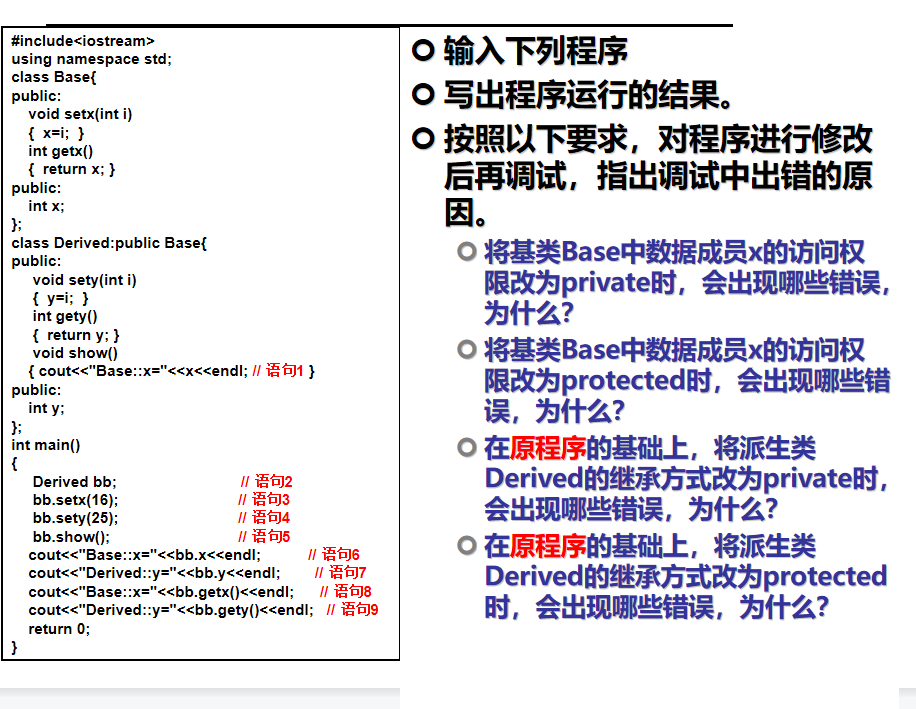
一、实验题目

上机实验5：派生类与继承



二、实验过程

———————————————原始代码：———————————————

#include<iostream>

using namespace std;

class Base{

public:

void setx(int i)

{ x=i; }

int getx()

{ return x; }

public:

int x;

};

class Derived:public Base{

public:

void sety(int i)

{ y=i; }

int gety()

{ return y; }

void show()

{ cout<<"Base::x="<<x<<endl; // 语句1 }

public:

int y;

};

int main()

