第七章：封裝與建構式

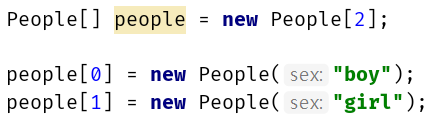
* 再談物件與類別

→類別class就是一個物件的基礎，裡面包含了物件本身的名稱、各種承載的屬性、與其方法，並利用這些特性與程式或其它物件互動

→物件的建立是以類別為架構，類別就是物件的基本單位

→程式在執行時的主體大部份是實體物件，而非類別

→類別不只能放數值，也能放物件、陣列、集合(陣列也可以以物件為單位→物件陣列

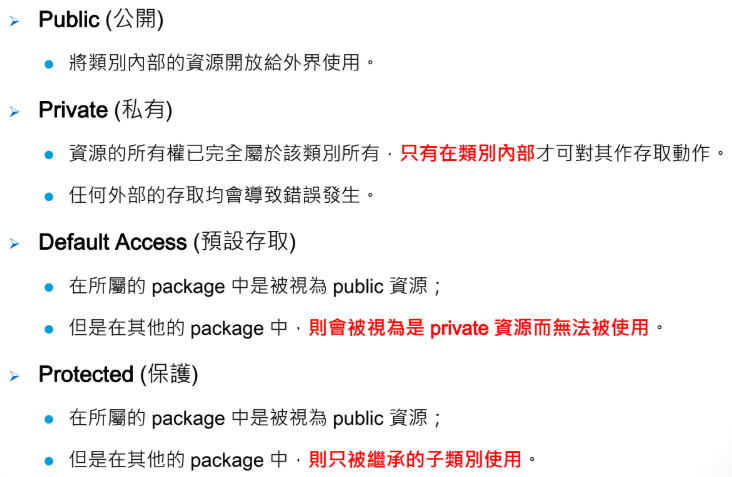


→People[]為people物件的陣列，每一個空間都是放一個people物件

1. 封裝

→為了保護重要或敏感的資料(如個人資訊、帳號、年紀、體重)不外流，僅能透過特定方法存取→封裝的意義

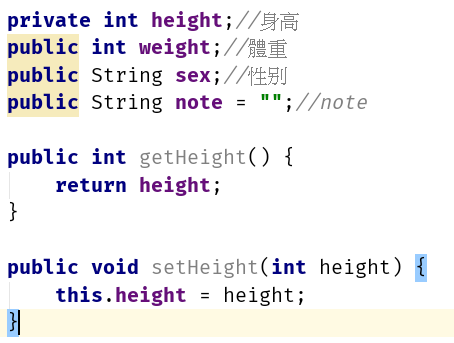
→透過修飾詞變更屬性層級，使得屬性本身不容易被其它類別改變，藉以達到保護資料本身的目的



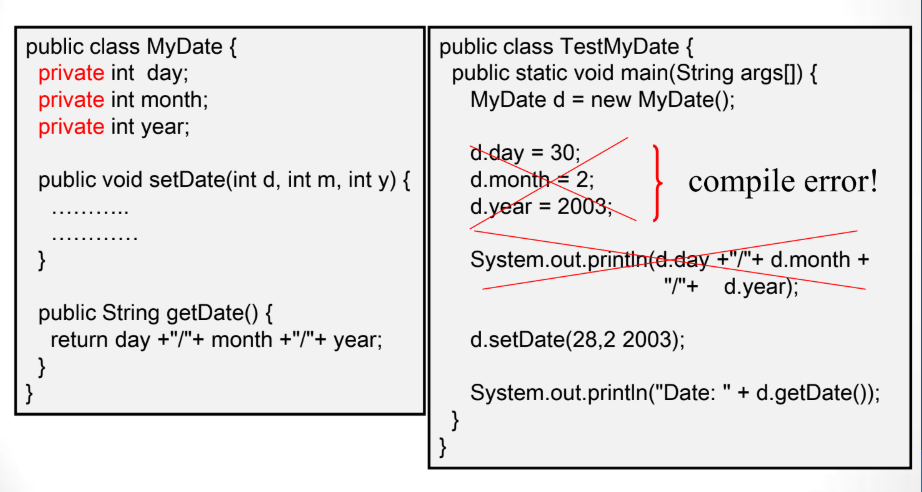
可存取權限：public > protected > default > private

