

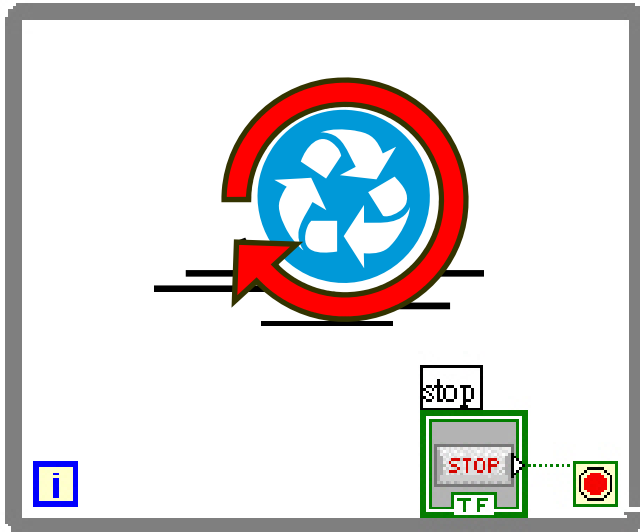
單元2：

重複及迴圈

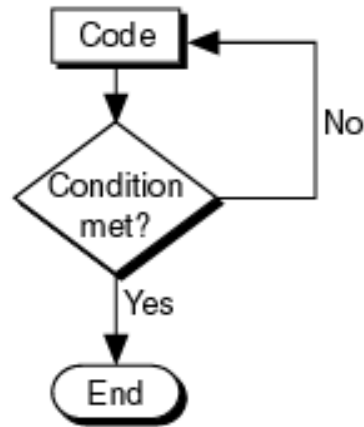
主題：

- a. While Loop
- b. For Loop

While Loops



LabVIEW While Loop



Flow Chart

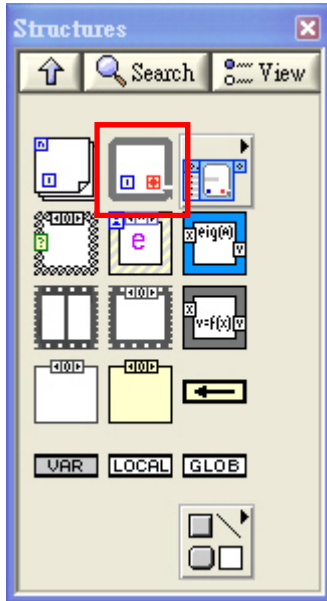
```
Repeat (code);  
Until Condition met;  
End;
```