{

Derived bb; // 语句2

bb.setx(16); // 语句3

bb.sety(25); // 语句4

bb.show(); // 语句5

cout<<"Base::x="<<bb.x<<endl; // 语句6

cout<<"Derived::y="<<bb.y<<endl; // 语句7

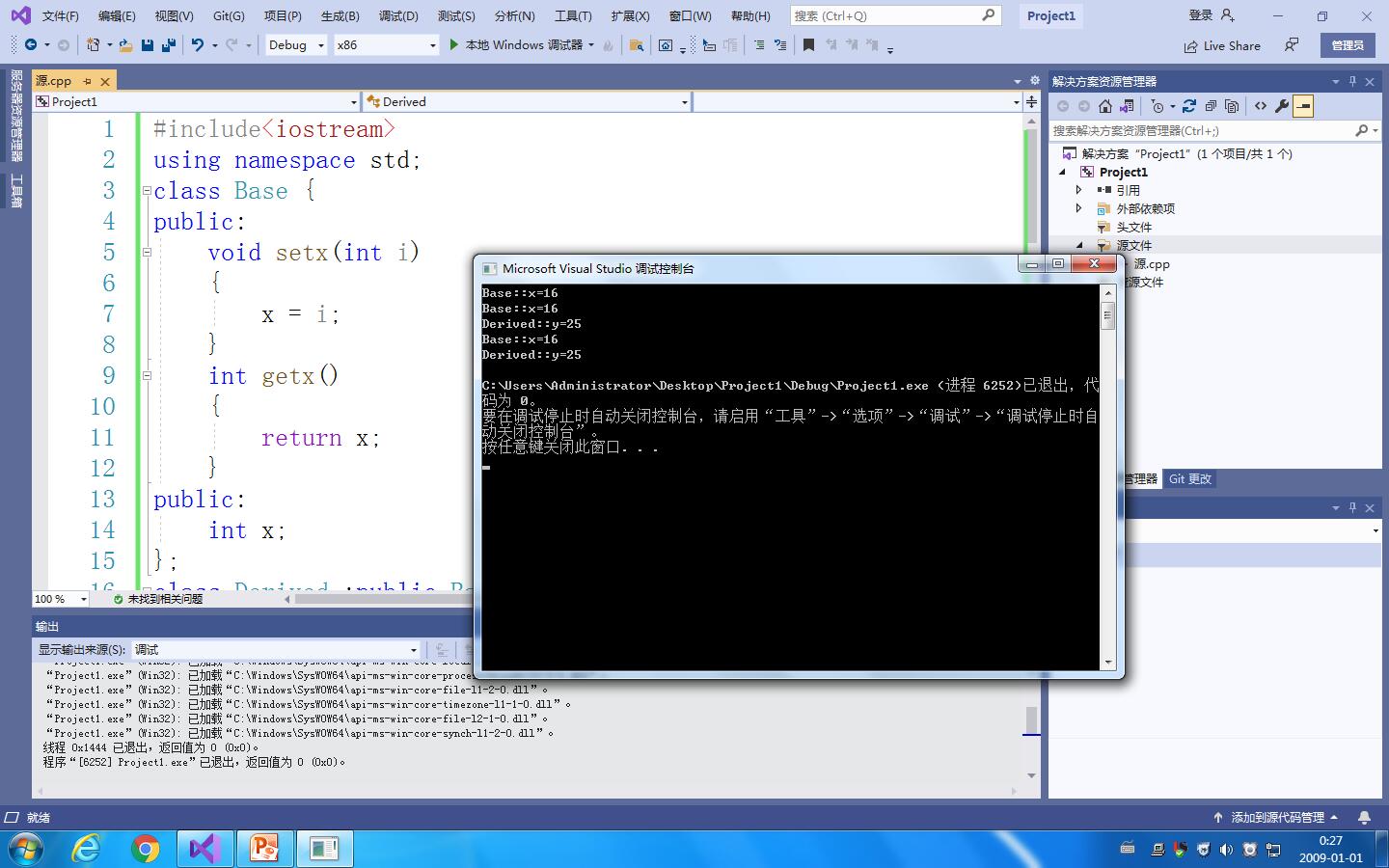
cout<<"Base::x="<<bb.getx()<<endl; // 语句8

cout<<"Derived::y="<<bb.gety()<<endl; // 语句9

return 0;

}

———————————————运行结果：———————————————



—————①将基类Base中数据成员x的访问权限改为private时—————

class Base{

public:

void setx(int i)

{ x=i; }

int getx()