* 封裝兩步驟

1. 屬性更改為private
2. 提供公開的方法存取，通常為getxxx或setxxx為一組



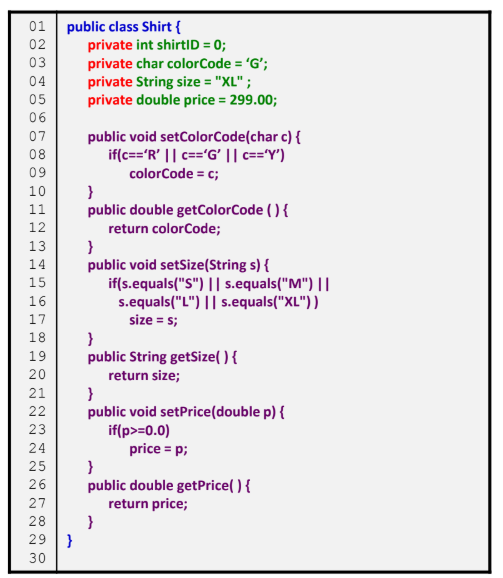
* 封裝後的物件存取



達到保護屬性的目的

PS：compile error的意思：程式在編譯初期就不會過，無法執行

* 典型的封裝實例



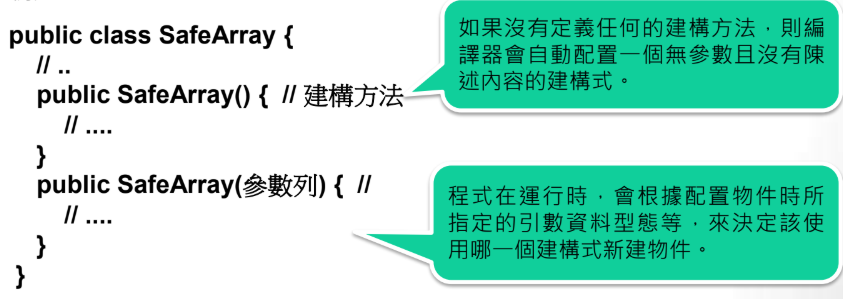
→每一個private的屬性搭配一組get與set方法來存取該屬性資料

1. 建構式

→建構式是做為一個類別的初始化動作

→本身其實就是一個方法：建立物件時會自動執行的方法

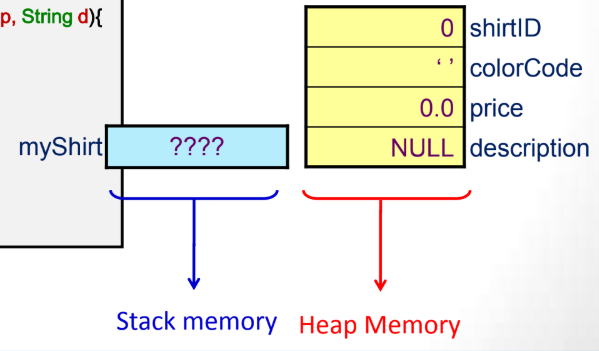
→名稱需與類別名稱相同，不需回傳值



預設建構式↓

也是一個方法多載的概念↑

* 複習物件中記憶體配置的位置



* 建構式vs方法

→性質相同，但呼叫時機不同，建構式只在一開始建立物件時呼叫，一般方法是物件建立後使用物件呼叫

→建構式名稱需要跟類別名稱一樣，方法不用

→建構式不需要回傳值，不需冠上回傳值的欄位，方法一定要有回傳值欄位

|  |  |  |
| --- | --- | --- |
|  | 建構式 | 方法 |
| 呼叫時機 | 物件建立初期 | 物件建立後/靜態方法隨時 |
| 命名 | 與類別名稱相同 | 獨立的名稱 |
| 回傳值 | 無 | 無/有 |
| 必須性 | 必要 | 非必要 |

