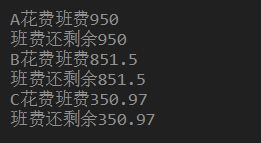
第四次实验报告

实验内容：  
#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(); //静态成员函数  
};  
float TStudent::ClassMoney=1000;  
void TStudent::InitStudent(string name)  
{  
 Name=name;  
}  
void TStudent::ExpendMoney(float money)  
{  
 ClassMoney-=money; //班费为原先的减去花费的  
 cout<<Name<<"花费班费"<<ClassMoney<<endl;  
}   
void TStudent::ShowMoney()   
{   
 cout<<"班费还剩余"<< 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;  
}  
程序结果：



感想心得：  
静态数据成员是可以让所有同类的对象共同使用一个数据成员，那么我们就能通过这个静态数据成员来观察到各个对象对他进行的操作，并且静态数据成员必须在类外定义：数据类型 类名：：变量名=初值；静态数据成员同时完全代替了全局变量，满足了面向对象的封装特性，并且一个类的静态成员函数与非静态成员函数不同，它不需要创建任何该类的对象就可以被调用。  
  
Copyright ©2021-2099 FangYuLiu. All rights reserved