面向对象面试题

1. 面向对象与面向过程有什么区别？

1. 出发点不同

面向对象方法是用符合常规思维的方式来处理客观世界的问题，强调把问题域的要领映射到对象及对象之间的接口上。面面向过程方法强调的则是过程的抽象化与模块化，它是以过程为中心构造或处理客观世界问题的。

2. 层次逻辑关系不同

面向对象方法则是用计算机逻辑来模拟客观世界中的物理存在，以对象的集合类作为处理问题的基本单位，尽可能地使计算机世界向客观世界靠拢，以使问题的处理更清晰直接，面向对象方法是用类的层次结构来体现类之间的继承和发展。而面向过程方法处理问题的基本单位是能清晰准确地表达过程的模块，用模块的层次结构概括模块或模块间的关系与功能，把客观世界的问题抽象成计算机可以处理的过程。

3. 数据处理方式与控制程序方式不同

面向对象方法将数据与对应的代码封装成一个整体，原则上其他对象不能直接修改其数据，即对象的修改只能由自身的成员函数完成，控制程序方式上是通过"事件驱动"来激活和运行程序。而面向过程方法是直接通过程序来处理数据，处理完毕后即可显示处理结果，在控制程序方式上是按照设计调用或返回程序，不能自由导航，各模块之间存在着控制与被控制、调用与被调用的关系。

4. 分析设计与编码转换方式不同

面向对象方法贯穿于软件生命周期的分析、设计及编码中，是一种平滑过程，从分析到设计再到编码是采用一致性的模型表示、即实现的是二种无缝连接。而面向过程方法强调分析、设计及编码之间按规则进行转换，贯穿于软件生命周期的分析、设计及编码中，实现的是一种有缝的连接

2. 面向对象有哪些特征？

面向对象的特征包括抽象、继承、封装和多态。

1)抽象。抽象就是忽略一个主题中与当前目标无关的邓些方面，以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面：一是过程抽象；二是数据抽象。

2)继承。继承是一种联结类的层次模型，并且允许和鼓励类的重用，它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生，这个过程称为类继承、新类继承了原始类的特性，新类称为原始类的派生类(子类)，而原始类称为新类的基类(父类)，派生类可以从它的基类郡里继承方法和实例变量，并且派生类可以修改或增加新的方法使之更适合特殊的需要。

3)封装。封装是指将客观事物抽象成类，每个类对自身的数据和方法实行保护：类可以把自己的数据和方法只让可信的类或者对象操作，对不可信的进行信息隐藏.

4)多态。多态是指允许不同类的对象对同一消息作出响应。多态包括参数化多态和包含多态。多态性语言具有灵活、抽象、行为共享、代码共享等优势，很好地解决了应用程序函数同名问题。

3. 面向对象的开发方式 有什么优点？

1)较高的开发效率。采用面向对象的开发方式，可以对现实的事物进行抽象，可以把现实的事物直接映射为开发的对象，与人类的思维过程相似，同时，由于面向对象的开发方式可以通过继承或者组合的方式来实现代码的重用，因此可以大大地提高软件的开发效率。

2)保证软件的鲁棒性。正是由于面向对象的开发方法有很高的重用性，在开发的过程中可以重用已有的而且在相关领域经过长期测试的代码，因此，自然而然地对软件的鲁棒性起到良好的促进作用。

3)保证软件的高可维护性。由于采用面向对象的开发方式，使得代码的可读性非常好，同时面向对象的设计模式也使得代码结构更加清晰明了。同时针对面向对象的开发方式，已有许多非常成熟的设计模式，这些设计模式可以使程序在面对需求的变更时，只需要修改部分的模块就可以满足需求，因此维护起来非常方便。

