0、

讲一些基础，主要是Operator Overloading和Templates

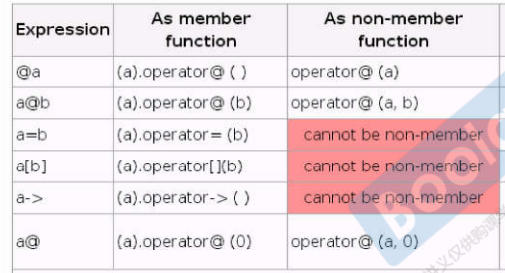
1、

符号不能被重载

分别是：\ :: \ . \ .\* \ ?: \ \*\* \ <> \ &| \

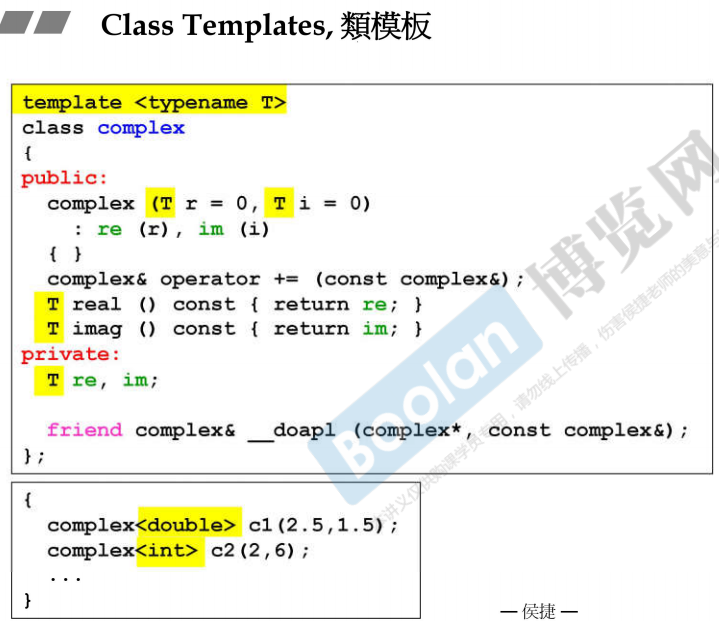
2、

操作符重载可以写成全局的函数也可以写成成员函数（在类里面）



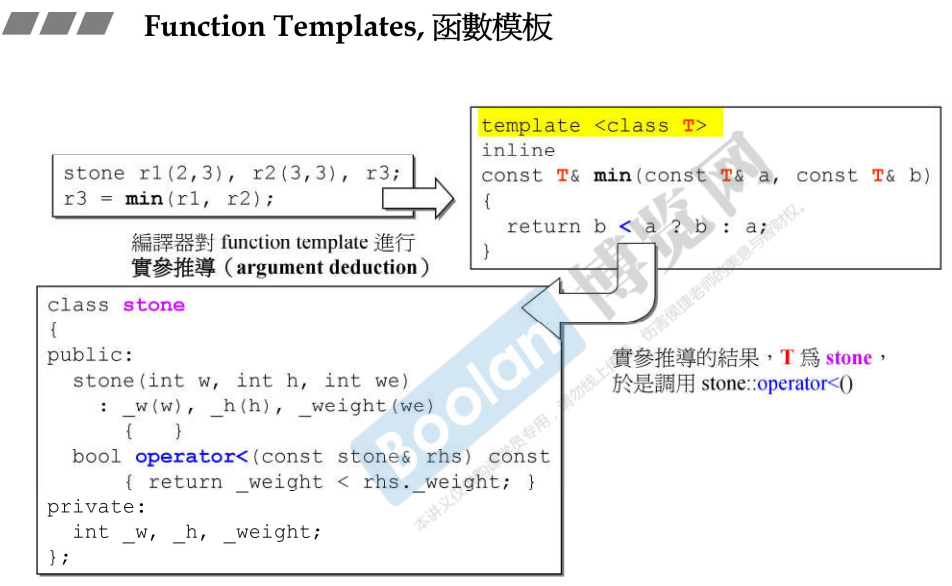
3、

类模板



4、

函数模板



这里的关键字class 和前面的 typename是一样的功能。

实参推导是指将r1和r2传入后编译器可以知道是stone类，既T就是stone

5、

成员模板，比较复杂，不去讲

6、

类模板：泛化、特化Specialization（全特化）

泛化：

template <class type> struct \_\_type\_traits { ……};

特化：

template <> struct \_\_type\_traits<int> { ……};

template <> struct \_\_type\_traits<double> { ……};

