#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;

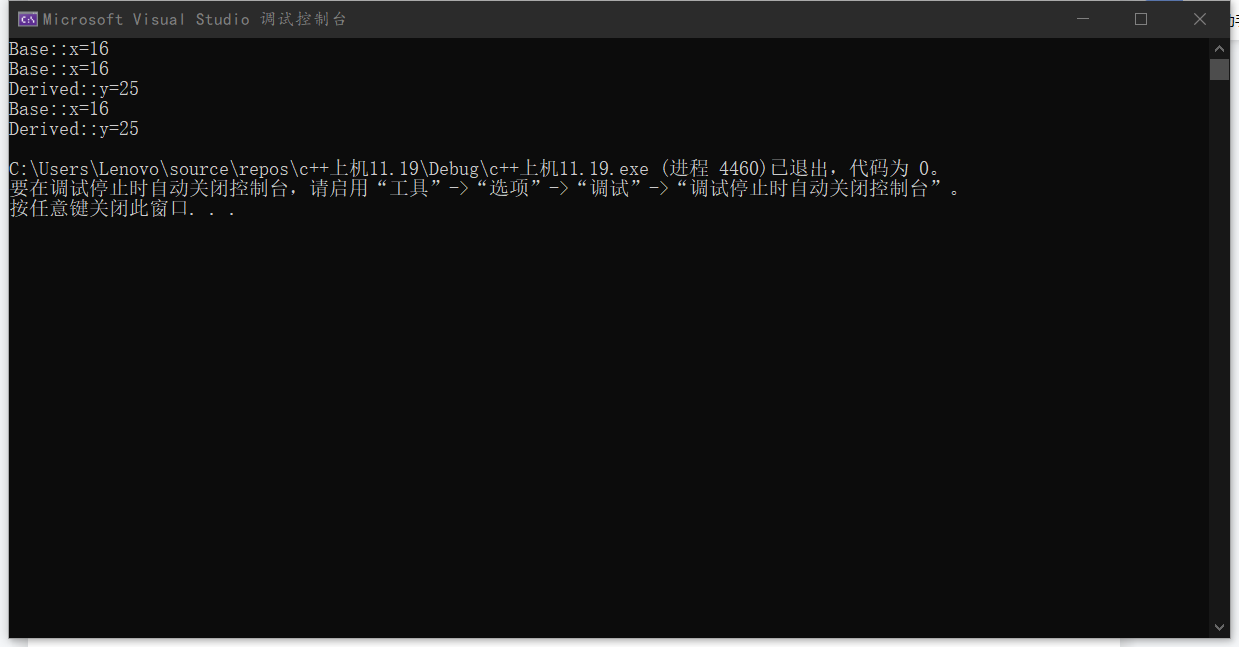
cout << "Derived::y=" << bb.gety() << endl;

return 0;

}

代码

结果和解释



1. 如果把base中的x改为private时，derived在继承时无法访问x，所以不能改为private型。
2. 如果把base中的x改为protected型时，x被保护起来，derived无法继承x。
3. 如果把继承类型改为private型时，derived继承base时，bb。Set函数无法访问
4. 如果把继承类型改为protected型时，derived继承时，bb。Set函数无法访问。

感悟：在使用继承函数，一定要注意对象是否时公有的可以被调用出来的类。

Copyright ©2021-2099 luanxiang wang.All rights reserved