4. 什么是继承？

继承是面向对象中的一个非常重要的特性。能过继承，了类可以使用父类中的一些成员变量与方法，从而能够提高代码的复用性，提高开发效率。继承主要有如下几个特性：

1．Java 语言不支持多重继承

2．子类只能继承父类的非私有(public 与 protected)成员变量与方法

3．当了类中定义的成员变量和父类中定义的成员变量同名时，子类中的成员变量会覆盖父类的成员变量，而不会继承。

4．当子类中的方法与父类中的方法 有相同的函数签名(相同的方法名，相同的参数个数与类型)时，子类什么覆盖父类的方法，而不会继承。

5. 组合和继承有什么区别？

组合和继承是面向对象中两种代码复用的方式。

组合是指在新类里面创建原有类的对象，重复利用已有类的功能。被称为 has-a 关系。

继承是面向对象的主要特性之一，它允许设计人员根据其他类的实现来定义一个类的实现。被称为 is-a 关系。

使用注意事项：

除非两个类之间是 is-a 关系，否则不要轻易使用继承，不要单纯地为了实现代码的重用而使用继承，因为过多地使用继承会破坏代码的可维护性。

不要仅仅为了实现多态而继承，如果类之间没有 is-a 的关系，可以通过实现接口与组合方式来达到相同的目的。

由于 Java 语言只支持间继承，如果 想同时继承两个类或多个类，在 Java 中是无法实现的。

在 Java 语言中，能使用组合就尽量不要使用继承。

6. 多态的实现机制是什么？

多态是面向对象程序设计中代码重用的一个重要机制，它表示当同一个操作作用在不同对象时，会有不同的语义，从而会产生不同的结果，例如，同样是执行"+"操作，"3+4"用来实现整数相加，而"3"+"4"却实现了字符串的连接。在 Java 语言中，多态主要有以下两种表现方式：

1)方法的重载(overload)。重载是指同一个类中有多个同名的方法，但这些方法有着不同的参数，因此在编译时就可以确定到底调用哪个方法，它是一种**编译时多态**。重载可以被看作一个类中的方法多态性。

2)方法的覆盖(overide)。子类可以覆盖父类的方法，因此同样的方法会在父类与子类中有着不同的表现形式。在 Java 语言中，基类的引用变量不仅可以指向基类的实例对象，也可以指向其子类的实例对象。同样，接口的引用变量也可以指向其实现类的实例对象，而程

序调用的方法在运行期才动态绑定(绑定指的是将一个方法调用和一个方法主体连接到一起)，就是引用变量所指向的具体实例对象的方法，也就是内存里正在运行的那个对象的方法，而不是引用变量的类型中定义的方法。通过这种动态绑定的方法实现了多态。由于只有在运行时才能确定调用哪个方法，因此通过方法覆盖实现的多态也可以被称为**运行时多态**，示例如下。

7. 重载和覆盖有什么区别？

重载(overload)和覆盖(overide)是 Java 多态性的不同表现方式。其中，重载是在一个类中多态性的一种表现，是指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。在使用重载时，需要注意以下几点：

