第八章

1. 对象的概念：①任何一个对象都有属性和行为这两个要素，对象由一组属性和一组行为构成。②一个系统中多个对象通过一定的渠道互相联系，想要摸个对象实现某个操作，需要向它传送相应信息。③任何对象都由数据与函数两部分组成，数据即“属性”，函数即操作代码，对数据进行操作以实现某些功能
2. 对象可以对信息做到封装的效果以保证信息隐蔽，即把实现功能的代码放在内部封装，只留下少量对外接口和外界联系
3. 程序设计中有“抽象”一词，类是对象的抽象，对象是类的特例，即类的具体表现形式（抽象的作用是表示同一类事物的本质）
4. 在已经拥有一些功能的代码上进行添加、修改以优化代码，叫做继承与重用（软件重用），可以达到方便效果
5. 多态性：多态性是类似的对象对相同信息做出不同反应，可以达到增加程序灵活性的效果
6. 程序设计者设计程序要考虑各种类和对象（将数据和操作封装在一起），以及怎样向对象发出信息
7. 在基于过程的结构化程序设计中（c语言）程序=算法+数据结构。基于对象和面向对象程序设计为：对象=算法+数据结构，程序=（对象\*n）+消息。消息的作用就是对对象的控制
8. 面向对象的软件开发过程
9. 面向对象分析，即先考虑系统改做什么
10. 面向对象设计，对每一个部分进行具体的设计，此时主要要把思路和方法形成
11. 面向对象编程，就是写出来
12. 面向对象测试，软件写好后要进行测试以发现错误
13. 面向对象维护，类似于bug修复，版本更新
14. 用结构体来类比对象的话，结构体类型（类），结构体（对象）
15. 声明与定义类类型的方法与声明结构体类型相似，区别在于类类型中分有private部分和public部分，且有函数
16. 定义类类型时没有指定private和public时，默认为private，但此时与外界没有接口
17. 在类外定义类的成员函数时，必须在函数名前加类名（限定）和“∷”（作用域限定符）
18. C++自动将类中函数内置，达到高效效果
19. 成员函数的储存方式：一个类定义的多个对象的成员函数对应的是同一个函数代码段，以达到节约内存的效果（c++内有this指针防止弄混）。注：①不论成员函数在类内定义还是类外定义，其储存方式是相同的，不占用对象储存空间。②inline函数只影响程序执行效率，与函数占用储存空间无关
20. 虽然成员函数没有放到对象的储存空间内，但逻辑上成员函数和数据是封装在一起的
21. 对象成员的引用类似结构体，用“.”，用指针引用对象成员用“->”，也可以用引用来访问对象成员
22. 外界只能通过公用函数对私有数据进行调用以实现功能，公用函数是类的对外接口，此为将接口与实现分离
23. 只要类的接口没有改变，对私有数据的修改不会引起程序其他部分的修改
24. ？？C++程序由三部分组成：①类声明头文件（后缀为.h）②类实现文件（后缀为.cpp）③类的事故题部分文件（即主文件，后缀为.cpp）
25. 系统提供的头文件中只包括对成员函数的声明而不包括成员函数的定义，否则会造成时间浪费
26. 类库有两种，一是c++本身有的，二是用户自己编的。（类库有两个组成部分，一是包括类声明的头文件，二是已经编译的成员函数的定义）
27. 面向对象程序设计中名词：类的成员函数叫“方法”；对成员函数发送命令叫做“消息”

第九章

1.如果一个类中所有成员都是公用的，可以在定义对象时对数据成员进行初始化，如果有private成员或protected成员就不可以用这种方法

2.构造函数可以对对象中的私有数据进行初始化，它是一种特殊的成员函数，不需用户调用，而是在建立对象时自动执行，且没有类型，没有返回值（使用注意：构造函数必须和类名同名，不用在定义构造函数时声明类型）

3.构造函数可以带参数，用来直接给数据初始化，使用方法为构造函数名（类型 形参，类型 形参...）

类名 对象名（实参，实参...）

4.构造函数还有参数初始化表的数据成员初始化方法，

构造函数名（参数表）：成员初始化｛构造函数体｝（注意：数组只能在函数体内定义不能再参数表中初始化，可以在类外对构造函数进行定义，但是要在构造函数名前加 类名∷）

1. 构造函数的重载：在一个类中定义多个构造函数，以他们的参数数量不同来进行区分，然后对数据进行不同的初始化（注意：1.构造函数名字要相同。2.没有给出实参的构造函数叫默认构造函数，一个类只能有一个默认构造哈桑农户，并且在定义对象时正确写法为time out;而不是time out();）
2. 构造函数中参数可以不用实参传递，直接指定为默认值（注意：声明构造函数时就指定默认值）
3. 析构函数的名字时类名前加~，作用是在撤销对象的内存前做清理工作，即最后一次使用对象之后做一些事情（注意：1.静态局部对象在调用函数时不释放，所以析构函数只在main函数结束或调用exit函数时才调用析构函数2.析构函数无参数，所以不能被重载）
4. 有关调用析构函数和构造函数的顺序：先构造的后析构，后构造的先析构

Std student[3]={

Std(1,2,3);

Std(4,5,6);

