电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 2017221302006

姓 名 周玉川

（实验） 课程名称 面向对象程序设计（Java）

理论教师 钱伟中

实验教师 冯月

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：周玉川 学号：2017221302006 指导教师：冯月**

**实验地点：信软楼西304 实验时间：2018.11.19**

1. **实验名称：类的继承、接口及访问控制**
2. **实验学时：2**
3. **实验目的：**

熟悉Java的类的继承基本操作，接口及访问控制，interface与abstract的区别和相同点，理解面向对象程序设计的基本思想。

1. **实验原理：**

学习书本第5章第六章中子类对超类的继承，abstract，接口interface，implement关键字，公共型public，缺省型default，保护型protect，私有型private。

1. **实验内容：**

1、完成第五章习题9、10编程。

2、完成第六章习题7、8编程。

3、有几何形状边数为n及可计算面积area的Shape类，其子类Triangle类及Rectangle类实现几何形状三角形和矩形面积area计算，利用前三个形状类实现柱体Pillar类的体积计算，并在PillarTest类中实现对某一柱体的体积计算。

4、创建学生成绩中所涉及的类：Student类、Teacher类、Course类，并由Grade类将Student类、Teacher类和Course类关联起来，由GradeTest类对以上四个类进行测试。

1. **实验器材（设备、元器件）：**

PC机

1. **实验步骤：**

1、创建工程

2、编辑程序

3、编译程序

4、调试程序

5、运行程序，分析结果

本实验4个小题目都分别按上述步骤进行。

1. **实验结果与分析（含重要数据结果分析或核心代码流程分析）**

1.第五章习题9、10编程

|  |
| --- |
| 5.9代码  package 第五章;  public class p5\_9 {  public static void main(String[] args) {  //实例化Person类对象person  Person person = new Person("周玉川",'b',19);  //查看date  System.out.println("the person's date: "+person.getDate());  //实例化Student类对象student  Student student = new Student();  //信息setDate提示  System.out.println("<----Set student's date---->");  student.setDate("zhouyuchuan",'b',19, 221302006, 221302);  //输出学生信息  System.out.println("the student's date: "+student.getDate());    }  }  //person类  class Person{  //三个成员变量  String name;  char sex;  int age;  //带参数构造函数  Person(String name, char sex, int age){  this.name = name;  this.sex = sex;  this.age = age;  }  //不带参数构造函数  Person(){  this("鑫鑫",'g',19);  }  //成员方法，设置信息  void seDate(String name, char sex, int age) {  this.name = name;  this.sex = sex;  this.age = age;  }  //成员方法，得到信息  String getDate() {  return ("Name:"+name+"\tSex:"+sex+"\tAge"+age);  }  }  //Person的子类Student类  class Student extends Person {  //5个成员变量，三个在父类中  int sID;  int classNo;  //不带参数构造函数  Student(){  super();  sID = 0000;  classNo = 00;  }  //带参数构造函数  Student(String name, char sex, int age,int sID, int classNo) {  super(name,sex,age);  this.sID = sID;  this.classNo = classNo;  }  //成员方法，setDate设置信息  void setDate(String name, char sex, int age, int sID, int classNo) {  super.seDate(name, sex, age);  this.sID = sID;  this.classNo = classNo;  }  //成员方法，获得信息getDate  String getDate() {  return (super.getDate()+"\tsID"+sID+"\tclassNo:"+classNo);  }  }  //结束End |
| 运行结果  the person's date: Name:周玉川 Sex:b Age19  <----Set student's date---->  the student's date: Name:zhouyuchuan Sex:b Age19 sID221302006 classNo:221302    结果正确，实验成功。 |
