实验内容1

了解三种不同的对象传递方式

程序代码

#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);

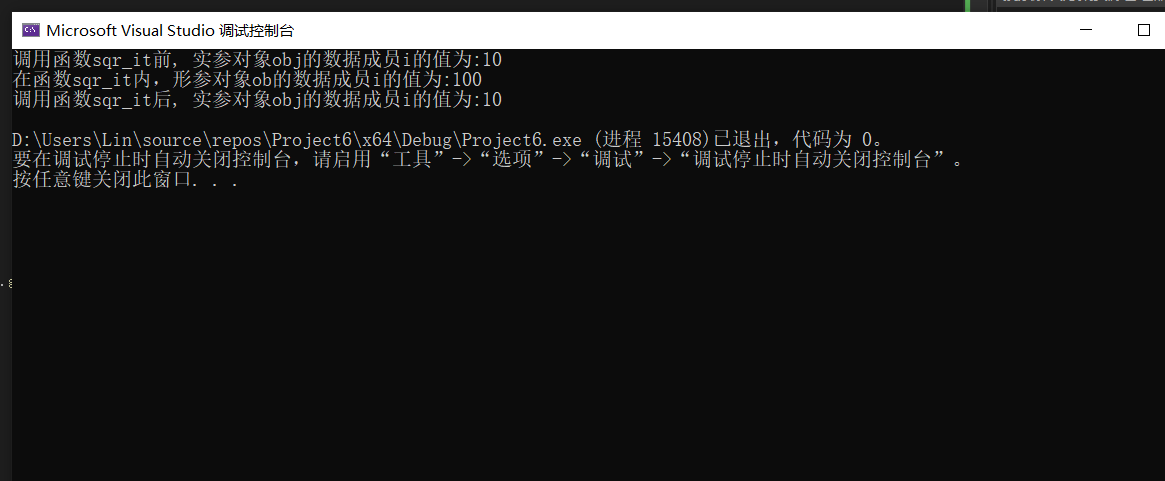
cout << "调用函数sqr\_it后, 实参对象obj的数据成员i的值为:";

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

return 0;

}

运行结果



实验内容2

* **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50,98.5,500.53，最后显示班费的余额；**

程序代码

#include <iostream>

#include <string>

using namespace std;

class TStudent

{

private:

char m\_Name[6];

static float m\_ClassMoney;

public:

void InitStudent(const char\*);

void ExpendMoney(const float);

static void ShowMoney();

};

float TStudent::m\_ClassMoney = 1000;

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

{

strcpy\_s(m\_Name, name);

}

void TStudent::ExpendMoney(const 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];

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

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

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

stu[0].ExpendMoney(50);

TStudent::ShowMoney();

stu[1].ExpendMoney(98.5);

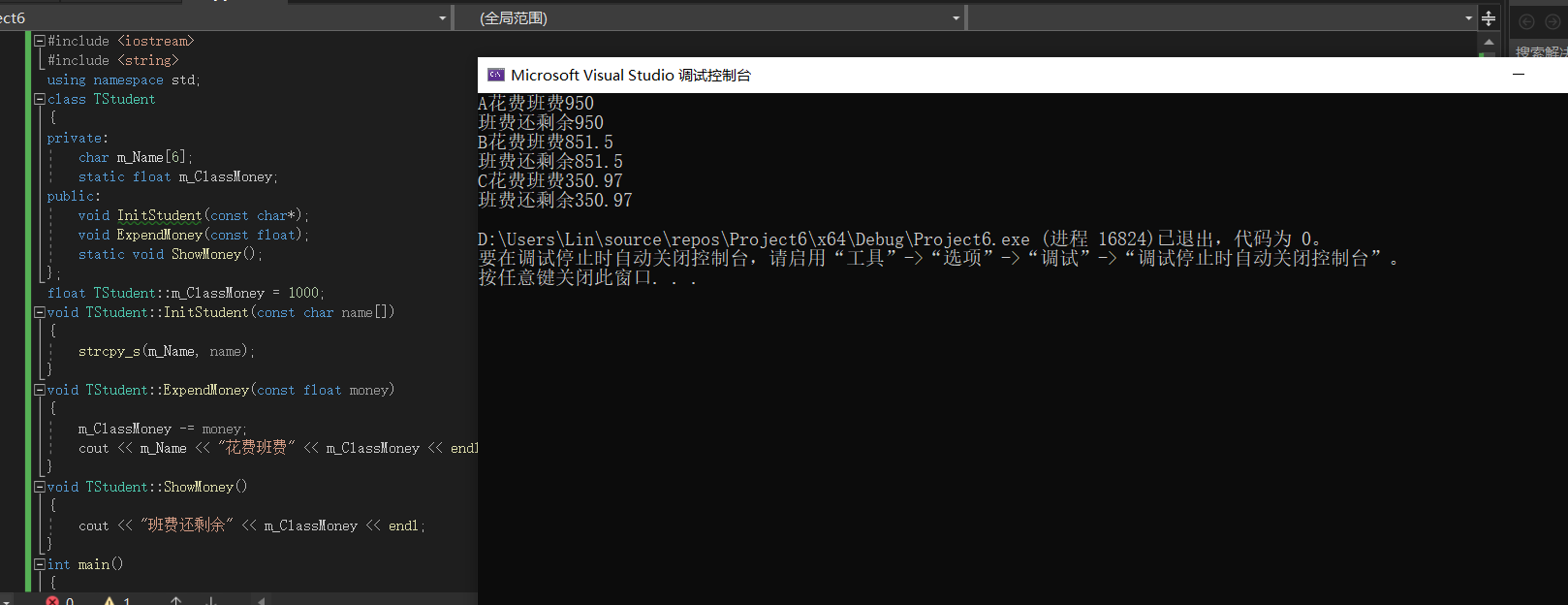
TStudent::ShowMoney();

stu[2].ExpendMoney(500.53);

TStudent::ShowMoney();

}

运行结果



感想心得

通过本次实验，我了解了静态成员的定义与作用，静态成员函数不访问普通数据成员，主要是为了访问和操作同类中的静态数据成员或全局变量。并且，静态数据成员必须在类外定义，它能满足面向对象的封装特性。记住要加static。