

《C++面向对象程序设计》模拟试题（7）

一、单选题(共 20 分，每题 2 分)

- 下列哪个不是 C++标识符的作用域：
(A) 块作用域 (B) 对象作用域 (C) 类作用域 (D) 名字空间作用域
- 假设有函数调用 `f(0)` 可以编译成功，则 `f` 的原型不可能是：
(A) `void f(int&)` (B) `void f(A a)` (C) `void f(A* a)` (D) `void f(const A*)`
- 下列函数原型声明正确的是：
(A) `float Volume(int x, y);` (B) `float Volume(int x, int& y=0);`
(C) `const float Volume(int, float);` (D) `const float Volume(int x=10, int y);`
- 类 A 的非静态成员函数中的隐含参数 `this` 的类型为：
(A) `const A&` (B) `const A *` (C) `A* const` (D) `A&`
- 下面关于静态成员的描述中，不正确的是：
(A) 静态成员函数可以直接访问所属类的全部数据成员。
(B) 静态数据成员的类型可以是其所属类。
(C) 静态数据成员可以被作为类成员函数的缺省实参。
(D) 静态成员可以被所属类及其派生类对象所共享。
- 下列关于友元的描述中，不正确的是：
(A) 友元函数可直接访问类中私有成员。
(B) 友元函数是成员函数，它被声明在类体内。
(C) 嵌套类可以成为其外围类的友元。
(D) 友元类中的所有成员函数都是友元函数。
- 在局部变量的作用域内可用下面哪种方法对被其隐藏(overwrite)的同名全局变量进行访问：
(A) 名字空间 (B) 虚基类 (C) 引用 (D) 作用域限定符
- 下列关于动态联编的描述中，不正确的是：
(A) 动态联编是以虚机制为基础的。
(B) 动态联编是编译时确定所调用的函数代码的。
(C) 动态联编是运行时确定所调用的函数代码的。
(D) 只有通过对象指针或对象引用来调用虚函数，才能实现动态联编。
- 假设有如下类定义 `class B { /*略*/ }; class A { B b; /*略*/ }; 则类 A、B 之间最可能的关系是：
(A) 依赖关系 (B) 聚合关系 (C) 组合关系 (D) 泛化关系`
- 下列关于虚函数的描述中，正确的是：
(A) 一个类中定义了纯虚函数,则必定是抽象类，并且不能含有数据成员。
(B) 类中可以定义不带 `const` 修饰的纯虚函数，但不能定义带 `const` 修饰的纯虚函数。
(C) 如果一个类中所有自定义成员都是纯虚函数，则这个类无意义，也不应定义这样的类。
(D) 虚函数表只存放虚函数的入口地址，但不存放函数的具体代码。

二、判断正误，对于你认为错误的论述，说明原因或举出反例。（共 20 分，每题 2 分）

- 1. 使用没有初始化的指针，是一种语法错误。
- 2. 运算符的优先级可以通过重载来改变。
- 3. 在形如 `x=expression;` 的赋值语句中，`x` 的值总是等于右端表达式 `expression` 的值。
- 4. 当定义一个类时，其数据成员的存储空间即被分配。
- 5. 类的私有成员只能由该类的成员函数直接访问。
- 6. 类的静态成员函数不能声明为常函数，构造函数和析构函数也不能。
- 7. 可以将基类对象看作派生类对象。
- 8. 继承用以改进数据隐藏和封装。
- 9. 通过单参构造函数可以把用户自定义类型的对象转换成内置类型的数据。
- 10. 抽象类中的所有成员函数必须声明为纯虚函数。

三、回答下列各题（每题 3 分，共 15 分）

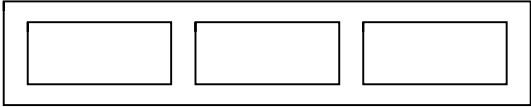
1. 下面类中两函数是重载函数么？如果是，如何决定调用哪个函数？

```
class A {
public: char fun(int x, int y);
       char fun(int x, int y) const;    };

```

2. 请编写例子程序，说明浅复制只复制指针数据成员的指针值。
3. 请举例说明哪些情况下必须使用构造函数的初始化列表进行显式初始化？（至少 4 种）
4. 请用面向对象方法分析动物(Animal)、虎(Tiger)、鸟(Bird)、企鹅(Penguin)、鸽子(Pigeon)、鸽群(Pigeon Group)、飞翔(Fly)之间的关系。
5. 现在把奶农和奶牛看作两个类，奶农可以给奶牛挤奶。若有 A、B 两类变化，A：奶农会有多种不同类型，如熟练奶农和实习奶农等；B：奶牛会有多种不同类型，如奶用型黑白花牛、中国荷斯坦奶牛等。若每类变化都需要改变挤奶行为的实现，请说明把挤奶这个行为分配给奶农，可以更方便地适应哪类变化？把挤奶这个行为分配给奶牛，可以更方便地适应哪类变化？如何适应？

四、(5 分)在制作绘图程序时，需要绘制各种图形元素，包括矩形、椭圆等。现需要增加一种图形元素—Grid，每个 Grid 本身就是一个矩形，但其文本内容一定为空，同时一个 Grid 还包含多个矩形，矩形个数由创建 Grid 时的参数指定，示例如图。请在下边给出的类的基础上，定义并实现类 Grid。

<pre>class Shape { public: virtual ~Shape(){} virtual void Draw()=0; }; </pre>	<pre>Class Ellipse:public Shape{ public: virtual ~ Ellipse () {} virtual void Draw () { /*略 */ } }; </pre>
<pre>class Rect:public Shape { public: Rect (const char * text) { /*略 */} virtual ~ Rect () {} virtual void Draw () { /*略 */ } }; </pre>	<div style="text-align: center;"> Grid 示例</div>

五、分别写出下面两个程序的运行结果（共 10 分）

1.

