

面向对象Java—实验报告

****

**实验三： 面向对象（二）**

**姓 名：**  於俊涛

**学 号： 202007020625**

**专业班级： 计算机203**

**系 别： 计算机系**

**学 院： 电子信息与人工智能学院**

**实验三 面向对象（二）**

1. **实验目的**

在集成开发环境下，实现类的继承及多态，实现抽象类、接口的定义，根据实际问题，运用抽象方法、接口及抽象类进行程序开发。

1. **实验目标**

(1)能够运用Java语言来实现类的继承及多态性、熟练使用包来扩展程序功能并能建立自己的包；

(2)能够灵活运用Java语言的抽象方法、接口、抽象类来编写应用程序解决实际问题。

1. **实验内容**

**3.1 实验环境**

Eclipse IDE for Java Developers

**3.2 具体实验内容**

1. 设计一个抽象类图形类，在该类中包含有至少三个抽象方法，分别用于求周长、求面积，以及对图形进行描述（如，“这是\*\*形状，周长是\*\*，面积是\*\*”），分别定义圆形类、长方形类、等边三角形类来继承图形类，实现上述三个方法，并创建实例验证。
2. 定义一个抽象接口，该接口用于求和与统计数量。定义两个奇数类与偶数类实现上述接口，分别可以求100以内的奇数的和，统计100以内的奇数个数，以及求100以内的偶数的和，统计100以内的偶数个数。
3. 定义一个抽象类动物类，同时定义抽象接口：进食接口、睡觉接口、思考接口、说话接口、行走接口、游泳接口、飞翔的接口。分别定义猫类、人类、鸟类、鱼类继承动物类，同时，根据实际情况，选择性实现上述接口，并创建实例验证。
4. 编写一个类，实现银行账户，并完成程序的测试：
5. 设置的属性有“账号”、 “储户姓名”、“地址”以及“存款余额”；
6. 设置的方法有“存款”、“取款”、“查询”、“计算利息”、“累加利息”；
7. 给类增加一个静态属性“最小余额”和一个用来修改这个最小余额属性的方法；
8. 给类增加一个静态属性“活期利息”和封装这个数据的对应方法；
9. **实验过程分析**
   1. **实验步骤**
10. 设计一个抽象类图形类，在该类中包含有至少三个抽象方法，分别用于求周长、求面积，以及对图形进行描述（如，“这是\*\*形状，周长是\*\*，面积是\*\*”），分别定义圆形类、长方形类、等边三角形类来继承图形类，实现上述三个方法，并创建实例验证。

实验代码：

abstract class Shape {

public abstract double calculatePerimeter ();

public abstract void describe ();

public abstract double calculateSquare ();}

class Round extends Shape {

private final double R;

public Round(double r) {R = r;}

@Override

public double calculatePerimeter() {return Math.PI \* this.R \* 2;}

@Override

public void describe() {System.out.println("这是一个半径为" + this.R + "的圆, 面积为" + calculateSquare() + ", 周长为" + calculatePerimeter());}

@Override

public double calculateSquare() {return Math.PI \* this.R \* this.R;}}

class Rectangle extends Shape {

private final double a;

private final double b;

public Rectangle(double a, double b) {

this.a = a;

this.b = b;}

@Override

public double calculatePerimeter() {return this.a \* 2 + this.b \* 2;}

@Override

public void describe() {System.out.println("这是一个长为" + this.a + ",宽为" + this.b + "的矩形, 面积为" + calculateSquare() + ", 周长为" + calculatePerimeter());}

@Override

public double calculateSquare() {return this.a \* this.b;}}

class EquilateralTriangle extends Shape {

private final double a;

public EquilateralTriangle(double a) {this.a = a;}

@Override

public double calculatePerimeter() {return this.a \* 3;}

@Override

public void describe() {System.out.println("这是一个变长为" + this.a + "的等边三角形, 面积为" + calculateSquare() + ", 周长为" + calculatePerimeter());}

