第十四章　继承和组成关系

1. 组成关系 composition syntax

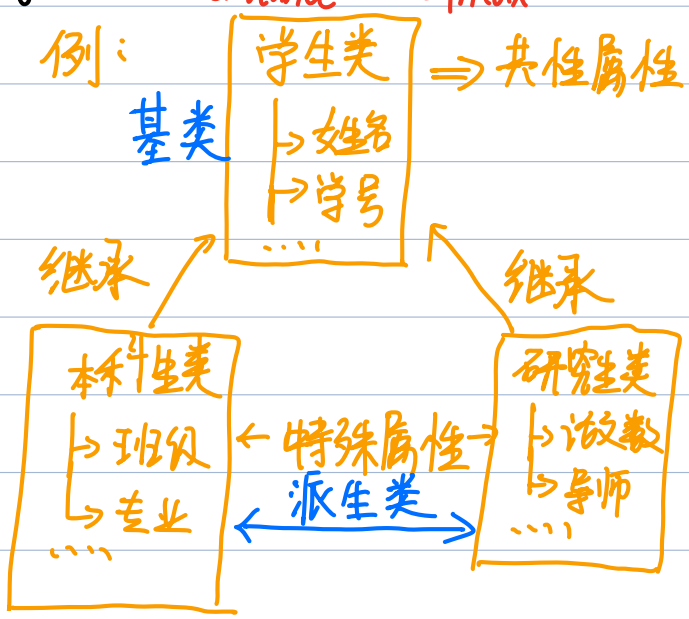
组成关系一般是指在一个类中嵌套定义其它类的对象做为成员变量，实现了两个类一内一外的组合应用

聚合的关系 'has a' Relationship

'is a' Relationship

2. 继承关系 inheritance syntax

例：



被继承的类称为基类（父类），其成员变量为共性属性；

继承的类称为派生类（子类），其成员变量为特殊属性；

子类可以访问父类的所有非私有成员变量或函数 → public protect

[相当于子类内包含]父类的全部内容.

语法：

class subclass : 继承类型 base-class

public/protected/private，默认为 private

{类体};

对于多继承类型，此处为父类列表

关于继承类型的说明：

① public：父类中的成员类型将继承给子类不改变

父		子
public	→	pubic
protected	→	protected
private	→	private

② protected：父类中的非私有成员将以protected类型继承给子类

父		子
public	→	protected
protected	→	protected
private	→	private

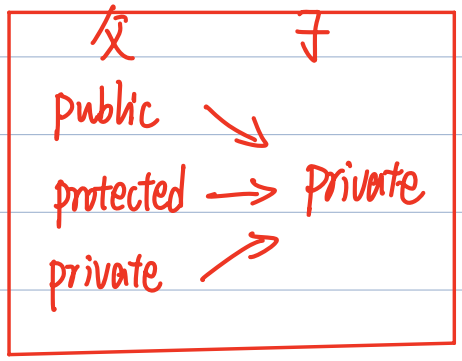
protected类型成员有这样的特点：

可以被该类及其子类的函数直接访问

无法被该类及其子类的对象直接访问

访问	public	protected	private
同一个类	yes	yes	yes
派生类	yes	yes	no
外部的类	yes	no	no

⑤ Private：父类中的成员类型将以 private 类型继承给子类



※ 注意：子类会继承父类的大多数成员，但不会继承

① 基类的构造函数、析构函数

因此在构建子类的构造函数和析构函数时

请在初始化列表中显式调用基类的构造/析构函数

类似于组合类中操作

```
class Shape
{
   public:
      Shape(int w,int h)
      {
        width=w;
        height=h;
      }
   protected:
      int width;
      int height;
};

// 派生类
class Rectangle: public Shape
{
   public:
      Rectangle(int a,int b)
      {
        Shape(a,b);
      }
```

错误

Shape 函数在子类中不存在

```
// 基类
class Shape
{
   public:
      Shape(int w,int h)
      {
        width=w;
        height=h;
      }
   protected:
      int width;
      int height;
};

// 派生类
class Rectangle: public Shape
{
   public:
      Rectangle(int a,int b):Shape(a,b)
      {

      }
```

正确

※ 子类在构建时的执行顺序：先是构建基类

注意：基类/内嵌类的构造参数需要从子类的构造函数中传入。

再构建子类内嵌套类

最后构建子类

子类在析构时的执行顺序：先析构子类

再析构子类内嵌套类

最后析构父类

② 父类的重载运算符和拷贝构造函数

因此在子类中使用父类的重载运算符时

请使用 this 指针和作用域 Base::

③ 基类的友元函数

因此请在子类中重新定义友元函数

3. 继承中的类型转换

父类

↑ upcasting

子类

在 C++ 中，子类可以自动隐式转换为父类，该行为是安全的、正确的

（因为子类中至少包含了所有父类的成员）

补充：C++ 中的几种类型转换

① 静态转换

static_cast<类型>（表达式）

该操作类似于 (类型)表达式

但会多一步安全检查

判断是否兼容

③常指针/常引用转换

const_cast <类型&/类型*> (表达式)

实现常量 → 非常量

非常量 → 常量 的转换

③动态指针/动态引用转换

dynamic_cast <类型&/类型*> (表达式)

该操作符多用于父类和子类的强制转化

且会进行转换安全检查. ↓要求类中有虚函数

④重译强制转换

reinterpret_cast <类型> (表达式)

该操作将强制将表达式进行类型转换

但不进行安全检查

4.多继承

即一个子类可以有多个父类，它继承了多个父类的特性

语法：class <子类名>：<方式1><父类名1>,<方式2><父类2>

{ <派生类体> }

父类列表

// 三种权：private
protected
public

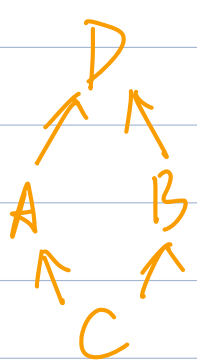
多继承子类可以以不同的方式继承各基类

※ 若多继承类的多个父类中有冲突的（重名…的）函数，会使得子类在调用函数时产生二意性。

Solution 1：用户在调用时加函数作用域
obj.A::Function( );

Solution 2：在派生类中自行定义一个同名函数而编译器在调用函数时会优先调用派生类内的同名函数

※ 菱形继承的问题
当有四个类以如下方式存在和继承



而当C类对象在调用D中的成员时，由于C内相当于存在两个D类对象，故二意性产生（一份自A，一个来自B）

Solution：使用虚继承关键字virtual

格式：`class 类名: virtual 继承方式 父类名`

```
class D{......};
class B: virtual public D{......};
class A: virtual public D{......};
class C: public B, public A{.....};
```

该关键字用在最派生类的父类上

virtual关键字声明使得当最派生类对象创建时，由此对象向上创造所有祖先类对象而非递归的由父类创造祖先类对象