

电子科技大学信息与软件工程学院

标 准 实 验 报 告

(实验) 课程名称: 面向对象程序设计 C++

电子科技大学教务处制表

学生姓名：罗悦 学 号：2016220304022 指导教师：钱伟中

实验地点：信软学院 实验时间：2018-12-12

一、实验室名称：信软学院实验室

二、实验项目名称： 多态与虚函数程序设计

三、实验学时：2 学时

四、实验原理：

本次实验通过虚函数实现运行时多态，统一接口。

五、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。

六、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：

```
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
};

```

(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。

(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。

七、实验器材（设备、元器件）：

PC 计算机、Windows 系列操作系统 、Visual Studio2013 软件

八、实验步骤：

- 1) 创建工程；
- 2) 添加头文件；
- 3) 定义基类，并在基类基础上定义各派生类；
- 4) 在各派生类中重新实现基类的成员函数；
- 5) 编写主程序，验证虚函数的多态性；
- 6) 编译链接，并运行程序；
- 7) 将基类的虚函数改为普通成员函数，运行程序，观察虚函数与普通成员函数重载的区别。

九、实验程序及结果分析：

实验程序：

Employee.h 头文件

```
#include<iostream>

#include<string>

using namespace std;

class Employee
{
    public:
        Employee(string n){
            totalNo=totalNo+1;

            salary=0;

            name=n;

            No=totalNo;

        }

        ~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;

class jingli:public Employee
{
    public:
```

```

    jingli(string n):Employee(n){}

    void pay(){
        salary=8000;
    }

    void display(){
        cout<<"员工身份:经理"<<endl;
        cout<<"编号:"<<No<<endl;
        cout<<"姓名:"<<name<<endl;
        cout<<"薪水:"<<salary<<endl;
    }
};

class jishu:public Employee
{
    public:
        jishu(string n,int t):Employee(n),time(t){}

        void pay(){
            salary=time*50;
        }

        void display(){
            cout<<"员工身份:技术人员"<<endl;
            cout<<"编号:"<<No<<endl;
            cout<<"姓名:"<<name<<endl;
            cout<<"薪水:"<<salary<<endl;
            cout<<"工作时长:"<<time<<endl;
        }

    private:
        int time;

```

```

};

class xiaoshou:public Employee
{
    public:
        xiaoshou(string n,int x):Employee(n),xs(x){}
        void pay(){
            salary=x*0.03;
        }
        void display(){
            cout<<"员工身份:销售人员"<<endl;
            cout<<"编号:"<<No<<endl;
            cout<<"姓名:"<<name<<endl;
            cout<<"薪水:"<<salary<<endl;
            cout<<"当月销售额:"<<xs<<endl;
        }
    private:
        int xs;
};

```

如上 Employee.h 头文件所示：我定义了一个基类 Employee，基类中包含着保护段成员变量编号、姓名、薪水和一个计算员工编号所使用的静态变量 totalNo，并且定义了构造函数对成员变量进行初始化，然后对 totalNo 变量进行加一操作。并且定义了两个纯虚函数 pay()和 display()，此时基类 Employee 为抽象函数。

之后我定义了三个派生类来继承基类 Employee，都为 public 继承。在 jishu 类中定义了其私有变量“time”来记录技术人员的工作时长，并用来计算薪水。在 xiaoshou 类中定义了其私有变量“xs”来记录当月的销售额。