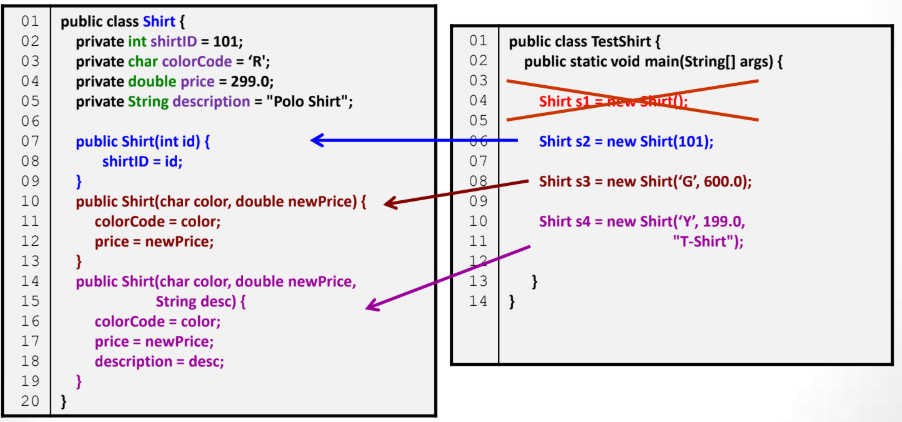
* 再談預設建構式

→類別裡面一定至少會有一個建構式

→如建立類別時沒有特別定義，預設會有一個預設建構式

→如建立了其它形式的建構式，則預設建構式不寫就不存在

* 建構式多載與預設建構子消失的實例



→有了自訂建構式後，預設建構式即會消失，如要使用就要自行再寫出來！

1. static成員

* static的概念

→就相常於公寓的客廳、樓梯間、交誼廳等，是大家可以一起使用的空間

→冠上static的變數、物件或方法即為全域變數、全域物件、全域方法

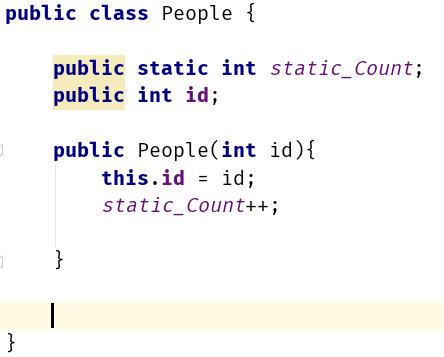
