电子科技大学信息与软件工程学院

**实验报告**

点名册序号 304-3 学 号 2015220103022

姓 名 张健顺

（实验） 课程名称 面向对象程序设计(C++)

理论教师 陈安龙

实验教师 陈安龙

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：张健顺 学号：2015220103022 指导教师：陈安龙**

**实验地点：信软楼西304 实验时间：16.10.25**

1. **实验名称：类和对象**
2. **实验学时：4**
3. **实验目的：**

1）理解类和对象的概念，掌握声明类和定义对象的方法；

2）掌握构造函数和析构函数的实现方法；

3）初步掌握使用类和对象编制C++程序；

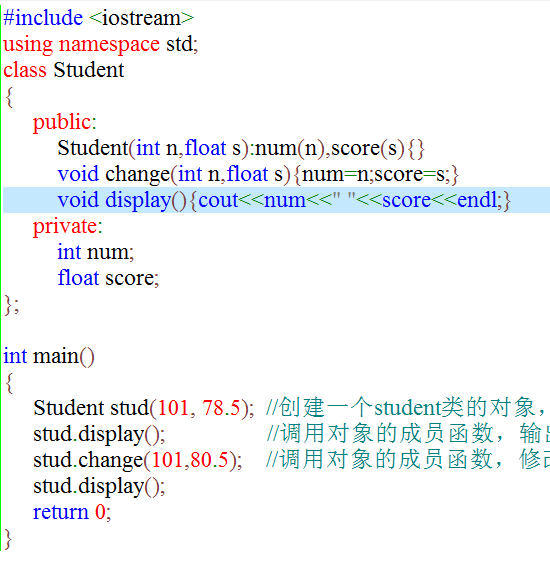
4）掌握使用对象、对象指针和对象引用作为函数参数的方法；

5）掌握静态数据成员和静态成员函数的使用方法；

6）理解友元的概念和掌握友元函数的使用方法。

1. **实验内容：**

1）阅读一段代码，分析其执行过程，然后上机运行，对比输出结果；修改上面的程序，增加一个fun函数，改写main函数。在main函数中调用fun函数，在fun函数中调用change和display函数。在fun函数中使用对象的引用（Student&）作为形参。



2）商店销售某一商品，商店每天公布统一的折扣（discount）。同时允许销售人员在销售时灵活掌握售价（price），在此基础上，对一次购10件以上者还可以享受9.8折优惠。现已知当天3个销货员销售情况为：

销货员号（num） 销货件数（quantity） 销货单价（price）

101 5 23.5

102 12 24.56

103 100 21.5

请编些程序，计算出当日此商品的总销售款sum以及每件商品的平均售价。要求用静态数据成员和静态成员函数。

3）设计一个立方体类Box，它能计算并输出立方体的体积和表面积。要求写出Box类的构造函数和析构函数。

4）阅读一段代码，将程序中的display函数不放在Time类中，而作为类外的普通函数，然后分别在Time和Date类中将display声明为友元函数。在主函数中调用display函数，display函数分别引用Time和Date两个类的对象的私有数据输出 年，月，日和时，分，秒。修改后上机调试和运行。