1)重载是通过不同的方法参数来区分的，例如不同的参数个数、不同的参数类型或不同的参数顺序。

2)不能通过方法的访问权限、返回值类型和抛出的异常类型来进行重载。

3)对于继承来说，如果基类方法的访问权限为 privae，那么就不能在派生类对其重载；如果派生类也定义了一个同名的函数，这只是一个新的方法，不会达到重载的效果。

覆盖是指派生类函数覆盖基类函数。覆盖一个方法并对其重写，以达到不同的作用。在使用覆盖时需要注意以下几点：

1)派生类中的覆盖方法必须要和基类中被覆盖的方法有相同的函数名和参数。

2)派生类中的覆盖方法的返回值必须和基类中被覆盖白方法的返回值相同。

3)派生类中的覆盖方法所抛出的异常必须和基类(或是其子类)中被覆盖的方法所抛出的异常一致。

4)基类中被覆盖的方法不能为 private 否则其子类只是定义了一个方法，并没有对其覆盖。

重载与覆盖的区别主要有以下几个方面：

1)覆盖是子类和父类之间的关系，是垂直关系；重载是同一个类中方法之间的关系，是水平关系。

2)覆盖只能由一个方法或只能由一对方法产生关系；重载是多个方法之间的关系。

3)覆盖要求参数列表相同；重载要求参数列表不同。

4)覆盖关系中，调用方法体是根据对象的类型(对象对应存储空间类型)来决定；而重

载关系是根据调用时的实参表与形参表来选择方法体的。

8. 抽象类(abstract class) 与接口(interface)有什么异同？

9. 内部类有哪些？

在 Java 语言中，可以把一-个类定义到另外一个类的内部，在类哏而的这个类就叫做内部

类，外面的类叫做外部类。在这种情况下，这个内部类可以被看作外部类的·个成员(`：i 类

的属性和方法类似)。还有-_一种类被称为顶层(top-j.·el)类，指的是类定义代码不嵌套在其

他类定义中的类口

需要注意的是，嵌套类(Ne 目 tedCs)守内部类(lnn·r(爿·H、)类似，只是嵌套类是

的说法，而内部类址 Java 的说法而已门内部类可以分为很多种，主要有以卜`4 种：静态

内部类(·tati.inn 贮 r 日 a··)、成员内部类(-lem 山 erinnt`rcl.··)、局部内部类油 n'alinn 曜`r 日找··)

和匿名内部类(arumym 口·`innor 川.··)它们的定义方法如下 J

口

« ·tanti outor(: l*\Hel I

Htatit\ « ·lt\HH inn{\r(: laRSJ I

c ·lar¢tl uutrrclupol

t ·ltul= oult ·rclat\s 1

/静态内部类 1

/成员内部类(博通内部垫)

publit ·voitl mrnlwrl,"um ·tion( ) I

glaset innerclass I

戶局部内部类

(1

静态内部类是指被声明为 static 的内部类，它可以不依赖于外部类实例而被实例化，而通常的内部类需要在外部类实例化后才能实例化。静态内部类不能与外部类有相同的名字，苡访问外部类的普通成员变量，只能访问外部类中的静态成员和静态方法(包括私有类

个静态内部类，如果去掉"static"关键字，就成为成员内部类。成员内部类为非部态内部类，它可以自由地引用外部类的属性和方法，无论这些属性和方法是静态的还是非静态的。但是它与一个实例绑定在了二起，不可以定义静态的属性和方法。只有在外部的衷苡产四化后，这个内部类才能被实例化。

成员

需要注意的是，非静态内部类中不能有静态

局部内部类指的是定义在一个代码块内的类，它的作用范围为其所在的代码块，是内部类中最少使用到的二种类型。局部内部类俅局部变量二样，不能被 public、protected、private 以

及 sta·ic 修饰，只康往右问方法中定义为 final 类型的局部变量。对二个静态内部类，去掉其声明

中的”static”关键字，将其定义移入其外部类的静态方法或静态初始化代码段中就成为了局

部静态内部类。对二个成员类，将其定义移入其外部类的实例方法或实例初始化代码中就成为

了局部内部类。局部静态内部类与静态内部类的基本特性相同。局部内部类与内部类的基本特

主相同。

匿名内部类是二种没有类名的内部类，不使用关键字 class、extend

`函数，它必须继承(exte·ds)其他类或实现其他接口。匿名内部类鑫遐代码更加简洁，

没有构

10. 如何获取父类的类名？

11. this 与 super 有什么区别？

在 Ja·a 语言中，this 用来指向当前实例对象，它的一个非常重要的作用就是用来区分鲫

的成员变量与方法的形参(当一个方法的形参与成员变量的名字相同时，就会覆盖成员量)。为了能够对 this 有一个更好的认识，首先创建一个类 Peopl” ，示例如下：

```
class People I

String name ;
/正确的写法

public People( String name) 1

this. Name-name ;

刀错误的写法
```

public People(String name) 1

name-name ;

上例中，第一个构造函数使用 this，

name

来表示左边的值为成员变量，而不是这个构造[

数的形式参数。对于第二个构造函数，由于在这个函数中形参与成员变量有着相同的名字，1

此对于语 name 一 name，等号左边和右边的两个 name 都代表的是形式参数。在这种情阳

只有通过 this 才能访问到成员变量。

“up”’ 可以用来访问父类的方法或成员变量。当子类的方法或成员变量与父类有相同%’