Pseudo Code

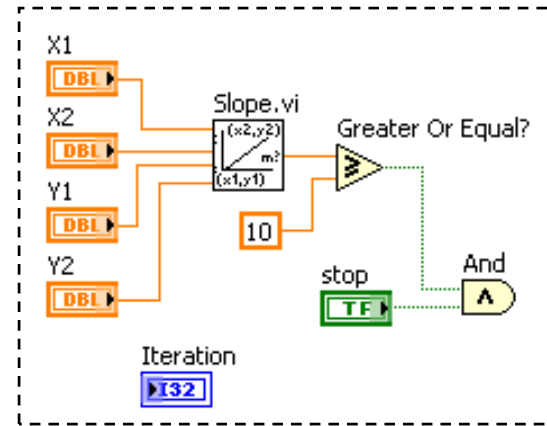
⚠ While Loop至少會執行一次

While Loops使用方法

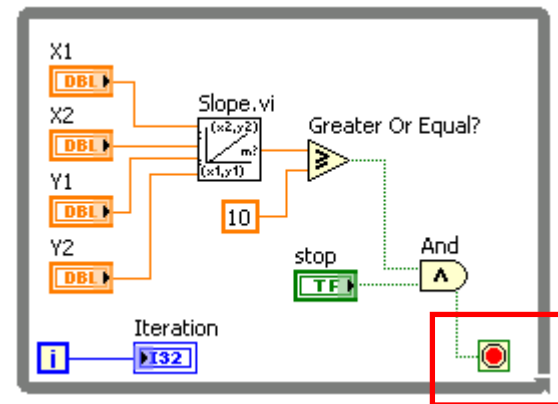
1. 選出While Loop



2. 用滑鼠框出欲重複進行的程式



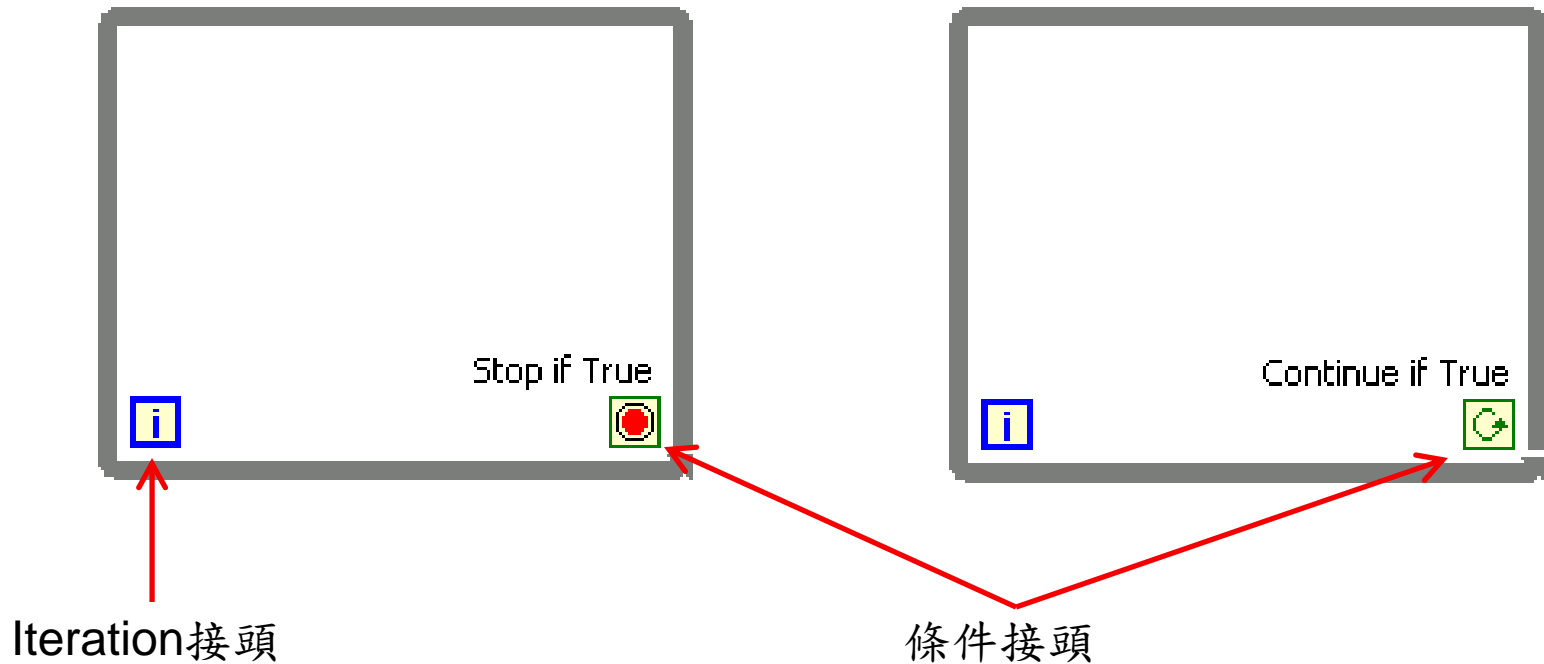
3. 完成剩下的部份



While Loop的停止條件

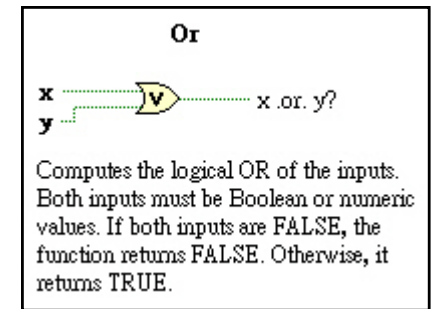
