***Q1***

**Encapsulation -封裝**

把功能與資料包裝起來成為一個物件，並定義物件的介面。讓外界的程式，透過物件所提供的介面，來與這個物件溝通。

優點:

1. 模組化 : 可將系統細分然後分配給不同團隊來進行工作
2. 程式再用 : 物件、類別可以重複的使用，不用再撰寫新的程式
3. 降低風險 : 程式可重複使用，減少撰寫程式錯誤的風險

***Q2***

**enum-列舉**

一般用來表示一組相同類型的常量。如性別、日期、月份、顏色等。

列舉能增加程式的可讀性，加速工程師對程式碼的理解且所占的記憶體小。

***Q3***

enum class Colors {

RED, YELLOW, BLUE, BLACK

}

fun main() {

val c01: Colors = Colors.RED

val c02: Colors = Colors.BLUE

val c03: Colors = Colors.BLACK

println("$c01, $c02, $c03")

}

***Q4***

**LinearLayout–線性布局**

線性可分為水平（horizontal）及垂直（vertical），在簡單的版面內較好用，除了線性的排列，還有weight可以讓 View依照比例佔滿整個水平方向或是垂直方向。

**RelativeLayout–相對布局**

可以透過相對位置來設定各個元件的位置，RelativeLayout 中的所有 View 都有相對關係，也因此較能處理處理各種解析度，避免A手機畫面正常，到B手機版面卻跑掉的情況發生。

***Q5***

**ConstraintLayout**

類似RelativeLayout，很適合用可視化的Design直接對各個元件進行拖移來做頁面編輯。

優點:

1. 布局扁平化，避免套用多個View層級
2. 執行性能提升!