

#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;

}

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

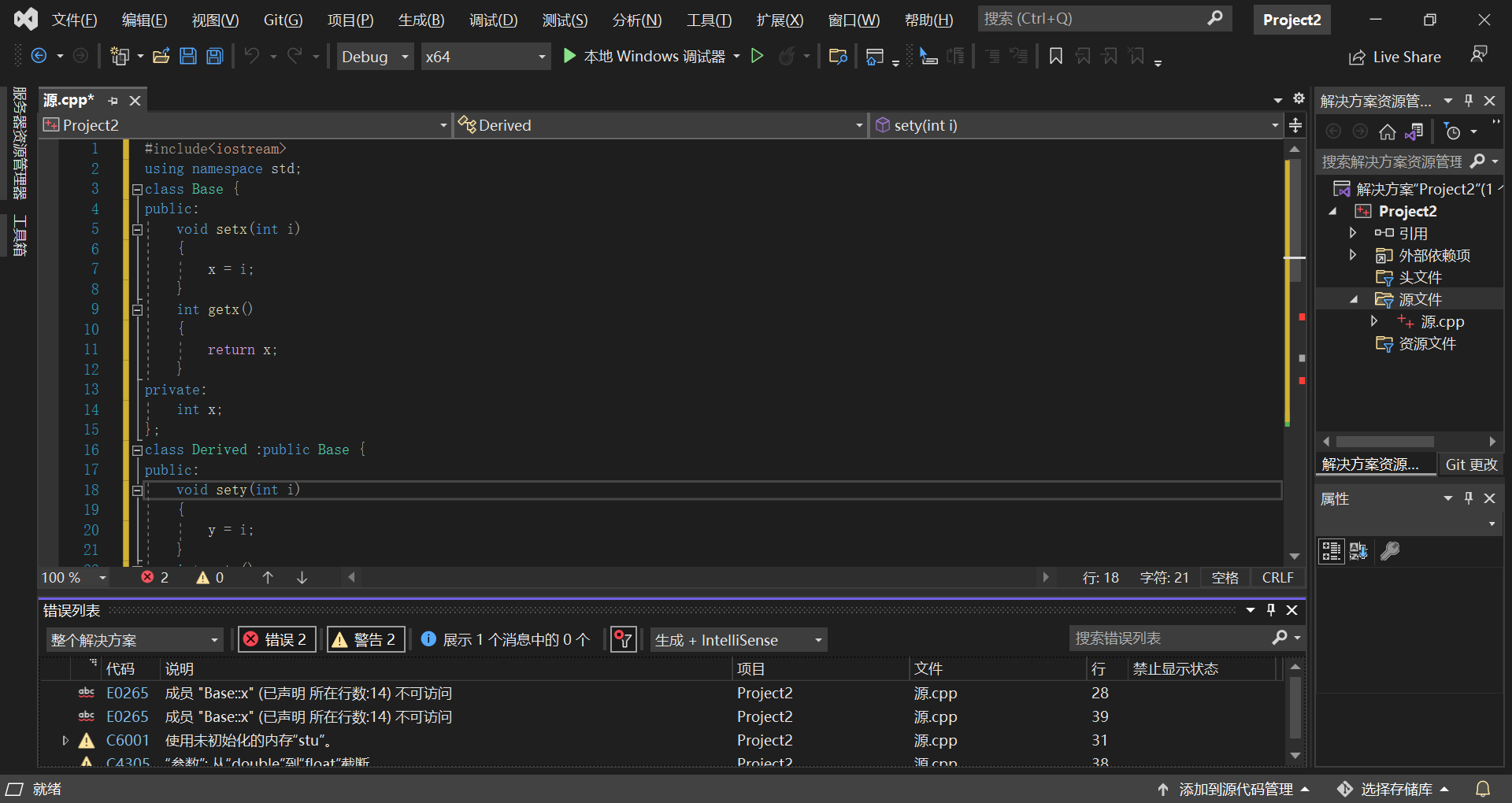
cout << "Derived::y=" << bb.y << endl; // 语句7