@Override

public double calculateSquare() {return this.a \* this.a \* 2 \* Math.sqrt(3);}}

public class first {

public static void main(String[] args) {

Shape shape1 = new Round(2);

Shape shape2 = new Rectangle(3, 4);

Shape shape3 = new EquilateralTriangle(3);

shape1.describe();

shape2.describe();

shape3.describe();}}

结果输出:

这是一个半径为2.0的圆, 面积为12.566370614359172, 周长为12.566370614359172

这是一个长为3.0,宽为4.0的矩形, 面积为12.0, 周长为14.0

这是一个变长为3.0的等边三角形, 面积为31.17691453623979, 周长为9.0

1. 定义一个抽象接口，该接口用于求和与统计数量。定义两个奇数类与偶数类实现上述接口，分别可以求100以内的奇数的和，统计100以内的奇数个数，以及求100以内的偶数的和，统计100以内的偶数个数。

实验代码：

interface Lecture {

public abstract void talked ();

public abstract void op\_homework ();}

class Student implements Lecture {

private final String name;

@Override

public void talked() {System.out.println(this.name + "回答问题");}

public Student(String name) {this.name = name;}

@Override

public void op\_homework() {System.out.println(this.name + "做作业"); }}

class Teacher implements Lecture {

private final String name;

@Override

public void talked() {System.out.println(this.name + "提出一个问题");}

public Teacher(String name) {this.name = name;}

@Override

public void op\_homework() {System.out.println(this.name + "布置了作业");}}

public class two {

public static void main(String[] args) {

Teacher teacher = new Teacher("Smith");

Student student = new Student("john");

teacher.talked();

student.talked();

teacher.op\_homework();

student.op\_homework();}}

结果输出：

Smith提出一个问题

john回答问题

Smith布置了作业

john做作业

1. 定义一个抽象类动物类，同时定义抽象接口：进食接口、睡觉接口、思考接口、说话接口、行走接口、游泳接口、飞翔的接口。分别定义猫类、人类、鸟类、鱼类继承动物类，同时，根据实际情况，选择性实现上述接口，并创建实例验证。

实验代码：

interface Countable {

public abstract int sum ();

public abstract int count ();}

// 偶数

class Even implements Countable {

@Override

public int sum() {

int sum = 0;

for (int i = 1; i <= 100; i++)

if (i % 2 == 0) sum += i;

return sum;}

@Override

public int count() {

int count = 0;

for (int i = 1; i <= 100; i++)

if (i % 2 == 0) count++;

return count;}}

// 奇数

class Odd implements Countable {

@Override

public int sum() {

int sum = 0;

for (int i = 1; i <= 100; i++)

if (i % 2 != 0)

sum += i;

return sum;

@Override

public int count() {

int count = 0;

for (int i = 1; i <= 100; i++)

if (i % 2 != 0)

count++;

return count;}}

public class three {

public static void main(String[] args) {

Even even = new Even();

System.out.println("100以内的偶数个数：" + even.count() + "个");

System.out.println("100以内的偶数和:" + even.sum());

Odd odd = new Odd();

System.out.println("100以内的奇数个数:" + odd.count() + "个");

System.out.println("100以内的奇数和:" + odd.sum()); }}

结果输出：

100以内的偶数个数：50个

100以内的偶数和:2550

100以内的奇数个数:50个

100以内的奇数和:2500

4.编写一个类，实现银行账户，并完成程序的测试：

1. 设置的属性有“账号”、 “储户姓名”、“地址”以及“存款余额”；
2. 设置的方法有“存款”、“取款”、“查询”、“计算利息”、“累加利息”；
3. 给类增加一个静态属性“最小余额”和一个用来修改这个最小余额属性的方法；
4. 给类增加一个静态属性“活期利息”和封装这个数据的对应方法；

实验代码：

abstract class Animal {public abstract void breathe ();}

class Cat extends Animal implements AbleToEat, AbleToSleep, Thinkable, Speakable, Walkable {

