**抽象類別與介面的差異**

**抽象類別**

1. 抽象類別的用途是在於該類別必須衍生子類別, 才能產生物件的場合。
2. 如果只是要防止建立特定類別的物件, 應該改成只為該類別定義一個private存取控制且不需參數的建構方法。
3. 這在該類別僅是作為提供 static 方法給其他類別使用、或是需在初始化時自動產生物件時特別有用。
4. 抽象類別不一定有抽象方法，但如果有抽象方法表示繼承的子類別必需照其規格去實作。

第2項跟第3項合起來是一個常見的Design Pattern叫做Singleton

<https://www.javaworld.com.tw/jute/post/view?bid=29&id=320334>

**介面**

介面全都是抽象方法，程式預設就是public abstract(不寫也沒差)。

介面可以讓毫無關係的類別來實作同一個介面。

**1.**

interface的方法全都是抽象方法  
abstract則有一般方法和抽象方法

**2.**

interface與abstract類別都不能直接用來建立物件實體，

必須由一個標準類別來繼承它，實作其抽象方法，

然後再以此建立物件實體。

**3.**

interface可多重繼承interface

abstract只能單一繼承

**4.**

interface沒有建構子

abstract有建構子

**5.**

interface沒有內建的super與this變數

abstract有內建的super與this變數  
**6.**

interface的成員變數全都是static final型式

abstract類別的成員變數比照一般類別

**7.**

interface內的方法只可封裝public、default

abstract內的方法只可封裝public、default、protected

抽象類別一定屬於繼承架構，而且一定是父類別，介面可以讓毫無關係的類別來實作同一個介面