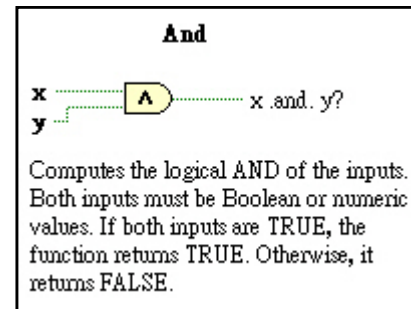
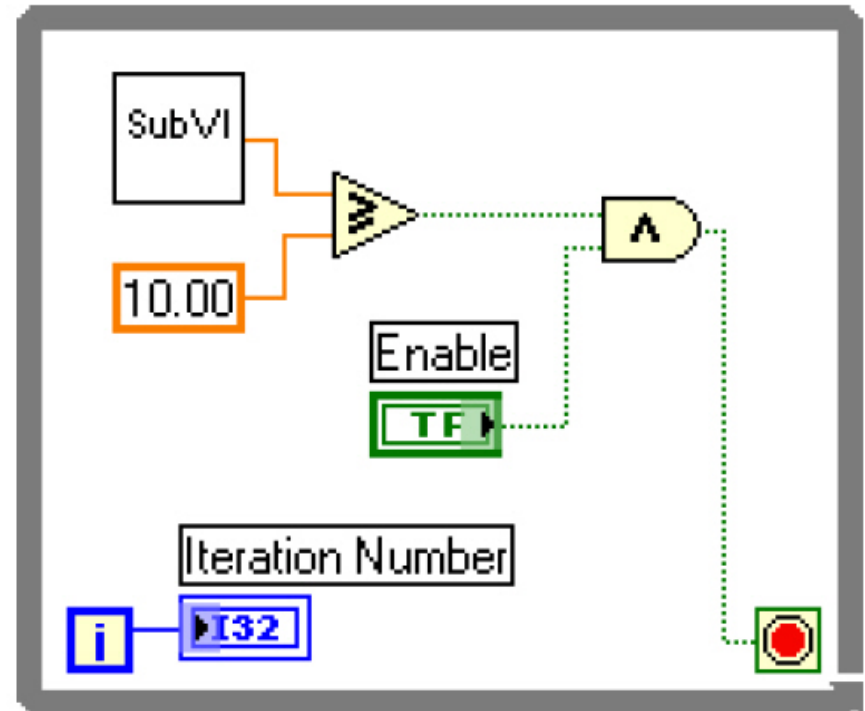
使用While Loop的時候，要設定While Loop的停止條件。

- 「Stop if True」：若True則停止迴圈
- 「Continue if True」：若True則繼續



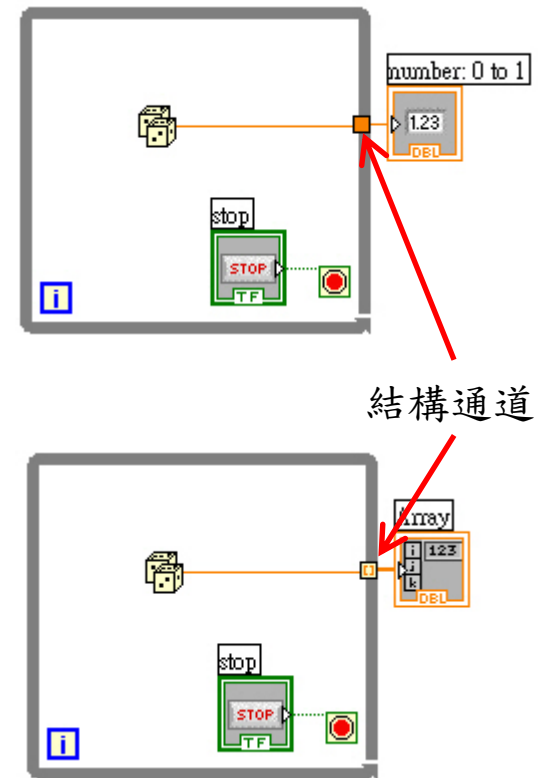
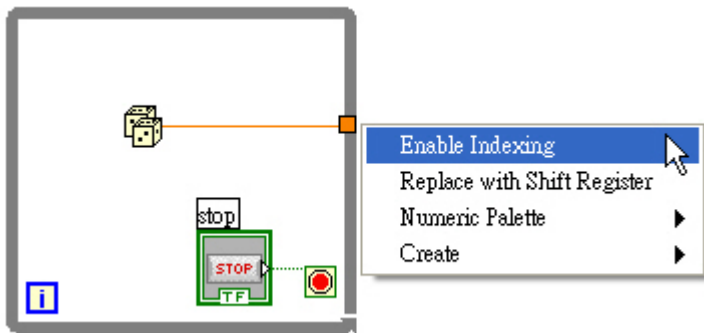
Example

