继承与派生

1.概念填空题

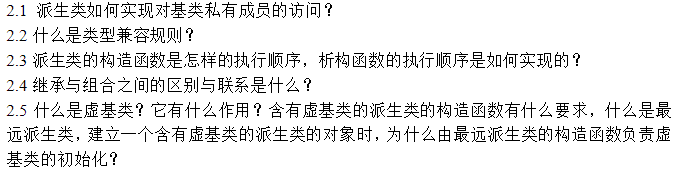
1.1在C++中，三种派生方式的说明符号为 public 、 private 、 protected 不加说明，则默认的派生方式为 私有 。

1.2当公有派生时，基类的公有成员成为派生类的 公有成员 ；保护成员成为派生类的 保护成员 ；私有成员成为派生类的 不可见 。当保护派生时，基类的公有成员成为派生类的 保护成员 ；保护成员成为派生类的 保护成员 ；私有成员成为派生类的 不可见 。

1.3 派生类的构造函数一般有3项工作要完成：首先 调用基类构造函数 ，其次 构造派生类的成员对象 ，最后 执行构造函数体内的内容 。

1.4多继承时，多个基类中的同名的成员在派生类中由于标识符不唯一而出现 二义性 。在派生类中采用 类名限定 或 同名隐藏规则 来消除该问题。

2.简答题



答：2.1：一是通过访问基类的公有成员函数来访问；二是将基类的私有成员改为保护成员。

2.2：即一个公有派生类的对象在使用上可以被当作基类的对象。

2.3派生类构造时，先调用基类的构造函数，调用顺序按照他们被继承时声明的顺序；再调用成员对象的构造函数，调用顺序按照他们在类中的声明顺序；最后执行构造函数体内的内容，析构函数的执行顺序与构造函数相反。

2.4 联系：都是用已有类来定义新的类，都是为了实现代码的重用；

区别：继承是“是”的关系，派生类与基类是特殊和一般的关系，组合是“有”的关系，表现的是整体和部分的关系。

2.5当在多条继承路径上有一个公共的基类，在这些路径中的某几条汇合处，这个公共的基类就会产生多个实例，若只想保存这个基类的一个实例，可以将这个公共基类说明为虚基类。

作用：避免二义性，在内存中只有一份

带有虚基类的派生类的构造函数的成员初始值表中含有虚基类的构造函数，并对其虚基类的子对象只初始化一次。

最远派生类是指在继承结构中建立对象时所指定的类，因为无法使用派生路径上的各个构造函数 ？

3．选择题

3.1下面对派生类的描述中，错误的是（ D ）。

A．一个派生类可以作为另外一个派生类的基类

B．派生类至少有一个基类

C．派生类的成员除了它自己的成员外，还包含了它的基类的成员

D．派生类中继承的基类成员的访问权限到派生类中保持不变

3.2下列对友元关系叙述正确的是（A ）。

A．不能继承

B．是类与类的关系

C．是一个类的成员函数与另一个类的关系

D．提高程序的运行效率

3.3当保护继承时，基类的（B ）在派生类中成为保护成员，不能通过派生类的对象来直接访问。

A．任何成员 B．公有成员和保护成员

C．公有成员和私有成员 D．私有成员

3.4设置虚基类的目的是（ B ）。

A．简化程序 B．消除二义性

C．提高运行效率 D．减少目标代码

3.5在公有派生情况下，有关派生类对象和基类对象的关系，不正确的叙述是（ C ）。

A．派生类的对象可以赋给基类的对象

B．派生类的对象可以初始化基类的引用

C．派生类的对象可以直接访问基类中的成员

D．派生类的对象的地址可以赋给指向基类的指针

3.6有如下类定义：

class MyBASE{

int k;

public:

void set(int n) {k=n;}

int get( ) const {return k;}

};

class MyDERIVED: protected MyBASE{

protected;

int j;

public:

void set(int m,int n){MyBASE::set(m);j=n;}

int get( ) const{return MyBASE::get( )+j;}

};

则类MyDERIVED中保护成员个数是（ B ）。

A．4 B．3 C．2 D．1

3.7程序如下：

#include<iostream>

using namespace std;

class A {

public:

A( ) {cout<<”A”;}

};

class B {public:B( ) {cout<<”B”;} };

class C: public A{

B b;

public:

C( ) {cout<<”C”;}

};

int main( ) {C obj; return 0;}

执行后的输出结果是（ D ）。

A．CBA B．BAC C．ACB D．ABC

3.8类O定义了私有函数F1。P和Q为O的派生类，定义为class P: protected O{…}； class Q: public O{…}。( C )可以访问Fl。

A． O的对象 B． P类内 C． O类内 D． Q类内

3.9有如下类定义：

class XA{

int x;

public:

XA(int n) {x=n;}

};

class XB: public XA{

int y;

public:

XB(int a,int b);

};

在构造函数XB的下列定义中，正确的是（ A ）。

A．XB::XB（int a，int b）：x(a)，y(b){ }

B．XB::XB（int a，int b）：XA(a)，y(b) { }

C．XB::XB（int a，int b）：x(a)，XB(b){ }

D．XB::XB（int a，int b）：XA(a)，XB(b){ }

4．写出程序运行结果

4.l#include<iostream>

using namespace std;

class B1{

public:

B1(int i){ cout<<”constructing B1 “<<i<<endl; }

~B1( ){ cout<<”destructing B1 “<<endl; }

};

class B2 {

public:

B2( ){ cout<<”constructing B3 \*”<<endl; }

~B2( ){ cout<<”destructing B3”<<endl; }

};

class C:public B2, virtual public B1 {

int j;

public:

C(int a,int b,int c):B1(a),memberB1(b) ,j(c){}

private:

B1 memberB1;

B2 memberB2;

};

int main( ){

C obj(1,2,3);

}

运行结果：

constructing B1 1

constructing B3 \*

constructing B1 2

constructing B3 \*

destructing B3

destructing B1

destructing B3

destructing B1

如果把virtual去掉，则结果将为：

constructing B3 \*

constructing B1 1

constructing B1 2

constructing B3 \*

destructing B3

destructing B1

destructing B1

destructing B3