**程序代码：**

1。

#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;

}

2.

#include <iostream>

#include <string>

using namespace std;

class TStudent

{

private:

string Name;

static float ClassMoney;

public:

void InitStudent(string);

void ExpendMoney(float);

static void ShowMoney();

};

float TStudent::ClassMoney = 1000;

void TStudent::InitStudent(string name)

{

Name = name;

}

void TStudent::ExpendMoney(float money)

{

ClassMoney -= money;

cout << Name << "花费班费" << ClassMoney << endl;

}

void TStudent::ShowMoney()

{

cout << "班费还剩余" << ClassMoney << endl;

}

int main()

{

TStudent stu[3];

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

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

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

stu[0].ExpendMoney(100);

TStudent::ShowMoney();

stu[1].ExpendMoney(300);

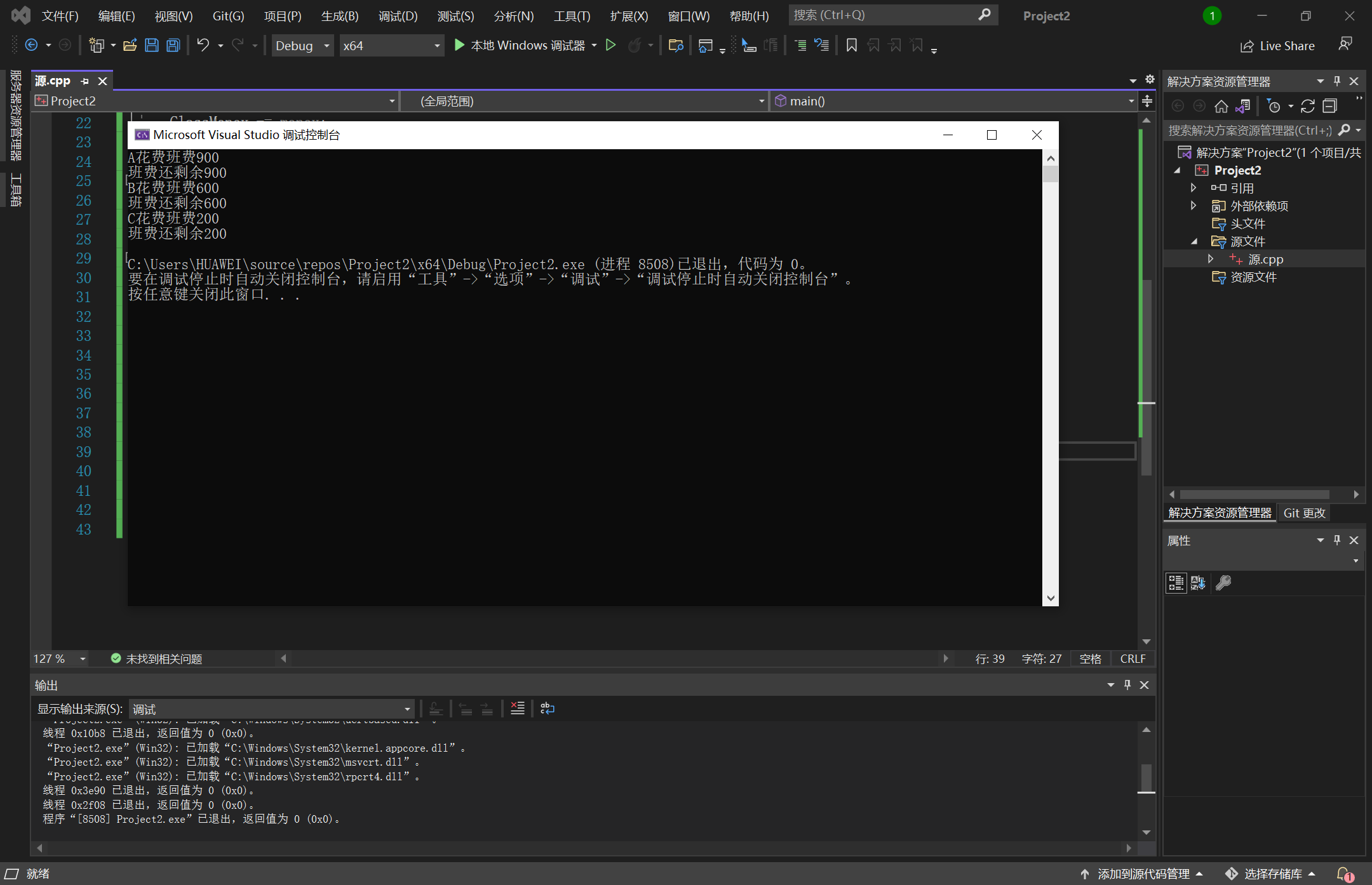
TStudent::ShowMoney();

stu[2].ExpendMoney(400);