| 5.10代码  package 第五章;  //抽象父类Person1类，可理解为基本类  abstract class Person1{  //三个成员变量  String name;  char sex;  int age;  Person1(){  name = "鑫鑫";  sex ='g';  age = 19;  }  //两个抽象函数  abstract void setData(String name, char sex, int age);  abstract String getData();  }  //子类Student1，可理解为扩展类  class Student1 extends Person1{  //成员变量，学号，专业  int sID;  String speciality;  //不带参构造函数  Student1(){  super();  sID=1302006;  speciality = "软件工程";  }  //带参数构造函数  Student1(String name, char sex, int age, int sID, String speciality){  this.name =name;  this.sex = sex;  this.age = age;  this.sID = sID;  this.speciality = speciality;  }  //实现父类抽象setDate函数  void setData(String name, char sex, int age) {  this.name =name;  this.sex = sex;  this.age = age;  }  //根据学生类成员变量扩展setData函数  void setData(String name, char sex, int age, int sID, String speciality) {  setData(name, sex, age);  this.sID = sID;  this.speciality = speciality;  }  //返回学生信息  String getData() {  // TODO Auto-generated method stub  return ("name: "+name+"\tsex: "+sex+"\tage: "+age+"\tsID: "+sID+"\tspeciality: "+speciality);  }    }  //子类Teacher,Person1的扩展类  class Teacher extends Person1{  //成员变量老师编号，部门  int tID;  String department;  //不带参构造函数  Teacher(){  super();  tID=110;  department = "教务处";  }  //带参数构造函数  Teacher(String name, char sex, int age, int sID, String department){  this.name =name;  this.sex = sex;  this.age = age;  this.tID = sID;  this.department = department;  }  //实现父类setData函数  void setData(String name, char sex, int age) {  // TODO Auto-generated method stub  this.name = name;  this.sex = sex;  this.age =age;  }  //扩展setDate函数  void seData(String name, char sex, int age, int sID, String department) {  setData(name, sex, age);  this.tID = sID;  this.department = department;  }  //返回老师信息  String getData() {  // TODO Auto-generated method stub  return ("name: "+name+"\tsex: "+sex+"\tage: "+age+"\tsID: "+tID+"\tdeaprtment: "+department);  }  }  public class p5\_10 {  public static void main(String[] agrs) {  //进行测试实验  Student1 student = new Student1("玉川", 'b', 19,1302028,"软件工程");  //Person1 person = new Teacher() ;//不能用抽象类实例化 or 抽象类不能实例化,which one,也只能用子类实例  Teacher teacher = new Teacher() ;  System.out.println("Student: "+student.getData()+"\n");  //System.out.println("Person or Teacher: "+person.getData()+"\n");  System.out.println("Teacher: "+teacher.getData()+"\nEND");  }  } |
| 运行结果  Student: name: 玉川 sex: b age: 19 sID: 1302028 speciality: 软件工程  Teacher: name: 鑫鑫 sex: g age: 19 sID: 110 deaprtment: 教务处  END  结果正确，实验成功。 |

