《C++面向对象程序设计》期末考试选择题部分答案解析

张玉帅（小奥）整理

2018.11.13

首发于：www.yushuai.xyz

欢迎关注

1.关于复制构造函数，下列说法正确的是

A.系统不会生成缺省复制构造函数，因此必须自己实现

B. 复制构造函数是形如 X::X(X)的函数

C. Myclass c1, c2;　c1.n = 1;　c2 = c1;第三句将会调用复制构造函数

D.调用函数 A Func() {　A a(4);　return a;　}时，将会调用 A 的复制构造函数

解析：

只要你不写构造函数，系统就会自动帮你生成缺省的构造函数，包括复制构造函数，故 A 错误。复制构造函数的格式是 **X::X(X&)**或 **X::X(const X &)**，故 B 错误。复制构造函数起作用主要有以下三种情况：

1)当用一个对象去初始化同类的另一个对象时；

2)如果某函数有一个参数是类 A 的对象，那么该函数被调用时，类 A 的复制构造函数将被调用；

3) 如果函数的返回值是类 A 的对象时，则函数返回时，A 的复制构造函数被调用。

由此可得，C 错误，D 正确（第 3 种情况）。

2.关于虚函数，下列说法不正确的是

A.不允许以虚函数作为构造函数

B.没有虚函数便无法实现多态

C.一般来讲，如果一个类中定义了虚函数，则不可将析构函数也定义为虚函数

D.不能用抽象类定义对象

解析：

在学习虚函数的时候，明确说过构造函数和静态函数不能使用 virtual，即不能成为虚函数，虚函数是实现多态的基础，在实现多态的时候，析构函数也经常会被定义为虚函数，抽象类是可以定义对象的。这是一个概念题。

3.关于 this 指针，以下说法不正确的是

A. static 成员函数内部不可以使用 this 指针

B. 在构造函数内部可以使用 this 指针

C. 在析构函数内部可以使用 this 指针

D. const 成员函数内部不可以使用 this 指针

解析：

一个对象的 this 指针并不是对象本身的一部分，不会影响 sizeof（对象）的结果。this 作用域是在类内部，当在类的非静态成员函数中访问类的非静态成员的时候，编译器会自动将

对象本身的地址作为一个隐含参数传递给函数。友元函数没有 this 指针，因为友元不是类的成员。只有成员函数才有 this 指针。静态成员函数是类的一部分，作用是为了处理静态数据成员，它没有 this 指针，静态成员函数可以直接访问该类的静态成员，但不能直接访问类中的非静态成员。构造函数、析构函数、const 成员函数都是成员函数，故可以使用 this 指针，静态成员函数因为其设计的原因，不可以使用 is 指针。

4.以下关于多态的说法那个不正确?

A.在成员函数中调用虚函数，是多态

B.通过“基类对象名.函数名"的方式调用虚函数，不是多态

C.多态的函数调用语句中，函数一定是虚函数

D.通过“基类引用名.函数名"的方式调用虚函数，是多态

解析：

多态的表现形式有以下几种：

1）通过基类指针调用基类和派生类中的同名虚函数时：

①若该指针指向一个基类的对象，那么被调用的是基类的虚函数；

②若该指针指向一个派生类的对象，那么被调用的是派生类的虚函数。

2）通过基类引用调用基类和派生类中的同名虚函数时：

①若该引用引用的是一个基类的对象，那么被调用的是基类的虚函数；

②若该引用引用的是一个派生类的对象，那么调用的是派生类的虚函数。

故 B、D 正确，A 错误。要实现多态，函数一定是虚函数，这是定义，故 C 正确。

5.map 的每个元素包括 KEY(first）和 VALUE(second)。关于 map 容器，下列哪种说法错误

A. map 支持下标运算符

B. map 的不同元素可以有相同的 VALUE

C. map 支持 STL 的 sort 算法

D. map 支持双向迭代器

解析：

map 的设计问题，不做解释，详情可以查看 map 的具体解析就会明白。

6.下列说法错误的是

A.可以在一个类的友元函数中使用 this 指针

B.每个类只有一个析构函数

C.抽象类至少包含一个纯虚函数

D.构造函数不可以是 virtual 函数

解析见第 3 题。

7.关于继承和派生的描述中，下列说法错误的是:

A.派生类的成员函数中，不能访问基类的 private 成员

B.在派生类的析构函数执行之前，会先调用基类的析构函数

C.派生类对象的地址可以赋值给基类指针

D.派生类可以有和基类同名同参数的成员函数

解析：

继承的权限如下：

若继承方式是 public，基类成员在派生类中的访问权限保持不变，也就是说，基类中的成员访问权限，在派生类中仍然保持原来的访问权限；

若继承方式是 private，基类所有成员在派生类中的访问权限都会变为私有(private)权限；

若继承方式是 protected，基类的共有成员和保护成员在派生类中的访问权限都会变为保护(protected)权限，私有成员在派生类中的访问权限仍然是私有(private)权限。

故 A 正确。C 正确，实践中已经实现。D 可以通过多态来实现。B 的正确过程应该是：构造函数先执行基类构造函数，再执行派生类的构造函数，析构函数先执行派生类的构造函数，再执行基类的构造函数。

8.以下哪种使用 std::sort 算法的方式是不合法的：

A. vector<int> a; …; sort(a.begin(), a.end());

B. bool b[99]; …; sort(b, b + 99);

C. string c = “2333”; …; sort(c.begin(), c.end());

D. list<int> d; …; sort(d.begin(), d.end());

解析：

A/B/C 均可以实现，list 应该使用 mylist.sort();这种格式来进行排序。

9.类 A 重载的运算符声明是 int operator<(A &other) const，那么以下说法中正确的是：

A.小于号左侧的 A 对象不可以是 const 的

B.小于号右侧的 A 对象不可以是 const 的

C.这个写法是错误的，因为小于号的返回类型必须是 bool

D.使用小于号的时候，other 参数处，传进来的对象实际上会被复制一次

解析：

该运算符使用时类似于 c.operator<(other)。小于号左侧的 A 对象就是这个表达式中的 c，是可以 const，也可以不 const 的，但是小于号右侧的 A 对象是不能为 const 的，因为若左侧的 A 对象不是 const，根据非 const 成员函数可以访问非 const 对象的非 const 数据成员、const 数据成员，但不可以访问 const 对象的任意数据成员，那么一旦右侧为 const 的，这个比较就无法进行了，故 A 错误，B 正确。

返回值为 bool 类型可以为 int，故 C 错误，声明中参数是引用，故不会被复制一次，D 错误。

10.以下 STL 中的函数模板哪个可以作用于 set

A. sort

B. random_shuffle

C. find

D. 都不行

解析：

见 set 的使用说明。