然后我重载了 pay()函数和 display()函数，分别用来计算该类中的成员的薪水和输出对象的基本信息。

.cpp 主程序

```
#include "stdafx.h"
#include <iostream>
#include <cmath>
#include "Employee.h"
using namespace std;

int main()
{
    string name;
    int time;
    int xs;
    cout<<"请输入经理的名字:"<<endl;
    cin>>name;
    Employee *p1=new jingli(name);
    p1->pay();
    p1->display();
    cout<<"-----"<<endl;

    cout<<"请输入技术人员的名字和工作时长:"<<endl;
    cin>>name;
    cin>>time;
    Employee *p2=new jishu(name,time);
    p2->pay();
    p2->display();
    cout<<"-----"<<endl;

    cout<<"请输入销售人员的名字和当月销售额:"<<endl;
```

```

    cin>>name;

    cin>>xs;

    Employee *p3=new xiaoshou(name,xs);

    p3->pay();

    p3->display();

    cout<<"-----"<<endl;

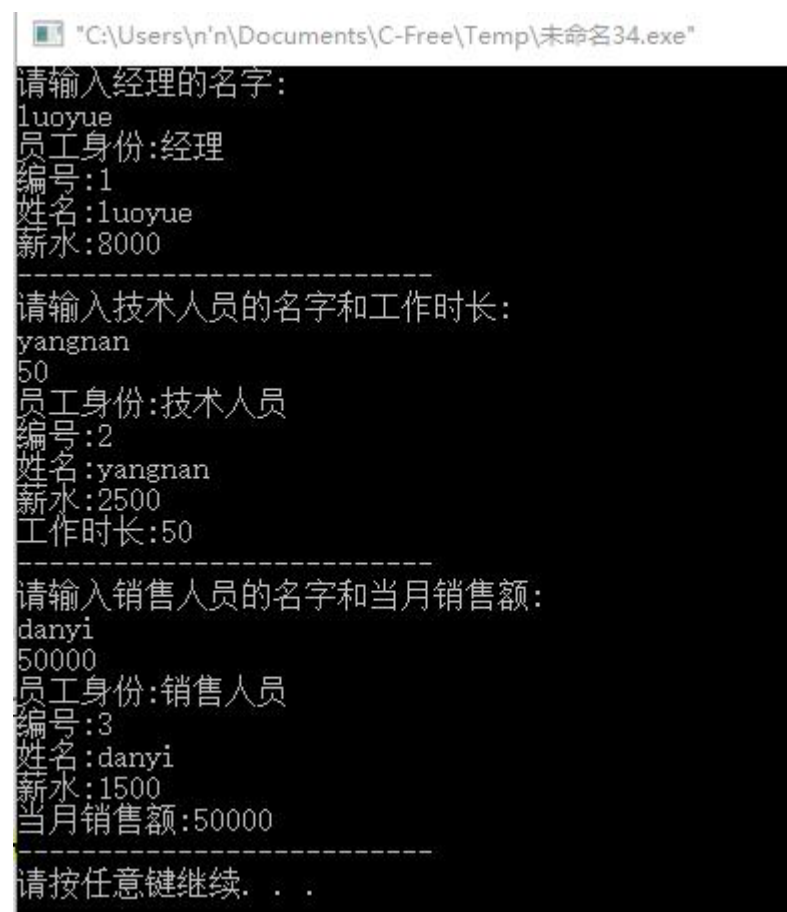
    return 0;

}

```

如上主函数所示：我定义了三个基类指针并分别用 jinli()、jishu()、xiaoshu() 三个构造函数进行初始化。经理对象只需要输入经理名字，技术人员需要输入名字和工作时长，而销售人员需要输入名字和当月销售额。

然后分别调用 pay()函数来计算薪水，之后调用 display()函数来输出对象的基本信息。输出结果如下图所示：



```

"C:\Users\n\n\Documents\C-Free\Temp\未命名34.exe"
请输入经理的名字:
luoyue
员工身份:经理
编号:1
姓名:luoyue
薪水:8000
-----
请输入技术人员的名字和工作时长:
yangnan
50
员工身份:技术人员
编号:2
姓名:yangnan
薪水:2500
工作时长:50
-----
请输入销售人员名字和当月销售额:
danyi
50000
员工身份:销售人员
编号:3
姓名:danyi
薪水:1500
当月销售额:50000
-----
请按任意键继续. . .

```

如上图所示：分别输出三个对象的基本信息。

//修改虚函数为普通成员函数

<pre>void pay (){}; //计算月薪 void display() {};</pre>	<pre> //显示人员信息</pre>
---	--------------------------------

结果分析：

当基类中的函数 `pay()` 和 `display()` 不再是虚函数后，用指针调用派生类的方法时，并不会达到目的，而是去调用了基类中的相应的函数，达不到我们预期的效果。

十、实验结论：

当基类中的函数被定义为纯虚函数后，此时基类便是一个抽象类，必须在其派生类中进行重写程序才不会报错，同时基类定义的指针便能调用派生类中的各个成员方法，而不是调用基类中的方法。

实现多态的方法：基类声明虚函数，然后子类重定义基类声明的虚函数，之后程序中用基类对象引用或指针调用虚函数。

函数重载是在同一个类中，相同名称不同形式参数的若干个函数，因此只要参数不同就可以调用这些同名称而不同内容的函数。虚函数是父类与子类中名称相同且参数相同的函数，因此在定义对象时，如果对象是父类的对象执行的是父类的虚函数，如果对象是子类的对象执行的是子类虚函数。根据以上的说明，两者在概念和使用方式上当然是不同的。如果是涉及的是同个类中的对象，那就是的重载。如果两个类是父类与子类的关系，调用的函数是它们都有的，那么是虚函数调用。

十一、总结及心得体会：

本次实验使我更加熟悉类 C++ 的基本语法和继承的使用方法，并且对纯虚函数有了更深的理解，纯虚函数的类在其派生类中必须定义自己这个函数的版本，而且纯虚函数是没有实际意义的，他的目的告知编译器派生类将会定义自己的版本。类中拥有纯虚函数表示这个类是抽象类，不存在此类的对象。而虚函数仅表示派生类可以定义自己的版本，但是基类也可以有意义，若没有定义自己的版本，

将使用基类的版本。

十二、对本实验过程及方法、手段的改进建议：

我认为本实验充分的利用了一些 C++ 的基本程序操作手段，无修改意见。

报告评分：

指导教师签字：