- While Loop會不斷執行，直到subVI輸出大於或等於10的值，以及 Enable控制器為True。
- 此範例有可能造成OS資源滿載，加個Delay的功能即可改善
- 一般盡量避免While Loop沒有停止條件，以避免形成無窮回圈。



結構通道 (Structure Tunnels)

- 資料可以透過通道傳入或傳出 While Loop
- 若通道是實心，則傳出最後一筆資料；若通道是空心，則資料以 Array 形式全部傳出
- 只有當 While Loop 停止後，資料才會透過結構通道傳出去



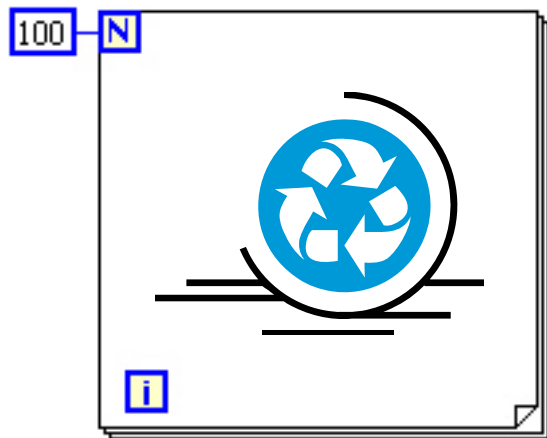
練習2.1：以While Loop作亂數產生圖

1. 開啟LabVIEW8.2，打開一個新的VI
2. 在人機介面拉出Waveform Chart
3. 在程式區拉出亂數產生器，並連接至Waveform Chart
4. 執行，並觀看結果
5. 將程式區的程式加入While Loop，在While Loop的停止條件處，按滑鼠右鍵，選擇「**Create » Control**」
6. 執行，並觀看結果
7. 在While Loop中加入Delay
8. 執行，並觀看結果

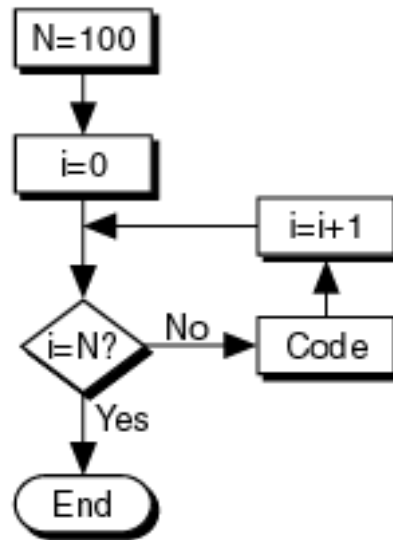
練習2.2：猜數字

1. 在程式區建立一個可以產生0~100之整數的亂數產生器
2. 建立While Loop，讓亂數一直產生
3. 停止條件設定為「當所產生的亂數等於20時，就停止迴圈」
4. 在「Iteration接頭」按滑鼠右鍵，選擇「**Create » Indicator**」，執行後就可以知道while loop跑了幾次
5. 修改while loop停止的條件，以及亂數產生的範圍，觀察while loop停止時所跑的迴圈數目有沒有不同
6. 把Iteration接頭的indicator放在while loop裡面跟while loop外面分別有什麼不同。為什麼？你較喜歡哪一種呈現方式？

