1. **What is Polymorphism? Try to explain in Mandarin.**

Polymorphism是某個東西擁有多種意義的意思，而這個東西的意義是由他的接收者所決定。又可分為兩種類型，Compile-time Polymorphism及Runtime Polymorphism。

Compile-time Polymorphism指的是當有多個method擁有相同名稱時，compiler會依據傳入的參數類型自動地決定我們要呼叫的method來進行compile。此概念的另一面即Method Overloading。

Runtime Polymorphism則是發生在不同Class間有繼承關係時。子類型可以override父類型現有的method，由子類型去決定原本的這個method該以怎麼樣的形式呈現。這種應用增加了程式的彈性，亦節省了我們編寫的時間。

1. **Here are the 7 important lifecycle methods of an Activity. Try to explain when they are called during the lifecycle of Activity.**

i. onCreate() => 當系統初次創建Activity時呼叫。

ii. onStart() => 在創建完成後，Activity會從Create狀態進入Start狀態，這個時候系統將會呼叫onStart。

iii. onResume() => 一旦onStart完成，Activity會進入Resume狀態，這個時候系統將會呼叫onResume。

iv. onPause() => 當Activity不在位於前台時(仍可見)被呼叫。

v. onStop() => 當Activity不可見時呼叫。

vi. onDestroy() => 當Activity被Destroy時呼叫，會被Destroy有兩種原因，Activity即將結束或Configuration Change。

1. **What is the Android Jetpack?**

Android Jetpack是一組library、tools及architectural guidance，它提供了良好的開發基礎，使得開發者能夠基於這個基礎而更專注在開發上。而Jetpack Components又由四種分類組成，Foundation、Architecture、Behavior及UI。

* Foundation Components提供了向後兼容及測試功能，並支援Kotlin語言。
* Architecture Components功能使得我們的App更強壯，並增加可維護性及可測試性，如提供Data Binding、Lifecycle及LiveData等。
* Behavior Components幫我們整合了標準的Android服務，如Notifications、Permissions、Sharing等。
* UI Components則提供了widgets及helper，如Fragment、Layout等，使得我們的App使用起來簡單且輕鬆。

1. **What is Coroutines? Why do we use it? Try to explain in Mandarin.**

在說明Coroutine前必須先理解一個概念，就是程式在執行時可分為同步及非同步。同步指的是所有執行的任務都在一條Thread上執行，也因此每個任務必須排隊等前一個任務完成後，才能夠被執行。這樣會導致一種情況，如果當中某個任務需要執行一個小時，那麼在他後面的任務就需要等到一個小時後才能被執行，這樣的方式無疑是有問題的，而非同步就是為了解決這個問題而存在，它將某些任務分配到另一個Thread上以避免過長的任務影響到使用者體驗。

為了完成非同步我們會需要建立新的Thread並使用Callback將結果引用給下一個被調用的method使用。但是這樣會導致Callback hell(看起來很酷的側V形)及控制權移轉的問題，也因此我們需要Coroutine來解決這個問題。

Coroutine由CoroutineScope、Suspend function及Dispatchers構成。它的作用與Thread + Callback類似，我們可以將其理解成輕量級的Thread。Coroutine位於Thread的內部，將Thread分為更小的單位，透過多個Coroutines我們可以達到非同步執行任務的目標，並在編寫時解決了Callback hell及控制權移轉的問題，讓程式碼更加簡單且好懂。