→static的成員其分配在記憶體的位置為global

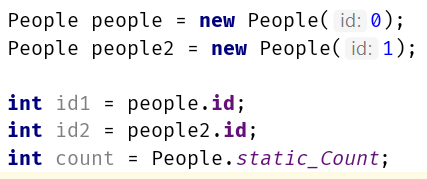
global

heap

stack

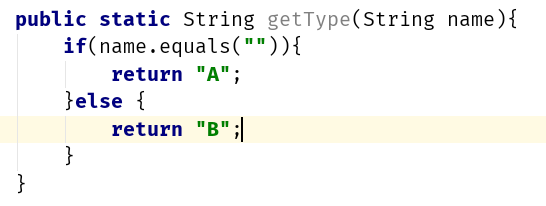
1. 靜態屬性





輸出結果：count = 2

1. 靜態方法



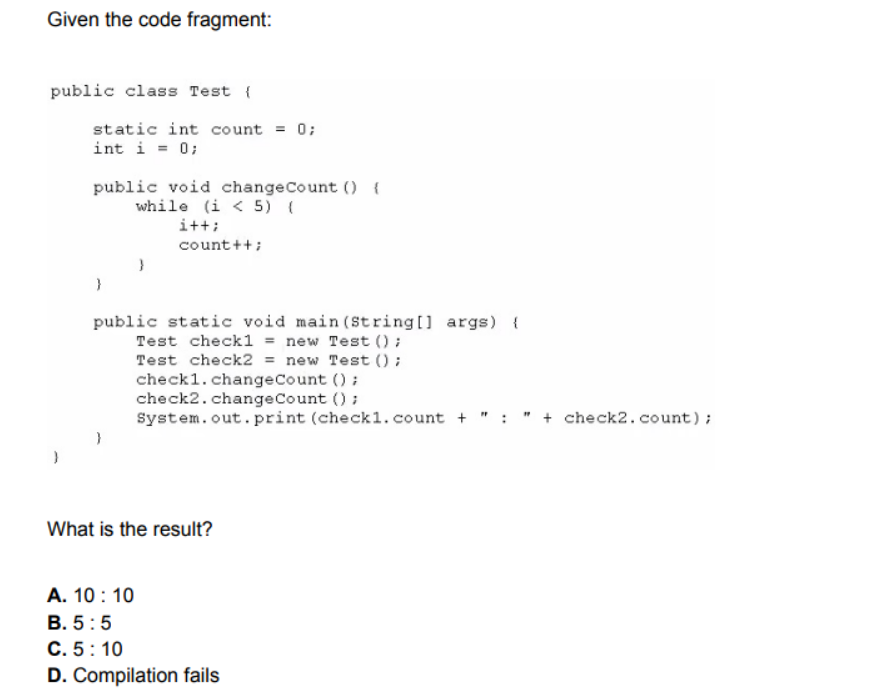
* 呼叫方式

→類別名稱+方法

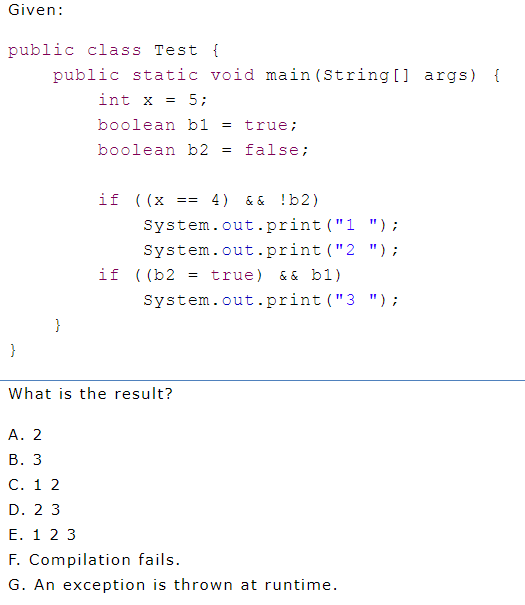


PS：物件名稱+方法名稱也可。

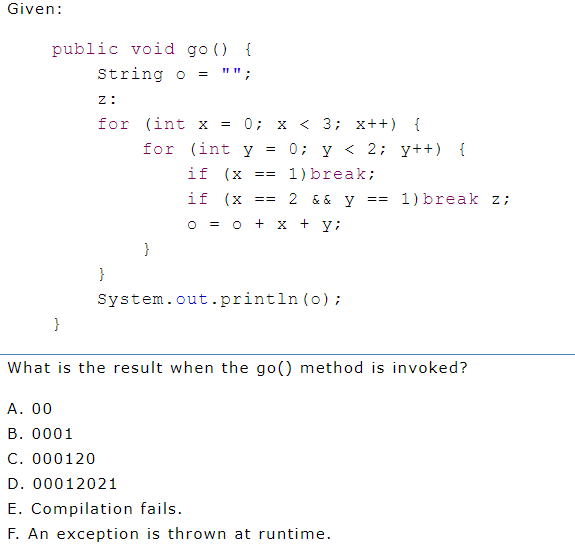
