**重庆交通大学信息科学与工程学院**

**综合性设计性实验报告**

**班 级： 计科1703**

**姓名 学号： 黄富灵 631707060308**

**实验项目名称： 实验四、继承与多态**

**实验项目性质： 验证性、设计性**

**实验所属课程： 《Java程序设计》**

**实验室(中心)：**

**指 导 教 师 ： 王桂平**

**实验完成时间： 2019年 3月 18日**

**一、实验概述**

**【实验目的】**

1. 熟练掌握构造器的定义和重载。

2. 熟练掌握类的继承。

3. 熟练使用访问控制符实现所需的封装要求。

4. 熟练掌握多态。

**【实验要求】**

1. 按照题目要求正确建立实验文件夹。

2. 使用IDE工具编写Java程序时，按照题目要求正确地建立项目、包、类等，正确地把相关文件保存在指定的位置。

3. 按题目要求记录实验结果。

**【实施环境】（使用的材料、设备、软件）**

JDK、NetBeans、Eclipse

**二、实验内容**

**每次实验前，请确认是否已经严格按实验一中的规范新建了实验所需的文件夹，是否已经按实验一中的规范新建了NetBeans和Eclipse项目。**

**【实验报告和代码提交】完成实验报告后，将“学号+姓名”文件夹压缩，发给班长，由班长汇总后再发到w\_guiping@163.com。课程结束后，再把所有实验报告打印一份交给老师存档。**

**实验任务一 构造器的定义和使用**

**第1题 定义丰富的构造器**

【实验内容】

改写实验三中的Student类，要求通过构造器重载的方式为Student类提供多种设置属性的方式，具体要求为：

1) 在ExpInNetBeans或ExpInEclipse项目中新建类Student，指定包名为“exp4”；

package exp4;

public class Student {

public int age; //年龄

public String name; //姓名

public String department; //院系名称

//要求以下定义多个构造器，例如初始化时如果没有提供年龄，则将年龄设为默认值18；

//如果没有院系名称，则将院系设为默认值"地信"；等等

}

2) 在项目中继续新建类TestStudent，指定包名为“exp4”；

3) 在TestStudent程序中充分地测试Student类。

【方案设计】

请将Student.java和TestStudent.java源程序的代码粘贴在下方。

回答：

public class Student {

public int age = 18; // 年龄

public String name; // 姓名

public String department; // 院系名称

Student() {

}

Student(int age, String name, String department) {

this.age = age;

this.name = name;

this.department = department;

}

Student(String name) {

this.name = name;

}

public void info() {

System.out.println("name:" + name + " age:" + age + " departmen:" + department);

}

}

public class TestStudent {

public static void main(String[] args) {

Student p1 = new Student(18, "Suwem","Computer science");

Student p2 = new Student(16, "Tom","Foreign language");

Student p3 = new Student();

Student p4=new Student("Susie");

Student[] Students = new Student[] { p1, p2, p3 ,p4};

for (Student each : Students) {

each.info();

}

}

}

【实验记录】

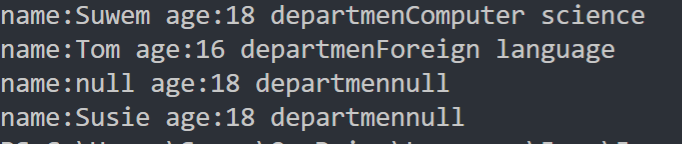
1. 请如实记录你在实验过程中出现的编译等错误(如发现错误无法解决，请举手)，对程序的错误进行修改并记录。如果没有发现错误则填“无”。

回答：

无

2. 请记录TestStudent程序的运行结果（即输出内容）。请把输出内容截图，只截输出窗口部分，并把截图粘贴在下方。

回答：



**实验任务二 继承和多态**

**第1题 继承和访问控制符**

【实验内容】

（面向对象设计）某项目涉及到的实体包括：

1) 乐器（instrument）：包含名称（name）和品牌（brand）属性，具有弹奏（play）方法。

2) 钢琴（Piano）：一种特殊的乐器，新增按键数（keyNum）属性，具有特有的弹奏（play）方法。

3) 小提琴(Violin)：一种特殊的乐器，新增琴弦数（chordNum）属性，具有特有的弹奏（play）方法。

4) 吉他（Guitar）：一种特殊的乐器，新增琴弦数（chordNum）属性，具有特有的弹奏（play）方法。

5) 手风琴（Accordion）：一种特殊的乐器，新增按键数（keyNum）属性，具有特有的弹奏（play）方法。

上述5个类的类图如下图所示，请在ExpInNetBeans或ExpInEclipse项目中实现这5个类，（方法的实现只需输出提示信息即可）；另外再新建一个Test1类，用于测试这5个类，要求充分测试多态。

