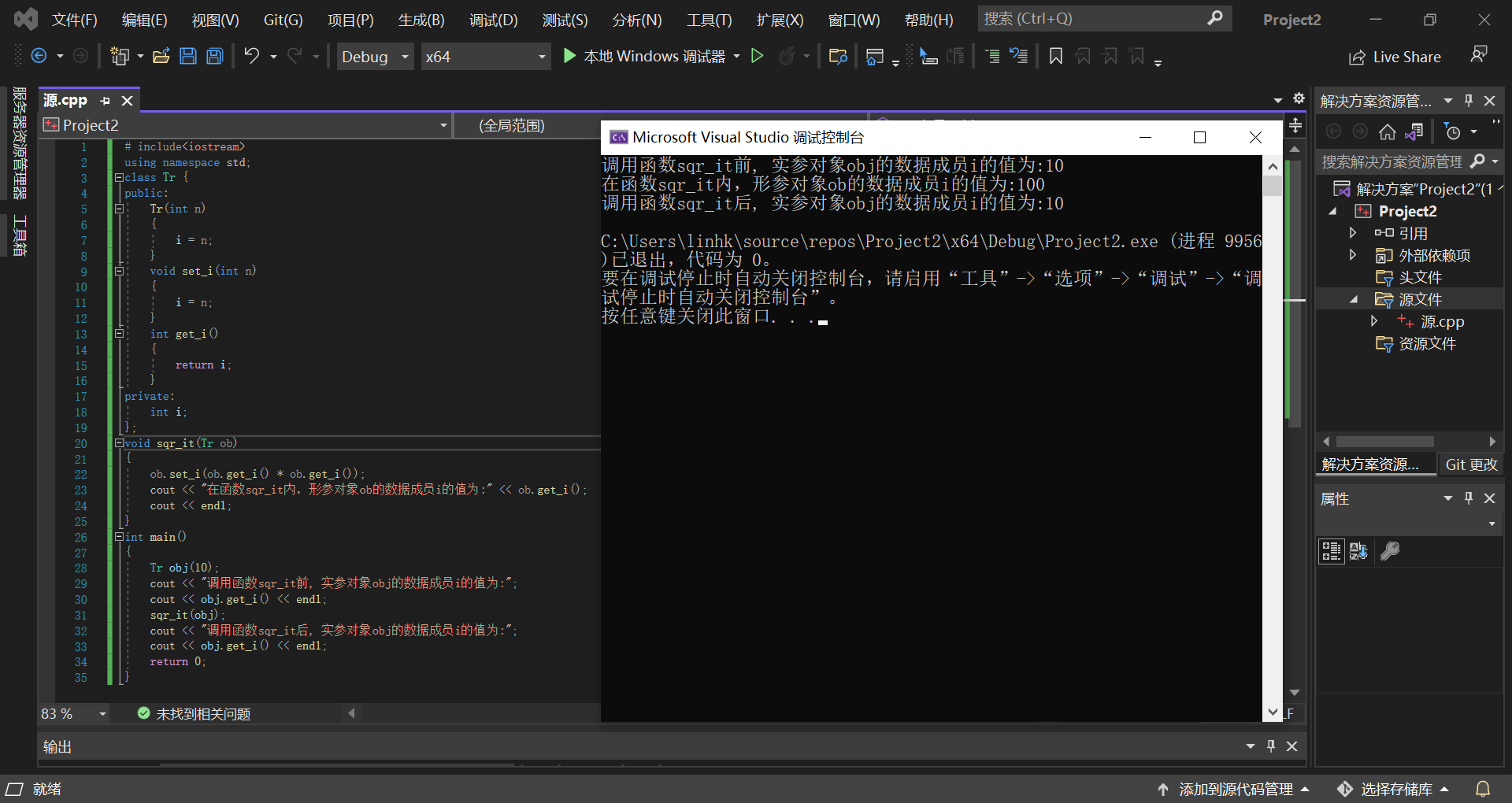
**程序代码（结果）**



# include<iostream>

using namespace std;

class Tr {

public:

Tr(int n)

{

i = n;

}

void set\_i(int n)

{

i = n;

}

int get\_i()

{

return i;

}

private:

int i;

};

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的数据成员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obj的数据成员i的值为:";