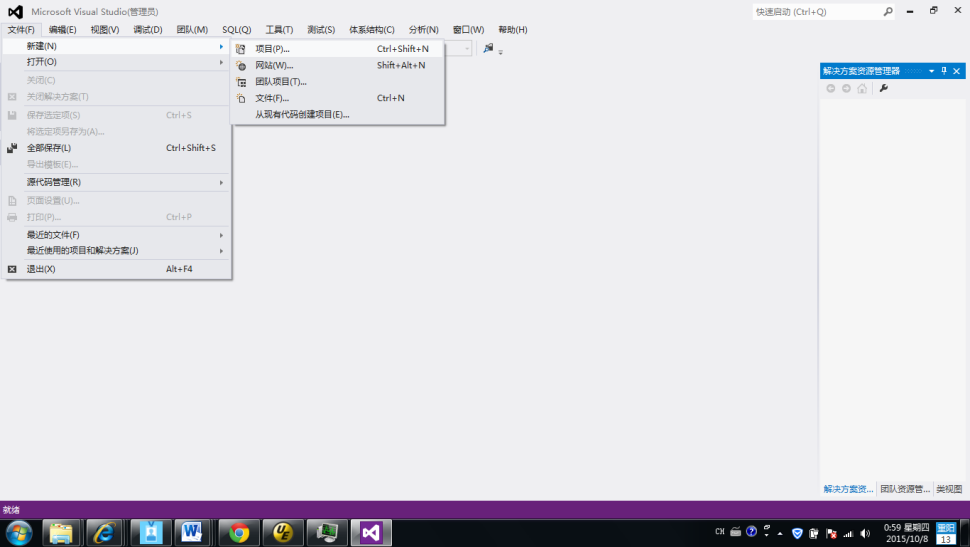
|  |
| --- |
| #include<iostream>  using namespace std;  class Date; //对Date类的提前引用声明  class Time //定义Time类  {  public:  Time(int, int, int); //构造函数  void display(Date &); //display是成员函数，形参是Date类对象的引用  private:  int hour;  int minute;  int sec;  };  class Date //声明Date类  {  public:  Date(int, int, int);  friend void Time::display(Date &); //声明Time中的display函数为友元成员函数  private:  int month;  int day;  int year;  };  Time::Time(int h, int m, int s) //类Time的构造函数  {  hour=h;  minute=m;  sec=s;  }  void Time::display(Date &d) //display的作用是输出年，月，日和时，分，秒  {  cout<<d.month<<"/"<<d.day<<"/"<<d.year<<endl;  cout<<hour<<":"<<minute<<":"<<sec<<endl;  }  Date::Date(int m,int d,int y) //类Date的构造函数  {  month=m;  day=d;  year=y;  }  int main()  {  Time t1(10,13,56); //定义Time类对象t1  Date d1(12,25,2004); //定义Date类对象d1  t1.display(d1); //调用t1中的display函数，实参是Date类对象d1  return 0;  } |

