Q1. Encapsulation(封裝):指class A的設計者可以指定其他的class能不能存取A的某一個member(組成)，java裡定義了四種

* private:只有A自己才可以存取, 使用keyword **private**
* package:只有和A同一個package的class才可以存取, 沒有相對應的keyword
* protected:只有同一個package或是A的子類別才可以存取, 使用keyword **protected**
* public:所有的class都可以存取, 使用keyword **public**

Kotlin 中則是

在應用程式套件裡面宣告的類別與函式，可以依照應用程式的架構，使用下列的關鍵字：

* private：只能在宣告的檔案中使用
* internal：只能在同一個模組中使用
* public：沒有使用private或internal的時候，預設為public，可以在應用程式任何地方使用

在類別與介面中宣告的屬性、建構式與函式，可以使用下面的關鍵字：

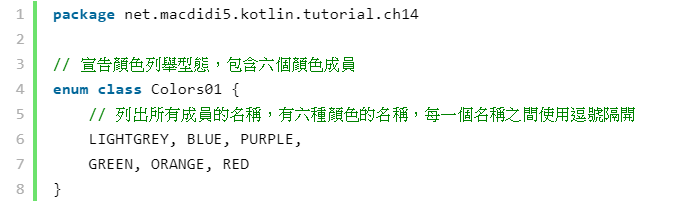
* private：只能在宣告的類別或介面中使用
* protected：比private多了可以在子類別或子介面中使用
* internal：只能在同一個模組中使用
* public：沒有使用private、protected或internal的時候，預設為public，可以在應用程式任何地方使用

因為有這樣的分別，資料在儲存時，或是在運行時，才能有效率的溝通。並且可以讓許多人一起維護、擴充功能而不會因為使用了相同的變數而導致程式崩潰。可讀性也因此增加。

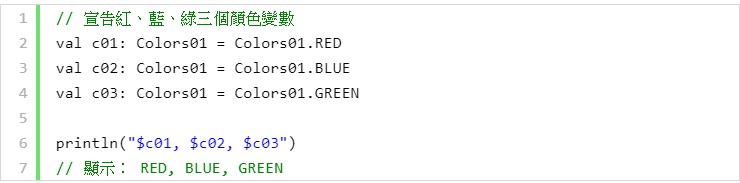
Q2. Enum(列舉型態)=> 設計應用程式的時候，除了基本的繼承架構外，通常也需要一些資料結構，例如產品分類、會員等級，或是項目的顏色。這類的應用通常是少數幾個固定的資料，使用變數或類別設計的話，會是比較麻煩的。Kotlin提供的列舉型態，是一種特殊的類別，非常適合用來設計這類資料結構。

Q3.

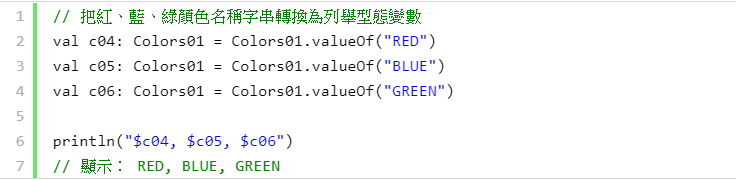
列舉類中的每一個列舉常量都是一個物件,並且他們之間用逗號分隔



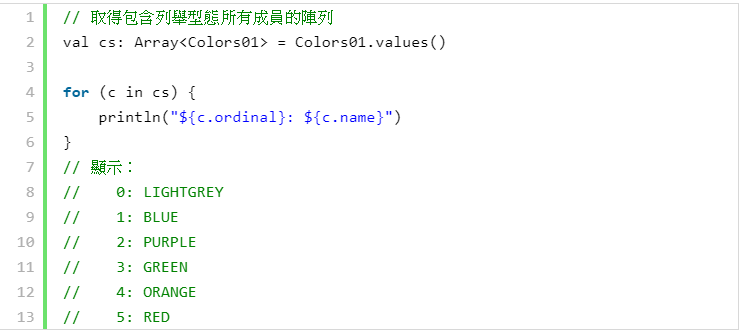
上面宣告的列舉類別Colors01，是一個特殊的類別，宣告與用法都跟一般的類別不一樣。列舉類別型態的變數，只能指定在類別中宣告好的成員名稱：



Kotlin會自動為列舉類別加入一些函式，例如把字串轉換為列舉類別型態，可以呼叫Kotlin自動加入的valueOf函式：



Kotlin自動加入的函式還有values，它可以傳回所有宣告在這個列舉類別型態裡面的成員，型態是列舉類別陣列。每一個列舉類別成員有Kotlin自動加入的ordinal與name兩個屬性，分別是成員的索引編號(Int)和名稱(String)：



Q4.

LinearLayout =>可以逐一對齊(垂直或水平)，並且整齊的表示在layout上

RelativeLayout =>與其父類、或其他元件的相關位置去觀看、排列。

Q5.

ConstraintLayout =>比較像是RelativeLayout，但是更靈活。因為他可以使用跟其他元件的相對位置定位與調整細小元件，具有相當大的靈活性。更容易且直觀的在佈局編輯器中使用。