<pre>class A { public: A(int m=0):n(m) { } void Add() { n++; } A * SP() { return this; } int TheVal(A *s) { return s->n; } private: int n; };</pre>	<pre>void main() { A *p[2]; A a1(1); A * pa2 = new A(a1); p[0]=a1.SP(); p[1]=pa2; p[0]->Add(); pa2->Add(); p[1]->Add(); cout<<p[1]->TheVal(p[0])<<endl; cout<<p[0]->TheVal(p[1])<<endl; delete pa2; }</pre>
--	---

2.

<pre>class Data { public: Data(int x=0) { Data::x=x; cout<<"Data()"<<endl; } Data(const Data&) { cout<<"Data(const Data&)"<<endl; } ~Data() { cout<<"~Data()"<<endl; } private: int x; };</pre>	<pre>class Child : public Base { public: Child() { cout<<"Child()"<<endl; } virtual ~Child() { cout<<"~Child()"<<endl; } private: Data d2; };</pre>
<pre>class Base { public: Base() { cout<<"Base()"<<endl; } Base(const Base&) { cout<<"Base(const Base&)"<<endl; } virtual ~Base() { } };</pre>	<pre>int main() { Child c1; Child c2(c1); return 0; }</pre>

六、(共 10 分) 分油问题是一个古典数学问题。问题可描述为： 现有 3 个容量均为整数的瓶子，初始时，容量最大的瓶子(容量为偶数)是满的，其它两个为空瓶。问： 如何通过这 3 个瓶子，将初始的油平分为两份？

例： 容量分别为 10 斤、7 斤、3 斤的三个瓶子，10 斤瓶子初始是满的。经过：10 斤倒 7 斤，7 斤倒 3 斤，3 斤倒 10 斤，7 斤倒 3 斤，3 斤倒 10 斤，7 斤倒 3 斤，10 斤倒 7 斤，7 斤倒 3 斤，3 斤倒 10 斤。这时，10 斤瓶和 7 斤瓶，就是各装 5 斤。

现需要编写一个程序验证分油过程是否正确，请回答：

1. 请给出 **Bottle** 类的定义和实现。
2. 利用上边的 **Bottle** 类，给出能够验证题中例子的 **main** 函数。

七、（10 分）一个正实数 R 总可以用唯一的连分数形式表示，即 $R = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$ ，其

中为 a_0 非负整数， a_i 为正整数 ($i > 0$)。下面给出了连分数(CFraction)类的部分定义和主程序代码，请给出 CFraction 类的完整定义和实现，使得给定的主程序能够输出预期结果(如果必要，可以在 CFraction 类中增加其它成员函数)。

```
class CFraction {
public:
    CFraction(const unsigned int data[ ], int theCount);
    ~CFraction( ) ;
    //取前 n 项对应分数的分子和分母, n>0 且 n≤count
    void GetFraction( int& num, int & den, int n) const;
private:
    const unsigned int count; //项数
    unsigned int * items; //各项的整数值
};

int main( ) {
    const unsigned int data[ ]={ 3,7,15,1,292,1,1,1,2};
    CFraction pi( data, sizeof(data) / sizeof(unsigned int) );
    int num, den; //分子和分母
    pi. GetFraction ( num,den,2);
    cout<<"前 2 项对应的分数="<<num<<"/"<<den<<endl; //预期输出: 22/7
    pi. GetFraction ( num,den,4);
    cout<<"前 4 项对应的分数="<<num<<"/"<<den<<endl; //预期输出: 355/113
    return 0;
}
```

八、（10 分）客户请小王编写一个从键盘读入字符并输出到打印机的程序。小王使用结构化的设计方法，编写了如下代码。该程序包含 3 个子程序，Copy 子程序从 ReadKeyboard 子程序中获取字符，并把字符传递给 WritePrinter 子程序。几个月后，客户说有时希望 Copy 程序能从纸带读入机中读入字符。并且，在未来可能还会从其它种类的输入设备中读入字符，也可能输出到其它种类的输出设备。小王希望他写出的代码既能满足上述要求，又不用每次都改写 Copy 的实现。请你帮小王重新设计 Copy 函数，添加必要的类，使得新设计能够满足小王的愿望。简要说明你的设计思想，给出实现代码。

```
void Copy( ) {
    int c;
    while ((c=ReadKeyboard( )) != EOF) //EOF 为宏定义的终结字符
        WritePrinter(c);
}
```

（全卷完）