调用：

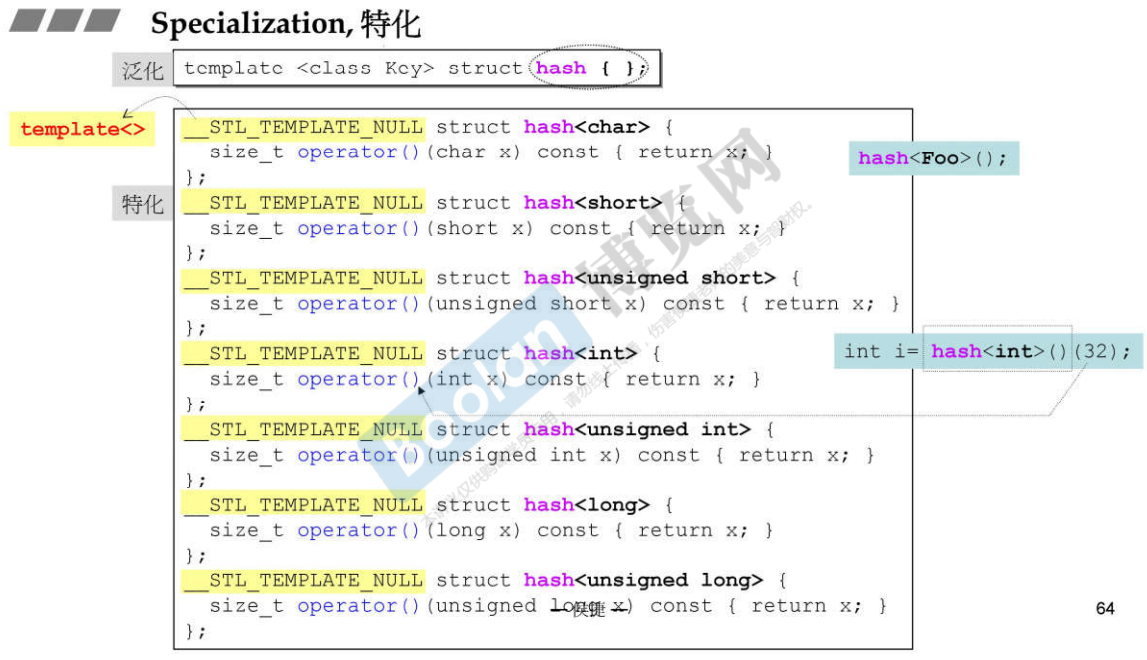
\_\_type\_traits<Foo>::has\_trivial\_destructor //泛化

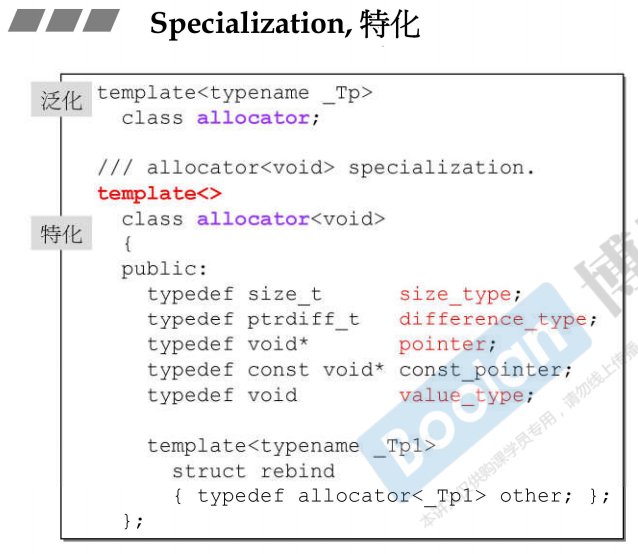
\_\_type\_traits<int>::has\_trivial\_destructor //特化

\_\_type\_traits<double>::has\_trivial\_destructor //特化

7、

一些例子，特化在标准库中存在很多例子

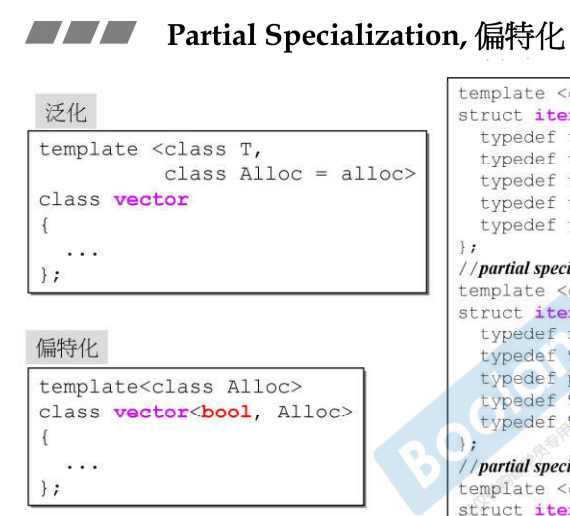




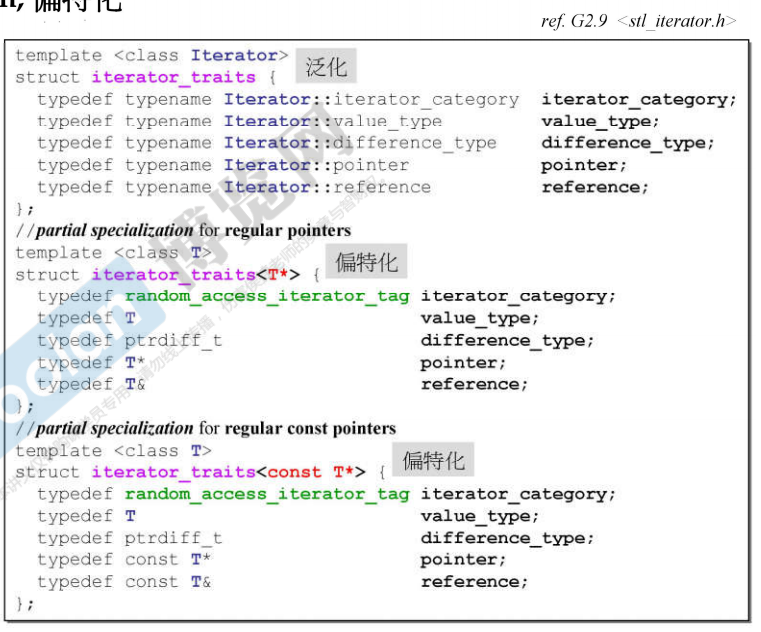
最重要是语法的形式，template<>

8、

偏特化，相比于泛化只有一些模板参数被使用，还有其他模板参数被直接指定。

这里是个数的偏特化

9、



范围的偏特化，传入指针，指向什么不重要