**继承作业**

1. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **以下关于继承条件下构造方法执行过程的代码的执行结果是（ A ）。（选择一项）** | |
|  | **class** Person {  **public** Person() {  System.*out*.println("execute Person()");  }  }  **class** Student **extends** Person {  **public** Student() {  System.*out*.println("execute Student() ");  }  }  **class** PostGraduate **extends** Student {  **public** PostGraduate() {  System.*out*.println("execute PostGraduate()");  }  }  **public** **class** TestInherit {  **public** **static** **void** main(String[] args) {  **new** PostGraduate();  }  } | |
|  |  |  |
|  | **A.** | execute Person()  execute Student()  execute PostGraduate() |
|  | **B.** | execute PostGraduate() |
|  | **C.** | execute PostGraduate()  execute Student()  execute Person() |
|  | **D.** | 没有结果输出 |

|  |  |  |
| --- | --- | --- |
| **2.** | **以下关于this和super关键字的说法错误的是（ BD ）。（选择二项）** | |
|  |  |  |
|  | **A.** | this关键字指向当前对象自身，super关键字指向当前对象的直接父类 |
|  | **B.** | 在main方法中可以存在this或super关键字，但不能同时存在。 |
|  | **C.** | this和super关键字都可以访问成员属性，成员方法和构造方法 |
|  | **D.** | 在一个类的构造方法中可以同时使用this和super来调用其他构造方法 |

|  |  |  |
| --- | --- | --- |
| **3.** | **给出下面的代码段，在代码说明//assignment x=a,y=b处写入如下（ D ）个代码是正确的。（选择一项）** | |
|  | **public** **class** Base {  **int** w, x, y, z;  **public** Base(**int** a, **int** b) {  x = a;  y = b;  }  **public** Base(**int** a, **int** b, **int** c, **int** d) {  // assignment x=a,y=b  w = d;  z = c;  }  } | |
|  |  |  |
|  | **A** | Base(a,b); |
|  | **B.** | x=a,y=b; |
|  | **C.** | this(a),this(b); |
|  | **D.** | this(a,b) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **4.** | **以下Java程序编译运行后的输出结果是（ A ）。（选择一项）** | | | |
|  | **class** HelloA {  **public** HelloA( ) {  System.*out*.println("HelloA");  }  { System.*out*.println("I'm A class");  }  **static** {  System.*out*.println("static A");  }  }  **public** **class** HelloB **extends** HelloA {  **public** HelloB( ) {  System.*out*.println("HelloB");  }  { System.*out*.println("I'm B class");  }  **static** {  System.*out*.println("static B");  }  **public** **static** **void** main(String[] args) {  **new** HelloB( );  }  } | | | |
|  |  |  | | |
|  | **A** | static A  static B  I'm A class  HelloA  I'm B class  HelloB | C. | static A  I'm A class  HelloA  static B  I'm B class  HelloB |
|  | **B.** | static A  static B  I'm A class  I'm B class  HelloA  HelloB | D | static A  static B  HelloA  HelloB  I'm A class  I'm B class |

|  |  |  |
| --- | --- | --- |
| **5.** | **下列选项中关于Java中super关键字的说法正确的是（ AD ）。（选择二项）** | |
|  |  |  |
|  | **A** | super关键字是在子类对象内部指代其父类对象的引用 |
|  | **B.** | super关键字不仅可以指代子类的直接父类，还可以指代父类的父类 |
|  | **C.** | 子类通过super关键字只能调用父类的方法，而不能调用父类的属性 |
|  | **D.** | 子类通过super关键字可以调用父类的构造方法 |

