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：16.11.15**

1. **实验名称：多态与虚函数程序设计**
2. **实验学时：4**
3. **实验目的：**

通过实验练习，让学生理解静态联编和动态联编两种不同的多态机制；理解赋值兼容原则及其实现方法；理解虚函数与普通函数重载的区别; 掌握通过虚函数实现运行时多态的方法;掌握纯虚函数的概念，以及抽象类的使用方法。

1. **实验内容：**

编写小型公司的工资管理程序。公司主要有3类人员：经理、技术人员和销售人员。要求存储并显示各员工的编号、姓名和月薪等基本信息。其中各类员工的月薪计算方法如下：

经理：固定月薪8000元；

技术人员：50元/小时；

销售人员：当月销售额的3%

编程完成如下任务：

（1）根据如上要求，设计基类Employee，包含各类员工共有的基本信息，以及静态数据成员totalNo，用于自动计算员工编号（员工编号从1开始）；基类定义构造函数和析构函数，另外还需要定义计算员工月薪和显示员工信息的成员函数，以规范各派生类的行为。 Employee类声明如下：

（2）分析其余各类人员的特征，设计合理的继承结构，并在Employee类的基础上派生出其余各类人员，每个类需要对从基类继承的虚函数进行重新定义。

class Employee

{

protected:

int No;

string name;

float salary;

static int totalNo； // 自动计算员工编号

public:

Employee (); //自动计算员工编号，姓名从键盘输入，工资初值为0

~Employee ();

virtual void pay ( )=0; //计算月薪

virtual void display( ) = 0; //显示人员信息

}；

（3）在主函数中通过基类指针指向各派生类对象，通过基类指针访问各派生类对象的成员函数，通过程序运行结果观察虚函数如何实现运行时多态。

（4）将基类的虚函数修改为普通成员函数，通过运行结果分析虚函数与普通函数重载的区别。

要求：

（1）理解静态联编和动态联编；

（2）理解赋值兼容原则；

（3）理解虚函数与普通函数重载的区别;

（4）掌握虚函数实现运行时多态的方法;

