## 实验3.3 面向对象基础编程接口和多态

|  |
| --- |
| 实验目的：   1. 掌握接口的定义和实现接口的方法 2. 掌握接口回调、多态的基本概念 |
| 实验数据记录及分析（或程序及运行结果）  1、试改写实验3.2实验1中的Pet类，其中move与speak方法定义为抽象方法。定义一个Pet类的子类Cat类。在Cat类中实现move和Speak方法。其中move方法返回值为“run”，speak方法返回值为“meow meow”。 编写含有main方法的公共类PetTest，在其方法体中创建一个名为myCat的Cat的实例。并依照实验1中PetTest类main方法中的相应格式输出结果。  **abstract** **class** Pet {  **protected** String name;  **public** Pet(String n) {  name = n;  }  **public** String getName() {  **return** name;  }  **public** **abstract** String move();  **public** **abstract** String speak();  **public** String toString() {  **return** "My pet " + name;  }  }  **class** Cat **extends** Pet {  **public** Pet(String n) {  name = n;  }  **public** String move() {  **return** "run";  }  **public** String speak() {  **return** "meow meow";  }  }  **class** PetTest {  **public** **static** **void** main(String[] args) {  Cat myCat = **new** Cat("Mao");  System.***out***.println(myCat.toString() + "\n" + "Speak: " + myCat.speak() + "\n" + myCat.move() + " "  + myCat.getName() + "\n");  }  }  2、如果将上一次实验（即实验3.2）中实验1的Pet类定义为接口，即所有相应方法全部为抽象方法，则Cat类该如何改动？试改写相关代码，并验证你的结果。  **interface** Pet {  **public** **abstract** String getName();  **public** **abstract** String move();  **public** **abstract** String speak();  **public** **abstract** String toString();  }  **class** Cat **implements** Pet {  String name;  **public** Cat(String n) {  name = n;  }  **public** String move() {  **return** "run";  }  **public** String speak() {  **return** "meow meow";  }  **public** String getName() {  **return** name;  }  **public** String toString() {  **return** "My pet" + name;  }  }  **class** PetTest {  **public** **static** **void** main(String[] args) {  Cat myCat = **new** Cat("Mao");  System.***out***.println(myCat.toString() + "\n" + "Speak: " + myCat.speak() + "\n" + myCat.move() + " "  + myCat.getName() + "\n");  }  }  3、编一个程序，包含以下文件。  （1）Shape.java文件，在该文件中定义接口Shape，该接口在shape包中。  属性：PI。  方法：求面积的方法area()。  **package** shape;  **public** **interface** Shape {  **double** ***PI*** = Math.***PI***;  **double** area();  }  （2）Circle.java文件，在该文件中定义圆类Circle，该类在circle包中，实现Shape接口。  属性：圆半径radius。  方法：构造方法；实现接口中求面积方法area()；求周长方法perimeter()。  **package** circle;  **import** shape.Shape;  **public** **class** Circle **implements** Shape {  **protected** **double** radius; // 半径  **public** Circle(**double** r) {  radius = r;  }  **public** **double** area() { // 实现Shape接口中的方法（这是必须的）  **return** ***PI*** \* radius \* radius;  }  **public** **double** perimeter() { // 定义求圆周长的方法  **return** 2 \* ***PI*** \* radius;  }  }  （3）“Cylinder.java”文件，在该文件中定义圆柱体类Cylinder，该类口在cylinder包中，继承圆类。  属性：圆柱体高度height。  方法：构造方法；求表面积方法area()；求体积方法volume()。  **package** cylinder;  **import** circle.Circle;  **public** **class** Cylinder **extends** Circle {  **double** height;  **public** Cylinder(**double** r, **double** h) {  **super**(r);  height = h;  }  **public** **double** area() {  **return** 2 \* ***PI*** \* radius \* radius + 2 \* ***PI*** \* radius \* height;  }  **public** **double** volume() {  **return** ***PI*** \* radius \* radius \* height;  }  }  （4）Main.java文件，在该文件中定义主类Main，该类在默认包中，其中包含主方法main()，在主方法中创建两个圆类对象cir1和cir2，具体尺寸自己确定，并显示圆的面积和周长；再创建两个圆柱体类的对象cy1和cy2，具体尺寸自己确定，然后分别显示圆柱体cy1和cy2的底圆的面积和周长以及它们各自的体积和表面积。  **import** circle.Circle;  **import** cylinder.Cylinder;  **public** **class** Main {  **public** **static** **void** main(String[] args) {  Circle cir1 = **new** Circle(120.5);  Circle cir2 = **new** Circle(183.8);  System.***out***.println("cir1.area: " + cir1.area());  System.***out***.println("cir1.perimeter: " + cir1.perimeter());  System.***out***.println("cir2.area: " + cir2.area());  System.***out***.println("cir2.perimeter: " + cir2.perimeter());  Cylinder cy1 = **new** Cylinder(27.3, 32.7);  Cylinder cy2 = **new** Cylinder(133.5, 155.8);  System.***out***.println("cy1.area: " + cy1.area());  System.***out***.println("cy1.volume: " + cy1.volume());  System.***out***.println("cy2.area: " + cy2.area());  System.***out***.println("cy2.volume: " + cy2.volume());  }  }  (5)不断变更Circle类中属性和方法的修饰关键字public、缺省、protected、private，在类Circle和Cylinder中测试访问权限，总结public、缺省、protected、private的封装特性。 实验小结： |