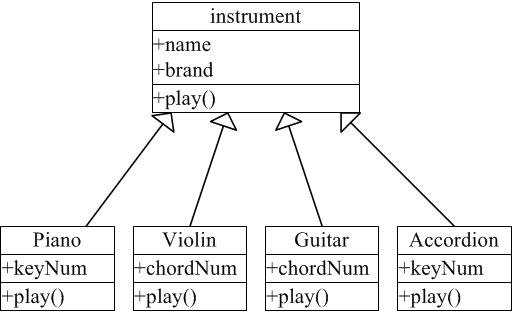


图1 类继承层次结构图

【方案设计】

请将所有源程序的代码粘贴在下方。

回答：

public class Instrument {

public String name;

public String band;

Instrument() {

}

Instrument(String name, String band) {

this.name = name;

this.band = band;

}

public void Play() {

System.out.println("can be played");

}

}

public class Accord extends Instrument {

public int keyNum;

Accord() {

}

Accord(String name, String band, int keyNum) {

super(name, band);

this.keyNum = keyNum;

}

public void Play() {

System.out.println("play Accord");

}

}

public class Guitar extends Instrument {

public int chordNum;

Guitar() {

}

Guitar(String name, String band, int chordNum) {

super(name, band);

this.chordNum = chordNum;

}

public void Play() {

System.out.println("play Guitar");

}

}

public class Piano extends Instrument {

public int keyNum;

Piano() {

}

Piano(String name, String band, int keyNum) {

super(name, band);

this.keyNum = keyNum;

}

public void Play() {

System.out.println("play piano");

}

}

public class TestInstrument {

public static void main(String[] args) {

Piano piano = new Piano();

Violin violin = new Violin();

Guitar guitar = new Guitar();

Accord accord = new Accord();

Instrument instrument = new Instrument();

instrument.Play();

instrument = piano;

instrument.Play();

instrument = violin;

instrument.Play();

instrument = guitar;

instrument.Play();

instrument = accord;

instrument.Play();

}

}

【实验记录】

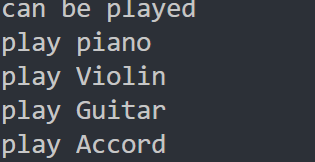
1. 请如实记录你在实验过程中出现的编译等错误(如发现错误无法解决，请举手)，对程序的错误进行修改并记录。如果没有发现错误则填“无”。

回答：

无

2. 请记录该程序的运行结果（即输出内容）。请把输出内容截图，只截输出窗口部分，并把截图粘贴在下方。

回答：



**第2题 人—教师—学生继承层次结构**

【实验内容】

（面向对象设计）某项目涉及到的实体包括：

1) 人（Person）：包含姓名（name）和身份证号（ID）属性，具有学习（learn）方法。

2) 教师（Teacher）：一种特殊的人，新增工号（Tno）属性，具有特有的学习（learn）方法。

3) 学生（Student）：一种特殊的人，新增学号（Sno）属性，具有特有的学习（learn）方法。

4) 中学生（MidStudent）：一种特殊的学生，具有特有的学习（learn）方法。

5) 大学生（UniStudent）：一种特殊的学生，新增专业（Major）属性，具有特有的学习（learn）方法。

上述5个类的类图如下图所示，请在ExpInNetBeans或ExpInEclipse项目中实现这5个类，（方法的实现只需输出提示信息即可）；另外再新建一个Test2类，用于测试这5个类，要求充分测试多态。

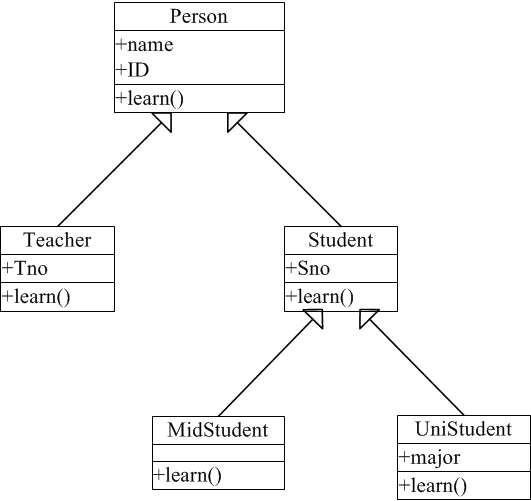


图1 类继承层次结构图

【方案设计】

请将所有源程序的代码粘贴在下方。

回答：

public class Person {

public String name;

public String ID;

public void learn() {

System.out.println("we can learn");