1. **判断题**
   1. 如果定义一个类时没有使用extends，则它的父类是java.lang.Object。（ **〇** ）
   2. 对子类实例化对象需要调用超类的构造函数进行初始化工作，完成对子类中的超类实例变量的初始化，则子类可以继承超类的构造函数。（ **×** ）
   3. 在Java中任何一个子类的构造方法都必须调用其父类的构造方法（包括隐式调用），并且调用父类的构造方法的语句必须是子类构造方法的第一条语句。（ **×** ）
   4. Java中的类不允许多重继承，一个类只能有有一个直接父类。（ **〇** ）
   5. Object类有一个public方法是toString( )，一个对象通过调用该方法可以获得该对象的字符串表示。（ **〇** ）
   6. 父类Person中定义了一个private void show()的方法，那么子类要重写这个方法时，方法的访问修饰符可以是默认的，protected或public。（ **〇** ）
   7. 运算符“==”用于比较引用时，如果两个引用指向内存同一个对象，则返回true。（ **〇** ）
2. **简答题**
   1. 方法重载和方法重写（覆盖）的区别。
   2. java.lang.Object类的六个常用方法的声明并说明其作用。
   3. 继承条件下子类构造方法的执行过程
   4. super关键字的作用和使用
   5. ==和equals()的联系和区别
3. **编码题**
   1. 编写应用程序，创建类的对象，分别设置圆的半径、圆柱体的高，计算并分别显示圆半径、圆面积、圆周长，圆柱体的体积。（7分）

**提示:**（1）编写一个圆类Circle，该类拥有：

* + - 1. 一个成员变量，radius（私有，浮点型）;//存放圆的半径；
      2. 两个构造方法

Circle（） //将半径设为0

Circle（double r ） //创建Circle对象时将半径初始化为r

* + - 1. 三个成员方法

double getArea（） //获取圆的面积

double getPerimeter（） //获取圆的周长

void show（） //将圆的关径、周长、面积输出到屏幕

（2） 编写一个圆柱体类Cylinder，它继承于上面的Circle类。还拥有：

* + 1. 一个成员变量，double hight （私有，浮点型）; //圆柱体的高；
    2. 构造方法

//创建Cylinder对象时将半径初始化为r,高度初始化为h

Cylinder（double r,double h）

* + 1. 成员方法

double getVolume（） //获取圆柱体的体积

void showVolume（） //将圆柱体的体积输出到屏幕

**public class** Circle {  
  
 **private double r**;  
 **private double pai**=3.14;  
  
 **public** Circle(){  
 }  
  
 **public** Circle(**double** r){  
 **this**.**r** = r;  
 }  
  
 **public double** getArea(){  
 **// double area = pai\*r\*r;  
 return pai**\***r**\***r**;  
 }  
  
 **public double** getPerimeter(){  
 **double** perimeter = 2\***pai**\***r**;  
 **return** perimeter;  
 }  
  
 **public void** show(){  
 System.***out***.println(**"圆的半径为："**+**r**);  
 System.***out***.println(**"圆的面积为："**+getArea());  
 System.***out***.println(**"圆的周长为："**+getPerimeter());  
 }  
  
}

**public class** Cylinder **extends** Circle {  
 **private double hight**;  
  
 **public** Cylinder(**double** r,**double** h){  
 **super**(r);  
 **this**.**hight** = h;  
 }  
  
 **public double** getVolume(){  
 **return super**.getArea()\***hight**;  
 }  
  
 **public void** showVolume(){  
 **super**.show();  
 System.***out***.println(**"圆柱体的体积为："**+getVolume());  
 }  
}

**public class** CircleTest {  
 **public static void** main(String[] args) {  
 Cylinder cylinder = **new** Cylinder(2,3);  
 cylinder.showVolume();  
 }  
}

* 1. 请使用面向对象的思想，设计自定义类，描述出租车和家用轿车的信息。

设定

1. 出租车类:

属性包括：车型，车牌，**所属出租公司**；方法包括：启动，停止

1. 家用轿车类：

属性包括：车型，车牌，**车主姓名**；方法包括：启动，停止

要求

1. 分析出租车和家用轿车的公共成员，提取出父类—汽车类
2. 利用继承机制，实现出租车类和家用轿车类
3. 编写测试类，分别测试汽车类，出租车类和家用轿车类对象的相关方法
4. 定义名为car的包存放汽车类，出租车类，家用轿车类和测试类