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

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

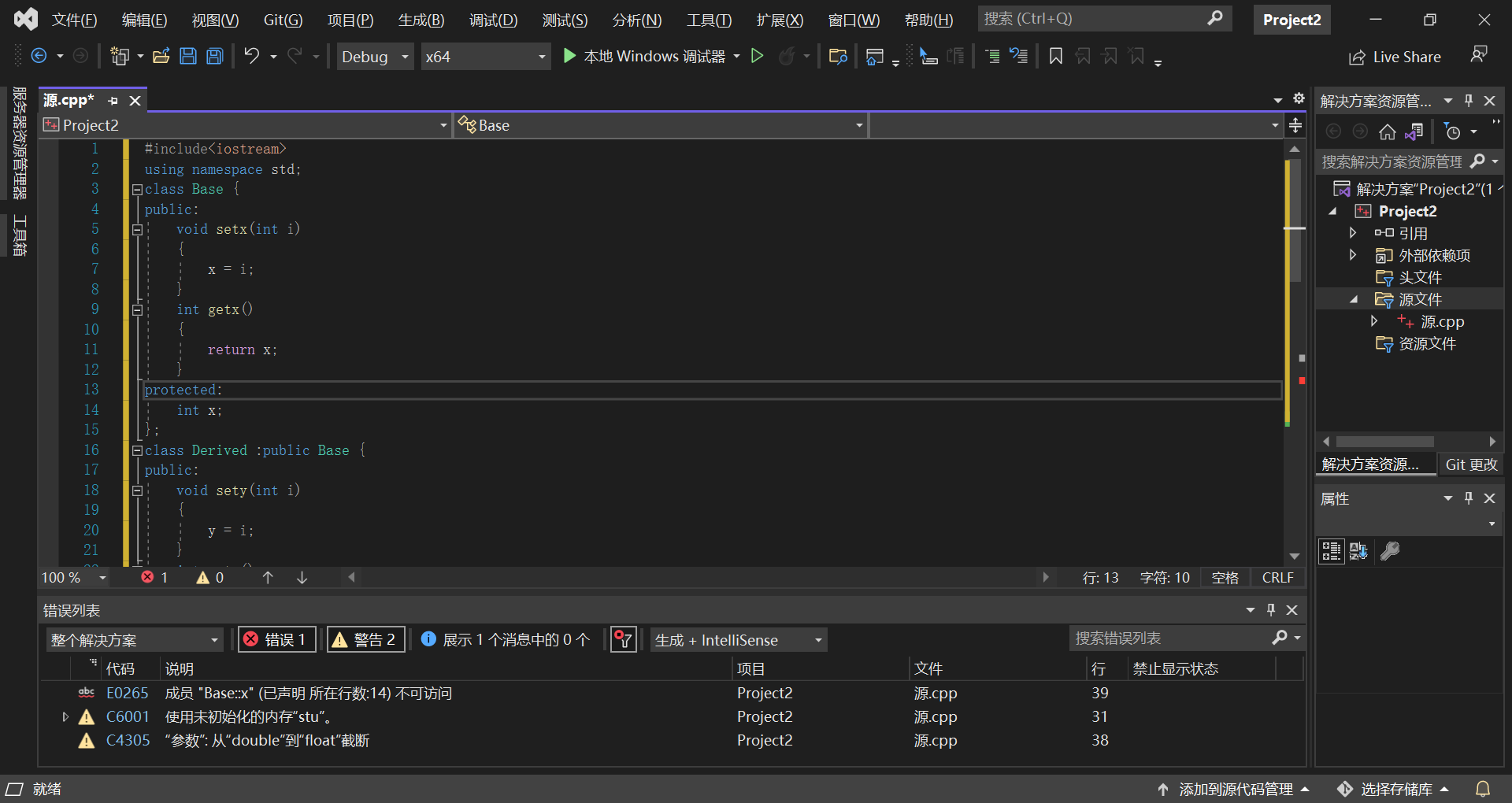
return 0;