@Override

public void breathe() {System.out.println("Cat is breathe"); }

@Override

public void eat() {System.out.println("Cat is eating");}

@Override

public void sleep() {System.out.println("Cat is sleeping");}

@Override

public void think() {System.out.println("Cat doesn't want to think!");}

@Override

public void speak() {System.out.println("Cat can't speak");}

@Override

public void walk() {System.out.println("Walking the cat please!");}}

class Fish extends Animal implements AbleToEat, AbleToSleep, Thinkable, Speakable, Walkable {

@Override

public void breathe() {System.out.println("Fish is breathe"); }

@Override

public void eat() {System.out.println("Fish is eating,fish die.");}

@Override

public void sleep() {System.out.println("Fish slept ,Cat is eating fish");}

@Override

public void think() {System.out.println("Fish doesn't want to think!");}

@Override

public void speak() {System.out.println("Fish speak blo blo!");}

@Override

public void walk() {System.out.println("Fish can't walk!");}}

class Humen extends Animal implements AbleToEat, AbleToSleep, Thinkable, Speakable, Walkable {

@Override

public void breathe() {System.out.println("Man is breathe");}

@Override

public void eat() {System.out.println("Man is eating");}

@Override

public void sleep() { System.out.println("Man is sleeping,the fish is ate by Cat");}

@Override

public void think() {System.out.println("The man stop thinking");}

@Override

public void speak() {System.out.println("The man :\" shut up!\"");}

@Override

public void walk() {System.out.println("The man is walking the cat!");}}

interface AbleToEat {public abstract void eat ();}

interface AbleToSleep {public abstract void sleep ();}

interface Thinkable {public abstract void think ();}

interface Speakable {public abstract void speak ();}

interface Walkable {public abstract void walk ();}

interface AbleToSwim {public abstract void swim ();}

interface Flyable {public abstract void fly ();}

public class four {

public static void main(String[] args) {

Cat cat = new Cat();

Fish fish =new Fish();

Humen person= new Humen();

cat.breathe();

fish.breathe();

person.breathe();

person.eat();

fish.eat();

cat.eat();

cat.sleep();

person.sleep();

fish.sleep();

person.think();

fish.think();

cat.think();

cat.walk();

fish.walk();

person.walk();}}

结果输出：

Cat is breathe

Fish is breathe

Man is breathe

Man is eating

Fish is eating,fish die.

Cat is eating

Cat is sleeping

Man is sleeping,the fish is ate by Cat

Fish slept ,Cat is eating fish

The man stop thinking

Fish doesn't want to think!

Cat doesn't want to think!

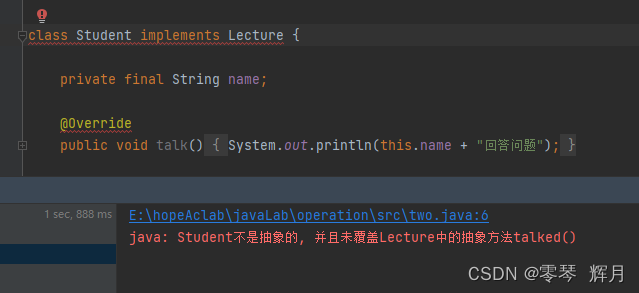
Walking the cat please!

Fish can't walk!

The man is walking the cat!

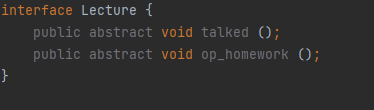
* 1. **错误分析**

在第二题的设计运行中遇到如下问题：



经检查是出错位置的函数名talk与定义的抽象方法名talked不一致，未覆盖到lecture中的talked方法.

**解决方案：**  
将函数名更改一致；

1. **实验****总结**

通过本次实验我能够运用Java语言来实现类的继承及多态性，并且能够灵活运用Java语言的抽象方法、接口、抽象类来编写应用程序解决实际问题。在实验中遇到了许多问题，但是在同学的帮助下得到了很好的解决。