（5）掌握纯虚函数的概念，以及抽象类的使用方法。

实验思考

（1）虚函数和普通函数重载的区别；

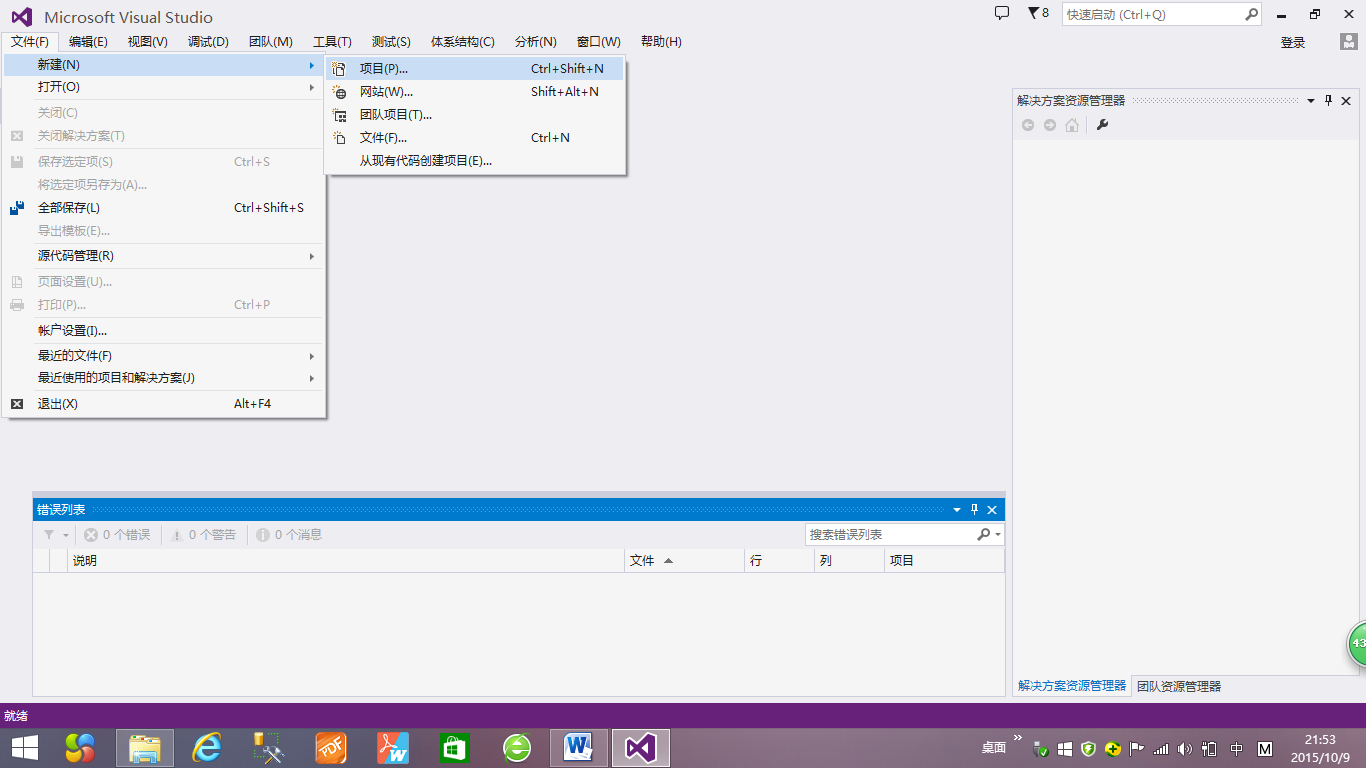
（2）虚函数如何实现多态。

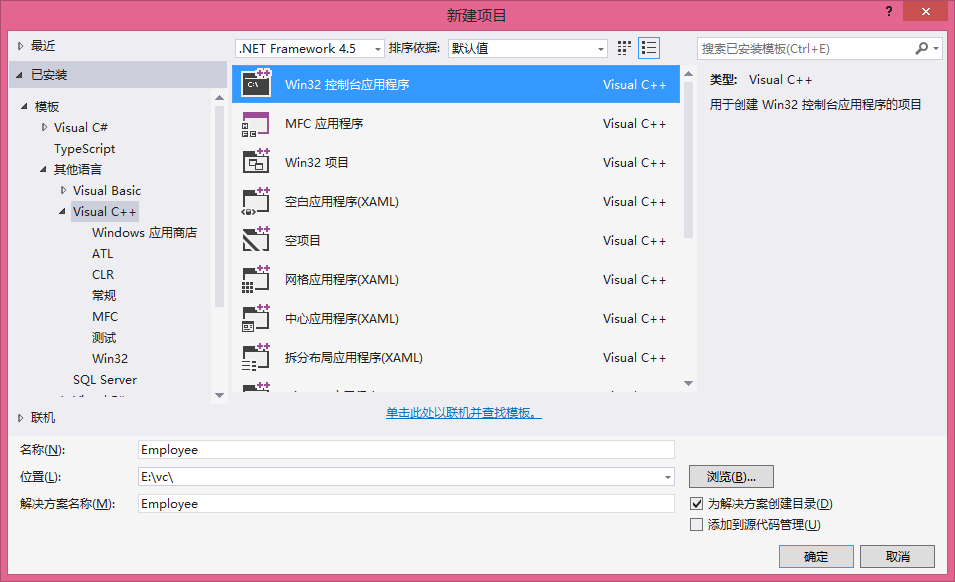
