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

Encapsulation(封裝)在物件導向程式語言中有兩種概念，一是如同其中文直譯，指的是將data及處理該data的method，綑綁封裝在一個單位裡面；另一個則是指限制方向的存取概念。

以Kotlin舉例的話，封裝的情況將會是，一個Class裡面有著variable及調用該variable的method，並在variable或method的前面，可能加上如private、protected等modifier來限制存取。

封裝所帶來的好處有：

1. 開發時將不同的功能封裝至不同的Class，我們的程式碼將更加地好讀。
2. 避免物件被其不希望的外部client存取。
3. 對client來說，將不會看到物件內部的全部細節，可以降低人為錯誤並增進開發效率。
4. **Try to explain what enum is and why we use them.**

Enum是一個擁有預先定義物件的Class，且這些物件數量是固定的。他能夠讓內部物件更好地被描述及閱讀，同時也是一種資料類型。

我們之所以使用Enum，最主要的原因就是因為他大大地提升了可讀性，避免了程式寫到後面越來越大，自己定義的東西太多太雜而不易閱讀。詳細案例將在第三題舉例。

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

在這邊我們將When Expression與Enum一起使用，進一步描述Enum的優點。這裡以flyDirection這個variable來控制要使用何種fly method，可以看到這樣的方式讓這個控制飛行方向的程式非常好懂。

1. enum class Direction {UP, DOWN}
2. fun flyUp() {}
3. fun flyDown() {}
4. fun main() {
5. var flyDirection = Direction.UP
6. when(flyDirection) {
7. Direction.UP -> flyUp()
8. Direction.DOWN -> flyDown()
9. }
10. }
11. **What are the differences between LinearLayout and RelativeLayout? Try to explain in detail.**

Android的畫面是由一個個的Views所組成，而這些Views將在ViewGruop裡組成一個群組，這些群組即組成畫面的最基礎單位。

在Android裡ViewGroup提供了多種Layouts，Linear Layout、Relative Layout、Constraintlayout、TableLayout、AbsoluteLayout、FrameLayout等。於此我們將討論LinearLayout及RelativeLayout。

LinearLayout將內部的Views進行垂直或水平排列成一排；RelativeLayout則是透過相對位置來設定Layout內各個Views的位置。

兩者皆是常用的Layout，最大的區別在於排版的風格。隨著App畫面的不同我們應該適當運用不同風格的Layout。在如登入畫面時，需要將Views如帳號、密碼及信箱等Edit Text直線排列的時候，LinearLayout會是一個好的選擇。然而在遇到較複雜排版的時候，若是使用多個LinearLayout去構成畫面將是很沒效率的，這時候就是使用RelativeLayout的最好時機。當然我們也可以將兩者一起使用，或嵌套對方，這一切都將視畫面的不同而決定。

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

如同第四題所提到的，在建構複雜的畫面時，我們會將LinearLayout及RelativeLayout一起使用，雖然能夠將畫面建構出來，但是將兩者組合、嵌套將是非常費工夫的事情。這時候ConstraintLayout的優點就顯示出來了，不需要將東西嵌套也可以完成這些複雜的畫面，而一些畫面的小細節，ConstraintLayout也提供了更好的工具，如Baseline、Chain等來幫助我們完成。

除處理複雜的畫面外，ConstriantLayout還有許多優點，他是透過拖放的方式來建構UI，在操作上十分直觀且順手。且因為它提供了較扁平的View Hierarchy，因此在表現複雜的畫面上也有著更好的效能，以上種種因素即我們之所以使用ConstraintLayout的主要原因。