Std(7,8,9);}

1. 存在对象数组,初始化方式：
2. 对象的指针就是对象空间的起始地址，定义的一般形式为 类名\*对象指针名
3. 定义指向对象成员函数的指针变量的方法有些特殊，指向普通函数的指针：类型名（\*指针变量名）（参数列表），而在定义指向对象函数成员指针变量时：类型名（类名∷\*指针变量名）（参数列表），使其指向一个共用函数的形式为：指针变量名=&类名∷成员函数名
4. 可以在定义对象时加const称为常对象。常对象中数据要有初值，且无法修改。（类名 const 对象名（实参表）=const类名...）

常对象只能调用它的常成员函数，不能调用普通成员函数（常成员函数不能修改常对象中数据的值）注：一定要修改某数据值时，可以在定义该数据时在类型名前加mutable

1. 在对象中的数据定义时加const称为常数据成员，只能用构造函数初始化表对其赋初值，其余任何函数不能对其赋值
2. 常成员函数只能调用本类中数据而不能修改，定义方法为：类型名 函数名（参数表）const

总结：

|  |  |  |
| --- | --- | --- |
| 数据成员 | 非const成员函数 | Const成员函数 |
| 非const数据成员 | 可以引用，可以改变值 | 可以引用，不可以改变值 |
| Const数据成员 | 可以引用，不可以改变值 | 可以引用，不可以改变值 |
| Const对象 | 不允许 | 可以引用，不可以改变值 |

1. 指针变量声明为const型，其指向就不可改变，形式为：

类名\*const指针变量名

1. 指向常对象的指针变量形式：const类型名\*指针变量名，注意：①常变量只能用指向常变量的指针指向它。②指向常变量的也可以指向普通变量，但此时不能通过指针改变其值。（此时只是用指针访问变量期间变量有常变量的特征，其他情况下变量仍可以改变）③time \*const p指向对象的常指针变量，const time \*p指向常对象的指针变量

总结：

|  |  |  |  |
| --- | --- | --- | --- |
| 形参 | 实参 |  | 改变变量值 |
| 指向非const变量指针 | 非const变量地址 | 合法 | 可以 |
| 指向非const变量指针 | const变量地址 | 非法 | / |
| 指向const变量指针 | const变量地址 | 合法 | 不可 |
| 指向const变量指针 | 非const变量地址 | 合法 | 不可 |

指向常对象的指针变量概念与此类似，注意：希望调用函数时对象的值不被修改，就把形参定义为为指向常对象的指针变量，并用对象的地址做实参。

1. 如果形参为变量的引用，实参为变量的值，那么函数中对形参中数据的改变实际就是对实参的改变

Const总结：

|  |  |
| --- | --- |
| 形式 | 含义 |
| Time const t | t是常对象，其值不能改变 |
| Void time ∷fun()const | Fun是time类中常成员函数，可以引用，但不能修改本类中数据成员 |
| Time \*const p | p是指向time类对象常指针变量，p的指向不能改变 |
| Const time \*p | p是指向time类常对象的指针变量，p指向的类对象的值不能通过p改变 |
| Const time &t1=t | t1是t的引用，t值不能改变 |

1. 可以用new运算符建立对象，用delete运算符撤销对象。New运算符非配空间后返回的是地址，可以用指针接收地址
2. 对象之间可以用=赋值（量对象必须是同类）注意：不能有动态分配的数据，不然会原地爆炸
3. 对象可以复制，形式为：类名 对象2（对象1），也可以用赋值的形式进行赋值：类型 对象2=对象1
4. 想要某变量在所有对象中值是一样的，可以把它定义成静态数据成员。形式为在定义时的类型前加static（①静态变量在函数结束时并不释放②静态成员可以初始化，但只能在类体外初始化，不能用参数初始化表对静态数据成员初始化，形式为：数据类型 类型∷静态数据成员名=初值）
5. 静态成员函数的定义形式也是在定义时加staic，静态成员函数是类的一部分而不是对象的一部分。若要在类外调用公用静态成员函数，要用“类名∷”也可以用“对象名.”调用。注意：静态成员函数没有this指针。为达方便，静态成员函数主要访问静态数据成员，不访问非静态成员
6. 在声明函数前加friend就是友元函数，友元函数可以访问对象中的私有数据，友元函数可以是一般函数，也可以是其他类的成员函数，使用方法p287
7. 在类名前加friend就是友元类，友元类中的所有函数都可以是另一个类的友元函数。①友元关系时单向的。②友元关系不能传递
8. 声明类模版时要加一行：template<class 类型参数名>，且定义类时原有的类型名都要换成类型参数名。可以理解为类模版是类的抽象，类是类模版的实例。
9. 声明类时class后面是模版名而不是类名，所以定义对象时应该在模版名后面加上类型名再写对象名，即：类模版名 <类型名> 对象名（参数表）

总结使用类模版的方法：

1. 先写一个实际的类
2. 将此类中打算改变的类型名该用一个自己指定的虚拟类型名
3. 在类声明前加一行temlate <class 虚拟类型参数>
4. 用类模版定义对象：类模版名<实际类型名>对象名（实参表）
5. 若在类外定义成员函数：

函数类型 类模版名<虚拟类型参数>∷成员函数名（函数形参表）｛｝