1. **实验器材（设备、元器件）：**

PC机一台，Windows 8.1，Visual Studio 2013

1. **实验步骤：**

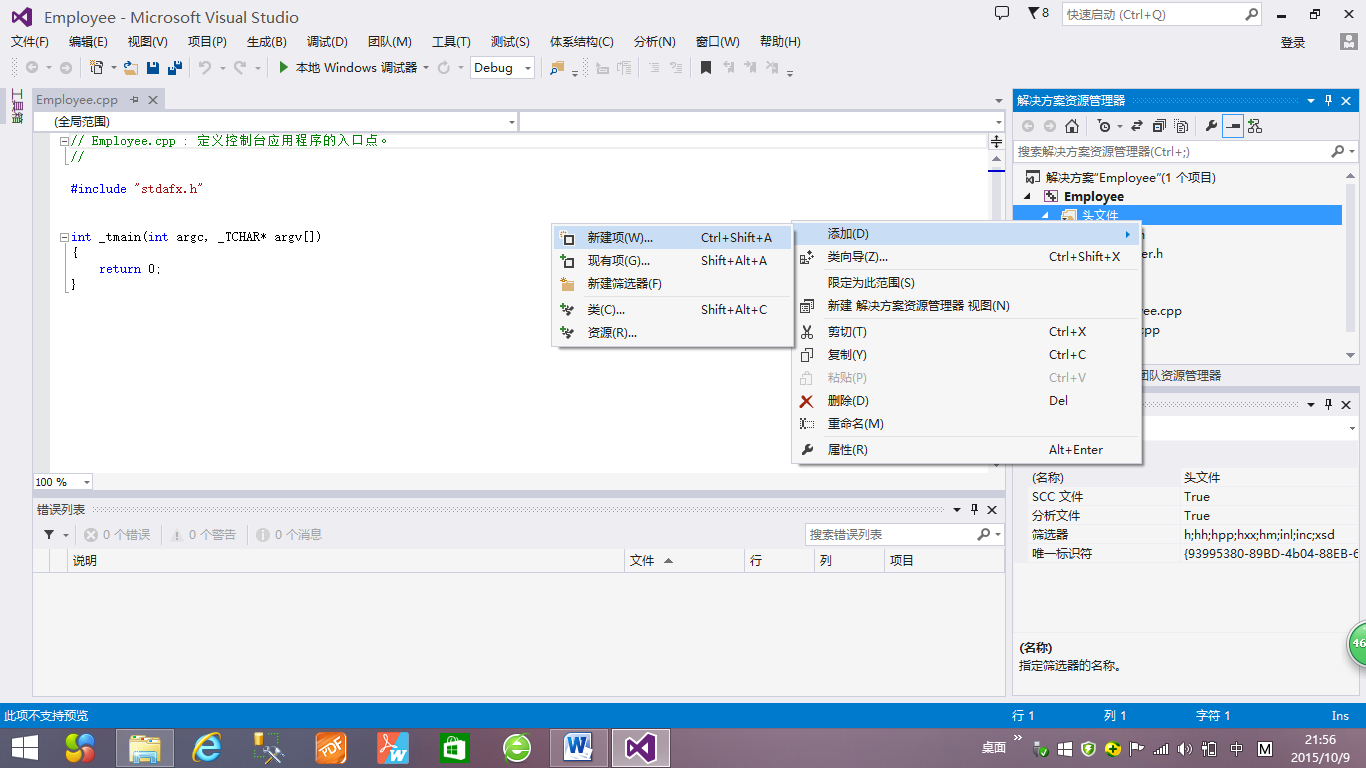
(1) 新建工程，名称设置为“Employee”，类型选择“Win32控制台应用”。

