1. 什麼是方法的重寫：

子類繼承父類以後，可以對父類中同名同參數的方法，進行覆蓋操作

1. 應用：

重寫以後，當創建子類對像以後，通過子類對象調用子父類中的同名同參數的方法時，實際執行的是子類重寫父類的方法。

1. 舉例：  
   class Circle{

public double findArea(){} //求圓面積

}

class Cyclinder extends Circle{

public double findArea(){} //求表面積

}

class Account{

public boolean withdraw( double amt){}

}

class CheckAccount extends Account {

public boolean withdraw( double amt){}

}

1. 重寫的規則：

方法的聲明： 權限修飾符 返回值類型 方法名(形參列表) throws 異常的類型{

//方法體

}

約定俗稱：子類中的叫重寫的方法，父類中的叫被重寫的方法

1 子類重寫的方法的方法名和形參列表與父類被重寫的方法的方法名和形參列表相同

2 子類重寫的方法的權限修飾符不小於父類被重寫的方法的權限修飾符

>特殊情況：子類不能重寫父類中聲明為private權限的方法

3 返回值類型：

>父類被重寫的方法的返回值類型是void，則子類重寫的方法的返回值類型只能是void

>父類被重寫的方法的返回值類型是A類型，則子類重寫的方法的返回值類型可以是A類或A類的子類

>父類被重寫的方法的返回值類型是基本數據類型(比如：double)，則子類重寫的方法的返回值類型必須是相同的基本數據類型(必須也是double)

4 子類重寫的方法拋出的異常類型不大於父類被重寫的方法拋出的異常類型  
（具體放到異常處理時候講）

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

子類和父類中的同名同參數的方法要么都聲明為非static的（考慮重寫），要么都聲明為static的（不是重寫）。

1. 面試題:

面試題：區分方法的重載與重寫

Ans:

1. 兩者的概念:
2. 重載和重寫的具體規則
3. 重載：不表現為多態性　重寫：表現為多態性
4. 4. 重載，是指允許存在多個同名方法，而這些方法的參數不同。編譯器根據方法不同的參數表，對同名方法的名稱做修飾。對於編譯器而言，這些同名方法就成了不同的方法。它們的調用地址在編譯期就綁定了。Java的重載是可以包括父類和子類的，即子類可以重載父類的同名不同參數的方法。所以：對於重載而言，在方法調用之前，編譯器就已經確定了所要調用的方法，這稱為“早綁定”或“靜態綁定”；而對於多態，只有等到方法調用的那一刻，解釋運行器才會確定所要調用的具體方法，這稱為“晚綁定”或“動態綁定”。

引用一句Bruce Eckel的話：“不要犯傻，如果它不是晚綁定，它就不是多態。”