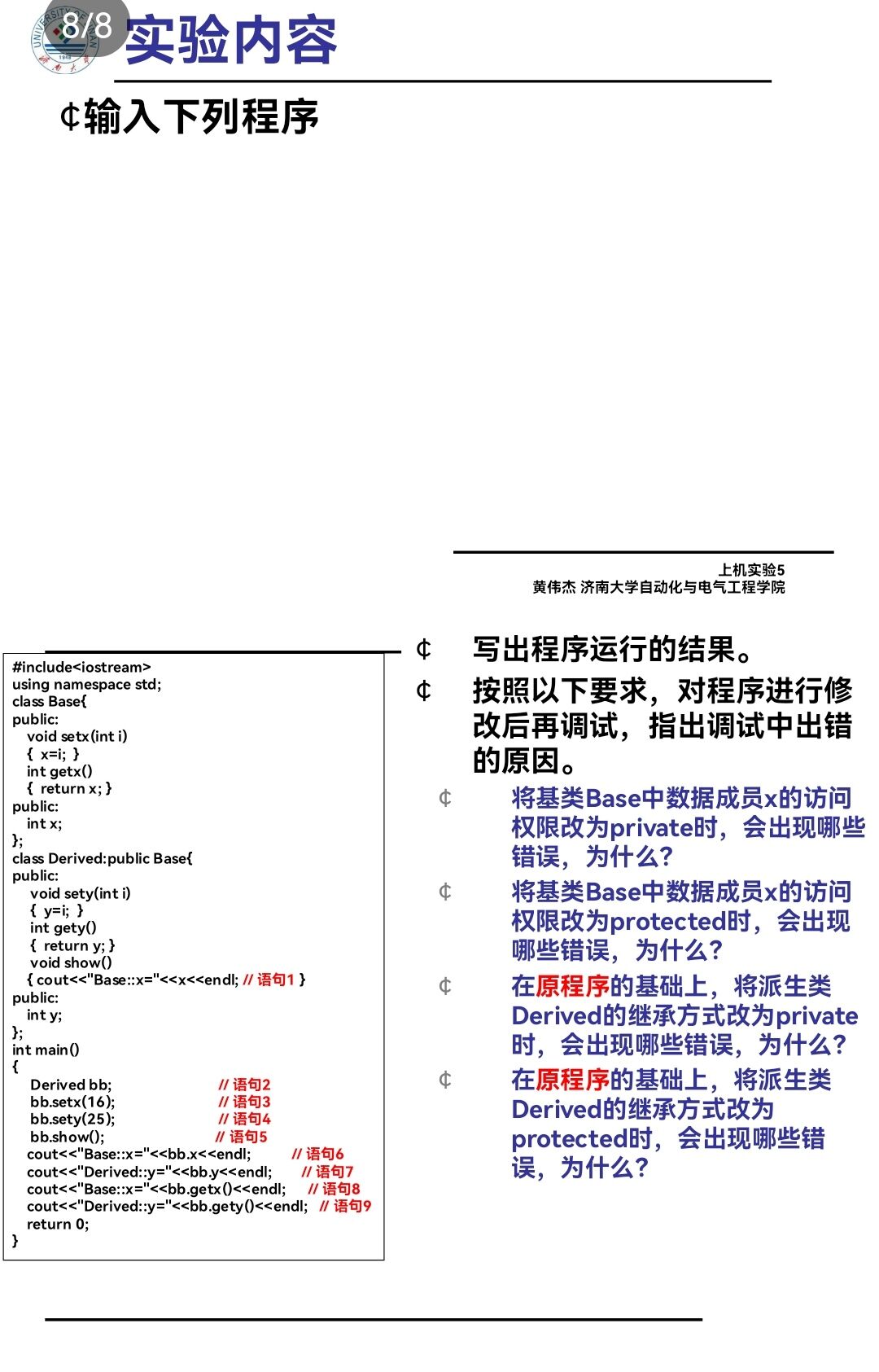
# 上机实验5



# 程序代码

#include<iostream>

using namespace std;

class Base {

public:

void setx(int i)

{

x = i;

}

int getx()

{

return x;

}

public:

int x;

};

class Deriverd :public Base {

public:

void sety(int i)

{

y = i;

}

int gety()

{

return y;

}

void show()

{

cout << "Base::x=" << x << endl;

}

public:

int y;

};

int main()