}



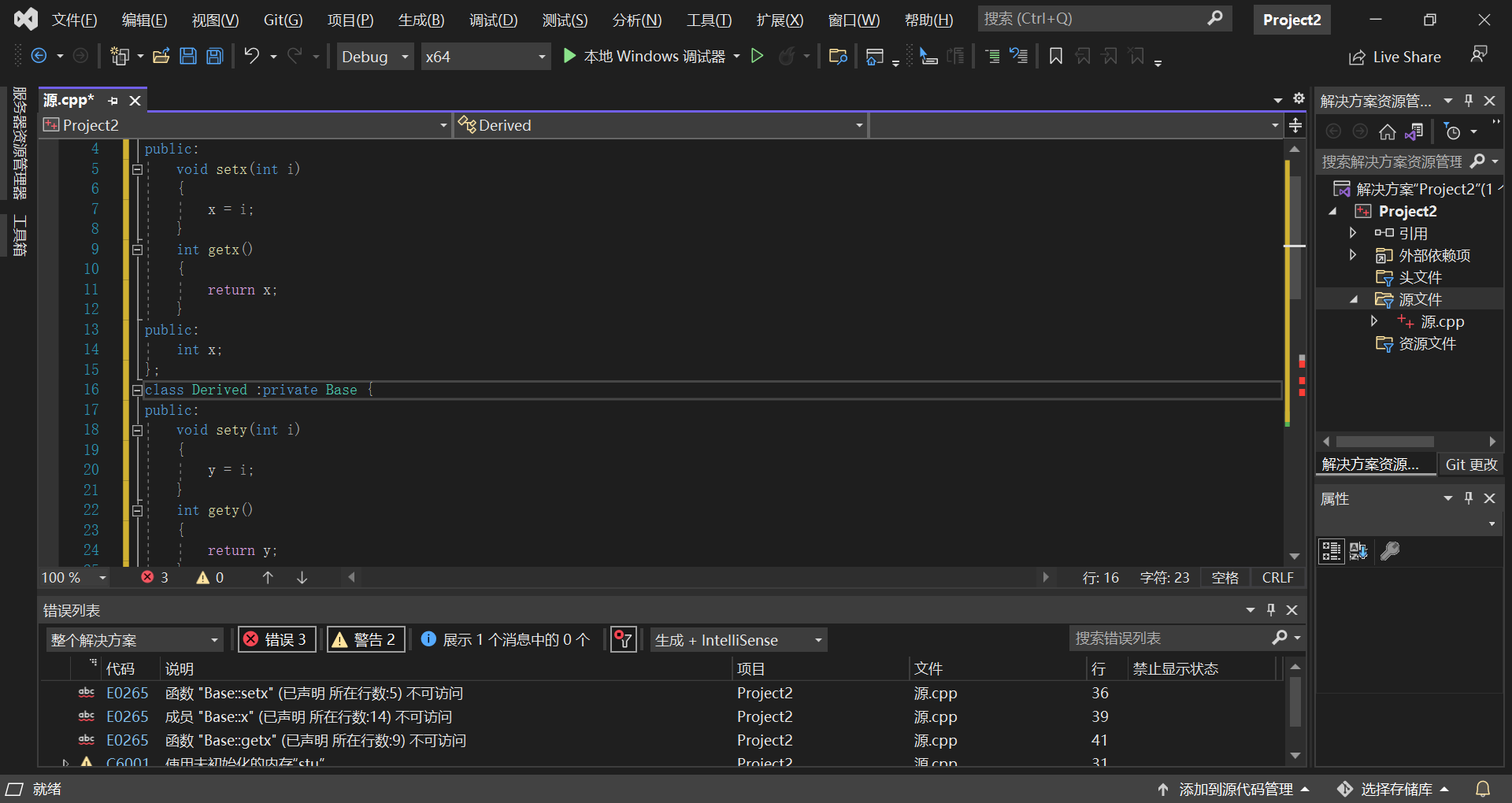
**将基类Base中数据成员x的访问权限改为private时，会出现哪些错误，为什么？**