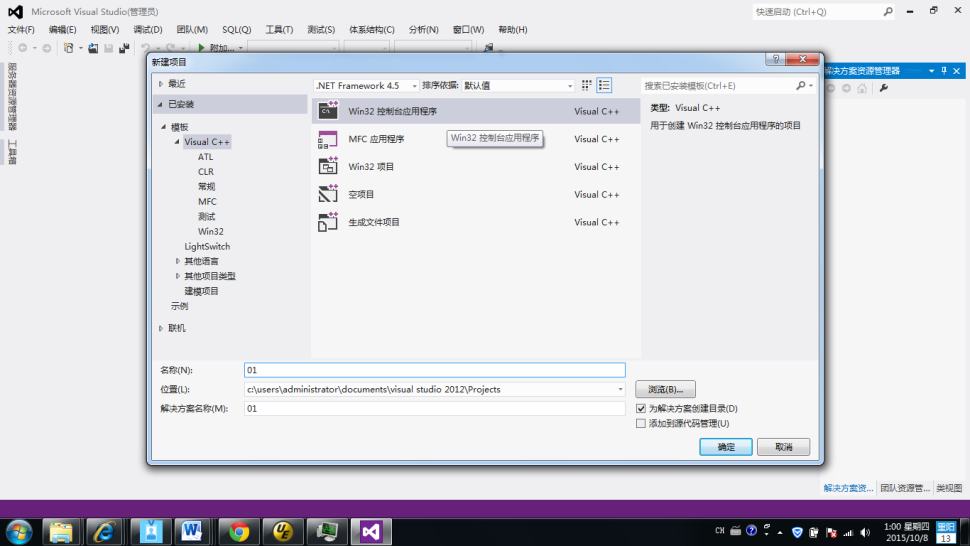
1. **实验器材（设备、元器件）：**

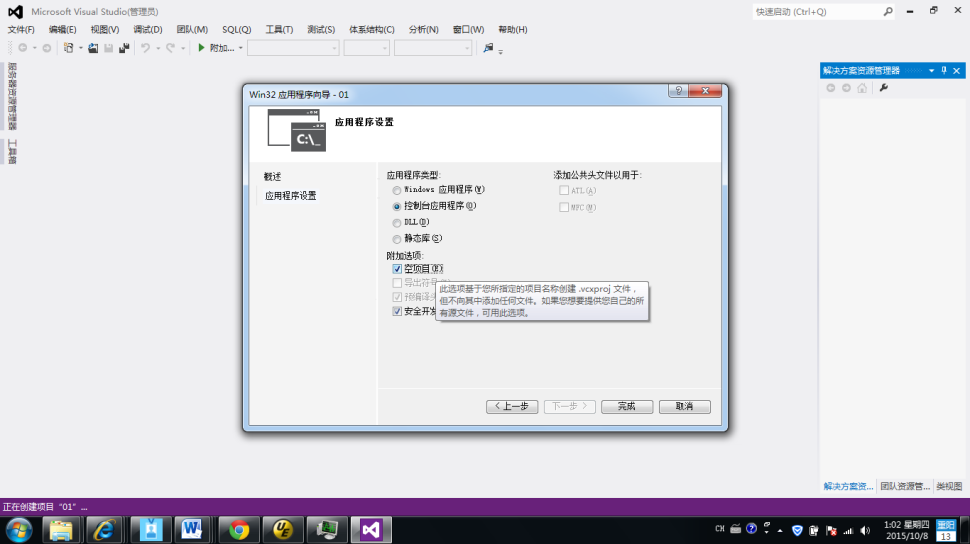
PC机一台，Windows 8.1，Visual Studio 2013

1. **实验步骤：**

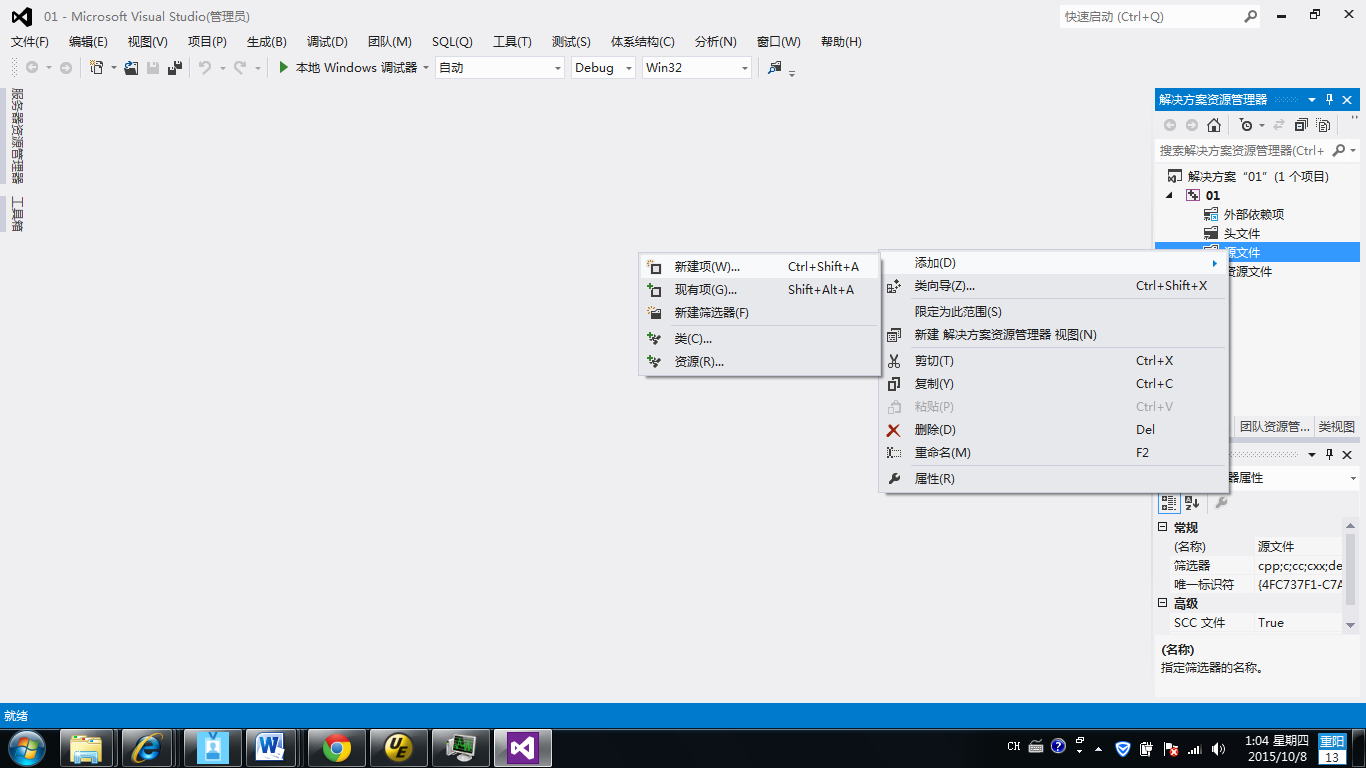
1、创建一个新的工程项目，选择项目的类型为“win32控制台应用程序”，主要选项如图所示。