For Loops



For Loop



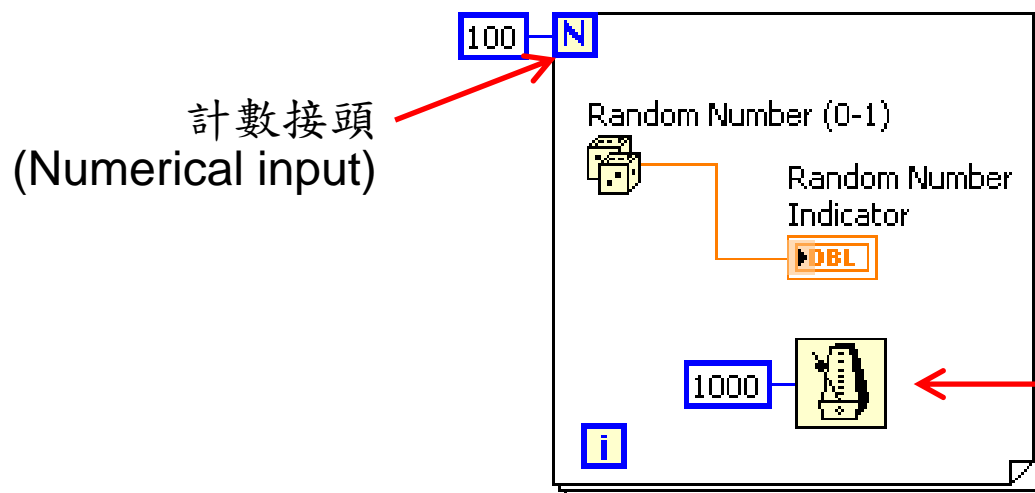
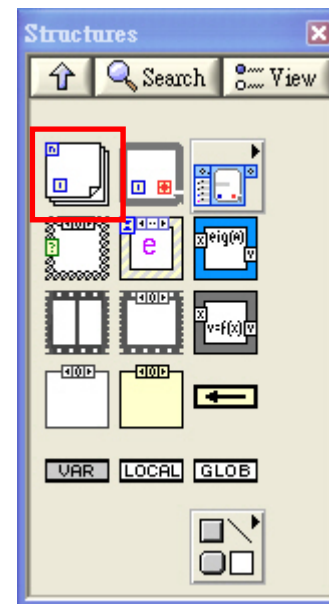
流程圖

```
N=100;  
i=0;  
Until i=N:  
    Repeat (code; i=i+1);  
End;
```

程式碼

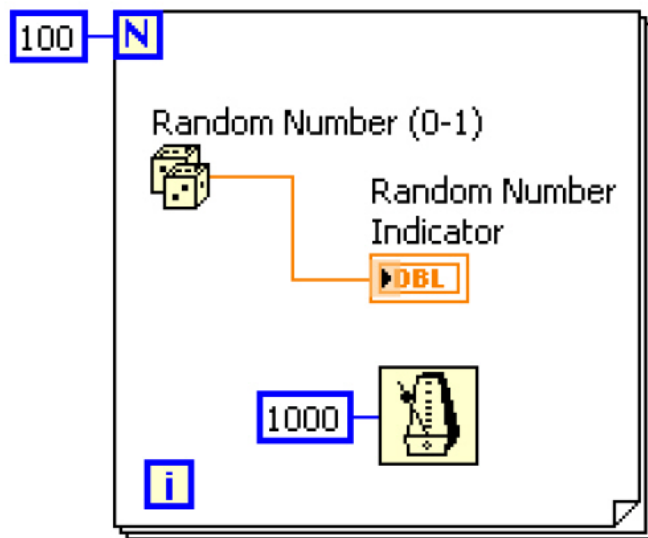
For Loops

- For Loop位於「**Functions»All Functions»Structure**」面板
- **i** Iteration接頭：顯示迴圈執行的次數
- **N** 計數接頭：For Loop執行的次數




練習2.3 – 產生亂數

- 以For Loop製作一個亂數產生器，亂數每1秒鐘產生一次，持續60秒。




Random Number (0-1)



number (0 to 1)


Produces a double-precision, floating-point number between 0 and 1, exclusively. The distribution is uniform.

Wait (ms)

milliseconds to wait 

Waits the specified number of milliseconds and returns the value of the millisecond timer. Wiring a value of 0 to the **milliseconds to wait** input forces the current thread to yield control of the CPU.

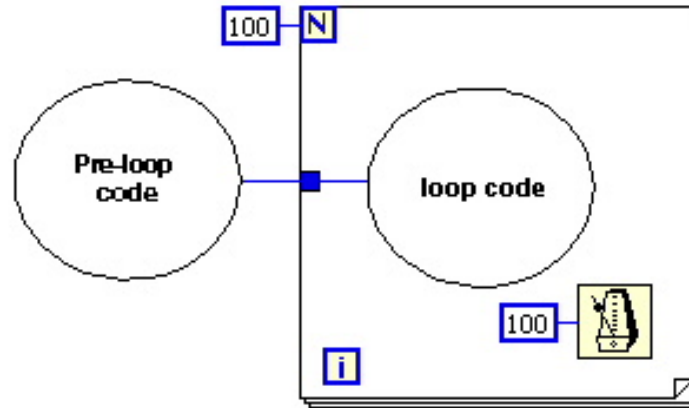
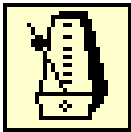
Wait Until Next ms Multiple

