上机实验4：了解三种不同的对象传递方式；

掌握静态成员的概念和使用

由关键字static 修饰说明的成员，称为静态成员（static class member）。

静态成员为所有对象共享，只有一份存于公用内存中。

静态成员包括静态数据成员和静态函数成员。

1. 将对象ob作为函数的形参

void sqr\_it(Tr ob) {

ob.set\_i(ob.get\_i() \* ob.get\_i());

cout << "在函数sqr\_it内，形参对象ob的数据成i的值为：" << ob.get\_i();

cout << endl;

}

int main() {

Tr obj(10);

cout << "调用函数sqr\_it前，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

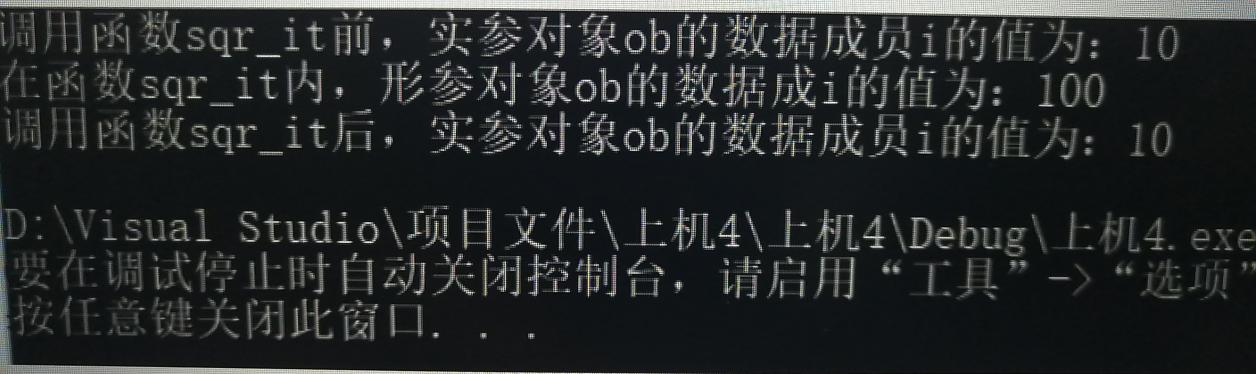
sqr\_it(obj);

cout << "调用函数sqr\_it后，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

return 0;

}



1. 将对象指针ob作为函数的形参

void sqr\_it(Tr \*ob) {

ob->set\_i(ob->get\_i() \* ob->get\_i());

cout << "在函数sqr\_it内，形参对象ob的数据成i的值为：" << ob->get\_i();

cout << endl;

}

int main() {

Tr obj(10);

cout << "调用函数sqr\_it前，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

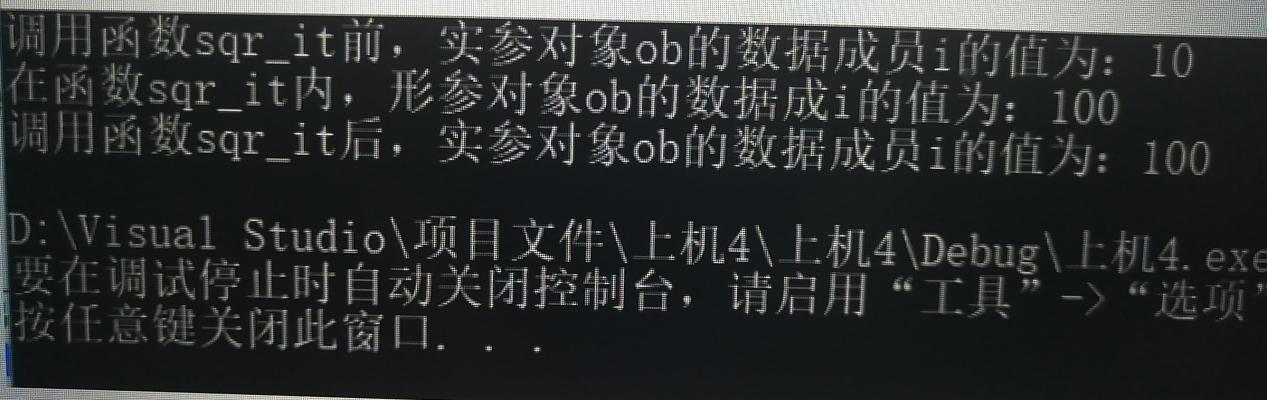
sqr\_it(&obj);

cout << "调用函数sqr\_it后，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

return 0;

}



1. 将对象地址作为形参

void sqr\_it(Tr& ob) {

ob.set\_i(ob.get\_i() \* ob.get\_i());

cout << "在函数sqr\_it内，形参对象ob的数据成i的值为：" << ob.get\_i();

cout << endl;