2.六章习题7、8编程。

|  |
| --- |
| 6.7代码  package 第六章;  // 接口Print  interface Print{  void print();  }  // 实现类1doPrint1  class doPrint1 implements Print{  public void print() {  // TODO Auto-generated method stub  int i,j;  for ( i =1 ; i< 9 ; i++)  for (j=1; j<i+1; j++)  System.out.print(j+"\*"+i+"="+i\*j+(i==j?'\n':' '));  }  }  // 实现类2doPrint2  class doPrint2 implements Print{  // 实现打印函数  public void print() {  // TODO Auto-generated method stub  System.out.println("I love Java!");  }  }  public class p6\_7 {  public static void main(String[] args) {  // 进行测试  //Print t = new Print() ; //接口不能被自己实例化  //Print t = new doPrint1(); //可以用实现类实例化  doPrint1 t1 = new doPrint1();  doPrint2 t2 = new doPrint2();  System.out.println("Print1:");  t1.print();  System.out.println("\nPrint2");  t2.print(); |
| 运行结果  Print1:  1\*1=1  1\*2=2 2\*2=4  1\*3=3 2\*3=6 3\*3=9  1\*4=4 2\*4=8 3\*4=12 4\*4=16  1\*5=5 2\*5=10 3\*5=15 4\*5=20 5\*5=25  1\*6=6 2\*6=12 3\*6=18 4\*6=24 5\*6=30 6\*6=36  1\*7=7 2\*7=14 3\*7=21 4\*7=28 5\*7=35 6\*7=42 7\*7=49  1\*8=8 2\*8=16 3\*8=24 4\*8=32 5\*8=40 6\*8=48 7\*8=56 8\*8=64  Print2  I love Java!  结果正确，实验成功。 |
| 6.8代码  package 第六章;  //接口Person类  interface Person{  //两个方法  void setData(String name, char sex, String birthday);  String getData();  }  //实现函数Student  class Student implements Person{  String name;  char sex;  String birthday;  int sID ;  String speciality;  public void setData(String name, char sex, String birthday) {  // TODO Auto-generated method stub  this.name = name;  this.sex= sex;  this.birthday= birthday;  }  void setData(String name, char sex, String birthday, int sID, String speciality) {  setData(name, sex, birthday);  this.sID = sID;  this.speciality = speciality;  }  public String getData() {  // TODO Auto-generated method stub  return ("name:"+name+"\tsex: "+sex+"\tbirthday: "+birthday+"\tsID: "+sID+"\tspeciality: "+speciality);  }  }  //测试公共类  public class p6\_8 {  public static void main(String[] args) {  //实例化student  Student student = new Student();  student.setData("周玉川", 'b', "20000429", 221302006, "软件工程");  System.out.println("Student's date: "+student.getData());  }  } |
| 运行结果  结果正确，实验成功。 |

3.第三题

|  |
| --- |
| 代码  package 额外;  import java.util.Scanner;  //超类shape类  abstract class Shape {  int n;  abstract double getArea();  }  //子类三角形类  class Triangle extends Shape {  double x, y, z;  // 构造函数  Triangle(double x, double y, double z) {  // 检查合法性  if ((x + y > z) && (x - y < z) && (y - x < z)) {  this.x = x;  this.y = y;  this.z = z;  } else {  System.out.println("Illegal input,usage:can't form a triangle");  System.exit(-1);  }  }  // 不带参数构造函数  Triangle() {  this(3, 4, 5);  }  // 设置数据  void setData(double x, double y, double z) {  if ((x + y > z) && (x - y < z) && (y - x < z)) {  this.x = x;  this.y = y;  this.z = z;  } else {  System.out.println("Illegal input,usage:can't form a triangle");  System.exit(-1);  }  }  // 返回面积  double getArea() {  // TODO Auto-generated method stub  double p = (x + y + z) / 2;  return (Math.sqrt((p - x) \* (p - y) \* (p - z)));  }  }  //子类长方形类  class Rectangle extends Shape {  double height, width;// 正方形的长和宽  // 不带参数构造器  public Rectangle() {  // TODO Auto-generated constructor stub  this(3, 4);  }  // 带参数  public Rectangle(double height, double width) {  this.width = width;  this.height = height;  }  // 设置数据  void setData(double height, double width) {  this.height = height;  this.width = width;  }  // 返回长方形面积  double getArea() {  // TODO Auto-generated method stub  return (height \* width);  }  }  //柱体类  class Pillar {  // 柱体底面边，以及高  int n;  double heigh;  // 构造器初始化成员变量  Pillar(int n, double h) {  this.n = n;  this.heigh = h;  }  // 返回柱体体积  public double getVolume() {  Scanner xx = new Scanner(System.in);  Shape shape = null;// 定义一个超类，用其子类实例化  if (n == 2) {  double w, d;// 长方形长和宽  w = xx.nextDouble();// 读入数据  d = xx.nextDouble();  shape = new Rectangle(w, d);// shape实例化为长方形类  } else if (n == 3) {  double a, b, c;// 三角形三个边  a = xx.nextDouble();  b = xx.nextDouble();  c = xx.nextDouble();  xx.close();  shape = new Triangle(a, b, c);  } else {  System.out.println("Error: 边数为3或者2。");  xx.close();  System.exit(-1);  }  // 返回体积，面积乘高  return (heigh \* (shape.getArea()));  }  }  //测试类  public class PillarTest {  public static void main(String[] args) {  Scanner x = new Scanner(System.in);  // BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));  // 提示信息  System.out.println("输入柱体底面的边数以及高，最后输入低面边的长度：");  int n = x.nextInt();// 读入数据，边数和高  double h = x.nextDouble();  // x.close();不能关掉  Pillar pillar = new Pillar(n, h);  // 输出柱体高度  System.out.println("\n柱体的体积为:" + pillar.getVolume());  x.close();  }  } |
| 运行结果  输入柱体底面的边数以及高，最后输入低面边的长度：  3 5  7 8 9  柱体的体积为:38.72983346207417    结果正确，实验成功。 |