{

Deriverd bb;

bb.setx(16);

bb.sety(25);

bb.show();

cout << "Base::x=" << bb.x << endl;

cout << "Derived::y=" << bb.y << endl;

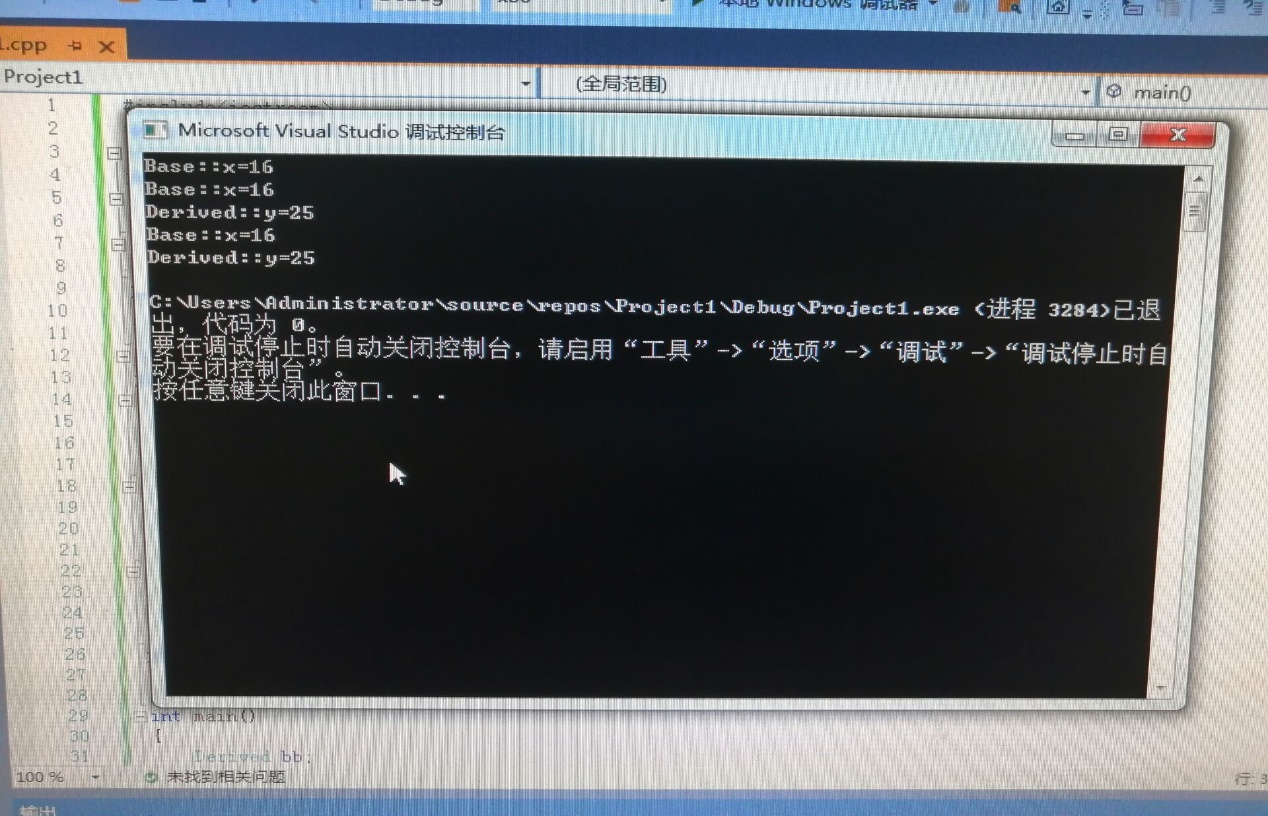
cout << "Base::x=" << bb.getx() << endl;