}

}

public class Student extends Person {

public String sno;

public void learn() {

System.out.println("student learn.");

}

}

public class Teacher extends Person {

public String tno;

public void learn() {

System.out.println("Teacher learn.");

}

}

public class TestPerson {

public static void main(String[] args) {

Person person = new Person();

Student student = new Student();

Teacher teacher = new Teacher();

person.learn();

person = student;

person.learn();

person = teacher;

person.learn();

}

}

【实验记录】

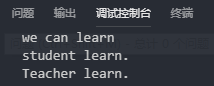
1. 请如实记录你在实验过程中出现的编译等错误(如发现错误无法解决，请举手)，对程序的错误进行修改并记录。如果没有发现错误则填“无”。

回答：

无

2. 请记录该程序的运行结果（即输出内容）。请把输出内容截图，只截输出窗口部分，并把截图粘贴在下方。

回答：



**第3题 人—用Java语言编写《出师表》**

【实验内容】

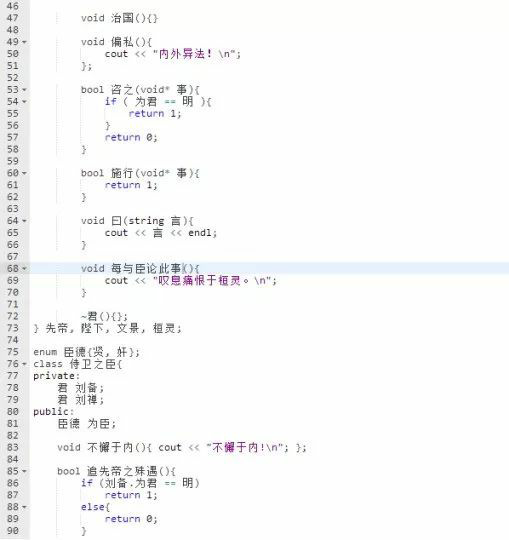
网上有人用C++语言实现了《出师表》(注：设计了多个类表示《出师表》中提及的人物，设计了属性和方法，类和类之间有继承关系，程序能输出《出师表》中的内容。)