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

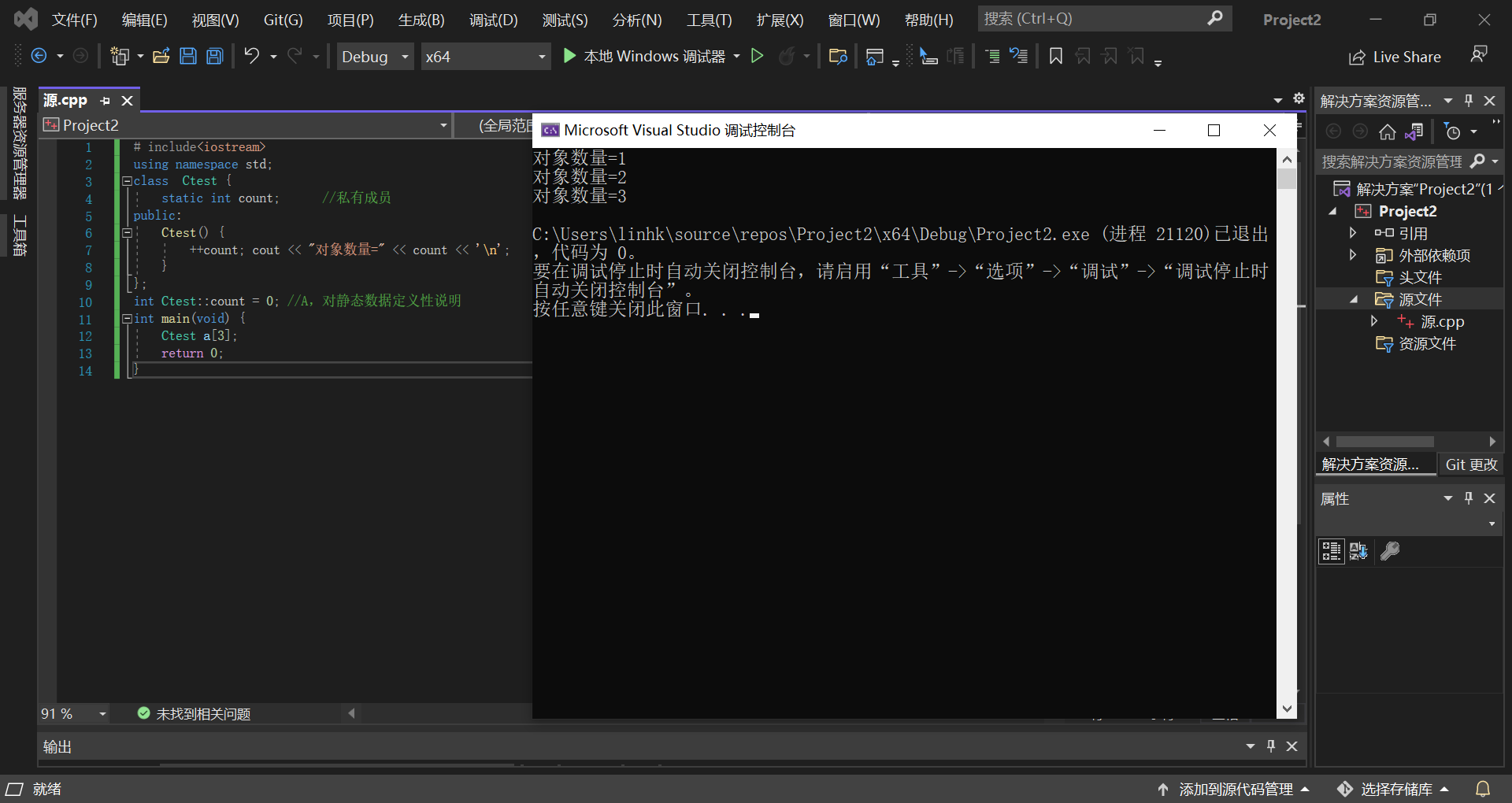
sqr\_it(obj);

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

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

return 0;

}



# include<iostream>

using namespace std;

class Ctest {

static int count; //私有成员

public:

Ctest() {

++count; cout << "对象数量=" << count << '\n';

}

};

