Week 3

1. What is Encapsulation? Try to list out the benefits of Encapsulation.

   Encapsulation 翻譯為封裝，有點像把一個類型的架構或資料打包並遮蓋起來，koltin 的封裝就程度來說有分為四種：private, protected, internal, public，差別是封裝完後該資料能夠被不同層級取得的權限。

   1. **private**：只能在該類型內被存取。
   2. **protected**：可以在該類型的內部或該類型的子類型中被存取。
   3. **internal**：可以被該文件或同一個 Module 的文件存取使用。
   4. **public**：可以被任意存取。

   封裝的好處，第一個是隱私，不讓架構被看到。再來是安全，架構不會被別人修改。也有人說封裝起來明確限制了可以修改的範圍，對於維護而言反而更為便利。

8. Try to explain what enum is and why we use them?

   enum (enumeration) 翻譯為列舉，在想要重複表達固定幾種狀態時，可以用列舉宣告不同狀態的代號，例如：想吃雞肉飯的人是 1，想吃魯肉飯的人是 2，想吃雞魯飯的人是 3，有了代號後就可以直接從代號得出對應的狀態。如此一來有一個大優點是：省空間。因為可以用簡單數值取代字串，數值僅佔用非常少甚至 1 bit 的資料空間，當資料量龐大時就會有很大差異。

9. How to use enum in Kotlin? Attach a sample code that uses enum in Kotlin.

```
enum class wantToEat(val status : Int) {
  雞肉飯(1),滷肉飯(2),雞魯飯(3)
}
```

10. What are the differences between LinearLayout and RelativeLayout? Try to explain in detail.

    LinearLayout 是以線性順序去做介面編排，由上至下，由左至右，將整個介面想像成一個外框，裡面再放入各種小框為物件，以 height 和 width 這兩個屬性決定外框大小，而小框的編排順序只要拉動 Component Tree 裡面的物件，就可以做調換或更動。RelativeLayout 則是以物件之間的相對位置來排列和決定彼此的順序，且最先宣告的物件會放在畫面的左上方， 若以 A 為第一個宣告的物件，則以 B 在 A 的右方多遠、C 在 A 的右下方多遠這樣的方式來做介面編排。

    兩者在介面編排上的出發點不太一樣，前者是一層一層的，後者是像樹圖以方向為線條多方發展。個人認為若以圖片區塊為主體的畫面，適合用 LinearLayout 來編排，但若是文字排列較多且有順序性，也沒有明顯區塊的畫面，則用 RelativeLayout 的編排方式可能更適合。

11. Try to explain the benefits of ConstraintLayout. Why should you use it?

    ConstraintLayout 算是結合了 LinearLayout 和 RelativeLayout 兩者的特色，以單一物件對於四個邊的距離為編排邏輯，且直接在 Design page 上用拖曳的方式操作更方便，單一物件設定四個方向的位置有點類似於 LinearLayout 在設定區塊外框的感覺，但同時在 A 物件想要跟隨 B 物件連動這樣的狀況下時，又可以用類似於 RelativeLayout 的物件相對關係去編排，因此靈活性比上述兩者更高。

    另外官方數據表示，ConstraintLayout 可以見少 LinearLayout 搭配 RelativeLayout 的階層數，對於運行的速度能提高 40% ！？