数据成员x是Base的私有成员，在类Base外的对象不能直接访问它，派生类的成员函数也不能访问它。



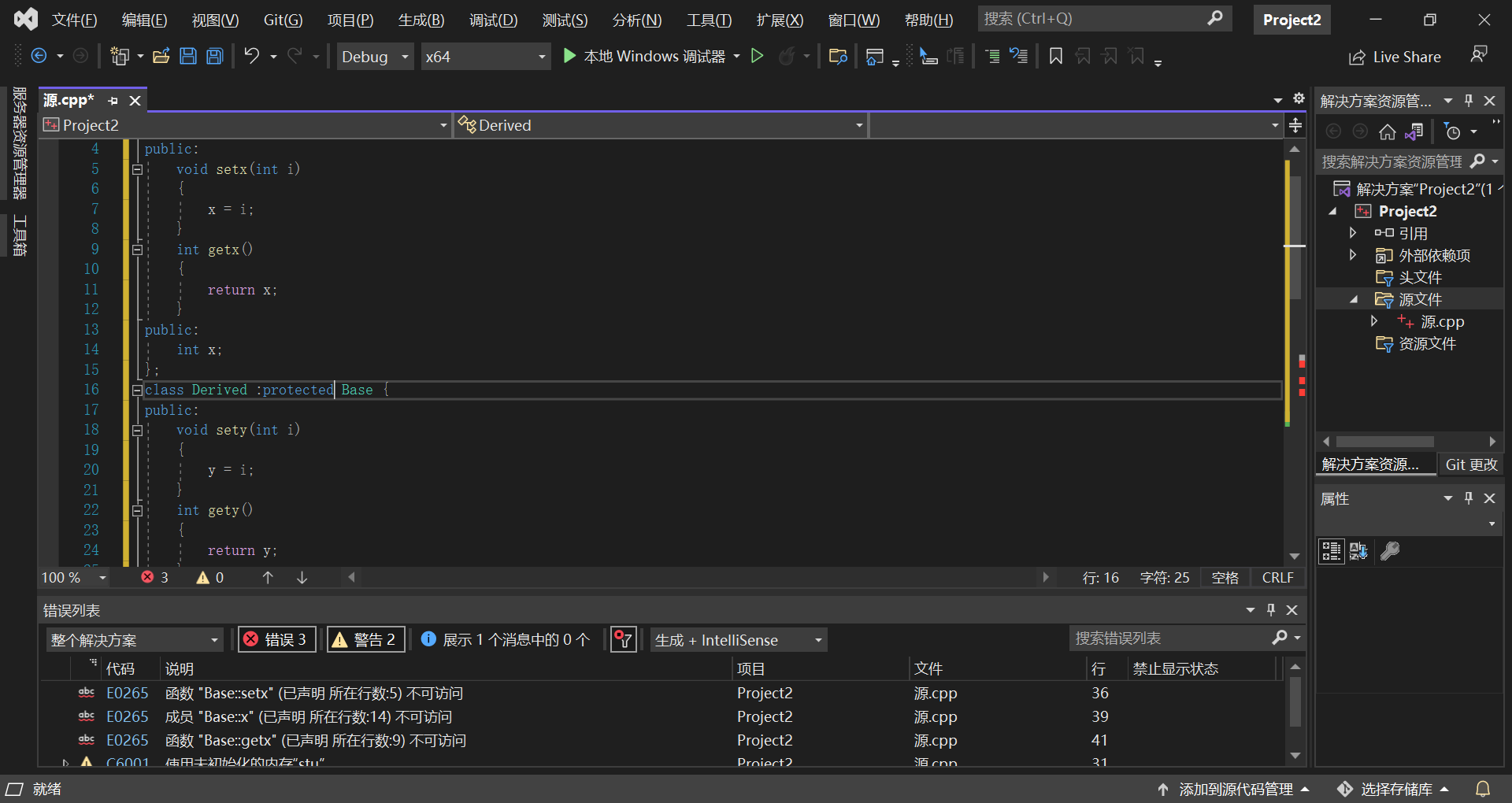
**将基类Base中数据成员x的访问权限改为protected时，会出现哪些错误，为什么？**

数据成员x是Base的保护成员，在类Base外基类的对象不能直接访问它。



**在原程序的基础上，将派生类Derived的继承方式改为private时，会出现哪些错误，为什么？**

私有继承时，基类Base的公有数据成员x，公有成员函数setx（）和gets（）作为派生类的Derived的私有成员，派生类的成员函数可以直接访问他们，但在类外部，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protected时，会出现哪些错误，为什么？**

保护继承时，基类Base的公有数据成员x，公有成员函数setx（）和gets（）作为派生类的Derived的保护成员，派生类的成员函数可以直接访问他们，但在类外部，派生类的对象无法访问。