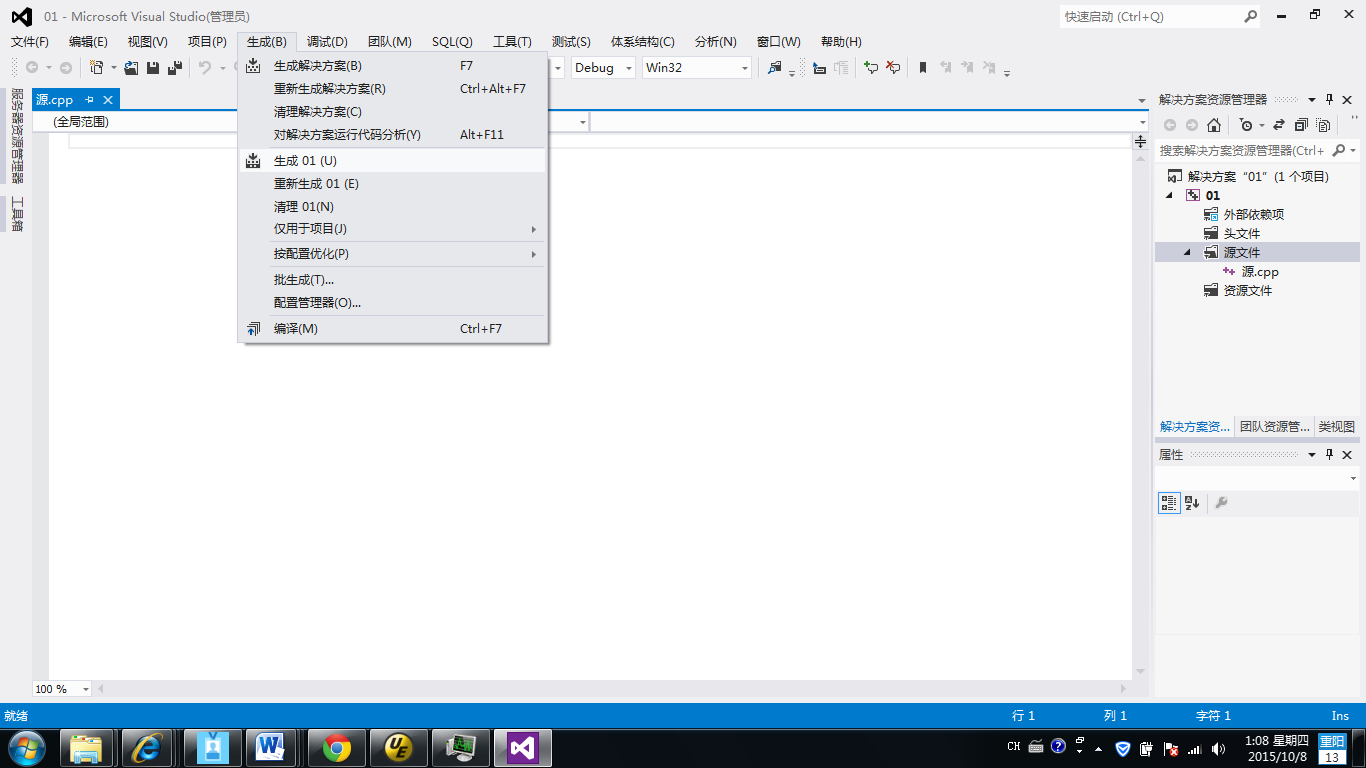


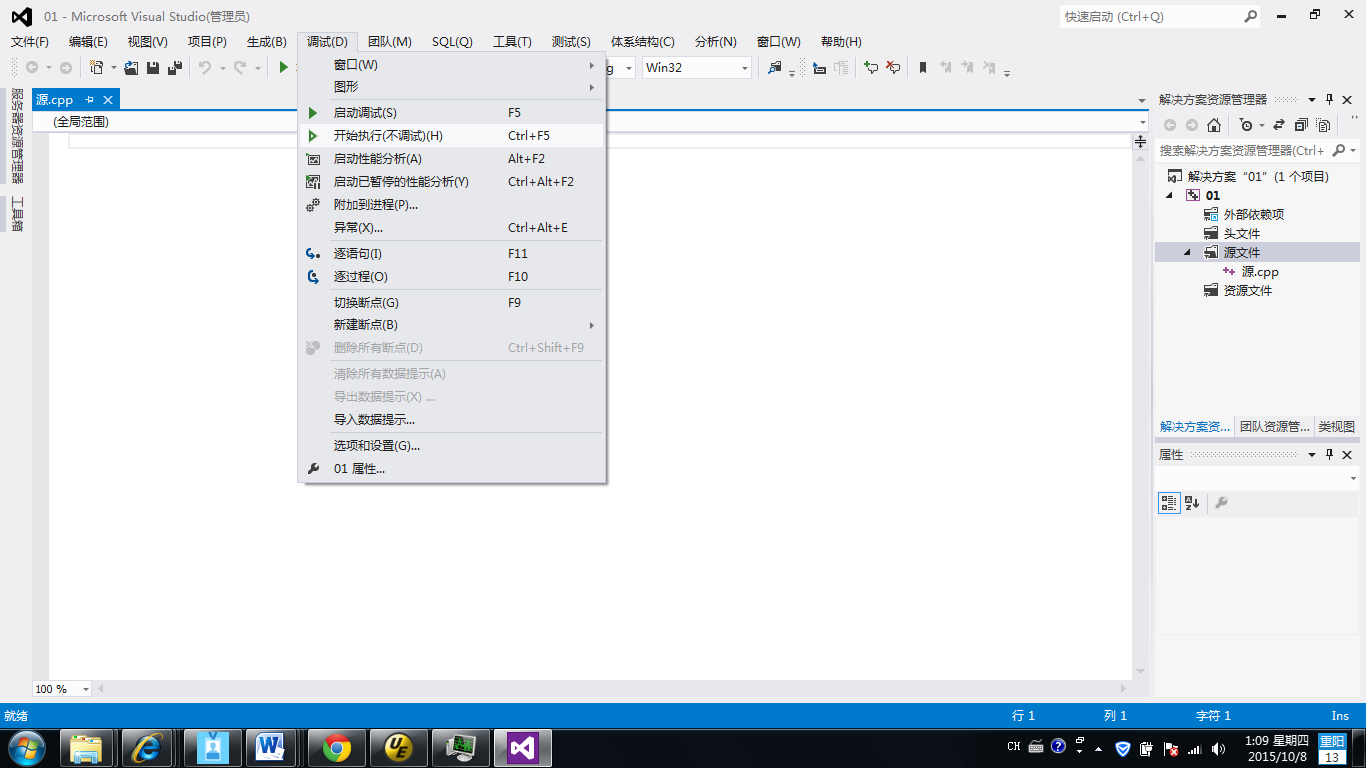


2、在创建完项目后，在项目的源文件中点击右键-》选择添加“新建项”-》选择“C++文件”，即可在新文件中输入源代码。



3、输入完源代码后，需要编译程序及运行程序。





1. 按照实验内容逐步完成实验操作
2. **实验结果与分析（含重要数据结果分析或核心代码流程分析）**

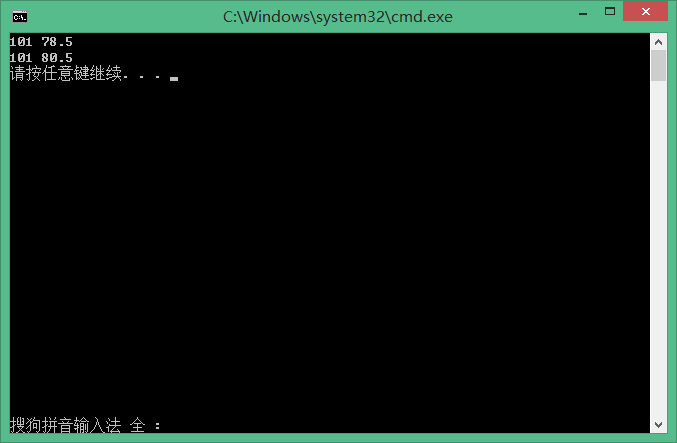
1. 原题目所给代码执行过程：

（1）创建一个Student对象stud并调用类中定义的构造函数进行初始化；

（2）调用对象的成员函数display()来输出初始化对象后的属性值；

（3）调用对象的成员函数change()来对对象的属性值进行修改；

（4）再次调用对象的成员函数display()来输出修改后的对象的属性值。



题目所给代码的两次输出（初始化、修改后）

修改程序：增加一个fun函数，改写main函数。在main函数中调用fun函数，在fun函数中调用change和display函数。在fun函数中使用对象的引用（Student&）作为形参。

#include<iostream>

using namespace std;

class Student{

public:

Student(int n, float s) :num(n), score(s){}

void change(int n, float s){

num = n; score = s;

}

void display(){