运行效果

我是车，我启动

乘客您好

我是景天出租车公司的，我的车牌是琼B250，您要去哪里？

我是车，我停止

目的地已经到了，请您付费下车，欢迎再次乘坐。

=================================================

我是车，我启动

我是根根，我的汽车我做主

我是车，我停止

目的地到了，我们去玩儿吧

**package** Car;  
  
**public class** Car {  
  
 **private** String **type**;  
 **private** String **plate**;  
  
 **public** Car(){  
 }  
  
 **public** Car(String type,String plate){  
 **this**.**type** = type;  
 **this**.**plate** = plate;  
 }  
  
 **public** String getType() {  
 **return type**;  
 }  
  
 **public void** setType(String type) {  
 **this**.**type** = type;  
 }  
  
 **public** String getPlate() {  
 **return plate**;  
 }  
  
 **public void** setPlate(String plate) {  
 **this**.**plate** = plate;  
 }  
  
 **public void** start(){  
 System.***out***.println(**"我是车，我启动"**);  
 }  
  
 **public void** stop(){  
 System.***out***.println(**"我是车，我停止"**);  
 }  
}

**package** Car;  
**import** Car.Car;  
  
**public class** Taxi **extends** Car {  
  
 **private** String **carCompany**;  
  
 **public** Taxi(){  
 }  
  
 **public** Taxi(String type,String plate,String carCompany){  
 **super**(type,plate);  
 **this**.**carCompany** = carCompany;  
 }  
  
 @Override  
 **public void** start(){  
 **super**.start();  
 System.***out***.println(**"乘客您好"**);  
 System.***out***.println(**"我是"**+**carCompany**+**"的，我的车牌是"**+getPlate()+**"，您要去哪里？"**);  
 }  
  
 @Override  
 **public void** stop(){  
 **super**.stop();  
 System.***out***.println(**"目的地已经到了，请您付费下车，欢迎再次乘坐。"**);  
 }  
}

**package** Car;  
**import** Car.Car;  
  
**public class** PrivateCar **extends** Car {  
 **private** String **ownerName**;  
  
 **public** PrivateCar(String type, String plate, String ownerName) {  
 **super**(type, plate);  
 **this**.**ownerName** = ownerName;  
 }  
  
 @Override  
 **public void** start(){  
 **super**.start();  
 System.***out***.println(**"我是"**+**ownerName**+**"，我的汽车我做主"**);  
 }  
  
 @Override  
 **public void** stop(){  
 **super**.stop();  
 System.***out***.println(**"目的地到了，我们去玩儿吧"**);  
 }  
}

**package** Car;  
  
**public class** CarTest {  
  
 **public static void** main(String[] args) {  
 Taxi taxi1 = **new** Taxi(**"SUV"**,**"琼B250"**,**"景天出租车公司"**);  
 taxi1.start();  
 taxi1.stop();  
 System.***out***.println(**"================================================="**);  
 PrivateCar privateCar1 = **new** PrivateCar(**"皮卡"**,**"黑A555"**,**"根根"**);  
 privateCar1.start();  
 privateCar1.stop();  
 }  
}

* 1. 某公司要开发新游戏，请用面向对象的思想，设计游戏中的蛇怪和蜈蚣精

设定

1. 蛇怪类:

属性包括：怪物名字，生命值，攻击力

方法包括：攻击，移动（曲线移动），补血（当生命值<10时，可以补加20生命值）

1. 蜈蚣精类：

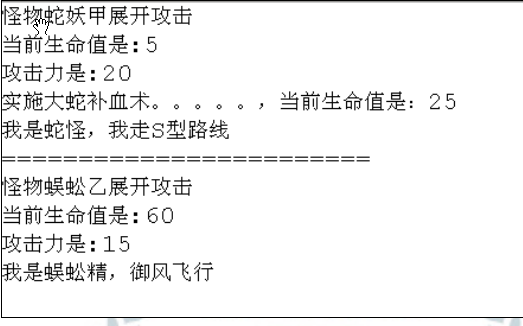
属性包括：怪物名字，生命值，攻击力

方法包括：攻击，移动（飞行移动）

要求

1. 分析蛇怪和蜈蚣精的公共成员，提取出父类—怪物类
2. 利用继承机制，实现蛇怪类和蜈蚣精类
3. 攻击方法，描述攻击状态。内容包括怪物名字，生命值，攻击力
4. 编写测试类，分别测试蛇怪和蜈蚣精的对象及相关方法
5. 定义名为mon的包存怪物类，蛇怪类，蜈蚣精类和测试类

运行效果



**package** Mon;  
  
**public abstract class** Monster {  
  
 **private** String **name**;  
 **private int lifeValue**;  
 **private int attackValue**;  
  
 **public** Monster(){  
 }  
  
 **public** Monster(String name, **int** lifeValue, **int** attackValue) {  
 **this**.**name** = name;  
 **this**.**lifeValue** = lifeValue;  
 **this**.**attackValue** = attackValue;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 **public int** getLifeValue() {  
 **return lifeValue**;  
 }  
  
 **public void** setLifeValue(**int** lifeValue) {  
 **this**.**lifeValue** = lifeValue;  
 }  
  
 **public int** getAttackValue() {  
 **return attackValue**;  
 }  
  
 **public void** setAttackValue(**int** attackValue) {  
 **this**.**attackValue** = attackValue;  
 }  
  
 **public abstract void** attack();  
  
 **public abstract void** move();  
}

**package** Mon;  
  
**public class** Snake **extends** Monster{  
  
 **public** Snake(String name, **int** lifeValue, **int** attackValue) {  
 **super**(name, lifeValue, attackValue);  
 }  
  
 @Override  
 **public void** attack(){  
 System.***out***.println(**"怪物蛇妖"**+**this**.getName()+**"展开攻击"**);  
 System.***out***.println(**"当前生命值是："**+**this**.getLifeValue());  
 System.***out***.println(**"攻击力是："**+**this**.getAttackValue());  
 **if**(getLifeValue()<10){  
 System.***out***.println(**"实施大蛇补血术，当前生命值是："**+(getLifeValue()+20));  
 }  
 }  
  
 **public void** move(){  
 System.***out***.println(**"我是蛇怪，我走S型路线"**);  
 }  
}

**package** Mon;  
  
**public class** Centipede **extends** Monster{  
  
 **public** Centipede(String name, **int** lifeValue, **int** attackValue) {  
 **super**(name, lifeValue, attackValue);  
 }  
  
 @Override  
 **public void** attack(){  
 System.***out***.println(**"怪物蜈蚣"**+**this**.getName()+**"展开攻击"**);  
 System.***out***.println(**"当前生命值是："**+**this**.getLifeValue());  
 System.***out***.println(**"攻击力是："**+**this**.getAttackValue());  
 }  
  
 **public void** move(){  
 System.***out***.println(**"我是蜈蚣精，御风飞行"**);  
 }  
}

**package** Mon;  
  
**public class** MonsterTest {  
 **public static void** main(String[] args) {  
 Snake snake1 = **new** Snake(**"甲"**,5,20);  
 snake1.attack();  
 snake1.move();  
 System.***out***.println(**"=================================="**);  
 Centipede centipede1 = **new** Centipede(**"乙"**,60,15);  
 centipede1.attack();  
 centipede1.move();  
 }  
}

1. **可选题**
   1. 请用面向对象的思想，设计自定义类描述演员和运动员的信息

设定

1. 演员类:

