1. Lambda表達式使用前後的對比：  
   舉例一：

@Test  
public void test1*(){* Runnable r1 = new Runnable*() {* @Override  
 public void run*() {* System.*out*.println*(*"我爱北京天安门"*)*;  
 *}  
 }*;  
 r1.run*()*;  
 System.*out*.println*(*"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"*)*;  
 Runnable r2 = *()* -> System.*out*.println*(*"我爱北京故宫"*)*;  
 r2.run*()*;  
*}*

舉例二：

@Test  
public void test2*(){* Comparator*<*Integer*>* com1 = new Comparator*<*Integer*>() {* @Override  
 public int compare*(*Integer o1, Integer o2*) {* return Integer.*compare(*o1,o2*)*;  
 *}  
 }*;  
 int compare1 = com1.compare*(*12,21*)*;  
 System.*out*.println*(*compare1*)*;  
 System.*out*.println*(*"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"*)*;  
 *//Lambda表达式的写法* Comparator*<*Integer*>* com2 = *(*o1,o2*)* -> Integer.*compare(*o1,o2*)*;  
 int compare2 = com2.compare*(*32,21*)*;  
 System.*out*.println*(*compare2*)*;  
 System.*out*.println*(*"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"*)*;  
 *//方法引用* Comparator*<*Integer*>* com3 = Integer :: *compare*;  
 int compare3 = com3.compare*(*32,21*)*;  
 System.*out*.println*(*compare3*)*;  
*}*

1. Lambda表達式的基本語法：  
   *\* 1.舉例： (o1,o2) -> Integer.compare(o1,o2);  
   \* 2.格式：  
   \* -> :lambda操作符 或 箭頭操作符  
   \* ->左邊：lambda形參列表 （其實就是接口中的抽象方法的形參列表）  
   \* ->右邊：lambda體 （其實就是重寫的抽象方法的方法體）*
2. 如何使用：分為六種情況

語法格式一：無參，無返回值



語法格式二：Lambda 需要一個參數，但是沒有返回值。



語法格式三：數據類型可以省略 ，因為可由編譯器推斷得出，稱為“類型推斷



語法格式四：Lambda 若只需要一個參數時，參數的小括號可以省略



語法格式五：Lambda 需要兩個或以上的參數，多條執行語句，並且可以有返回值



語法格式六：當 Lambda 體只有一條語句時，return 與大括號 若有，都可以省略



總結六種情況：

->左邊：lambda形參列表的參數類型可以省略(類型推斷)；

如果lambda形參列表只有一個參數，其一對()也可以省略，

而如果沒有參數或以個以上則一對()不可以省略

->右邊：lambda體應該使用一對{}包裹；

如果lambda體只有一條執行語句（可能是return語句），

可省略這一對{}和return關鍵字，{}和return關鍵字必須一起省略