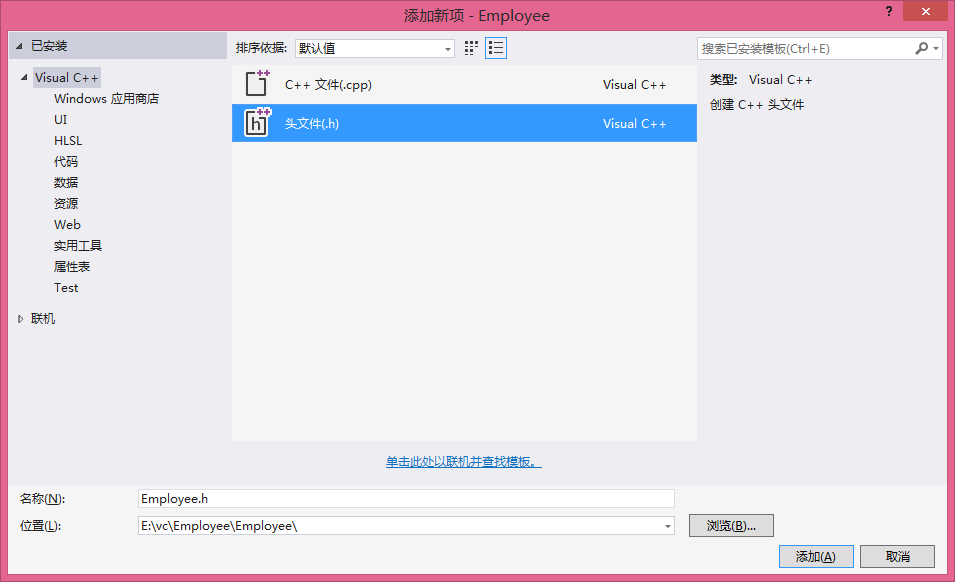




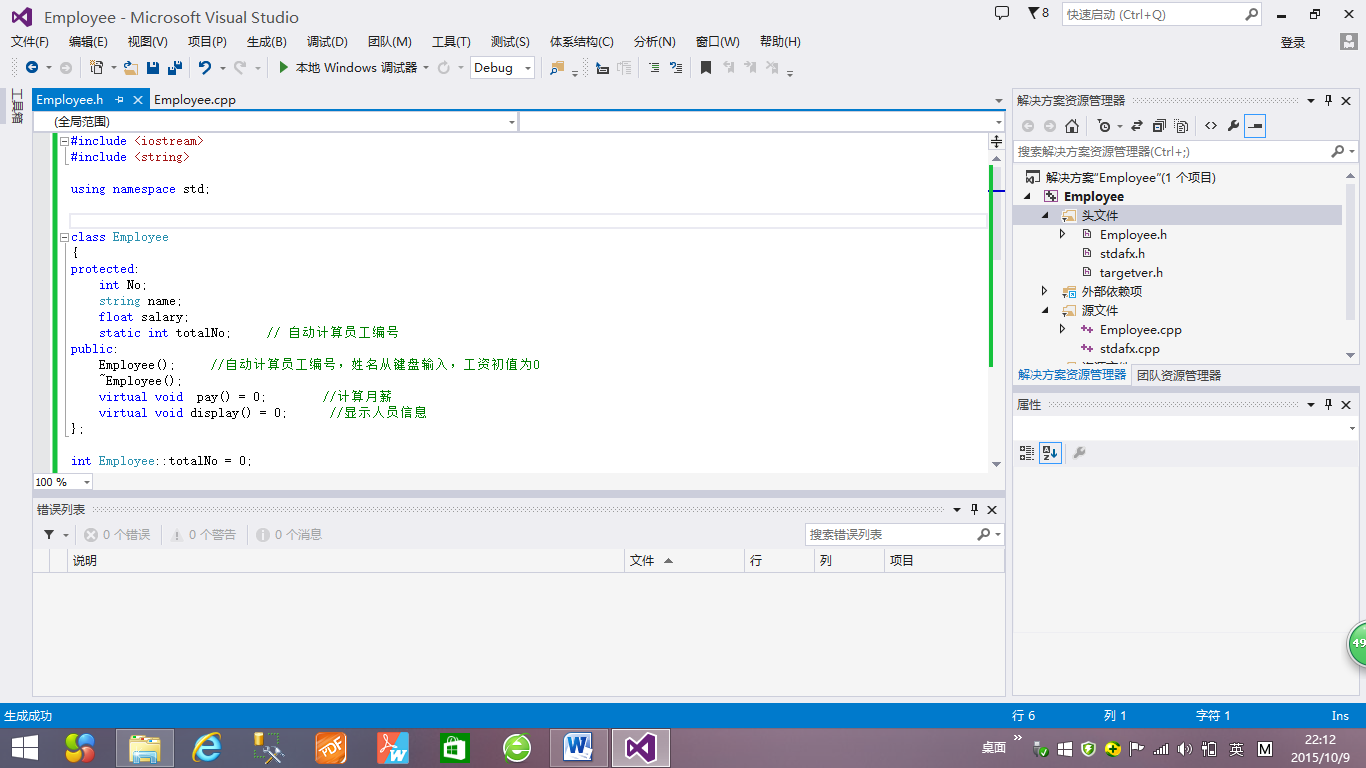
(2) 按应用程序向导选择控制台应用程序(Console application)，预编译头(Precompiled header)，安全开发声明周期(SDL)检查。

