**1. java类是否可以多继承，怎么实现多继承？**

答：java没有多继承，但可以通过接口的形式来达到多继承的目地。

**2.比较两个String总是false，但是它们明明都是"abc" ！**

答：比较String一定要使用equals或equalsIgnoreCase方法，不要使用 == ！

　　==比较的是两个引用（变量）是否指向了同一个对象，而不是比较其内容

**3.int 和 Integer 有什么区别**

答：Java 提供两种不同的类型：引用类型和原始类型（或内置类型）。

Int是java的原始数据类型，Integer是java为int提供的封装类。Java为每个原始类型提供了封装类。

原始类型封装类,booleanBoolean,charCharacter,byteByte,shortShort,intInteger,longLong,floatFloat,doubleDouble

引用类型和原始类型的行为完全不同，并且它们具有不同的语义。

引用类型和原始类型具有不同的特征和用法，它们包括：大小和速度问题，这种类型以哪种类型的数据结构存储，

当引用类型和原始类型用作某个类的实例数据时所指定的缺省值。对象引用实例变量的缺省值为 null，而原始类型实例变量

的缺省值与它们的类型有关

**4.定义类A 和类B 如下：**

class A {

int a=1;

double d=2.0;

void show(){

System.out.println("Class A: a="+a +"\td="+d);

}

}

class B extends A{

float a=3.0f;

String d="Java program.";

void show(){

super.show( );

System.out.println("Class B: a="+a +"\td="+d);

}

}

(1) 若在应用程序的main 方法中有以下语句：

A a=new A();

a.show();

则输出的结果如何？

(2) 若在应用程序的main 方法中定义类B 的对象b：

A b=new B();

b.show();

则输出的结果如何？

答：输出结果为：

1）Class A: a=1 d=2.0

2）Class A: a=1 d=2.0

Class B: a=3.0 d=Java program

**5.如下：**

public void testString(){

String str1 = "abc";

String str2 = "abc";

String str3 = new String("abc");

System.out.println(str1 == str2);//true

System.out.println(str1 == str3);//false

System.out.println(str1.equals(str3));//true

str1 = "abc";

String str4 = "abc";

String str5 = "a" + "bc";

System.out.println(str1 == str5);//true

String str6 = (str2 + str4).intern();

System.out.println(str1 == str6);//false ---->true

}

**6.重载（overload)和重写(overried，有的书也叫做“覆盖”）的区别？**

Overload的方法是否可以改变返回值的类型？

答：方法的重写Overriding和重载Overloading是Java多态性的不同表现。重写Overriding是父类

与子类之间多态性的一种表现，重载Overloading是一个类中多态性的一种表现。如果在子类中定

义某方法与其父类有相同的名称和参数，我们说该方法被重写 (Overriding)。子类的对象使用这

个方法时，将调用子类中的定义，对它而言，父类中的定义如同被"屏蔽"了。如果在一个类中定

义了多个同名的方法，它们或有不同的参数个数或有不同的参数类型，则称为方法的重载(Overloading)。

Overloaded的方法是可以改变返回值的类型

**7.写出程序结果**

class Test{

Test() {

System.out.println("Test");

}

}

class Demo extends Test{

Demo() {

//super();

System.out.println("Demo");

}

public static void main(String[] args) {

new Demo();

new Test();

}

}

Test

Demo

Test

**8.【练习题】1.继承**：

定义一个名为Vehicles（交通工具）的基类，该类中应包含String类型的成员属性brand（商标）和color（颜色），还应包含成员方法run（行驶，在控制台显示“我已经开动了”）和showInfo（显示信息，在控制台显示商标和颜色），并编写构造方法初始化其成员属性。

编写Car（小汽车）类继承于Vehicles类，增加int型成员属性seats（座位），还应增加成员方法showCar（在控制台显示小汽车的信息），并编写构造方法。 编写Truck（卡车）类继承于Vehicles类，增加float型成员属性load（载重），还应增加成员方法showTruck（在控制台显示卡车的信息），并编写构造方法。 在main方法中测试以上各类。

**【练习题】2.继承**：

编写一个类Calculate1，实现加、减两种运算，然后，编写另一个派生类Calculate2，实现乘、除两种运算。

**【练习题】3.包、继承、构造函数**

编写一个控制台应用程序，完成下列功能，并回答提出的问题。

1) 创建一个类A，在构造函数中输出“A”，再创建一个类B，在构造函数中输出“B”。

2) 从A继承一个名为C的新类，并在C内创建一个成员变量B。不要为C创建构造函数。

3) 在Main方法中创建类C的一个对象，写出运行程序后输出的结果。

4) 如果在C中也创建一个构造函数输出“C”，整个程序运行的结果又是什么？

**【练习题】4.继承**：

建立三个类：居民、成人、官员。居民包含身份证号、姓名、出生日期，而成人继承自居民，多包含学历、职业两项数据；官员则继承自成人，多包含党派、职务两项数据。要求每个类的字段都以属性的方式对外提供数据输入输出的功能。

**【练习题】5.覆盖、继承**

建立一个汽车Auto类，包括轮胎个数，汽车颜色，车身重量、速度等成员变量。并通过不同的构造方法创建实例。至少要求： 汽车能够加速，减速，停车。 再定义一个小汽车类Car，继承Auto，并添加空调、CD等成员变量，覆盖加速，减速的方法

**【练习题】6.类的继承，super**

1、写一个名为Account的类模拟账户。该类的属性和方法如下图所示。该类包括的属性：账号id，余额balance，年利率annualInterestRate；包含的方法：访问器方法（getter和setter方法），返回月利率的方法getMonthlyInterest()，取款方法withdraw()，存款方法deposit()。

|  |
| --- |
| Account |
| private int id  private double balance  private double annualInterestRate |
| public Account (int id, double balance, double annualInterestRate ) |
| public int getId()  public double getBalance()  public double getAnnualInterestRate()  public void setId( int id)  public void setBalance(double balance)  public void setAnnualInterestRate(double annualInterestRate)  public double getMonthlyInterest()  public void withdraw (double amount)  public void deposit (double amount) |

写一个用户程序测试Account类。在用户程序中，创建一个账号为1122、余额为20000、年利率4.5%的Account对象。使用withdraw方法提款30000元，并打印余额。

再使用withdraw方法提款2500元，使用deposit方法存款3000元，然后打印余额和月利率。

**提示：**在提款方法withdraw中，需要判断用户余额是否能够满足提款数额的要求，如果不能，应给出提示。

运行结果如图所示

**2、创建Account类的一个子类CheckAccount代表可透支的账户，该账户中定义一个属性overdraft代表可透支限额。在CheckAccount类中重写withdraw方法，其算法如下**：

如果（取款金额<账户余额），

可直接取款

如果（取款金额>账户余额），

计算需要透支的额度

判断可透支额overdraft是否足够支付本次透支需要，如果可以

将账户余额修改为0，冲减可透支金额

如果不可以

提示用户超过可透支额的限额

**要求：**写一个用户程序测试CheckAccount类。在用户程序中，创建一个账号为1122、余额为20000、年利率4.5%，可透支限额为5000元的CheckAccount对象。

使用withdraw方法提款5000元，并打印账户余额和可透支额。

再使用withdraw方法提款18000元，并打印账户余额和可透支额。

再使用withdraw方法提款3000元，并打印账户余额和可透支额。

**提示：**

（1） 子类CheckAccount的构造方法需要将从父类继承的3个属性和子类自己的属性全部初始化。

父类Account的属性balance被设置为private，但在子类CheckAccount的withdraw方法中需要修改它的值，因此应修改父类的balance属性，定义其为protected。