millisecond multiple 

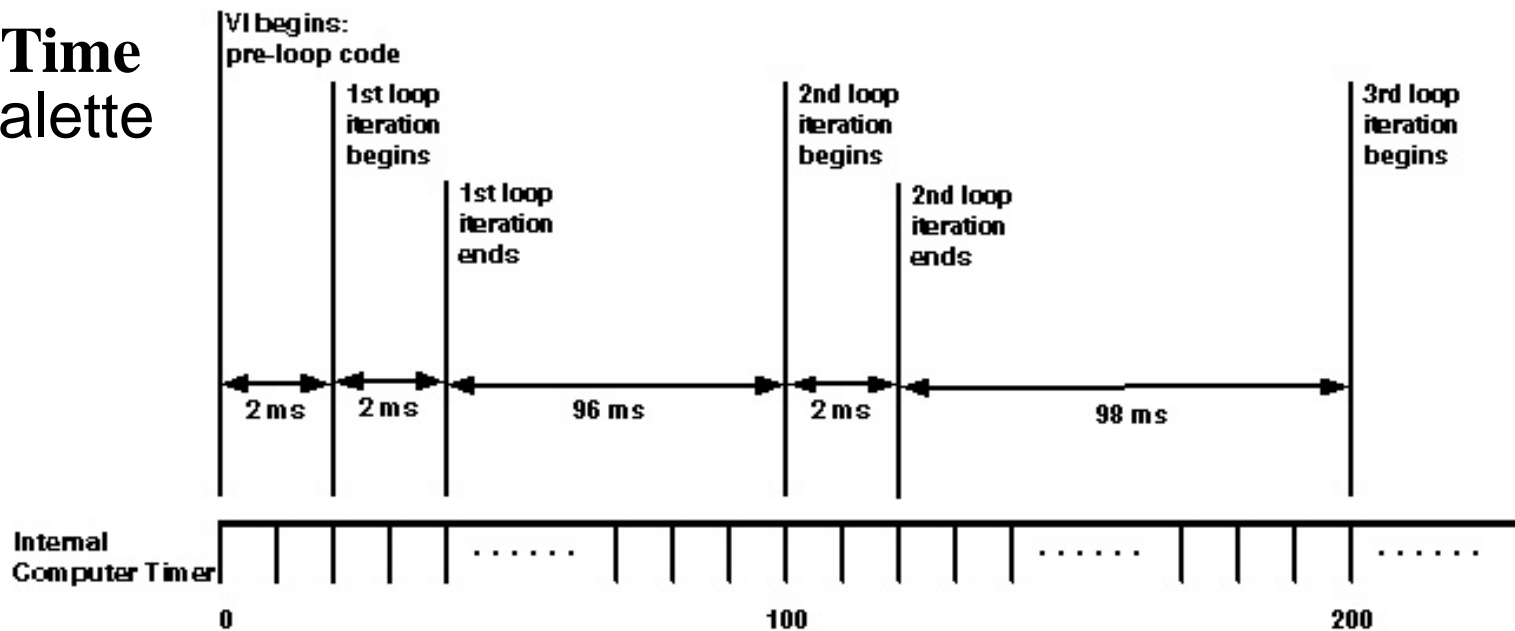
Waits until the value of the millisecond timer becomes a multiple of the specified **millisecond multiple**. Use this function to synchronize activities. You can call this function in a loop to control the loop execution rate. However, it is possible that the first loop period might be short. Wiring a value of 0 to the **millisecond multiple** input forces the current thread to yield control of the CPU.