请用Java语言改写下述程序。

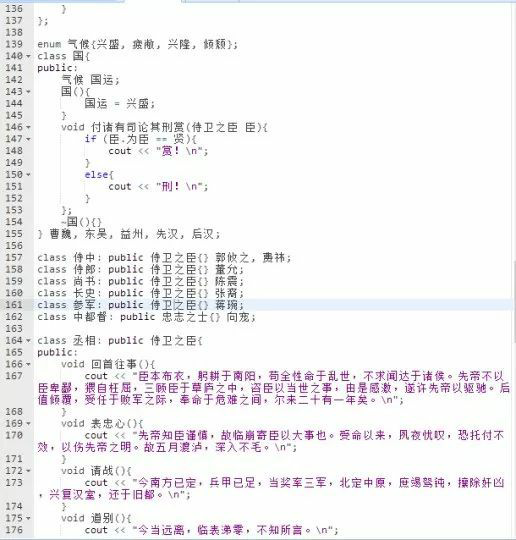
【注】Java语言和C++语言在实现面向对象思想上有些细微的差别，需特别注意。

http://www.sohu.com/a/214876788\_160850

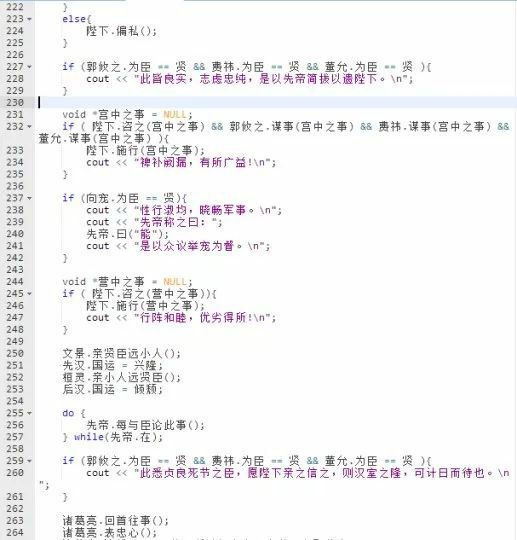




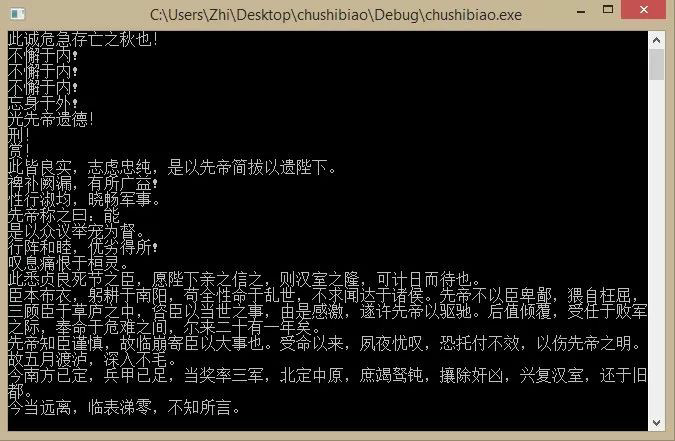








以下是上述程序的输出。



【方案设计】

请将所有源程序的代码粘贴在下方。

回答：

*/\**

*\* @Coding: utf-8 \n*

*\* @Despcription: \n*

*\* @Author: Suwen SP \n*

*\* @LastEditors: Suwen SP\n*

*\* @Date: 2019-03-25 18:32:57*

*\* @LastEditTime: 2019-03-25 18:33:33*

*\*/*

public class 君 {

public enum 王道 {

明, 昏

};

public boolean 在;

public 王道 为君;

public 君() {

在 = true;

为君 = 王道.明;

}

public boolean 创业(double *percentage*) {

if (percentage < 0.5) {

在 = false;

System.out.println("中道崩殂！");

return false;

} else {

System.out.println("兴复汉室，还于旧都！");

return true;

}

}

public void 开张圣听() {

System.out.println("光先帝遗德！");

}

public void 恢弘志士之气() {

}

public void 妄自菲薄() {

}

public void 引喻失义() {

System.out.println("塞忠谏之路！");

}

public void 亲贤臣远小人() {

为君 = 王道.明;

}

public void 亲小人远贤臣() {

为君 = 王道.昏;

}

public void 治国() {

}

public void 偏私() {

System.out.println("内外异法！");

};

public boolean 咨之() {

if (为君 == 王道.明)

return true;

else

return false;

}

public boolean 施行() {

return true;

}

public void 曰() {

System.out.println("言");

}

public void 每与臣论此事() {

System.out.println("叹息痛恨于柩灵。");

}

}

public class 国 {

侍卫之臣 s = new 侍卫之臣();

public enum 气候 {

兴盛, 疲敝, 兴隆, 倾颓

};

public 气候 国运;

public 国() {

国运 = 气候.兴盛;

}

public void 付诸有司论其刑赏(*侍卫之臣* *臣*) {

if (臣.为臣 == s.为臣.贤)

System.out.println("赏！");

else

System.out.println("刑！");

}

}

public class 侍卫之臣 {

君 j = new 君();

public enum 臣德 {

贤, 奸

}

private 君 刘备;

private 君 刘禅;

public 臣德 为臣;

public void 不懈于内() {

System.out.println("不懈于内！");

}

public boolean 追先帝之殊遇() {

return true;

}

public boolean 报之于陛下() {

return true;

}

public boolean 谋事() {

return true;

}

}

public class 忠烈之士 {

君 j = new 君();

public enum 臣德 {

贤, 奸

};

private 君 刘备;

private 君 刘禅;

public 臣德 为臣;

public void 忘身于外() {

System.out.println("忘身于外！");

};

public boolean 追先帝之殊遇() {

return true;

}

public boolean 报之于陛下() {

return true;

}

public boolean 谋事() {

return true;

}

}

public class 诸葛亮 extends 侍卫之臣 {

public void 回首往事() {

System.out.println(

"臣本布衣，躬耕于南阳，苟全性命于乱世，不求闻达于诸侯。先帝不以臣卑鄙，猥自枉屈，三顾臣于草庐之中，咨臣以当世之事，由是感激，遂许先帝以驱驰。\n后值倾覆，受任于败军之际，奉命于危难之间，尔来二十有一年矣。");

}

public void 表忠心() {

System.out.println("先帝知臣谨慎，故临崩寄臣以大事也。受命以来，夙夜忧叹，恐托付不效，以伤先帝之明。故五月渡泸，深入不毛。");

