#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;

bb.setx(16);

bb.sety(25);

bb.show();

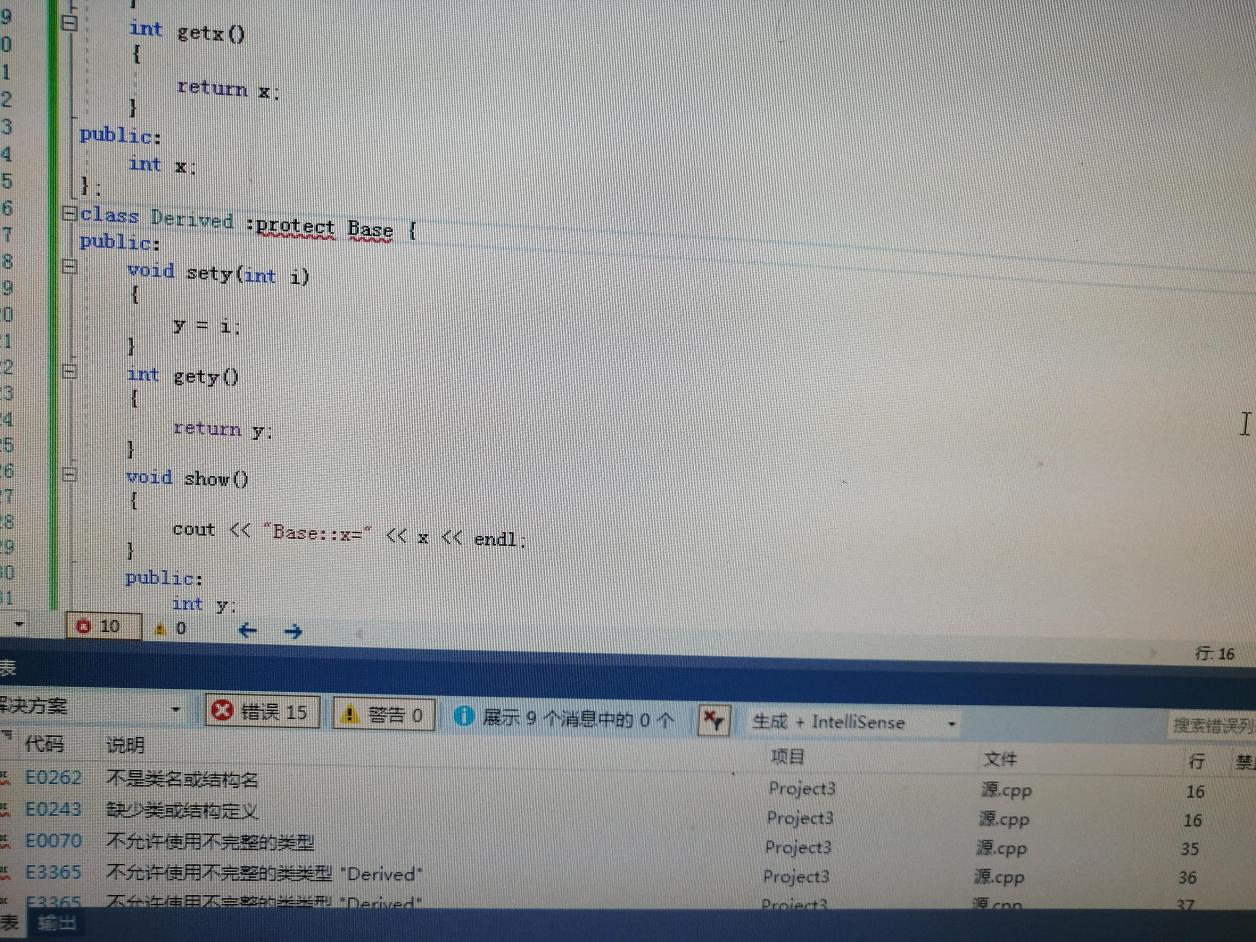
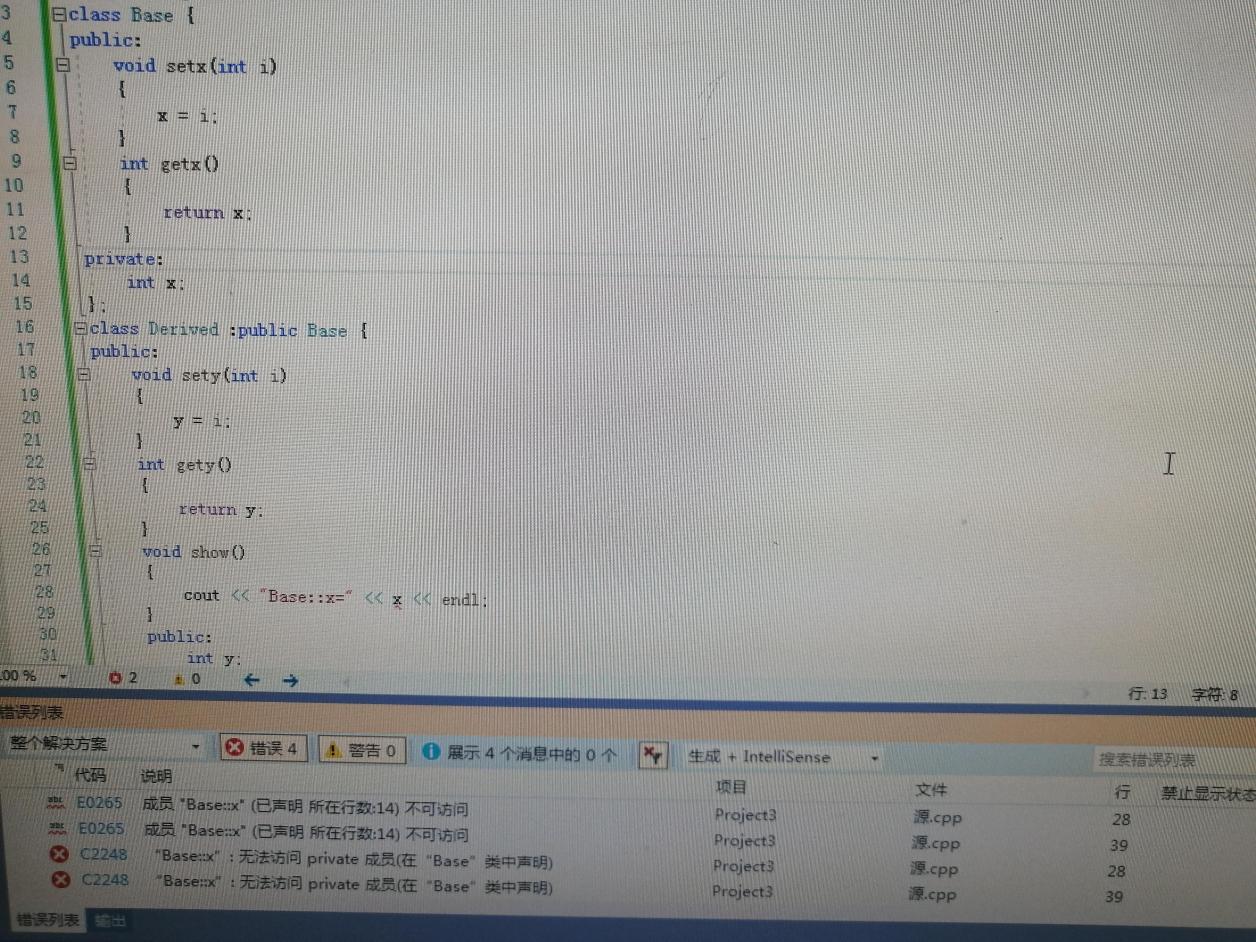
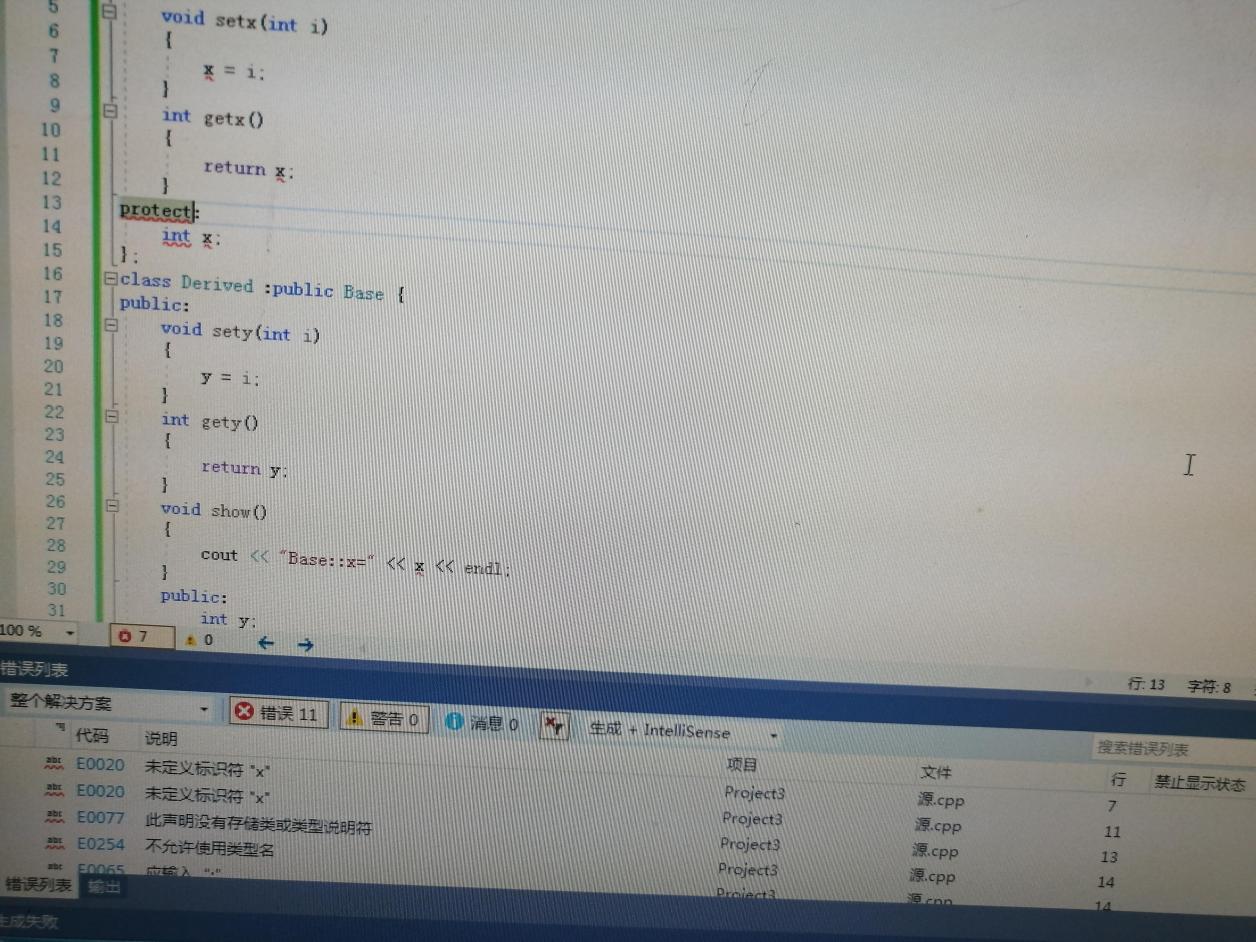
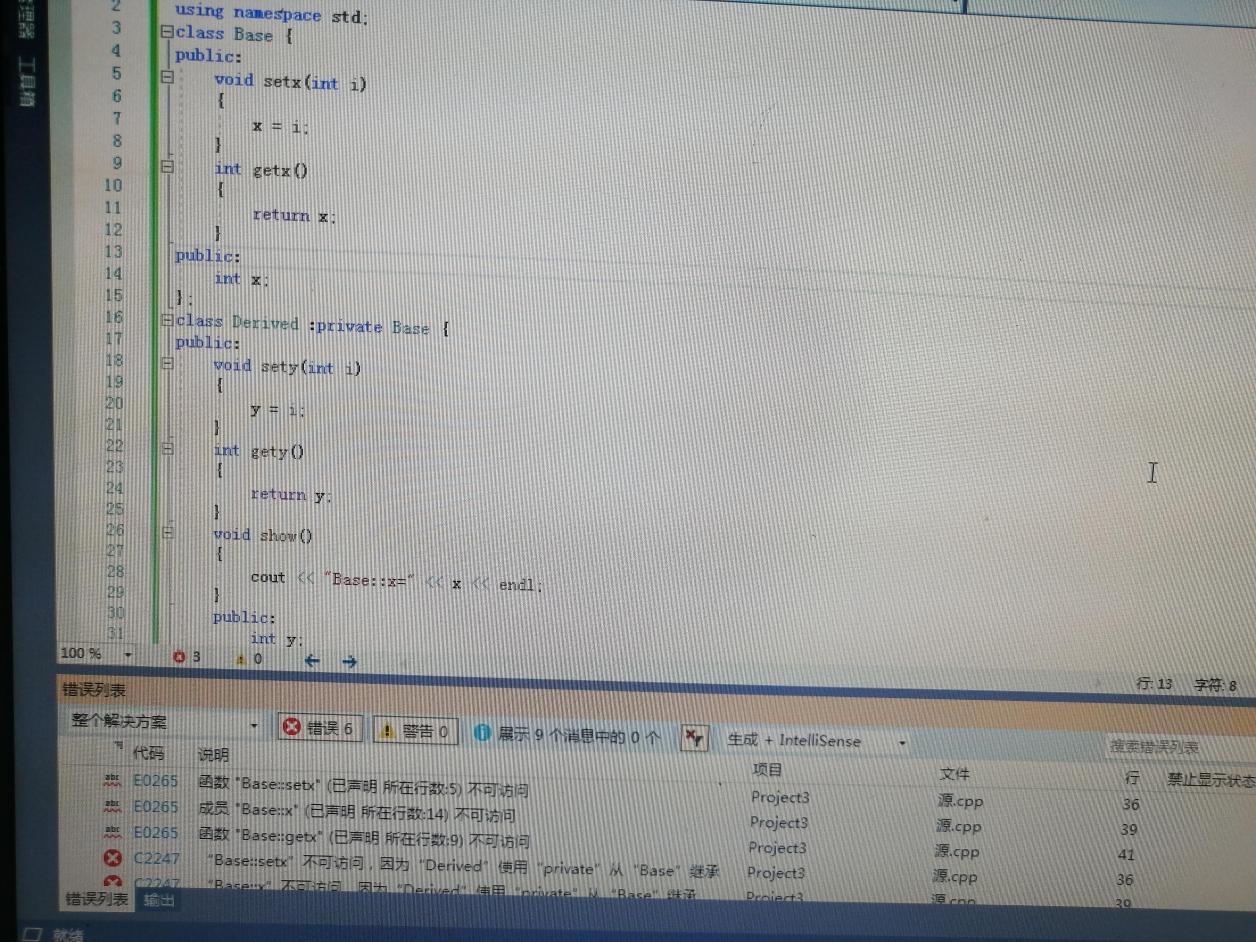
cout << "Base::x=" << bb.x << endl;

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

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

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

return 0;

}

上机实验5：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？

因为：**基类的成员函数访问权限改为private 只有内部可以访问 外部 和派生类都不可以访问**

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？

因为：**基类中的protected 不能被访问**

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？

因为：**为私有成员，不能被直接访问；基类中的公有成员x在派生类中为私有成员，不能被直接访问；基类的成员getx函数在派生类中为私有成员，不能被直接访问**

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？

因为：**基类的setx成员函数在派生类中为保护成员，不能被直接访问；基类中的公有成员x在派生类中为保护成员，不能被直接访问；基类的getx成员函数在派生类中为保护成员，不能被直接访问**

收获：通过这节课的学习，我学习到了私有成员，公有成员和保护成员之间的访问规则。基类的私有成员，无论哪种继承方式都不能访问。公有继承不改变基类的公有和保护成员的访问限制。私有继承将基类的公有和保护成员都变成私有。保护继承将基类的公有和保护成员变成保护。