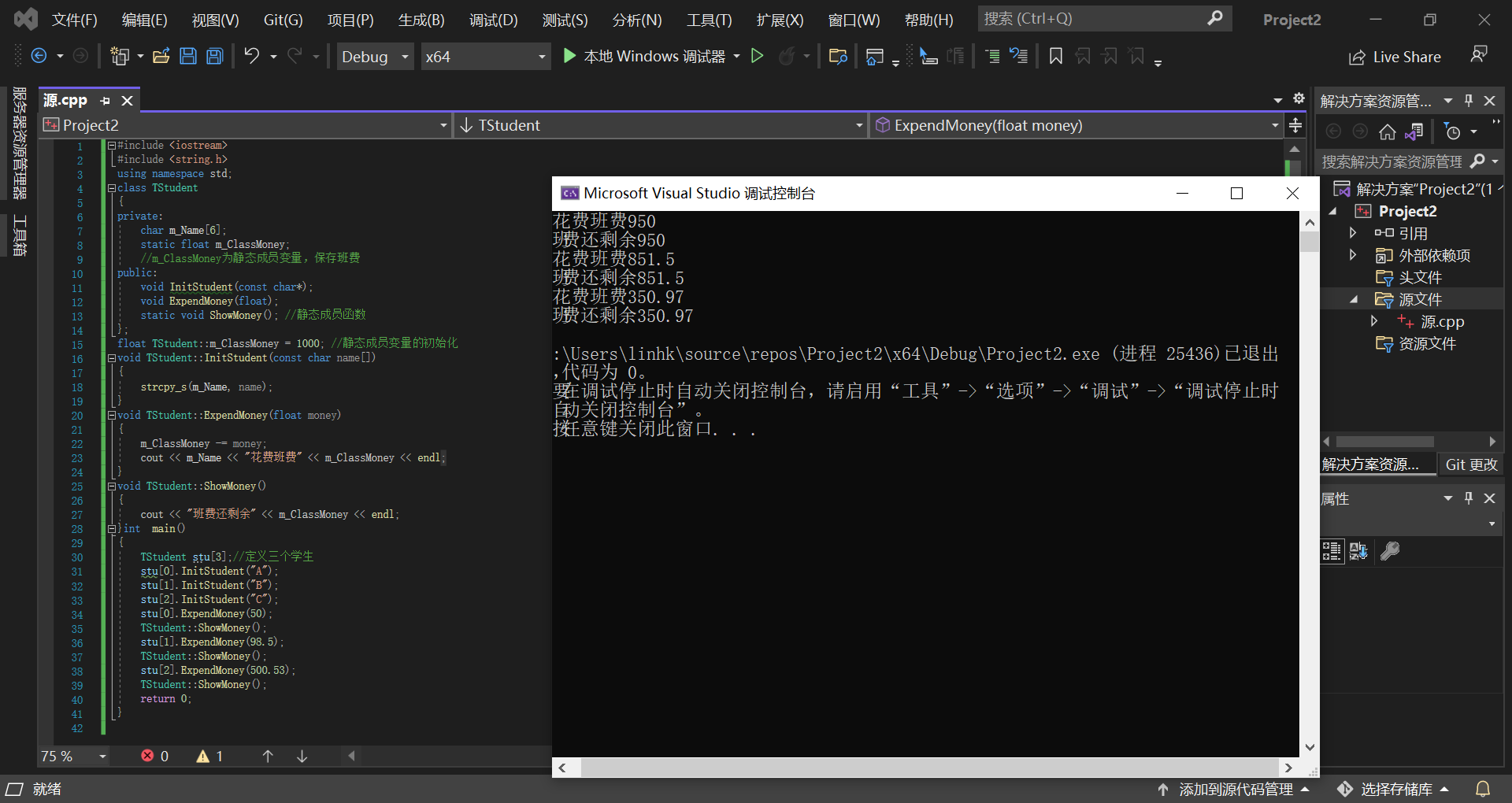
int Ctest::count = 0; //A，对静态数据定义性说明

int main(void) {

Ctest a[3];

return 0;

}



#include <iostream>

#include <string.h>

using namespace std;

class TStudent

{

private:

char m\_Name[6];

static float m\_ClassMoney;

//m\_ClassMoney为静态成员变量，保存班费

public:

void InitStudent(const char\*);

void ExpendMoney(float);

static void ShowMoney(); //静态成员函数

};

float TStudent::m\_ClassMoney = 1000; //静态成员变量的初始化

void TStudent::InitStudent(const char name[])

{

strcpy\_s(m\_Name, name);

}

void TStudent::ExpendMoney(float money)

{

m\_ClassMoney -= money;

cout << m\_Name << "花费班费" << m\_ClassMoney << endl;

}

void TStudent::ShowMoney()

{

cout << "班费还剩余" << m\_ClassMoney << endl;

}int main()

{

TStudent stu[3];//定义三个学生

stu[0].InitStudent("A");

stu[1].InitStudent("B");

stu[2].InitStudent("C");

stu[0].ExpendMoney(50);

TStudent::ShowMoney();

stu[1].ExpendMoney(98.5);

TStudent::ShowMoney();

stu[2].ExpendMoney(500.53);

TStudent::ShowMoney();

return 0;

}

**感想心得**

进一步加深对类和对象的理解，掌握几种对象传递的使用方法，掌握静态成员的概念和使用，在实验中声明用const的步骤还是不够熟练，老是出现格式错误