1. java異常處理的抓拋模型

　一、異常的處理：抓拋模型

過程一："拋"：程序在正常執行的過程中，一旦出現異常，就會在異常代碼處生成一個對應異常  
 類的對象。

並將此對象拋出。

一旦拋出對像以後，其後的代碼就不再執行。

關於異常對象的產生：

1. 系統自動生成的異常對象

2. 手動的生成一個異常對象，並拋出（throw）

過程二："抓"：可以理解為異常的處理方式：1. try- catch-finally 2. throws

1. 異常處理方式一：try- catch-finally

2.1使用說明:

try{

//可能出現異常的代碼

}catch(異常類型1變量名1){

//處理異常的方式1

}catch(異常類型2變量名2){

//處理異常的方式2

}catch(異常類型3變量名3){

//處理異常的方式3

}

....

finally{

//一定會執行的代碼

}

說明：

1. finally是可選的。

2.使用try將可能出現異常代碼包裝起來，在執行過程中，一旦出現異常，就會生成一個對應異

　常類的對象，根據此對象的類型，去catch中進行匹配

3.一旦try中的異常對象匹配到某一個catch時，就進入catch中進行異常的處理。一旦處理完

成，就跳出當前的try-catch結構（在沒有寫finally的情況）。繼續執行其後的代碼

4. catch中的異常類型如果沒有子父類關係，則誰聲明在上，誰聲明在下無所謂。

catch中的異常類型如果滿足子父類關係，則要求子類一定聲明在父類的上面。否則，報錯

5.常用的異常對象處理的方式： 1 String getMessage() 2 printStackTrace()

6.在try結構中聲明的變量，再出了try結構以後，就不能再被調用

7. try -catch-finally結構可以嵌套

總結：如何看待代碼中的編譯時異常和運行時異常？

體會1：使用try-catch-finally處理編譯時異常，使得程序在編譯時就不再報錯，但是運行時仍

可能報錯。相當於我們使用try-catch-finally將一個編譯時可能出現的異常，延遲到運行時出現。

體會2：開發中，由於運行時異常比較常見，所以我們通常就不針對運行時異常編寫try-catch-finally了。針對於編譯時異常，我們說一定要考慮異常的處理。

* 1. finally的再說明：

1.finally是可選的

2.finally中聲明的是一定會被執行的代碼。即使catch中又出現異常了，try中有return語

句，catch中有return語句等情況。

3.像數據庫連接、輸入輸出流、網絡編程Socket等資源，JVM是不能自動的回收的，我們需要   
 自己手動的進行資源的釋放。此時的資源釋放，就需要聲明在finally中。

* 1. [面試題]

final / finally / finalize 這三者的區別?

類似：  
throw 和 throws

Collection 和 Collections

String / StringBuffer / StringBuilder

ArrayList / LinkedList

HashMap / Link HashMap

重寫 / 重載

結構不相似的:

抽象類 / 接口

== / equals()

sleep() / wait()

1. 異常處理方式二：

"throws + 異常類型"寫在方法的聲明處。指明此方法執行時，可能會拋出的異常類型。

一旦當方法體執行時，出現異常，仍會在異常代碼處生成一個異常類的對象，此對象滿足throws後異常類型時，就會被拋出。異常代碼後續的代碼，就不再執行！

1. 對比兩種處理方式：

try-catch-finally:真正的將異常給處理掉了。

throws的方式只是將異常拋給了方法的調用者。 並沒有真正將異常處理掉。

1. 體會開發中應該如何選擇兩種處理方式？

5.1 如果父類中被重寫的方法沒有throws方式處理異常，則子類重寫的方法也不能使用

throws，意味著如果子類重寫的方法中有異常，必須使用try-catch-finally方式處理。

* 1. 執行的方法A中，先後又調用了另外的幾個方法(B、C)，這幾個方法是遞進關系執行的。我

們建議這幾個方法使用throws的方式進行處理。而執行的方法A可以考慮使用try-catch-

finally方式進行處理。

例:

A(){

try{

B(){}

C(){}

}catch(){}

finally{}

}

B(){}//使用throws

C(){}//使用throws

1. 補充：

方法重寫的規則之一：

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