2006-2007 学年第 1 学期

2005 级《C++面向对象程序设计》期末考试试题（A 卷）

考试时间：2007 年 1 月 13 日

班级 _____ **学号**_____ **姓名**_____

✧ 本试卷满分 100 分；
✧ 请将答案写在答题纸上，写明题号，不必抄题，字迹工整、清晰；
✧ 请在答题纸和试题纸上都写上你的班级，学号和姓名，交卷时请将试题纸、答题纸和草纸一并交上来。

一、单选题(共 10 分，每题 1 分)

1. C++中解决命名冲突的机制是：D
   (A) 虚基类　(B) 虚函数　(C) 函数重载　(D) 名字空间
2. 若类 A 的一个对象所占的内存空间中包含虚函数表的入口地址,则: C
   (A) 类 A 不能有静态数据成员　(B) 类 A 中公有的成员函数一定是虚的
   (C) 类 A 中至少有一个成员函数是虚的　(D) 类 A 的析构函数一定是虚的
3. 任意一个类，析构函数的个数最多是：B
   (A) 不限个数　(B) 1　(C) 2　(D) 3
4. 下列关于 this 指针的说法,哪个是正确的: D
   (A) this 指针一定指向常量型数据　(B) this 指向的数据不可更改
   (C) 静态成员函数中也可以访问 this 指针　(D) this 指针本身可直接作为成员函数的返回值
5. 在类定义中,为说明成员的访问权限,private, protected, public 可以出现次数为：A
   (A)次数没有具体限定　(B) 每种至多一次
   (C) **public** 至少一次　(D) 每种至少一次
6. 下面哪种定义方式是正确的，并且使得 p 可以作为函数 void f(A* const pp);的实参：A
   (A) A * p = new A;　(B) A a; A* p = a;
   (C) const A* p = new A;　(D) A a; const A* p = a;
7. obj 是类 A 的一个对象，执行语句 const A& aA= obj; ，则下列说法正确的是： C
   (A) 类 A 的拷贝构造函数会被调用　(B) 类 A 的赋值函数会被调用
   (C) &aA 的值就是 &obj　(D) 语句 obj.f( )；等价于语句 aA.f( )；
8. 下面关于访问类 A 的私有数据成员的说法，错误的是： C
   (A) 类 A 的友元函数可以访问类 A 的私有成员。
   (B) 类 A 的友元类中的非静态成员函数可以访问类 A 的私有成员。
   (C) 类 A 的嵌套类中的非静态成员函数可以访问类 A 的私有成员。
   (D) 类 A 中的非静态成员函数可以访问类 A 的私有成员。
9. 类 A 中有唯一的一个成员函数 f，且 f 是公有的静态或非静态成员函数，对于类 A 的一个对象 a,执行语句 a.f(100);成功，那么 f 的函数原型不可以是：B
   (A) A& f(int, int=50);　(B) void f(int&) ;
   (C) const A * f(const int);　(D) A f(const int&);

10. 下面关于类的成员函数描述不正确的是：A

(A) 静态成员函数内可以直接访问类的非静态成员数据

(B) 静态成员函数内可以直接访问类的静态成员数据

(C) 非静态成员函数可以直接访问类的非静态成员数据

(D) 非静态成员函数可以直接访问类的静态成员数据

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

1. 重载流操作符<<和>> 时，如果第一个参数的类型为 ostream 和 istream，那么这个重载函数既可以用于标准输入输出流，也可以用于文件流上。

对，ostream 和 istream 是标准输入输出流、文件流、字符串流的基类

2. 在同一个类中，可以定义重载的成员函数 void f(int);和 virtual void f(int); 。

错，这属于重复定义

3. 抽象类不会产生实例，所以不需要有构造函数。

错，被派生时需要它的构造函数

4. 类 A 有一个非静态的成员函数 f，其函数原型是：void A::f( ) const，则该函数被调用时，一定是通过类 A 或类 A 的某后裔类的一个用 const 修饰符说明的常量对象调用的。