4.第四题

|  |
| --- |
| 代码  package challenge;  /\*  \* 创建学生成绩中所涉及的类：Student类、Teacher类、Course类，  \* 并由Grade类将Student类、Teacher类和Course类关联起来，  \* 由GradeTest类对以上四个类进行测试。  \*/  //学生类  class Student {  String name;  char sex;  int age;  String speciality;  int sID;  // 构造器  Student(String name, char sex, int age, int sID, String speciality) {  this.name = name;  this.sex = sex;  this.age = age;  this.sID = sID;  this.speciality = speciality;  }  }  //老师类  class Teacher {  String name;  char sex;  int age;  int tID;  String department;  // Teacher类的构造器  Teacher(String name, char sex, int age, int sID, String department) {  this.name = name;  this.sex = sex;  this.age = age;  this.tID = sID;  this.department = department;  }  }  //课程类  class Course {  String name;  int cID;  // 构造器  Course(String name, int cID) {  this.name = name;  this.cID = cID;  }  }  //成绩类  class Grade {  double grade;  Student s;  Teacher t;  Course c;  // 构造器,把学生老师课程类关联在一起  Grade(Student s, Teacher t, Course c, int grade) {  this.grade = grade;  this.c = c;  this.t = t;  this.s = s;  }  }  //测试公共类  public class GradeTest {  public static void main(String[] args) {  // TODO Auto-generated method stub  System.out.println("Initialization.....");  Student student = new Student("玉川", 'b', 19, 1302028, "软件工程");  //实例化老师学生课程成绩类  Teacher teacher = new Teacher("钱伟中", 'b', 32, 110, "教务处");  Course course = new Course("java语言程序设计", 521);  Grade grade = new Grade(student, teacher, course, 98);  //输出结果  System.out.println("关联后信息为：\n" + "Student:" + grade.s + "\nTeacher:" + grade.t + "\nCouse: " + grade.c  + "\nGrade: " + grade.grade);  }  } |
| 运行结果  Initialization.....  关联后信息为：  Student:challenge.Student@7852e922  Teacher:challenge.Teacher@4e25154f  Couse: challenge.Course@70dea4e  Grade: 98.0    结果正确，试验成功。 |

1. **总结及心得体会：**

1.一定要认真听课，并且课下要练习写代码。

2.勤能补拙，学习新的语言每天都会碰到新东西，保持兴趣。

3.多看书，不会百度。

4.继承是java多态性的表现之一。

5.interface类比于c语言中函数的声明，但是更强大。

6.访问控制让java变得更安全。等级更加严密。

7.不能用抽象类实例化 or 抽象类不能实例化,which one,也只能用子类实例

8.接口不能被自己实例化, 可以用被她的实现类实例化。

1. **对本实验过程及方法、手段的改进建议：**

1.还是有些简单，希望有难一些的选做题。

2.希望多与学生交流，能把平时不会的而且百度不到的问题反馈给老师

3.希望老师能在课下布置一些和生活相关的任务，多培养同学一些兴趣和情趣。

**报告评分：**

**指导教师签字：**