Ps：可以開始做一點OCA的練習題加深基礎觀念



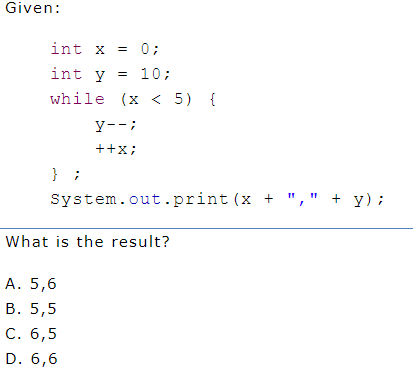
Ans: A



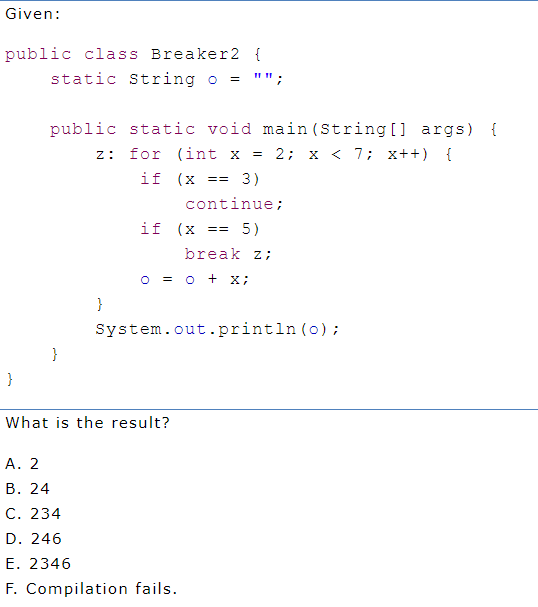
Ans: D



ANS: C

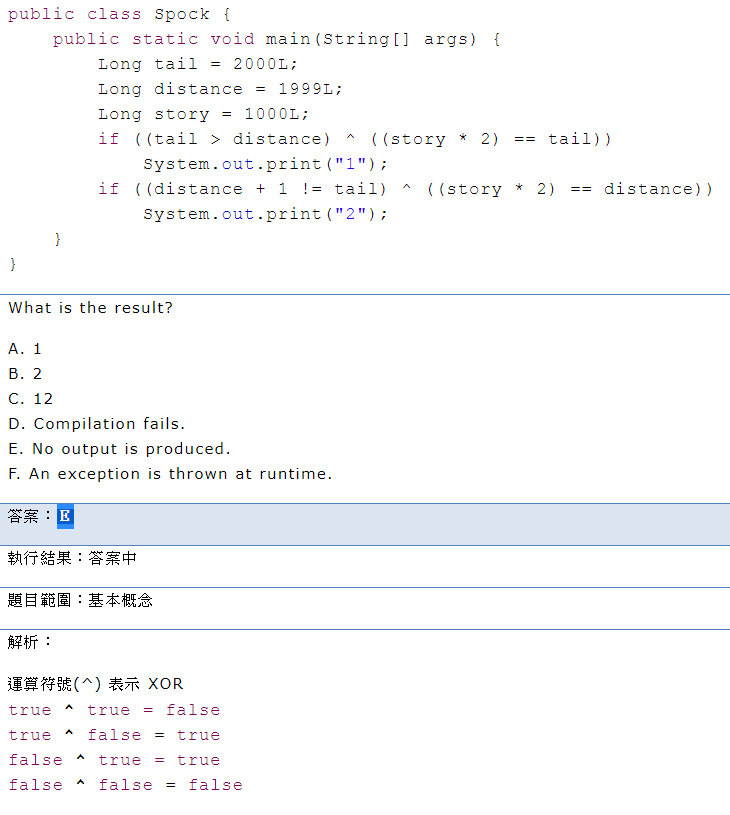


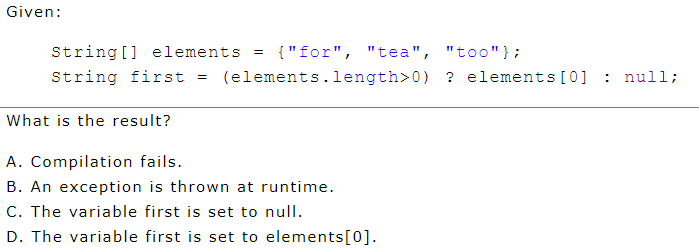
ANS: B



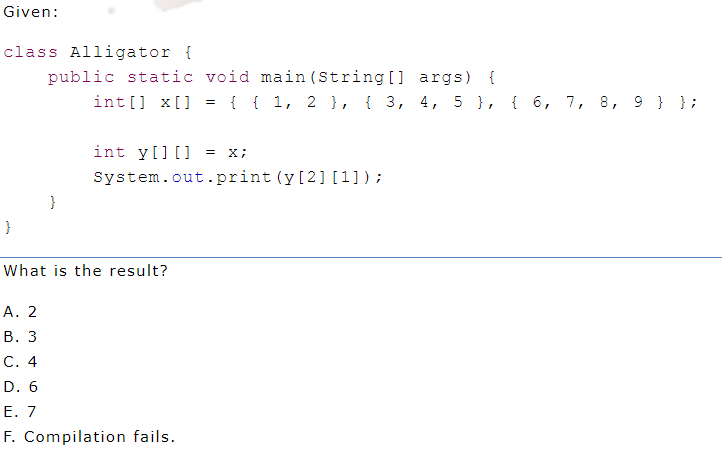
ANS: B

\*補充知識：^ XOR運算子規則





Ans: D



Ans: E int[][]的特殊寫法