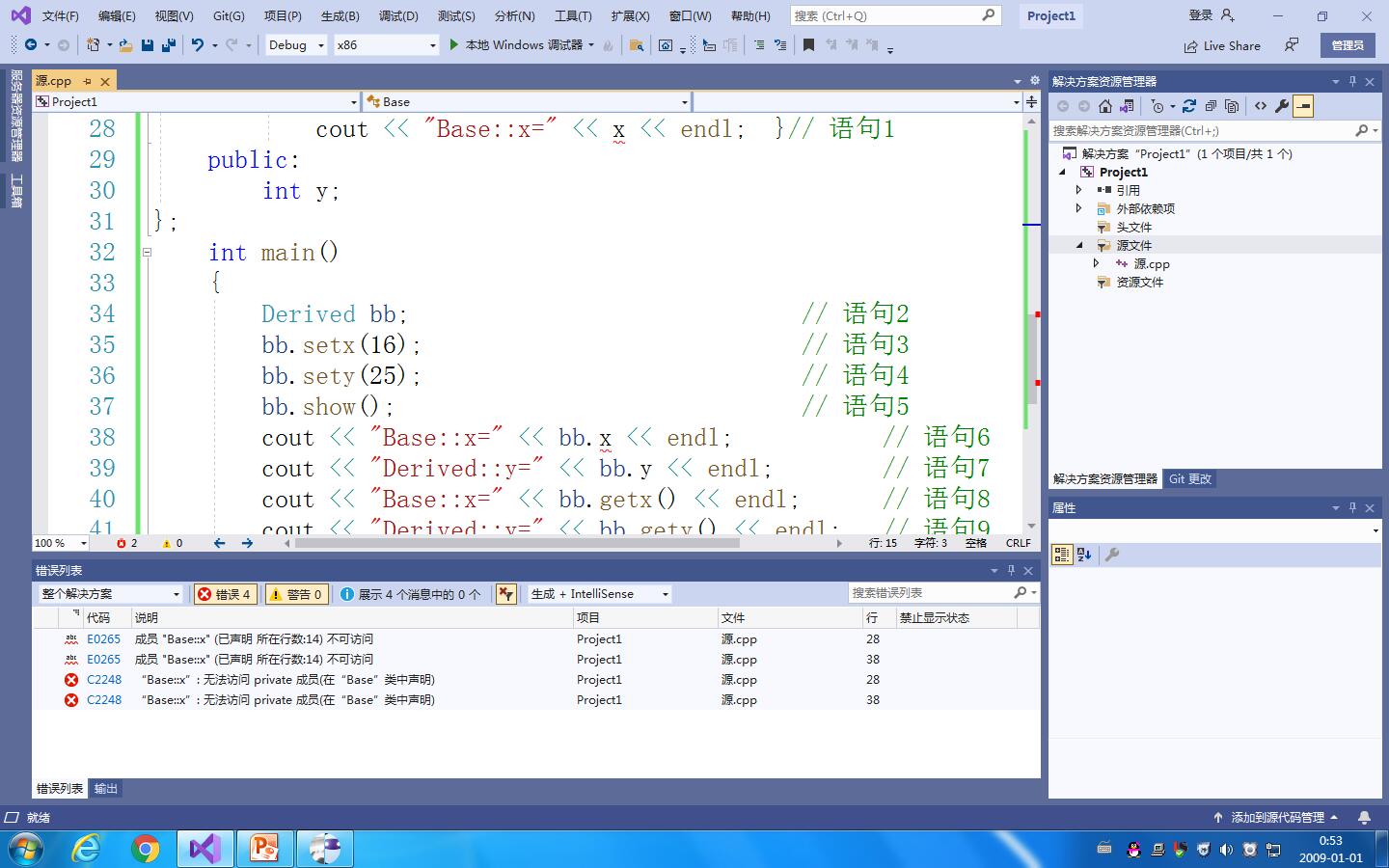
{ return x; }

private:

int x;

};

———————————————①运行结果：——————————————



————②将基类Base中数据成员x的访问权限改为protected时—————

class Base{

public:

void setx(int i)