属性包括：姓名，年龄，性别，毕业院校，代表作

方法包括：自我介绍

1. 运动员类：

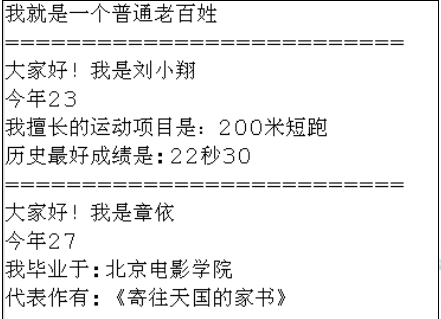
属性包括：姓名，年龄，性别，运动项目，历史最好成绩

方法包括：自我介始

要求

1. 分析演员和运动员的公共成员，提取出父类—人类
2. 利用继承机制，实现演员类和运动员类
3. 编写测试类，分别测试人类，演员类和运动员类对象及相关方法
4. 定义名为act的包存人类，演员类，运动员类和测试类

运行效果



**package** act;  
  
**public class** People {  
 **private** String **name**;  
 **private int age**;  
 **private** String **gender**;  
  
 **public** People(){  
 }  
  
 **public** People(String name,**int** age,String gender){  
 **this**.**name** = name;  
 **this**.**age** = age;  
 **this**.**gender** = gender;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 **public int** getAge() {  
 **return age**;  
 }  
  
 **public void** setAge(**int** age) {  
 **this**.**age** = age;  
 }  
  
 **public** String getGender() {  
 **return gender**;  
 }  
  
 **public void** setGender(String gender) {  
 **this**.**gender** = gender;  
 }  
  
 **public void** selfIntroduce(){  
 System.***out***.println(**"我就是一个普通老百姓"**);  
 }  
}

**package** act;  
  
**public class** Actor **extends** People{  
  
 **private** String **graduateSchool**;  
 **private** String **works**;  
  
 **public** Actor(){  
 }  
  
 **public** Actor(String name, **int** age, String gender, String graduateSchool, String works) {  
 **super**(name, age, gender);  
 **this**.**graduateSchool** = graduateSchool;  
 **this**.**works** = works;  
 }  
  
 @Override  
 **public void** selfIntroduce(){  
 System.***out***.println(**"大家好！我是"**+getName());  
 System.***out***.println(**"今年"**+getAge());  
 System.***out***.println(**"我毕业于:"**+**graduateSchool**);  
 System.***out***.println(**"代表作有:"**+**works**);  
 }  
}

**package** act;  
  
**public class** Athlete **extends** People{  
  
 **private** String **sportEvent**;  
 **private** String **bestScore**;  
  
 **public** Athlete(String name, **int** age, String gender, String sportEvent, String bestScore) {  
 **super**(name, age, gender);  
 **this**.**sportEvent** = sportEvent;  
 **this**.**bestScore** = bestScore;  
 }  
  
 @Override  
 **public void** selfIntroduce(){  
 System.***out***.println(**"大家好！我是"**+getName());  
 System.***out***.println(**"今年"**+getAge());  
 System.***out***.println(**"我擅长的运动项目是:"**+**sportEvent**);  
 System.***out***.println(**"历史最好成绩是:"**+**bestScore**);  
 }  
}

**package** act;  
  
**public class** PeopleTest {  
 **public static void** main(String[] args) {  
 People p = **new** People();  
 p.selfIntroduce();  
 System.***out***.println(**"============================"**);  
 Actor actor1 = **new** Actor(**"张译"**,27,**"男"**,**"北京电影学院"**,  
 **"<寄往天国的家书>"**);  
 actor1.selfIntroduce();  
 System.***out***.println(**"============================"**);  
 Athlete athlete1 = **new** Athlete(**"刘小翔"**,23,**"男"**,**"200米短跑"**,**"22秒30"**);  
 athlete1.selfIntroduce();  
 }  
}