1,多态的体现?

2,多态的好处?

3,多态的弊端?

4,多态的前提?

5,多态举例?

6,多态在子父类中的成员上的体现的特点：

7.【代码题必做】

1). 定义书写功能接口(writeAction)  
    a)    抽象方法void  write()   
2). 定义笔类(Pen),实现writeAction接口,重写write()方法，方法内不需要编写内容；  
    a)    属性: Price 价格（Float),color颜色(String 类型);  
    b)    提供空参,满参构造方法和set/get方法;  
3）. 定义画笔类(Brush)  继承 Pen   
    a)    重写Pen中write()方法： 输出格式: “现在使用的是价格为XXX的YYY颜色的画笔”；  
    b)    提供空参,满参构造方法  
4). 定义铅笔类(Pencil)  继承 Pen   
    a)     重写Pen中write()方法: 输出格式: “现在使用的是价格为XXX的YYY颜色的铅笔”；  
   b)    提供空参,满参构造方法  
5). 定义画家类（Painter)  
    a)    属性: name姓名(String)  
    b)    对象方法 : 绘画方法(void draw(Pen p)),首先输出“画家XX正在绘画，”，然后调用p对象的write方法  
    c)    提供空参,满参构造方法;set/get方法  
6）. 测试类Test  
    a)    提供main方法  
    b)    在main方法中  
            i.    创建Brush类对象b1,价格赋值为5.8,颜色赋值为红色；  
            ii.    创建Pencil类对象p1,价格赋值为2.5,颜色赋值为黑色；  
            iii.    创建Painter对象,姓名赋值为王流秋；调用两次Painter的draw方法，分别传入b1和p1；进行测试；

8.【代码题必做】

根据要求完成以下功能：

①定义接口Study，要求如下：

定义抽象方法： void studyLine()

②定义类Line，包含空参构造，满参构造和以下成员变量

就业班emplymentClass（String 型）

学费   money (int 型)

生成所有成员变量set/get方法

③ 定义类Student，实现Study接口，包含空参构造，满参构造和以下成员变量

 姓名 name（String 型）

成绩 score  (double 型 )

所报班级 Line lin

生成所有成员变量set/get方法

定义成员方法：void exam(),实现以下功能：判断学员成绩yyy，如果大于60分，打印”您的成绩为yyy,考试通过”,否则打印”您的成绩为yyy,考试不通过”

实现抽象方法 void studyLine(),输出：输出”您报名的就业班为：zzz,应缴纳学费：sss元。”

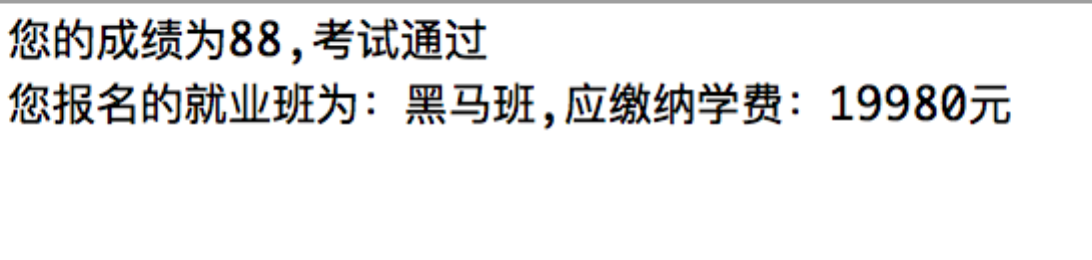
**PS：,yyy是**score**值,zzz是line类的emplymentClass值,sss是line类的**money**值**