{ x=i; }

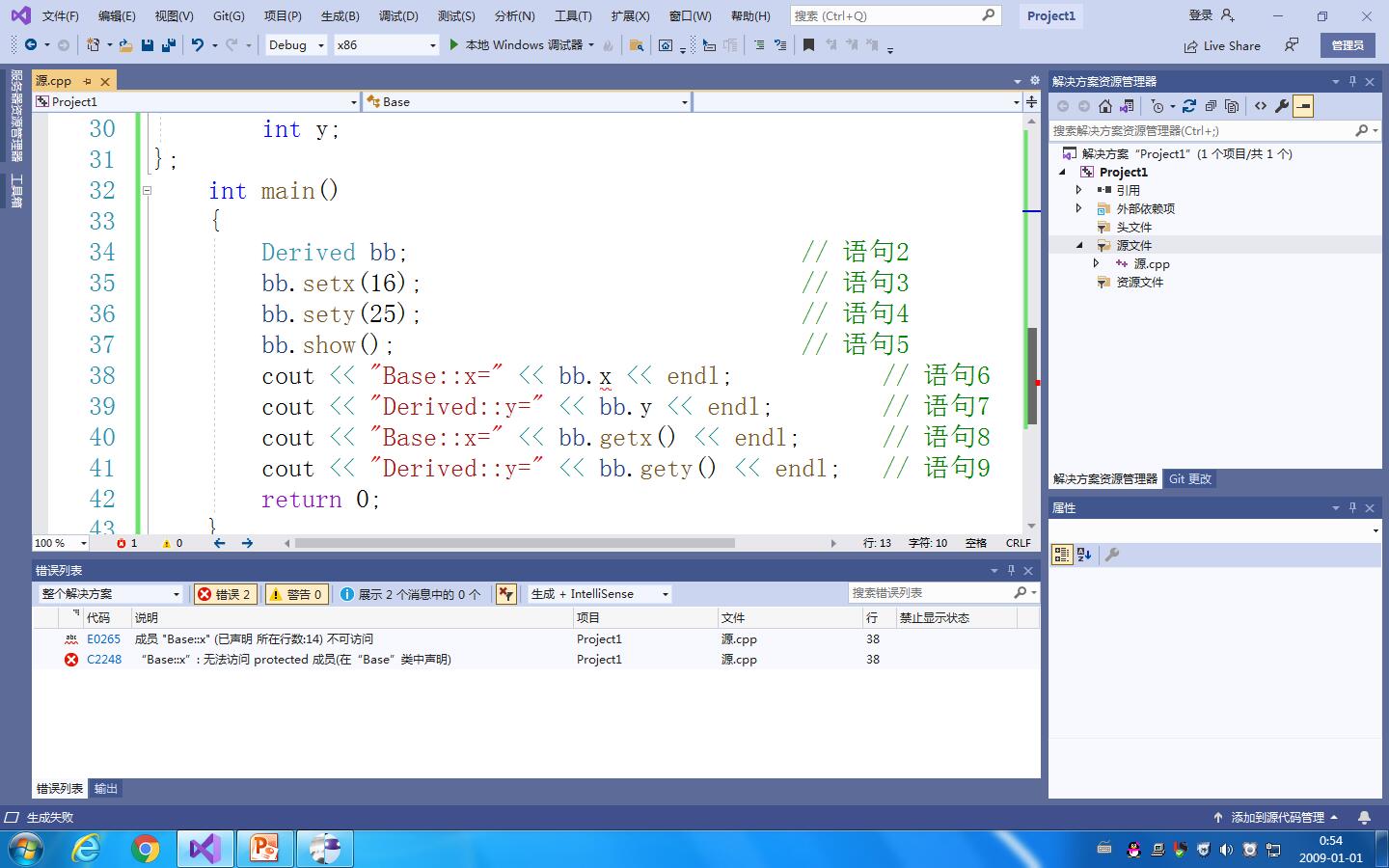
int getx()

{ return x; }

Protected:

int x;

};

——————————————②运行结果：———————————————

——③在原程序的基础上，将派生类Derived的继承方式改为private时———

class Derived:private Base{

public:

void sety(int i)

{ y=i; }

int gety()

{ return y; }

void show()

