**Chpater 15 友元，异常**

1. **友元类 friend void Remote**

**友元函数：类的拓展坞**

**友元类：类中所有方法都可以访问原始类中的私有成员和保护成员，非is-a, has-a**

类中某些函数可以像友元一样（某些不是友元）

1. Eg. 电视机和遥控器
2. **先声明/定义，后使用 （难点）**
3. **互相依赖时，采用前向声明forward declaration**
4. **互为友元类/共同的友元**
5. 互为友元类：class TV & class Remote，分别有成员 friend class Remote和friend class TV
6. 部分是友元函数：不是全部函数都有必要成为友元函数拉低效率，具体实现可以参考listing15.4， listing15.5， listing15.6
7. 共同的友元：一个函数需要分别存取恋歌不同类别的私有资料，相对更合理的做法是是此函数成为两个类别的友元。

Eg: class Analyzer;

Class Probe

{

Friend void sync (Analyzer & a, const Probe & p); //sync a to p

Friend void sync (Probe & p, const Analyzer & a); //sync p to a

};

Class Analyzer

{

Friend void sync (Analyzer & a, const Probe & p); //sync a to p

Friend void sync (Probe & p, const Analyzer & a); //sync p to a

};

//define the friend functions

Inline void sync (Analyzer & a, const Probe & p)

{

…

}

Inline void sync (Probe & p, const Analyzer & a)

{

…

}

1. **嵌套类**

**一个类声明在另一个类中**

1. **嵌套类与包含的不同：**

嵌套类不创建类成员，只是定义了一种类型

包含是将某个类的成员作为另一个类的成员

1. **嵌套类避免了名称的重复**
2. **包含类只能显示地访问嵌套类对象的公有成员**
3. **嵌套类的存取**与基本的private，protected，public规则无异

其declaration的位置决定了它的使用范畴

1. **异常exception**

**异常处理一定会降低程序的运行速度**

1. **解决问题的办法之一：呼叫abort()**

函数abort() 位于cstdlib中，典型实现是向标准错误流发送消息abnormal program termination至标准错误流standard error steam并终止程序

如果需要清除档案缓冲区的资料则可呼叫exit()

1. **解决问题的办法之二：return错误码，使用函数的返回值来指出问题**
2. 一般返回输入的字符的ASCII码，但是遇到程序终结时则返回特殊值EOF
3. **异常如何处理问题**
4. C++异常是对程序运行过程中发生异常的一种响应，通常包括三步：

引发异常throw an exception（try块包含可能出现异常的代码块）；

捕获异常，并提供刚处理程序（catch块 ）；

使用try块

1. Throw表示引出异常，紧随其后的值（字符串/类对象）指出了异常的特征
2. Catch区块：

Catch表示异常的拦截，catch后的括号内标示它要return的类型，大括号内是处理异常的代码

1. Try区块：

标示一段可能产生特定异常的代码块，之后是一个或多个catch块。

Try区块从关键字try开始，后面是大括号包住的代码表示会处理其产生的异常

1. **将对象作为异常类型**

一般来说，函数丢出的异常属于对象。其中一个好处在于，可以使用不同的异常形态，来分辨产生异常的不同的函数和情况，然后决定使用哪个catch块

1. **栈解退unwinding the Stack**

Try块内函数多次调用，就一直往回弹，查看是否接下来的一步是catch，如果是就执行，如果不是，就继续往回弹

相当于在一个多重包含之中寻找一个线头

1. **抛出异常时，编译器通常会创建一个临时对象的拷贝，即使异常定义时和catch块中使用的是引用**
2. **异常对象之间有继承关系**时，需要注意catch块的顺序，应该以继承的逆序来安排，即派生类应该在前，基类在后->主要是由于基类指针可以访问派生类对象
3. **可以定义catch块是省略号，以适配任何异常**
4. **异常之exception类**
5. **Exception可以作为异常处理的基类**，用户代码也可以直接抛出exception对象，定义在exception库里
6. Stdexcept头文件
7. Logic\_error（公有继承自exception类）

Domain\_error定义域

Invalid\_error函数值传递的异常

Length\_error字符串长度不够

Out\_of\_bounds指示索引错误

1. Routine\_error（公有继承自exception类）

Range\_error值域

Overflow\_error下溢，计算结果比浮点型可以表示的最小值还小

Underflow\_error上溢，计算结果比浮点型可以表示的最大值还大

1. **New头文件**

**对于new引发的异常问题，让new引发bad\_alloc异常（由exception类公有派生）**

1. **Unexpected exception**

这种情况没有包含的try块，没有匹配的catch块，通常会被识别为为拦截的异常uncaught exception，预设的做法是结束运行。