④ 定义测试类Test，在main方法实现以下功能：

创建一个Line对像lintest,为其赋值:就业班为：黑马班，学费为：19980。

Student对象stu，使用set方法为其赋值：姓名为张三，成绩为88，所报班级：lintest。

调用exam(),studyLine()两个方法。实现效果如下：



9.【代码题必做】

按照要求完成以下内容：

1).创建People（人）的抽象类，包含：

        a）成员属性：姓名（name），String类型。

                国籍（country），String类型。

        b）有参无返回值的抽象方法：void speak(String str)。

2).创建Movement（运动）接口，包含：

        a）无参无返回值的抽象方法：void exercise()。

3).创建Man（男人）类，要求：

    a）.继承People（人）类，实现speak方法，在方法内打印“姓名为X的Y人在说着 Z”，其中X为男人的名字，Y为男人的国籍，Z为当前方法的参数

    b）.完成满参构造

    c）.实现Movement接口，实现exercise方法，在方法内打印：“X正在锻炼中！”其中，X为男人的名字。

    d）.创建成员方法：life()，在方法内依次调用speak(String str)方法和exercise()方法。其中speak方法的参数自行创建。

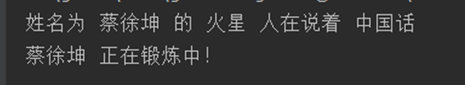
4).创建测试类，在测试类的main方法中完成：

    a).创建一个Man对象 Kk，属性赋值：姓名为“蔡徐坤”国籍为“火星”。

    b).调用这个对象的life()方法。

程序运行后的打印示例如图：

示例：



|  |
| --- |
| 10.【代码题必做】  按要求定义接口和类,完成以下功能：  a.定义电视类TV，要求包含：      1)成员变量:名称 name（ String 型）；      2)构造方法:空参、满参构造方法；      3)成员方法:void play(String channel ) ,方法内,打印输出 "xx电视正在播放xxx节目"（channel表示电视节目）；    b.定义接口RedRay，要求 ：      1)定义抽象方法： void controlTV(TV tv)；      2)定义默认方法： void connecting()，方法内,打印输出"外接设备,连接成功, 可以使用红外线"；    c.定义遥控器类RemoteControl,包含:      1)成员方法: void  controlTV(TV tv):方法内输出"遥控器控制电视",然后提示用户键盘录入一个字符串作为电视节目信息,调用tv的play方法,将节目信息作为参数传入；    d.定义手机类Phone,实现RedRay接口，要求：      1)重写controlTV方法：           \* 方法内输出"手机开启红外功能,控制电视"；           \* 提示用户键盘录入电视节目信息并接受；           \* 调用电视play方法,将节目信息作为参数传入；    e.定义测试类Test，在main方法实现以下功能：       \* 创建TV对象；       \* 创建RemoteControl对象,调用其controlTV的方法；       \* 创建Phone对象,调用其connecting方法和controlTV方法；  示例如图：  点击放大查看 |
| 11.【代码题必做】  按照要求完成以下内容：      创建Animal（动物）抽象类，包含：                  成员属性：姓名（name），String类型。                  有参无返回值的抽象方法：void speak (String str)。     创建Sports（运动）接口，包含：                无参无返回值的抽象方法：void swimming ()。     创建Dog（狗）类，要求：              继承Animal（动物）类，实现speak方法，在方法内打印“X说：Y”，其中X为狗的名字，Y为当前方法的参数。              完成满参构造。              实现Sports接口，实现swimming方法，在方法内打印：XXX狗刨中！其中，X为狗的名字。              创建成员方法：goPlay()，在方法内依次调用speak(String str)方法和swimming ()方法。其中speak方法的参数自行创建。      创建测试类，在测试类的main方法中完成：              通过满参构造创建一个姓名为“旺财”Dog类对象。              调用这个对象的goPlay()方法。  程序运行后的打印示例如图：  点击放大查看 |
| 12.【代码题必做】  ①定义接口Teach，要求如下：  **定义抽象方法  void** teaching();  ②定义类Teacher，实现接口Teach，包含空参构造，满参构造和以下成员变量  姓名 name（String 类型）  学科 course（String类型）   实现  抽象方法teaching()，并输出**"我是XX老师，我教YY学科”**   XX表示姓名，YY表示学科  ③定义Student类，包含空参构造，满参构造和以下成员变量  姓名 name（String类型）  分数 score（int 类型）  老师Teacher  **teacher**  定义方法getMyResult（），无参无返回值，并输出“我是学生XX，我的老师是YY，他教我ZZ学科”  XX是学生的姓名，YY是老师的姓名，ZZ是老师所教学科  判断学员分数是否大于等于60分，如果大于等于60分，则输出“继续加油哦”  如果小于60分，则输出**"成绩太低了，老师XX让我回去罚写100遍"，xx是老师的姓名**  ④在main方法中，定义教师类、学生类并赋值  调用teaching()、getMyResult（）方法。  示例：  点击放大查看 |