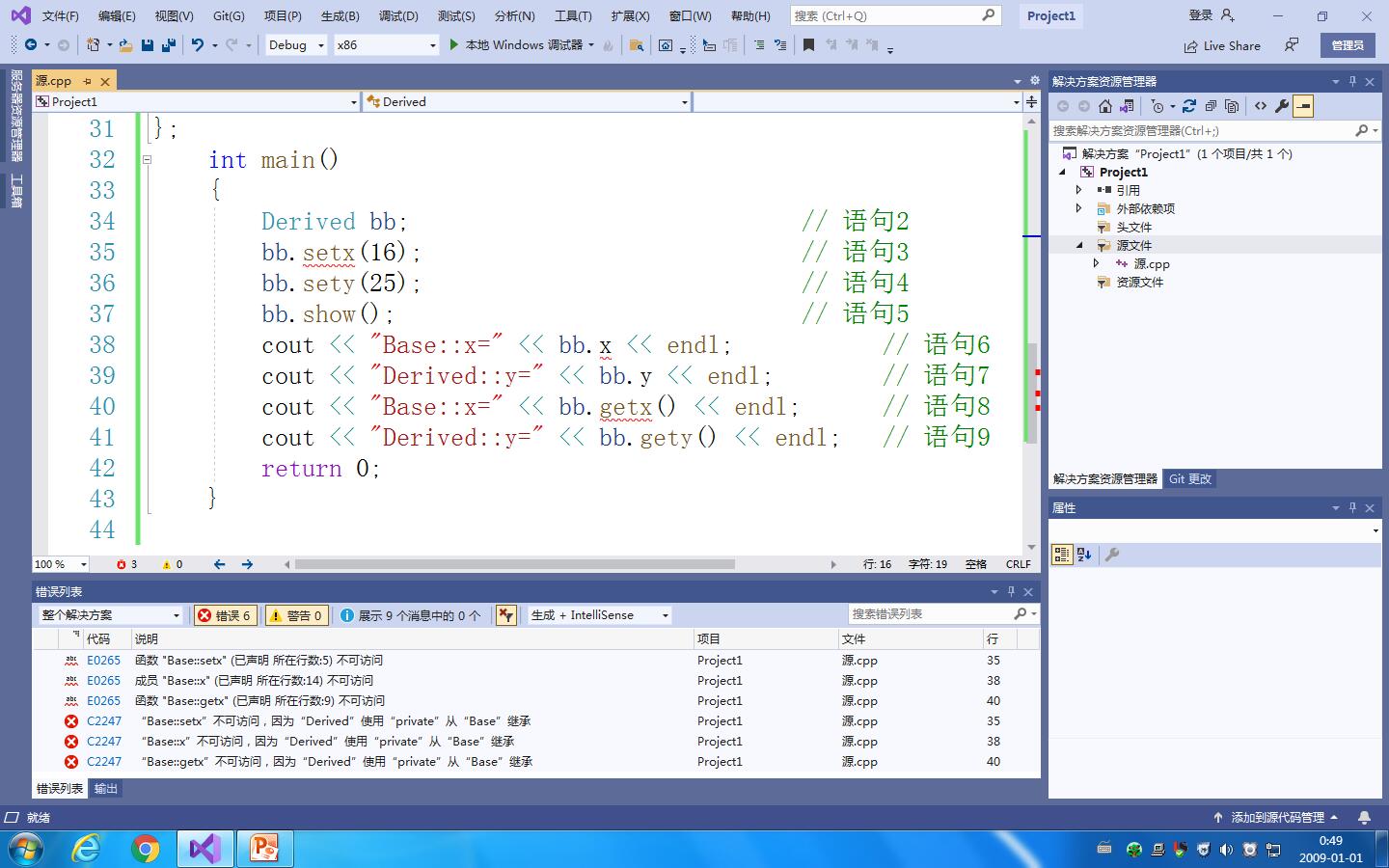
{ cout<<"Base::x="<<x<<endl; // 语句1 }

public:

int y;

};

——————————————③运行结果：———————————————



——④在原程序的基础上，将派生类Derived的继承方式改为protected时——

class Derived:protected Base{

public:

void sety(int i)

{ y=i; }

int gety()

{ return y; }

void show()

