## 基类和派生类

### 一、基类

**基类的特点**：必须已经定义而不仅仅是声明，才能被当成基类使用。分为直接基类和间接基类。定义一个虚析构函数。

**终极类**：不能作为基类的类，定义时在类名后面加上一个关键字final。

**虚析构函数**：不执行任何实际操作。552.15.7.1

**虚函数**： 派生类需要自己新定义与自己相关的特定操作来覆盖旧定义。

**虚函数特点**：当用指针或者引用调用虚函数时，可能执行基类版本或者某个派生类版本的函数。

**虚函数定义**：在函数前面加上关键字 virtual 。虚函数不能是构造函数，也不能是静态函数。

virtual 声明只能出现在类内声明，而不是类外定义

**受保护成员**：由 protected 受保护的访问运算符修饰，被修饰成员能被派生类访问，其他用户禁止访问。

### 二、派生类

**派生类列表**：用于指明派生了继承了哪些基类，每个基类都有一个访问说明符修饰，形式为：class Derived : pubic Base1 {\\*....\*\}。

**单继承和多重继承**：

**虚函数的覆盖**：如果派生类中不重新定义函数，那么派生类会直接继承其在基类中的版本。定义时可以加virtual，用于显式地注明它使用某个成员函数覆盖了它继承的虚函数。538.15.3

**派生类的组成部分**：一个含有自己定义的成员的子对象、一个与该派生类继承的基类对应的子对象。若果是多重继承，那么就有多个基类子对象。

**派生类到基类的类型转换**：编译器隐式转换，意味着派生类对象的引用可以作为基类类型的引用使用，指针也和引用一样能用。

**派生类的构造函数**：需要借助基类的构造函数初始化基类的成员，然后再初始化自己的成员。派生类构造函数通过构造函数初始化列表来将实参传递给基类的构造函数，否则基类将进行默认初始化。

**派生类的静态成员**：和基类中的静态成员是同一个。

**派生类的声明**：用法和类的声明一样，不能包含派生列表。

### 三、基类与派生类的类型转换

**目的**：将基类的指针或引用绑定到派生类对象上，并且不存在基类向派生类的转换。

**静态类型**：在编译时已知，它是变量声明时的类型或表达式生成的类型。

**动态类型**：变量或表达式表示的内存中的对象中的类型，知道运行时才可知。

NOTE：静态类型和动态类型只会在***表达式***是引用或者指针指针时才会发生。

**派生类对象向基类对象转换**：可以在派生类内部定义一个显示类型转换函数，也可以利用基类的拷贝构造函数。不管怎么样，派生类的独有成员会丢失。

**基类向派生类转换**：可以用static\_cast强制转换，但基类中可能会存在一些未初始化变量

## 虚函数