（3）参照下列图例，添加头文件Employee.h

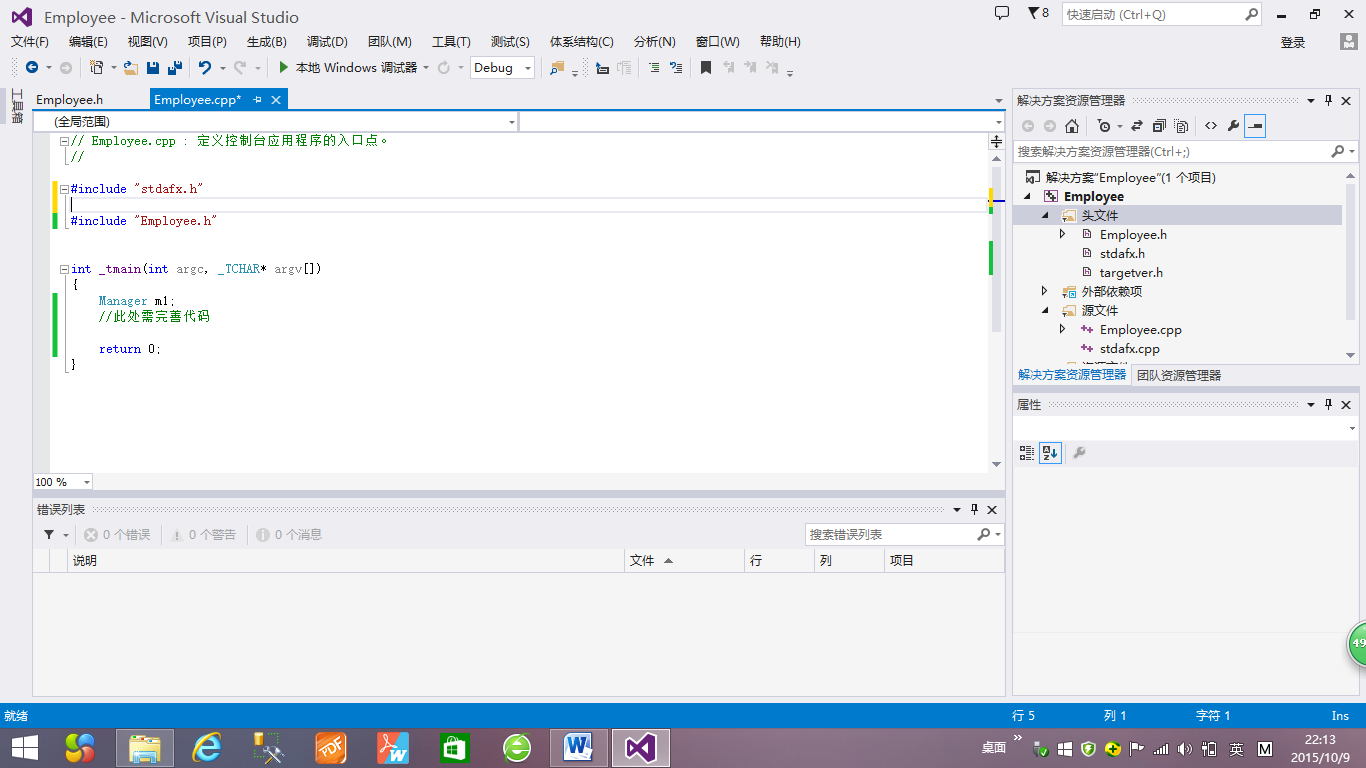




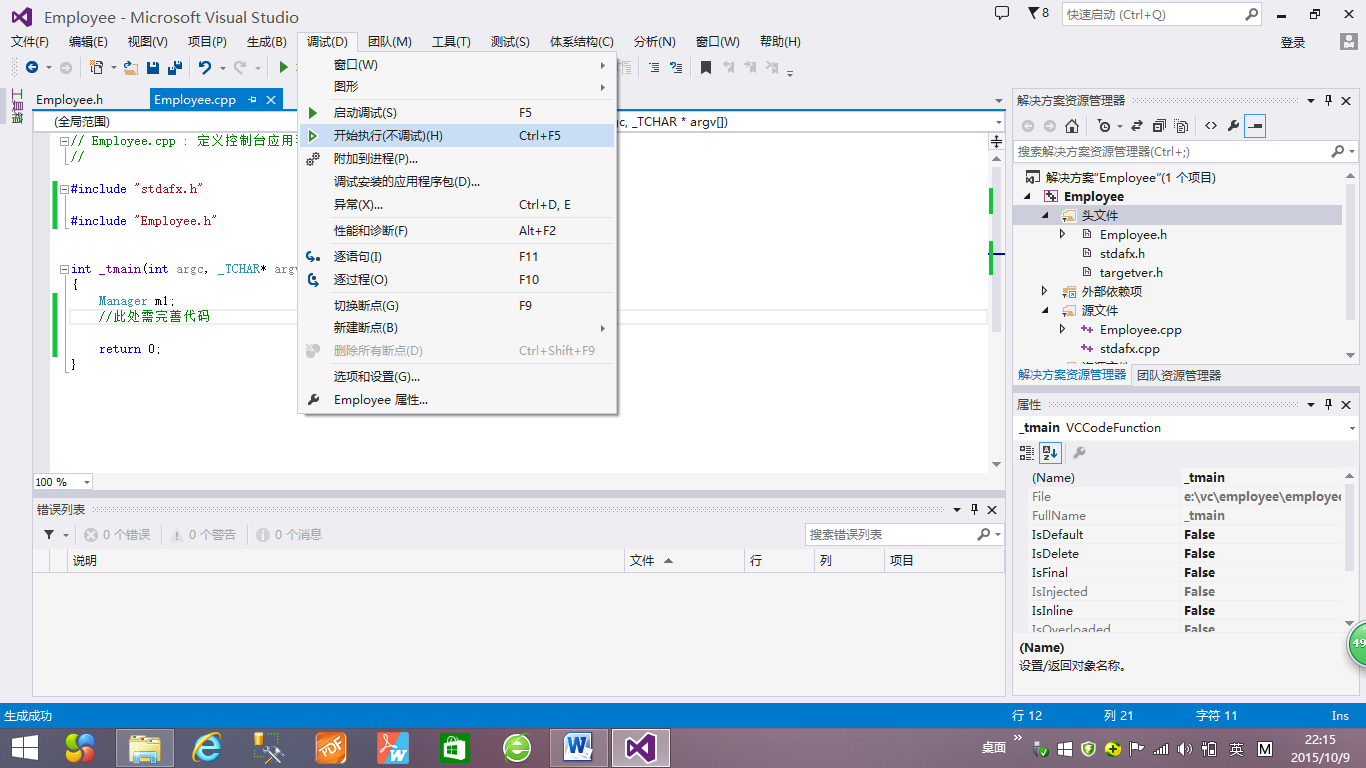
（4）在头文件中定义Employee类，并在此基础上派生出其他各类人员的类型，实现各类的成员函数。