错，常函数可以由变量对象或常量对象调用

5. 异常必须在其产生的当前函数中捕获，而不能在外层函数中捕获该异常。

错，可以在外层捕获，并且这是最常见的用法

6. 只要程序中没有 A a1 = a2; 和 A a1(a2); 形式的代码，类 A 的拷贝构造函数就不会被调用。

错，参数传递或函数返回时也调用拷贝构造函数

7. 在 protected 继承方式下，派生类对象的指针不能直接转换成指向基类对象的指针。

对，否则基类中的公有成员由不可见变为可见，权限被放大

8. 若静态成员函数中调用了一个函数 f，那么 f 一定不是虚函数。

对，静态成员函数不能是虚函数，因为虚函数入口需要在保存在对象中的虚函数表中，而静态成员函数不属于对象。

9. 若要实例化一个含有引用型数据成员的类，那么只能使用构造函数初始化列表来初始化该数据成员。

对，没有别的办法

10. 构造函数的函数体中，不能使用 return 语句；但在实现该类的自动类型转化函数时，必须有 return 语句。

对

三、回答下列各题（每题 4 分，共 20 分）

1. 举例说明 static 关键字的用法和相应目的(至少 3 种)。

f(){ static int a;…} 函数体内的静态变量，每次调用该函数时值保持不变

static int a; 全局的静态变量，约束作用域为所在文件

class A {static int a;…}; A 的静态成员，类似全局变量，需用 A::a 访问

2. 举例说明类的数据成员在哪些情况下必须在初始化列表中进行初始化(至少 3 种)。

基类不提供无参的构造函数

成员对象不提供无参的构造函数

有常量成员或引用成员

3．举例说明虚拟继承的作用和目的。

虚拟继承的目的是使基类在派生类中只保留一个副本

从而避免二义性

4．举例说明成员函数 A& f( ) const;和成员函数 A& f( ); 的区别。

A& f( ) const 是常函数，隐含的 this 指针是常指针，因此在 f 中不能修改对象成员的值。

举例略

5．有类 A 的对象 a，任意给出一种解决方案，使得程序支持下面的表达式：

a=10+a;

```
class A {
    A(int);  //转换构造函数
    friend const A operator+(const A,const A); //重载+
};
```

四、指出下列程序代码中存在的错误或不足，说明原因。（每题 5 分，共 10 分）

1.

```
#include<iostream.h>
class A  {
public:
 virtual ~A( ) { }
 virtual void f( )
  { cout<<"A::f( )"<<endl; }
 virtual void g( )
  { cout<<"A::g( )"<<endl; }
};
```

```
class B:public A
{
public:
 virtual void g( )
   { cout<<"B::g( )"<<end
   l;}
 virtual void k( )

{ cout<<"B::k( )"<<endl;}
};
```

```
void main( )
{
      A * p= new B;
      p->f( );
      p->g( );
      p->k( );
      delete p;
}
```

答：函数 k 在 A 中没有定义,执行 p->k( );时要根据 p 的类型在 A 中查 k 的信息

2.

```
#include<iostream.h>
#include<string.h>
class A;
class B:public A {
public:
    B(const char* info){
      m_buf=new char[256];
      strcpy(m_buf,info);  }
   ~B( ) { delete[] m_buf; }
   virtual void output( ) { cout
<< m_buf; }
private:
  char * m_buf; };
```

```
class A {
public:
   ~A( ) { }
   virtual void output( ) { }
};
void main()
{
   A*pa = new B("hello!") ;
   pa->output( );
   delete pa;
}
```

答：A 的析构函数应定义为虚函数，否则 B 的析构函数不会被调用，m_buf 也不会被释放

五、写出下面程序的运行结果（每题 5 分，共 10 分）

1.

```
#include <iostream.h>
class A {
public:
   A( ):count(1) {}
   virtual ~A( ) {}
   virtual  A*  Copy(  )
const = 0;
   virtual  void  Out(  )
const = 0;
protected:
   int count;
};
```

```
class B:public A {
public:
   ~B(   )      {  --count;
Out( );  }
   virtual A* Copy( ) const
{
      B *p = new B(*this);
      ++p->count;
      return p;
   }
  virtual void Out( ) const
  { cout << count << endl; }
};
```

```
void main( )
{
 {
  B b;
  A* a1=&b;
  a1->Out( );
  a1          =
a1->Copy( );
  a1->Out( );
  delete a1;
 }
}
```