}

int main() {

Tr obj(10);

cout << "调用函数sqr\_it前，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

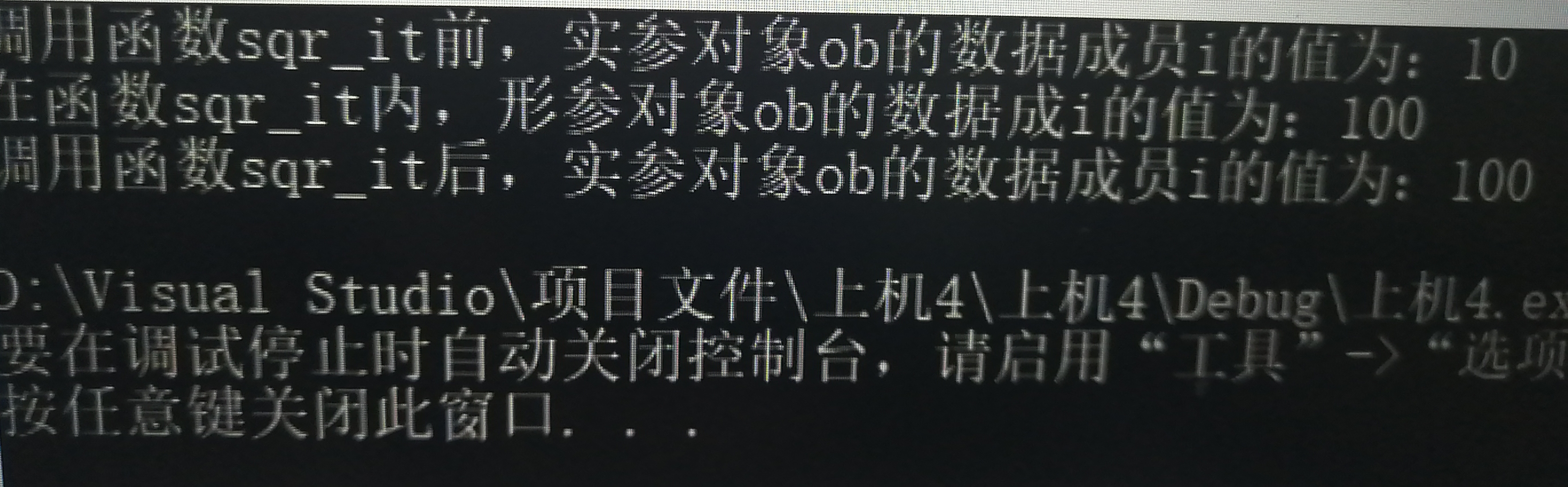
sqr\_it(obj);

cout << "调用函数sqr\_it后，实参对象ob的数据成员i的值为：";

cout << obj.get\_i() << endl;

return 0;

}



实验part2

#include<iostream>

using namespace std;

class Ctest {

static int count;

public:

Ctest() {

++count;

cout << "对象数量 = " << count << '\n';

}

};

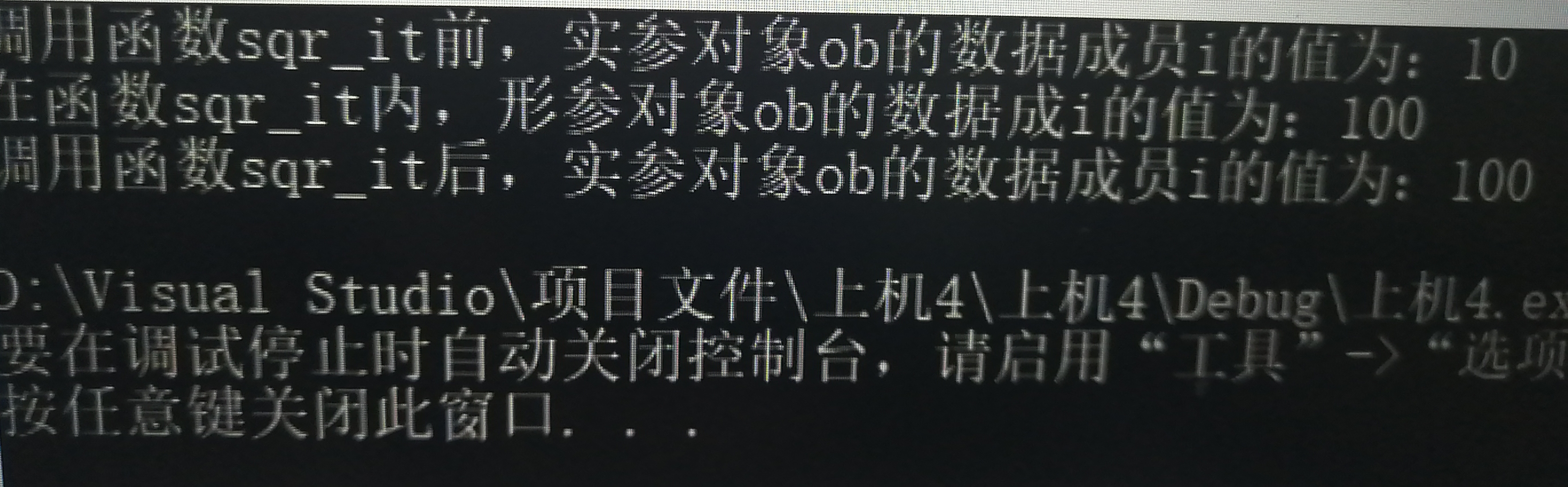
int Ctest::count = 0;

int main(void) {

Ctest a[3];

return 0;

}



1. 上机心得：上机时静态数据成员定义时前要加static，静态数据成员的初始化与普通数据成员不同。静态数据成员初始化应在类外单独进行，而且应在定义对象之前进行。一般在main()函数之前、类声明之后的特殊地带为它提供定义和初始化。