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

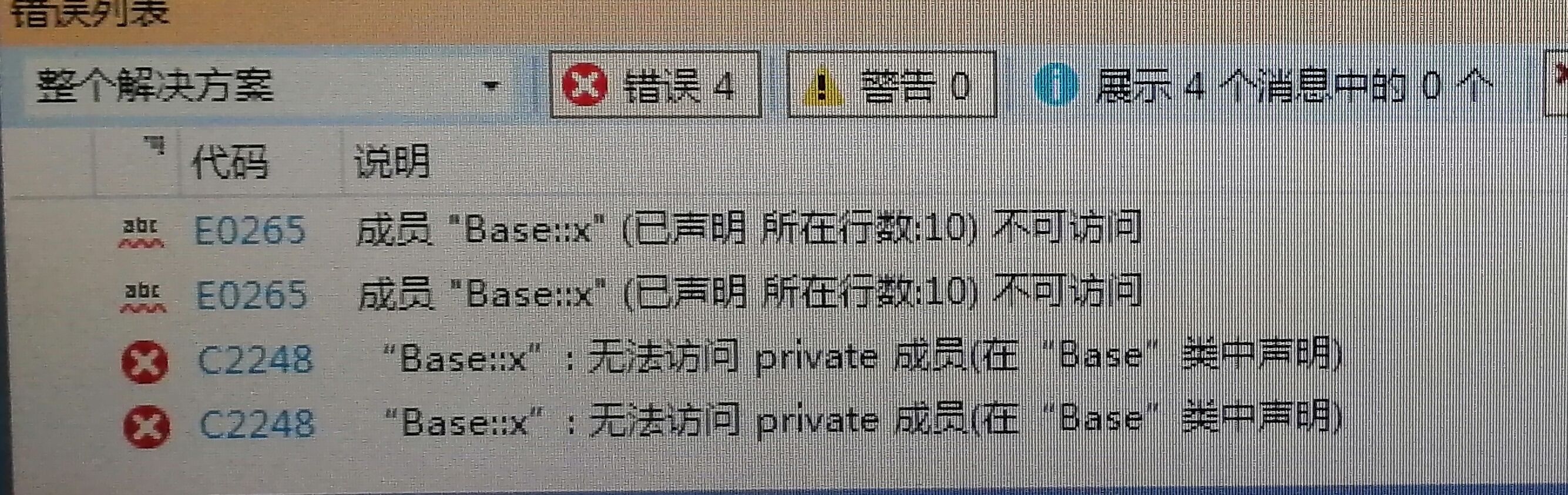
return 0;