答：

1

2

1

0

2.

```
#include <iostream.h>
class A  {
public:
    A(int n):num(n)    { Out( ); }
    A(const  A&  rhs):num(rhs.num)
{Out( );}
    void Out( ) {cout<<num<<endl; }
public:
    int num;
};
```

```
class B:public A  {
public:
     B(A& a) :obj(a),A(1) { }
     void Out( ) {  obj.Out( ); }
private:
     A  obj;
};
```

```
void main( ) {
   A   a(8);
   B   b1(a);
   B   b2(b1);
   b2.Out();
}
```

答：

8

1

8

1

```
8
8
-------------
```

六、阅读下面两个类的定义和部分实现代码，完成 3 个问题。（共 10 分）

```
#include<iostream.h>
class A  {
public:
   A(int n):value(n) {  }
   void Display( ) const
  {cout<<"Value            =
"<<value<<endl; }
private:
     int value;
};
```

```
class B {
public:
  B(int n);
  void   Display(    )
const
     {  aA.Display();
 }
private:
     A  aA;
};
```

```
int main( )
{
    B b1(1);
    b1.Display( )
;
    B b2(2);
    b2.
Display( );
    return 0;
}
```

1. [3 分]实现类 B 的构造函数，使得程序的输出为：

   Value=1

   Value=2

答：B(int n):aA(n)

2. [3 分]若 main 函数中增加了语句 B b3(b1); 针对本例，说明是否有必要以公有方式自定义并实现类 B 的拷贝构造函数，为什么？

   答：不需要，因为类 B 及基类 A 中不存在引用或指针成员，使用默认的拷贝构造函数就可以。

3. [4 分]在不改动类 A 和 main 函数的前提下，以继承的方式重新定义并实现类 B，使得程序的输出结果不变。

答：

```
class B:public A {
public:
  B(int n):A(n);
  void Display( ) const
     { A::Display(); }
};
```

七、(共 20 分，每问题 10 分) 某程序员为了灵活地对各种的给定的曲线函数 f(x)画出其曲线图形，设计并部分实现了一个曲线类 curve，该类的成员数据中，count 代表坐标点的个数，pxs 代表的数组存放这些坐标点的横坐标，pys 代表的数组存放利用 f(x)计算得到的这些坐标点的纵坐标。由于不同曲线的计算公式 f(x)是不同的，该程序员希望曲线函数的种类可以通过继承 curve 类的方式任意增加，增加一个新的 f(x)时不改变 curve 类中的内容，也不改变利用 curve 类进行图形绘制的算法。已部分完成的 curve 类定义和实现如下：

```
class  curve  {
public:
    void setPxs( ) {/*把获取的横坐标数据存放在 pxs 代表的数组中，并为 count 置值*/ }
    double* getPxs( ) const { return pxs;}
    int getCount( ) const { return count;}
```

```
    double* getPys( ) const ;
private:
    double* pxs;
    double* pys;
    int count
};
```

1、请按照该程序员的设计意图给出成员函数 getPys 的完整实现。实现过程中，可以为 curve 类增加其它成员。可以假设 setPxs 函数已经完整实现，不需要考虑曲线的绘制和显示。

答：

```
class  curve  {
public:
    void setPxs( ) {/*把获取的横坐标数据存放在 pxs 代表的数组中，并为 count 置值*/ }
    double* getPxs( ) const { return pxs;}
    int getCount( ) const { return count;}
    double* getPys( ) const ;
    virtual double f(double)=0;
private:
    double* pxs;
    double* pys;
    int count
};
double* curve::getPys( ) const
{
    if(pys==NULL) pys=new double[count];
   for(int i=0;i<count;i++)
      pys[i]=f(pxs[i]);
   return pys;
}
```

注：严格来讲，还应在析构函数中释放 pxs 和 pys，但这部分不作为要考的语法点，故此处略掉

2、以曲线函数：f(x)=3*x*x+2*x+1.为例，从 curve 派生一个类 curve1，并用文字说明其它的函数（如：main 函数）如何利用基类 curve 中的 getPys 函数获取该曲线的纵坐标值。

答：

```
class curve1 : public curve
{
 virtual double f(double x) {return3*x*x+2*x+1; }
};
```