cout <<"修改代码后的输出结果："<< num << " " << score << endl;

}

private:

int num;

float score;

};

void fun(Student& stu) //传入student引用

{

stu.change(101, 80.5);

stu.display();

}

int main()

{

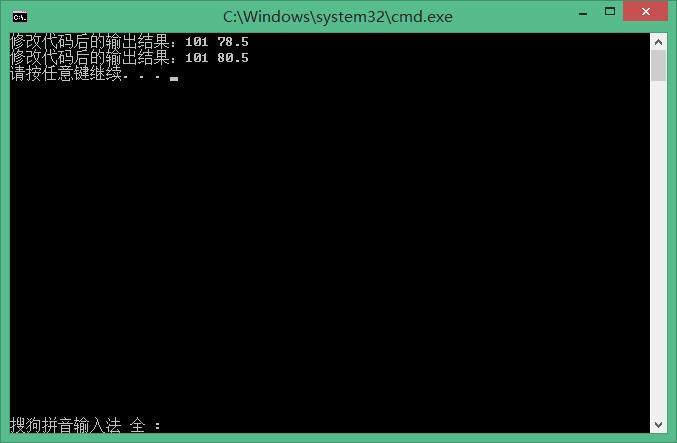
Student stud(101, 78.5);

stud.display();

fun(stud);

return 0;

}

****

修改代码后的输出结果

2.解决问题过程：

（1）设计出SalesMan类（对应的有销货员号、销货件数以及销货单价的成员变量同时设计出相应的统计方法）；

（2）根据题目要求使用静态数据成员和静态成员函数从而达到各个对象共享成员变量以及成员方法的效果，同时可自定义折扣：

static void dispaly();

double SalesMan::discount = 0.05;

double SalesMan::sumSalePrice = 0;

int SalesMan::sumSaleQuantity = 0;

1. SalesMan中相应方法，源码如下：

class SalesMan{

public:

SalesMan(){}

SalesMan(int num, int quantity, double price){

singleQuantity = quantity;

if (quantity>10){singlePrice = quantity\*price\*0.98;} else {singlePrice= quantity\*price;}