Wait Functions

Wait Until Next ms Multiple

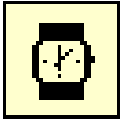


Functions»Time & Dialog palette



Wait Functions

Wait (ms)

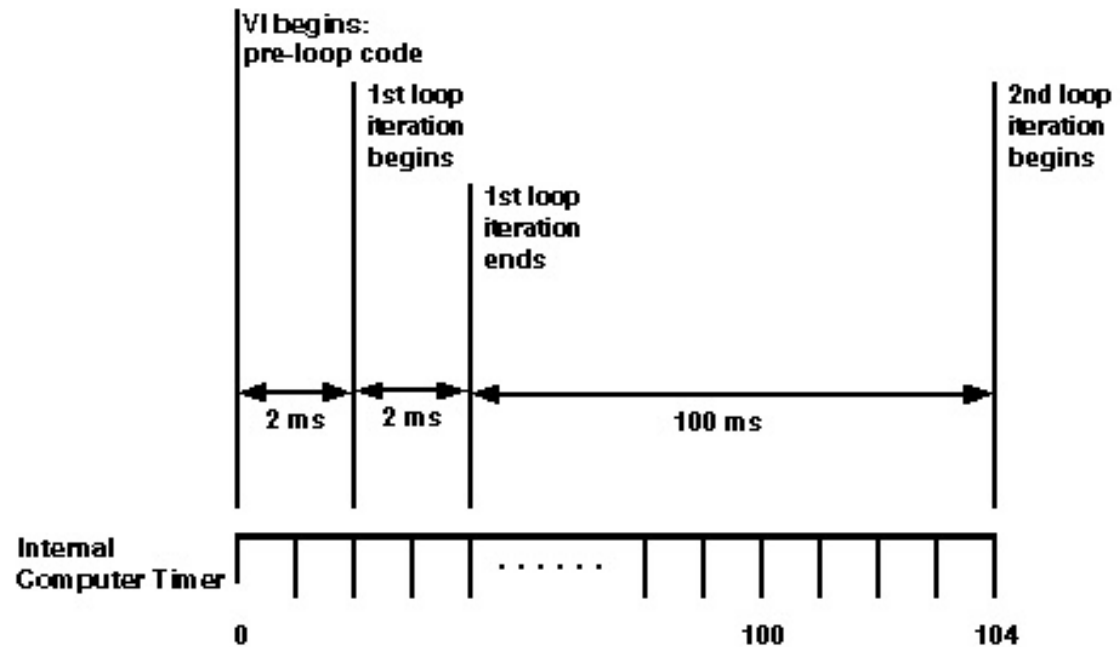
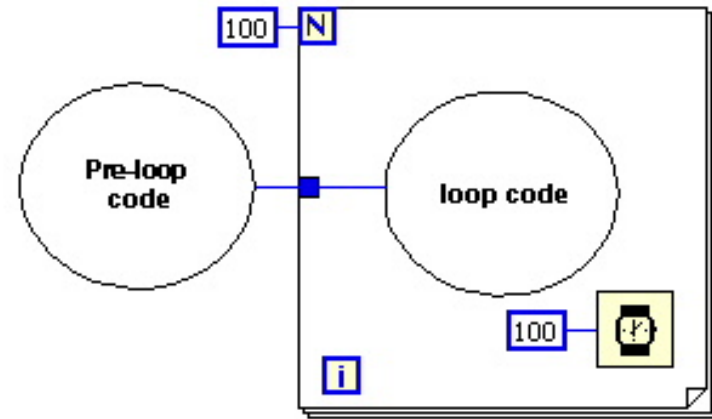


