例：

template<class type,int width>

//type为类型参数，width为非类型参数

class Graphics;