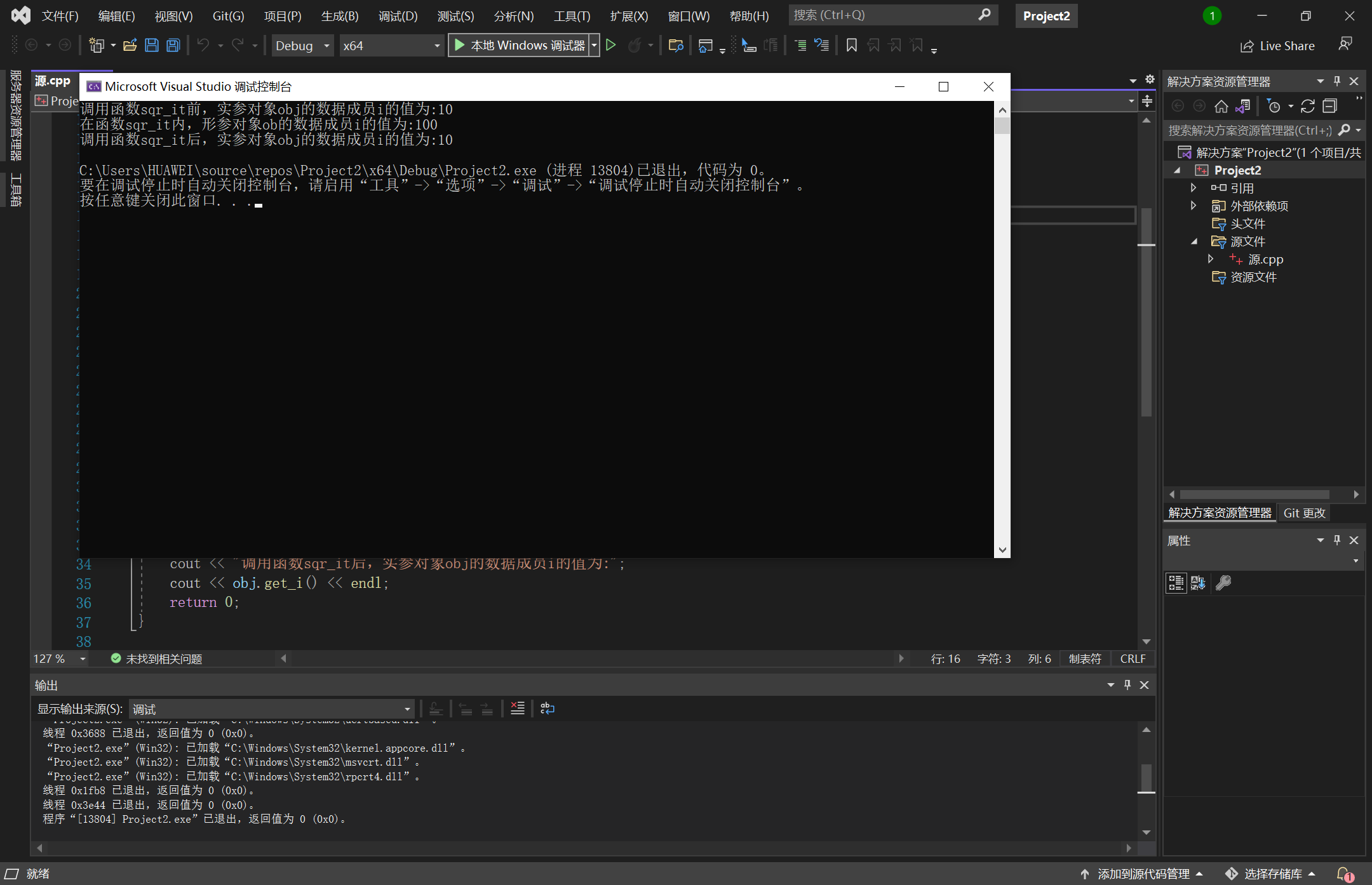
TStudent::ShowMoney();

return 0;

}

**程序结果：**

2



**感想心得：**

这次学到了C++ 标准库提供了大量的程序可以调用的内置函数。例如，函数 strcat () 用来连接两个字符串，函数 memcpy () 用来复制内存到另一个位置。函数还有很多叫法，比如方法、子例程或程序。并且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。创建一个TStudent类，完成以下功能：该类包括三个函数：InitStudent、ExpendMoney和ShowMoney；采用float m\_ClassMoney变量作为静态变量，用于存储班级的班费，初始值设为1000；InitStudent (char name[])主要负责完成学生姓名的初始化；ExpendMoney(float money)主要完成班费的花销计算；ShowMoney()主要完成班费余额显示；主程序中分别定义A、B、C三个学生，每个学生为一个对象，每个学生分别消费班费100，300，400最后显示班费的余额。让我在C++编程的学习中更进一层。

针对不同的函数调用模式，要函数名前的类型是返回值类型，后面跟的是参数类型和参数名称，调用的使用通过函数名调用，并根据参数类型填写参数即可，得到返回类型的返回值。有相对应的函数调用模式。普通的成员函数调用、指针函数调用、对象函数调用方式。

Copyright 2021-2099 Xiaojian. All rights reserved