}

public void 请战() {

System.out.println("今南方已定，兵甲已足，当奖率三军，北定中原，庶竭驽钝，攘除奸凶，兴复汉室，还于旧都。");

}

public void 道别() {

System.out.println("今当远离，临表涕零，不知所言。");

}

}

public class 侍郎 extends 侍卫之臣 {

}

public class 侍中 extends 侍卫之臣 {

}

public class 都督 extends 忠烈之士 {

}

public class Main {

public static void main(String[] *args*) {

君 先帝 = new 君();

君 陛下 = new 君();

君 文景 = new 君();

君 柩灵 = new 君();

国 曹魏 = new 国();

国 东吴 = new 国();

国 益州 = new 国();

国 先汉 = new 国();

国 后汉 = new 国();

侍中 郭攸之 = new 侍中();

侍中 费祎 = new 侍中();

侍郎 董允 = new 侍郎();

都督 向宠 = new 都督();

先帝.创业(0.49);

益州.国运 = 益州.国运.疲敝;

System.out.println("此诚危急存亡之秋也！");

if (郭攸之.追先帝之殊遇() && 郭攸之.报之于陛下()) {

郭攸之.不懈于内();

}

if (费祎.追先帝之殊遇() && 费祎.报之于陛下()) {

费祎.不懈于内();

}

if (董允.追先帝之殊遇() && 董允.报之于陛下()) {

董允.不懈于内();

}

if (向宠.追先帝之殊遇() && 向宠.报之于陛下()) {

向宠.忘身于外();

}

if (陛下.为君 == 陛下.为君.明) {

陛下.开张圣听();

陛下.恢弘志士之气();

} else {

陛下.妄自菲薄();

陛下.引喻失义();

}

陛下.治国();

boolean 宫中 = false;

boolean 府中 = false;

boolean 陟罚 = false;

boolean 臧否 = false;

assert (宫中 == 府中);

assert (陟罚 == 臧否);

侍卫之臣 作奸犯科者 = new 侍卫之臣();

作奸犯科者.为臣 = 作奸犯科者.为臣.奸;

侍卫之臣 为忠善者 = new 侍卫之臣();

为忠善者.为臣 = 为忠善者.为臣.贤;

陛下.为君 = 陛下.为君.明;

if (陛下.为君 == 陛下.为君.明) {

益州.付诸有司论其刑赏(作奸犯科者);

益州.付诸有司论其刑赏(为忠善者);

} else

陛下.偏私();

郭攸之.为臣 = 郭攸之.为臣.贤;

费祎.为臣 = 费祎.为臣.贤;

董允.为臣 = 董允.为臣.贤;

if (郭攸之.为臣 == 郭攸之.为臣.贤 && 费祎.为臣 == 费祎.为臣.贤 && 董允.为臣 == 董允.为臣.贤)

System.out.println("此皆良实，志虑忠纯，是以先帝简拔以遗陛下。");

if (陛下.咨之() && 郭攸之.谋事() && 费祎.谋事() && 董允.谋事()) {

陛下.施行();

System.out.println("裨补缺漏，有所广益！");

}

向宠.为臣 = 向宠.为臣.贤;

if (向宠.为臣 == 向宠.为臣.贤) {

System.out.println("性行淑均，晓畅军事。");

System.out.println("先帝称之曰：");

System.out.println("能");

System.out.println("是以众意举宠为督。");

}

if (陛下.咨之()) {

陛下.施行();

System.out.println("行阵和睦，优劣得所！");

}

文景.亲贤臣远小人();

先汉.国运 = 先汉.国运.兴隆;

柩灵.亲小人远贤臣();

后汉.国运 = 后汉.国运.倾颓;

do {

先帝.每与臣论此事();

} while (先帝.在);

if (郭攸之.为臣 == 郭攸之.为臣.贤 && 费祎.为臣 == 费祎.为臣.贤 && 董允.为臣 == 董允.为臣.贤) {

System.out.println("此悉贞良死结之臣，愿陛下亲之信之，则汉室之隆，可计日而待也。");

}

诸葛亮 z = new 诸葛亮();

z.回首往事();

z.表忠心();

z.请战();

z.道别();

}

}

【实验记录】

1. 请如实记录你在实验过程中出现的编译等错误(如发现错误无法解决，请举手)，对程序的错误进行修改并记录。如果没有发现错误则填“无”。

回答：

无

2. 请记录该程序的运行结果（即输出内容）。请把输出内容截图，只截输出窗口部分，并把截图粘贴在下方。

回答：

