1. 继承与派生
2. 派生类（is\_a）

吸收基类成员、改造基类成员、添加新成员

Class car: public vehicle{

Int load;

Public:

……

}

派生类会继承基类的成员函数与数据成员类型，并且可以增加新的自己的数据成员

1. 继承方式

//继承方式影响的是子类对继承的对象的访问属性，而不影响父类的访问权限

|  |  |  |  |
| --- | --- | --- | --- |
| 继承方式 | 基类访问属性 | 基类在派生类中的访问属性 | 类外的成员函数 |
| 公有继承（public） | + | + | + |
| 私有继承（private） | + | - | - |
| 保护继承（protected） | + | + | - |

1、派生类访问基类的数据成员时，可以访问public中的成员函数与protected中的数据成员，但是不能访问private中的数据成员

解决的方法是在基类中增加访问自己private数据的public函数，之后它可以被继承至派生类中，派生类可以通过此函数来访问基类的private

2、当派生类中定义了一个与基类函数同名的函数时，访问时会优先访问新定义的函数，可以使用作用域标识符在函数前加上作用的类来使用基类的函数

3、派生类的初始化与析构：

派生类的构造函数不能从基类继承。

派生类的初始化不能仅仅为新增的参数赋值，也要为基类中的数据成员赋值

Housing（int f ,int r ,int a,int p, int e）

:building（f , r , a）

//冒号后的初始化列表应使用类名而非数据名

1. 多继承

1、一个派生类可以同时继承多个基类，且多级继承时下级会继承所有上级类的函数 Class C: public A , public B{……}

2、但是多继承可能会导致二义性问题

二义性：在多条继承路径上有一个公共基类（菱形◇继承关系），下级派生只对直接基类初始化，但是调用直接基类时会间接调用根部基类的构造函数，从而导致根部基类被多次初始化，产生数据冗余，从而导致数据访问时的二义性乃至多义性

1. 二义性的解决方法：虚基类

——将公共基类设为虚基类，在整个类家族中只被初始化一次

虚基类的初始化：虚基类的初始化需要单独进行

首先调用虚基类的构造函数（比普通基类的构造优先级更高），且调用时层层递进（构造虚基类时候可能会需要构造其他类）若有多个，按照声明顺序

虚基类的声明：

Class bed: virtual public furniture{……}

1. 组合类（has\_a）
2. 当类A成为类B的组合类时，类B可以访问类A的public函数

类B可以具有类型为类A的数据成员

但类B不可以直接访问类A的数据成员

1. 组合类的初始化使用对象名来初始化而非类名

派生类的初始化才使用类名来初始化

1. 相较于父对象，子对象更后被构造，更后被析构

即基类--->派生类--->子对象

五、类与类的关系

1、依赖关系（uses-a）

类A的成员函数的参数类型或返回值类型为类B，则类A依赖于类B

在简化表达中使用横线表示依赖关系，用数字表示对应数量，用\*表示无限

1. 关联关系

Ex. Student(5..60)--------\*course(1..3)-------(1)teacher

表示一个课程有5~60个学生，一个学生可以选无数门课程，

一个课程对应一个教师，一个教师可教授1~3门课程。

1. 继承关系（is-a）
2. 组合关系（has-a ◇------）
3. 聚合关系（is-a type of ◆------）

聚合关系中的组成部分可以独立于组合之外存在

而组合关系不可以，比如“name”不可以独立于“person”存在

==>继承与组合的优点在于“渐增式开发（incremental development）”

即先创建少部分功能的类，再通过继承与组合引进新代码，而不是一次性地构造它

这样的操作有益于程序的维护，只需要修改出问题的部分即可

//一个类既是组合类又是派生类时

构造函数的初始化：

基类—>子对象—>派生类新增成员

六、小结与注意

Attention1、继承方式与其使用特点

Attention2、派生类构造函数初始化

Attention3、虚基类的使用

Attention4、子对象的初始化

Attention5、指针、引用与直接赋值

指针和引用类型只是要求了基地址和这种指针所指对象的内存大小，与对象的类型无关，相当于把指向的内存解释成指针或引用的类型。

而把一个派生类对象直接赋值给基类对象，就牵扯到对象的类型问题，编译器就会回避之前的的虚机制。从而无法实现多态。