方式一：繼承Thread類的方式：

1. 創建一個繼承於Thread類的子類
2. 重寫Thread 類的run() 🡪將此線程執行的操作聲明在run()中
3. 創建Thread類的子類對象
4. 通過此對象調用start()：1.啟動當前線程 2. 調用當前線程的run()

說明兩個問題：

問題一：我們啟動一個線程,必須調用start(),不能調用run()的方式啟動線程。

問題二：如果再啟動一個線程，必須重新創建一個Thread子類的對象，調用此對象的start()。不然會報IllegalThreadStateException異常。

方式二：實現Runnable接口的方式：

1. 創建一個實現了Runnable接口的類
2. 實現類去實現Runnable中的抽象方法：run()
3. 創建實現Runnable類的對象
4. 將此對象作為參數傳遞到Thread類的構造器中，創建Thread類的對象
5. 通過Thread類的對象調用start()

兩種方式的對比：

開發中：優先選擇：實現Runnable接口的方式

原因：1. 實現的方式沒有類的單繼承性的侷限性。

2. 實現的方式更適合來處理多個線程共享數據的情況。

聯繫：public class Thread implements Runnable

相同點：兩種方法須都要重寫run()，將線程要執行的邏輯聲明在run()中。

目前這兩種方式，想要啟動線程，都是調用的Thread類中的start()。

**Thread類中的常用的方法：**

1. start():啟動當前線程；調用當前線程的run()

2. run(): 通常需要重寫Thread類中的此方法，將創建的線程要執行的操作聲明在此方法中

3. currentThread():靜態方法，返回執行當前代碼的線程

4. getName():獲取當前線程的名字

5. setName():設置當前線程的名字

6. yield():釋放當前cpu的執行權

7. join():在線程a中調用線程b的join(),此時線程a就進入阻塞狀態，直到線程b完全執行完以後，

線程a才結束阻塞狀態。

8. stop():已過時。當執行此方法時，強制結束當前線程。

9. sleep(long millitime):讓當前線程“睡眠”指定的millitime毫秒。在指定的millitime毫秒時間內，當前線

程是阻塞狀態。

10. isAlive():判斷當前線程是否存活

線程的優先級：

1.

MAX\_PRIORITY：10

MIN \_PRIORITY：1

NORM\_PRIORITY：5 -->默認優先級

2.如何獲取和設置當前線程的優先級：

getPriority():獲取線程的優先級

setPriority(int p):設置線程的優先級

說明：

高優先級的線程要搶占低優先級線程cpu的執行權。但是只是從概率上講，高優先級的線程高概率的情況下被執行。並不意味著只有當高優先級的線程執行完以後，低優先級的線程才執行。

線程通信：wait() / notify() / notifyAll() :此三個方法定義在Object類中的。

補充: 線程的分類 : 一種是守護線程、一種是用戶線程。