}

# 程序结果

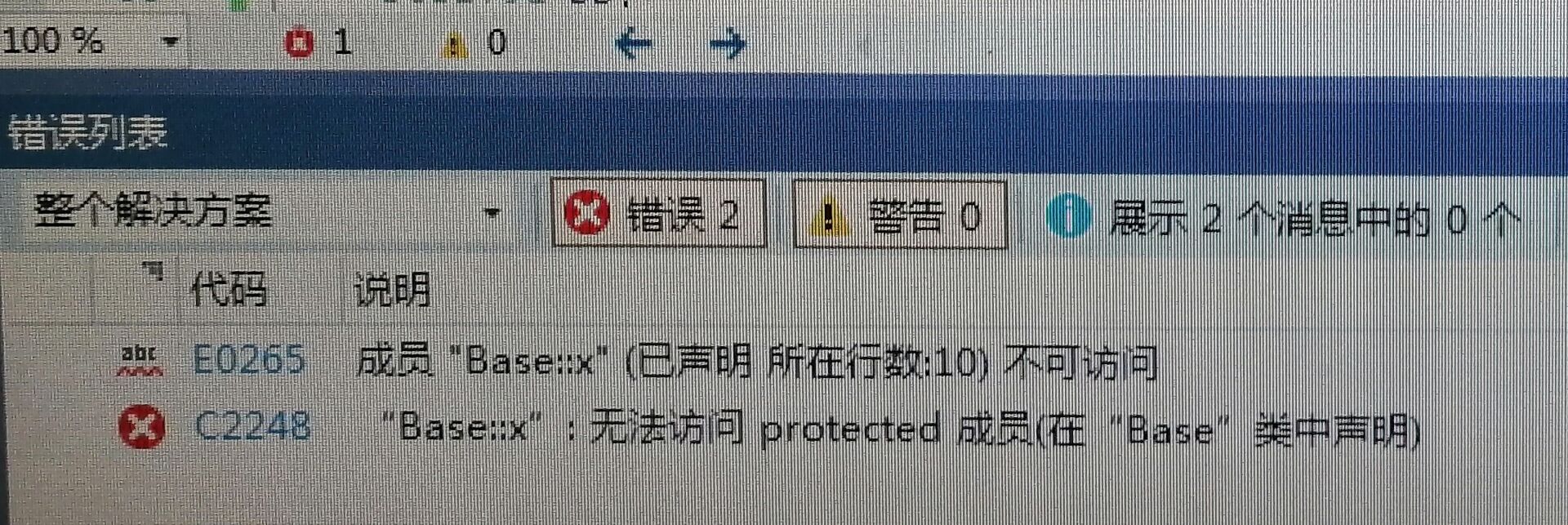


将基类Base中数据成员x的访问权限改为private时



原因：派生类函数不能直接访问基类中的私有成员。

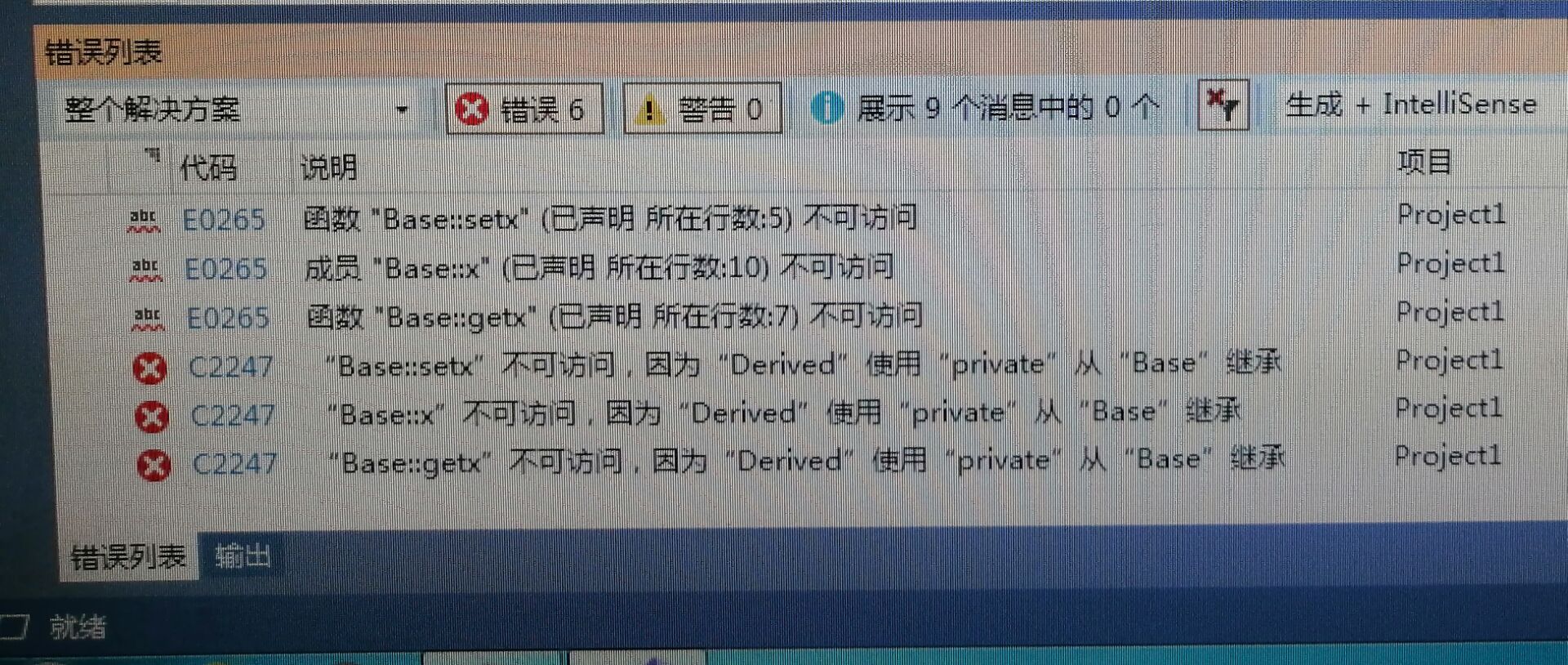
将基类Base中数据成员x的访问权限改为protected时



原因：基类的protected成员在派生类内部的其他成员可以直接访问它们，但是在类外部通过派生类的对象无法访问。

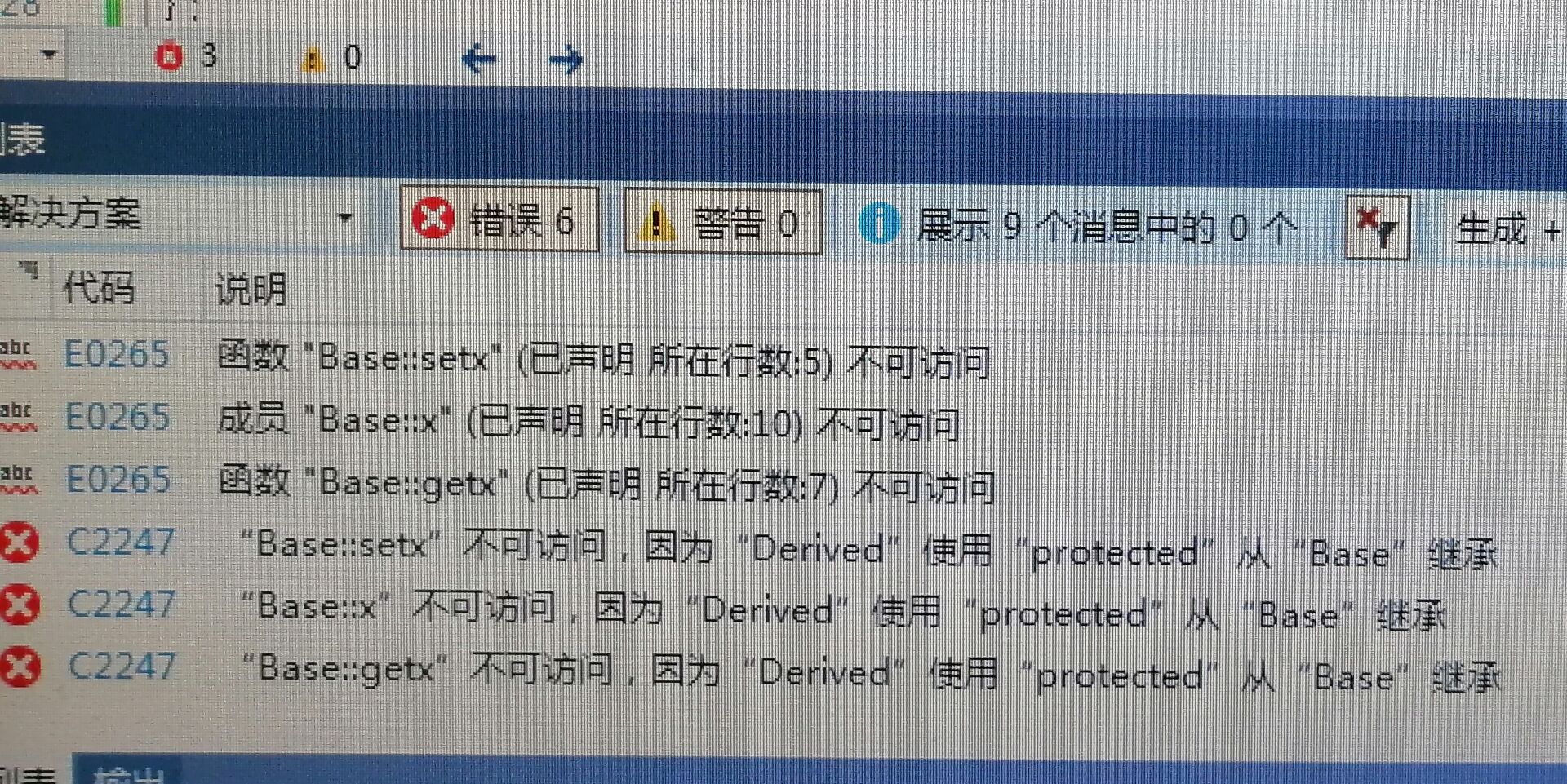
在原程序的基础上，将派生类Derived的继承方式改为private时

#### 



原因：基类中的成员通过 private 方式继承,在派生类中都为私有成员,派生类的其他成员可以直接访问它们,类外的对象都不能直接访问。

在原程序的基础上，将派生类Derived的继承方式改为private时



原因：基类的 public 成员和 protected 成员通过 protected 继承到派生类中都作为派生类的 protected 成员,派生类的其他成员可以直接访问它们,但是类的外部使用者不能通过派生类的对象来访问它们。

# 感想心得

通过这次实验，我掌握了派生类的声明方法和派生类构造函数的定义方法，还掌握了在不同方式下，基类成员在派生类中的访问属性：（1）无论哪种继承方式,基类中的私有成员在派生类中是不可直接访问的。(2)公有继承时,基类所有公有成员在派生类中仍以公有成员的身份出现。私有继承时,公有成员以私有成员身份出现。保护继承时,公有成员以保护成员身份出现。(3)公有继承时,所有保护成员在派生类中仍以保护成员的身份出现。私有继承时,保护成员以私有成员的身份出现。保护继承时,保护成员以保护成员的身份出现。

Copyright 2021-2099 Hubing. All rights reserved