singlePrice \*= 0.95;

}

void total(){

sumSalePrice += singlePrice;

sumSaleQuantity += singleQuantity;

}

static void display(){

cout << "销售总额：" << sumSalePrice << endl << "平均售价："

<< sumSalePrice / sumSaleQuantity<<endl;

}

private:

int singleQuantity;

double singlePrice;

static int sumSaleQuantity;

static double discount;

static double sumSalePrice;

};

1. 控制台输入在main函数中对应的实现：

int main(){

SalesMan S[10]; int m，num，quantity; double price;

cout << "售货员人数：" ; cin >> m;

cout << "销货员号 销货件数 销货单价" << endl;

for (int i = 0; i < m; i++){

cin >> num >> quantity >> price;

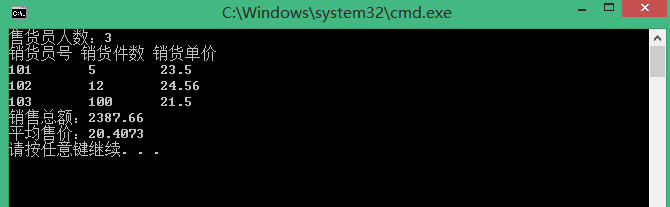
SalesMan temp(num, quantity, price); S[i] = temp;}

for (i = 0; i < m; i++) S[i].total();