（5）在Employee.cpp中编辑主程序。

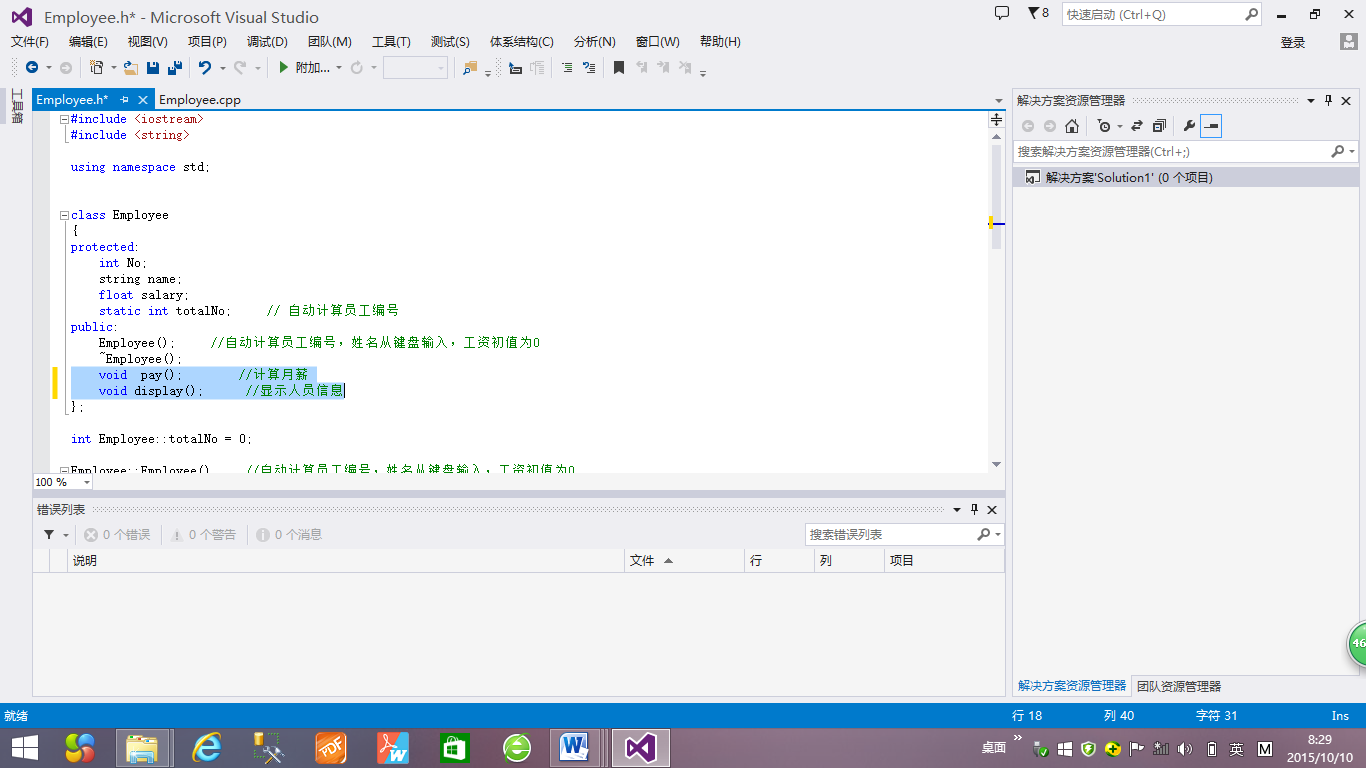


（6）调试程序，修改编译/连接错误



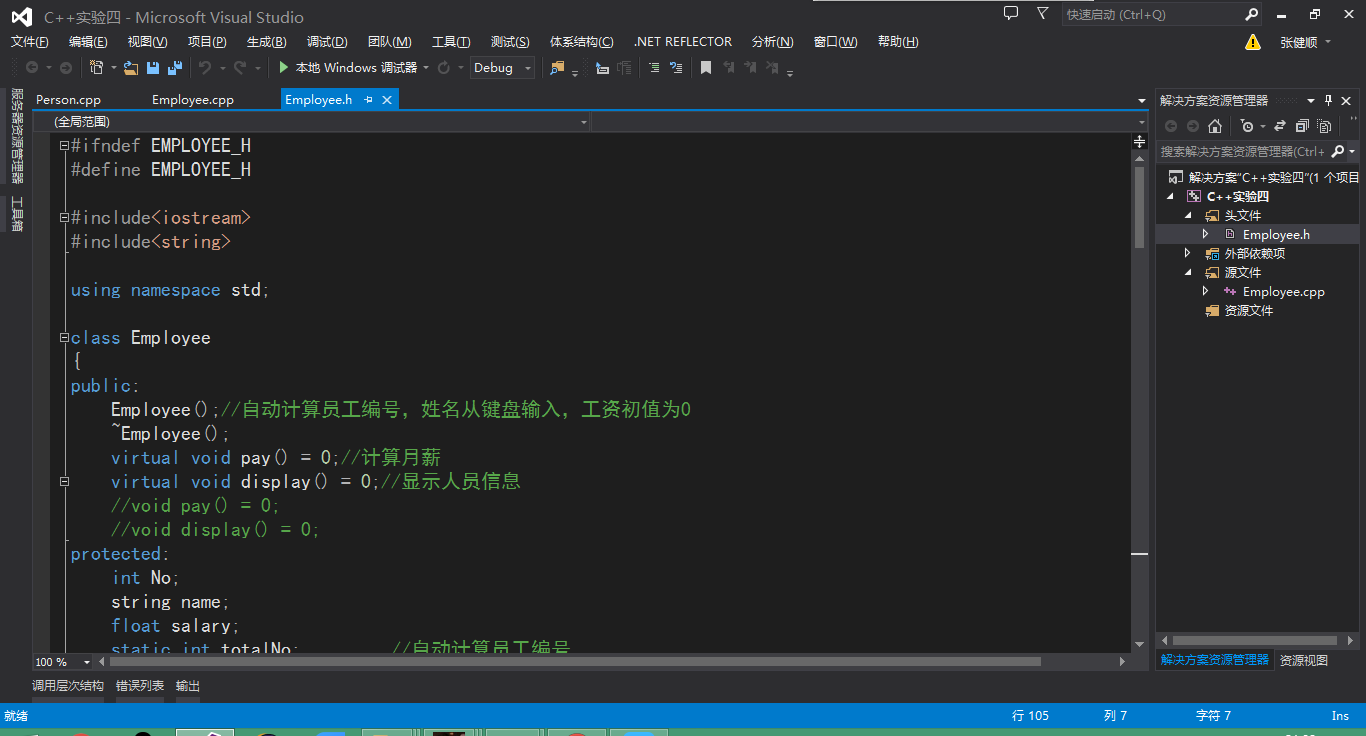
（7）运行程序，验证虚函数的多态性。

（8）将基类虚函数修改为普通成员函数，运行程序，分析虚函数与普通函数重载的区别。



**七、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）创建工程，创建头文件；



1. 在头文件中定义Employee类，并在此基础上派生出其他各类人员的类型，实现各类的成员函数。其中将计算月薪和输出人员信息的函数定义为虚函数。

class Employee

{

public:

Employee();//自动计算员工编号，姓名从键盘输入，工资初值为0

~Employee();

virtual void pay() = 0;//计算月薪

virtual void display() = 0;//显示人员信息

protected:

int No; string name; float salary;

static int totalNo; //自动计算员工编号

};

int Employee::totalNo = 0;

Employee::Employee(){

salary = 0; cout << "Please enter the name:";

cin >> name; totalNo++; No = totalNo;

}

Employee::~Employee(){}

void Employee::pay(){

cout << "基类pay函数执行" << endl;

}

void Employee::display(){

cout << "基类display函数执行" << endl;

}

其中构造函数可以自定义输入人员姓名，同时定义了静态变量员工编号，将会在每一次Employee构造函数的调用中进行编号的自动计算。

派生类：

A、经理（Manager）类（定义私有成员变量monthSalary）

class Manager : public Employee

{

public:

Manager() :Employee(){

monthPay = 8000; //经理固定月薪8000

}

void pay(){

salary = monthPay;//月薪

}

void display(){

cout << "编号为" << No << "的经理人员 " << name

<< " 本月薪水为" << salary << endl;

}

private:

float monthPay;

};

B、技术人员（Technicist）类(定义了私有成员变量hours和hourPay)

class Technicist : public Employee

{

public:

Technicist() :Employee(){

hourPay = 50;//技术人员：50元每小时

}

void pay(){

cout << "Please enter the work hours of technicist:";

cin >> hours;

salary = hours\*hourPay;//工作时长\*每小时的工资

}

void display(){

cout << "编号为" << No << "的技术人员 " << name << " 本月薪水为" << salary << endl;

}

private:

float hours; //工作时长

float hourPay; //每个小时支付薪水

};

对于技术人员，除开自定义输入的个人信息以外还将自定义输入工作时长。

C、销售人员（SalesMan）类（定义了私有成员变量TotalSale和percantage）

class SalesMan : public Employee

{

public:

SalesMan() :Employee(){

percantage = 0.03;//销售人员：当月销售额的百分之三

}

void pay(){

cout << "Please enter the total sales of the salesman:";

cin >> TotalSale;

salary = TotalSale\*percantage;//销售额\*提成比例

}

void display(){

cout << "编号为" << No << "的销售人员 " << name << " 本月薪水为" << salary << endl;

}

private:

float percantage; //提成比例

int TotalSale; //销售额

};

1. 在Employee.cpp中编辑主程序

int main()

{

cout << "Manager:";

Manager manager;

cout << "Technicist:";

Technicist technicist;

cout << "SalesMan:";

SalesMan salesman;

Employee \*p1 = &manager,\*p2=&technicist,\*p3=&salesman;

p1->pay();

p1->display();

p2->pay();

p2->display();

p3->pay();

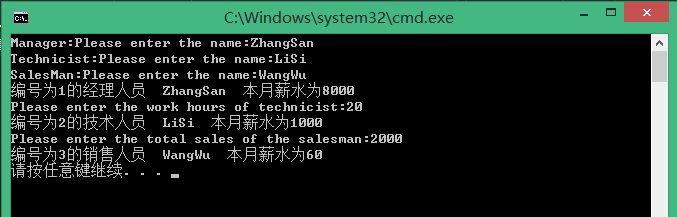
p3->display();

return 0;

}

p1,p2,p3均为父类指针，然后分别指向三个不同派生类的对象

**运行结果：**



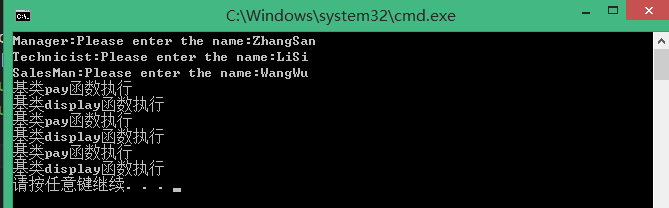
由于调用到了派生类各自的构造函数，所以需要自定义输入Manager、Technicist、SalesMan的姓名，又因为将基类的pay()和display()函数声明为了虚函数，所以再利用基类指针声明派生类对象并调用相应函数的的时候会调用派生类中自定义的函数，所以需要再自定义输入技术人员工作时长以及销售人员销售额。按照程序中设定好的值进行相关计算。

（4）将基类虚函数修改为普通成员函数，运行程序，分析虚函数与普通函数重载的区别。

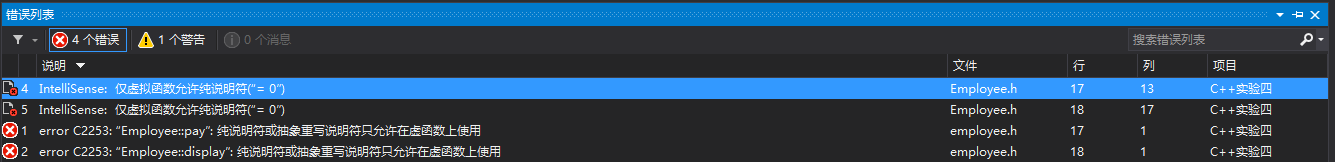
void pay() ; //注意此时不能再将纯说明符加入到函数末尾

void display() ; //仅虚拟函数允许使用纯说明符

**运行结果：**



由运行结果可知：

由于声明为父类指针的作用，而函数为普通成员函数，故调用对象相应的构造函数的时候会调用父类的构造函数和析构函数，由于构造函数和虚构函数是在派生类的构造函数中和虚构函数中调用过，但无法调用派生类各自相应的pay()函数和display()函数，只能调用基类的pay()和display()函数。

**八、总结及心得体会：**

（1）虚函数与普通函数重载的区别：

A.函数重载可以用于非成员函数和类的成员函数，而虚函数只能用于类的成员函数

B.函数重载可用于构造函数，而虚函数不能用于构造函数

C.重载的函数必须具有相同的函数名，函数类型可以相同也可以不同，但函数的参数个数和参数类型二者中至少有一个不同，否则在编译时无法区分。而虚函数则要求同一类族中的所有虚函数的函数名，函数类型，函数的参数个数和参数类型都全部相同，否则就不是重定义了，也就不是虚函数了

D.函数重载是在程序编译阶段确定操作的对象的，属于静态关联。虚函数是在程序运行阶段确定操作对象的，属于动态关联。

（2）纯虚函数与抽象类

纯虚函数是一个在基类中说明的虚函数，它在该基类中没有定义，要求任何派生类都必须定义自己的版本。

如果一个类至少有一个纯虚函数，那么就该称该类为抽象类。

**九、对本实验过程及方法、手段的改进建议：**

可以通过在Employee类中设置纯虚函数的方法将Employee类定义为抽象类，当然也要对其中的函数进行相应的改造。

**报告评分：**

**指导教师签字：**