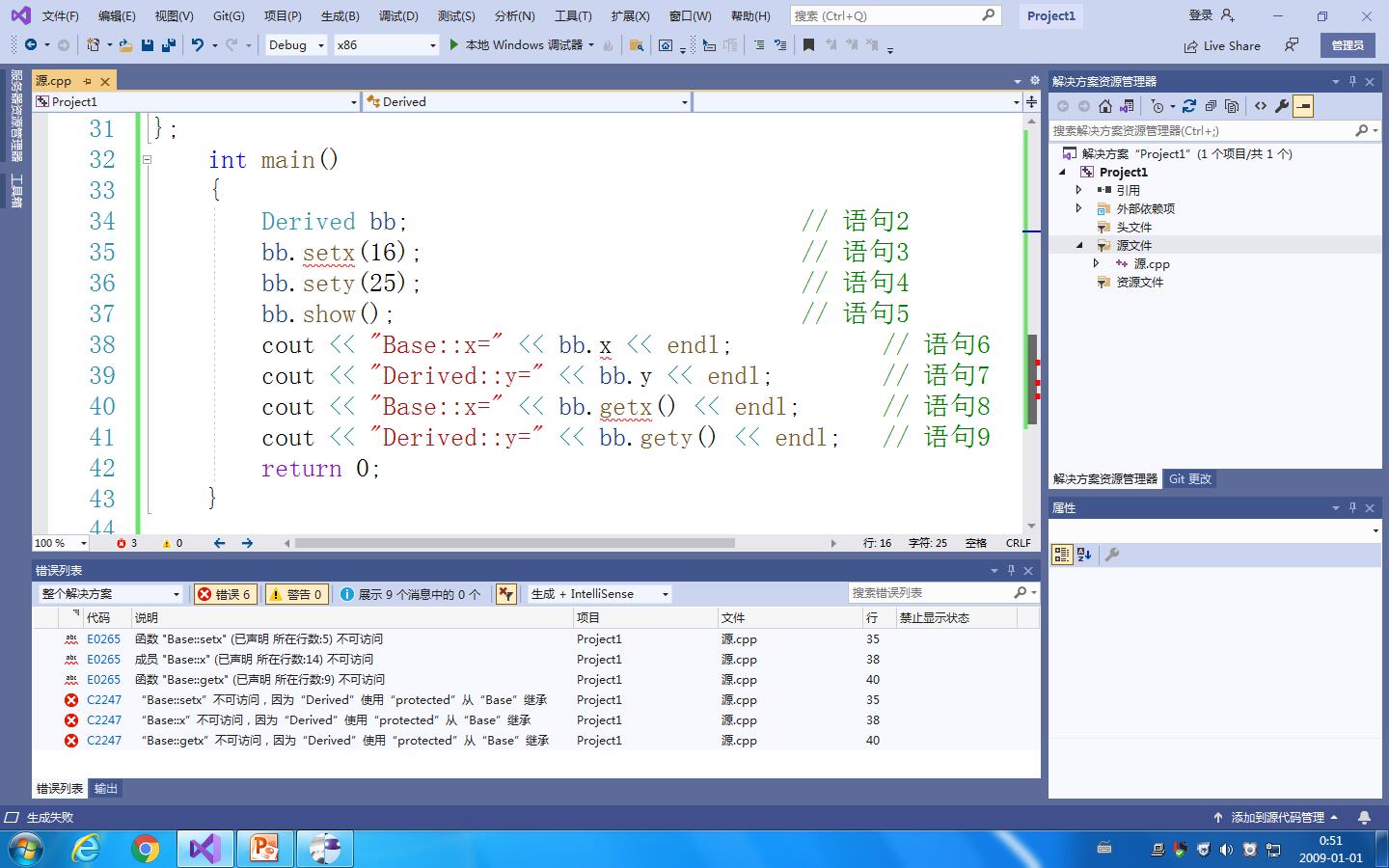
{ cout<<"Base::x="<<x<<endl; // 语句1 }

public:

int y;

};

——————————————④运行结果：———————————————



三、问题分析与感想

1、将基类Base中数据成员x的访问权限改为private时，x不能实现派生类内访问和对象访问。

2、将基类Base中数据成员x的访问权限改为protected时，x无法在类外访问。

3、将派生类Derived的继承方式改为private时，基类的成员函数和变量不能在类外访问。

4、将派生类Derived的继承方式改为protected时，基类的成员函数和变量不能在类外访问。

电自2002许吴瑕 202030310293

Copyright ©2021-2099 WuxiaXu. All right reserved