SalesMan::display();

return 0;

}



输出结果

3.设计出的Box类（将其初始化为Box box(1,2,3);并在main函数中输出相应属性值以及调用求体积和求表面积的方法后的输出）

class Box{

public:

Box(int l, int w, int h) :length(l), width(w), height(h){

cout << "调用了构造函数！" << endl;

}

~Box(){

cout << "调用了析构函数"<< endl;

}

int Volume(){return height\*length\*width;}

int Superficialarea(){

return 2 \* (height\*width + height\*length + length\*width);

}

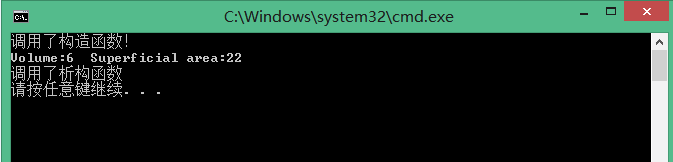
private:

int length;

int width;

int height;

};



输出结果

4.将程序中的display函数不放在Time类中，而作为类外的普通函数，然后分别在Time和Date类中将display声明为友元函数

void display(Date &d,Time &t){

cout << d.month << "/" << d.day << "/" << d.year << endl;

cout << t.hour << ":" << t.minute << ":" << t.sec << endl;

}

class Time{

public:

Time(int, int, int); //构造函数类外声明

friend void display(Date &,Time &); //形参是Date类对象的引用

private:

int hour; int minute; int sec;

};

class Date{

public:

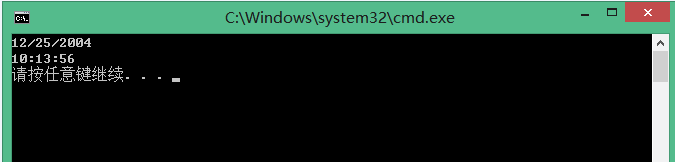
Date(int, int, int);

friend void display(Date &,Time &); //display函数为友元成员函数

private:

int month; int day; int year;

};



输出结果

1. **总结及心得体会：**

（1）根据需要解决的问题，确定好数据模型，并声明以及定义该数据模型对应的属性以及相应的方法从而定义出类，而对象则是类的具体定义与实现；

（2）构造函数主要用于对象的定义以及初始化的场景下，而复制构造函数则主要应用于对一个相同类型的对象的复制过程中以及某些函数中需要传递对象作为参数时需要调用复制构造函数；

（3）静态数据成员则主要应用于同类型多个对象共享一个数据成员，且均能对其进行读写；静态数据成员属于类，被该类的素有对象共享；

（4）需要在外部频繁访问类的某些私有成员，或者其他类需要和本类共享一个函数时则需要将函数或者某些类声明为友元函数与友元类。

（5）函数参数中传递引用和传递值/指针是完全不同的参数结合方式；前者传递给函数的是对象本身；而后两者传递的是对象的副本，或者是对象的地址，一旦涉及到对对象的修改时，操作的实体分别是对象本身和对象的复制品，从而使得结果会有所不同。

（6）对类中成员变量以及函数的定义可以在类外进行，但需要加上类名以及”::”进行限定。

（7）静态数据成员只能初始化一次，不能再次初始化，但可以对其做修改。

**九、对本实验过程及方法、手段的改进建议：**

（1）每个程序中涉及到的类的声明以及定义都可以放在头文件（.h文件）中，再在主程序中（.cpp文件）中引入该头文件，便于代码的管理；

（2）对于类的构造函数可以使用默认参数，防止初始化过程中遗漏某些参数，同时也能在主程序中无参地进行初始化，利于数据封装；

（3）构造函数和析构函数中可以输出相关函数执行语句，从而便于理解对象的产生与销毁的过程；

（4）对于某些需要用户输入大量数据的程序（例如第二题中的超市售价管理系统），可以考虑使用文件操作，利用文件的读写来进行信息的录入与写出。

**报告评分：**

**指导教师签字：**