Functions»Time & Dialog palette

Time Delay

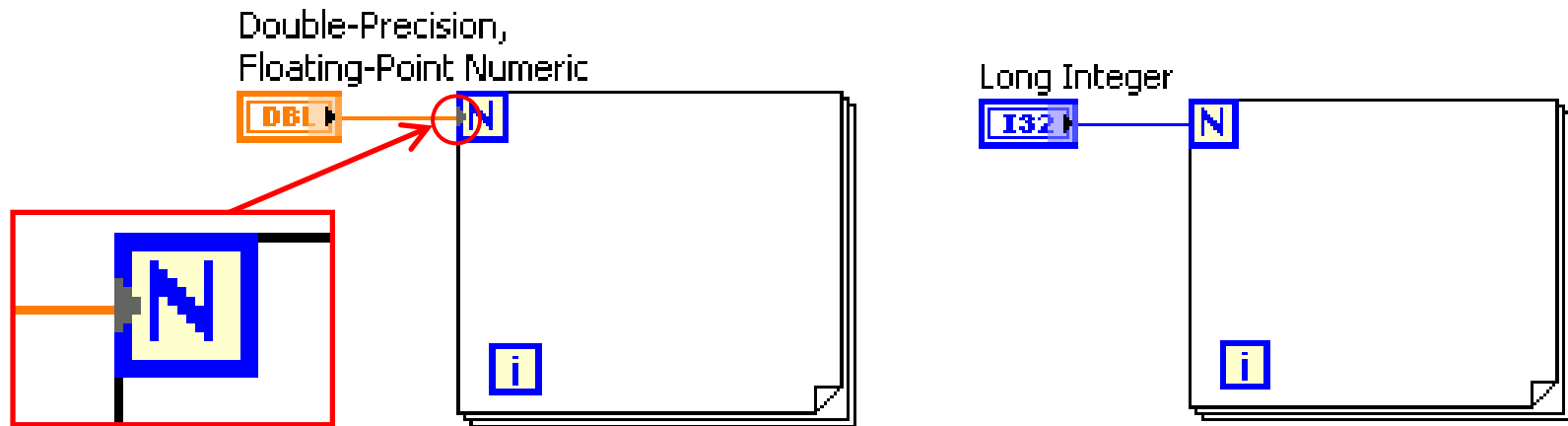


Functions»Time & Dialog palette



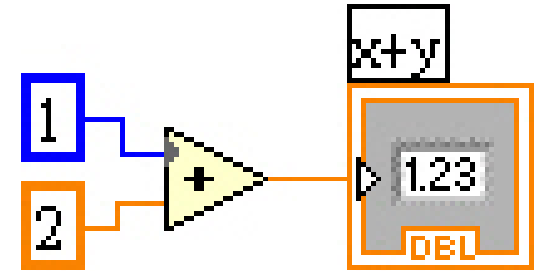
數值轉換 (Numeric Conversion)

- 當兩個以上不同格式的數值連接到函數上時，LabVIEW最自動轉換為較長或較大的格式。
- For Loop的計數接頭會自動轉換成「長整數」
- 如果接頭出現簡約點(coercion dot)，表示有經過數值轉換

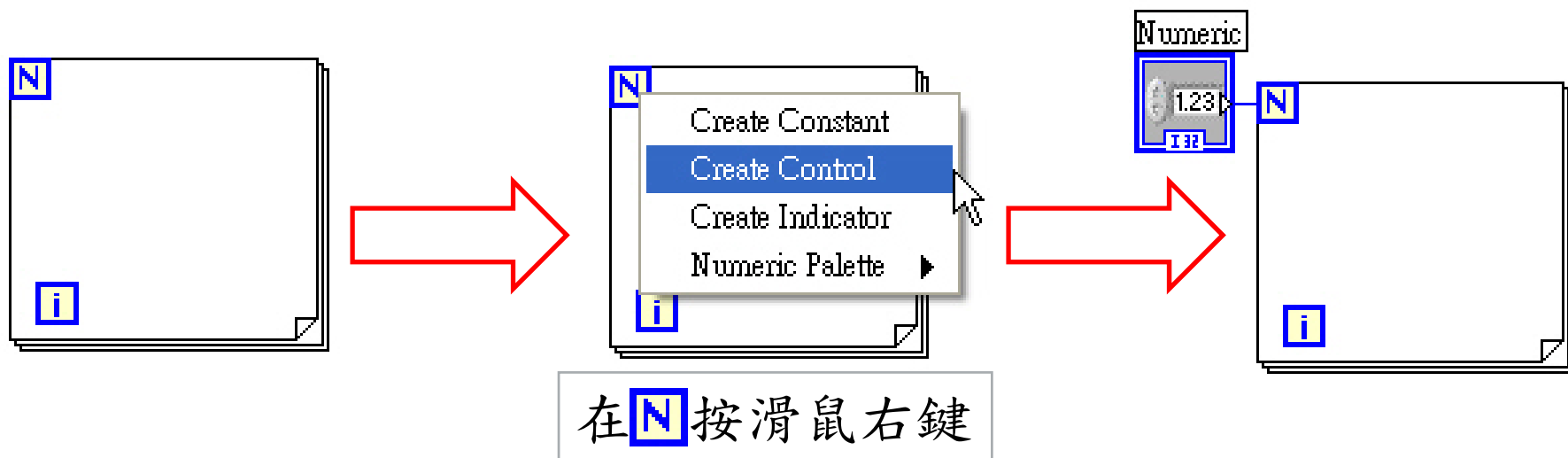


數值轉換 (Numeric Conversion)

- LabVIEW 選擇使用較多位元的資料格式
- 如果位元數相同，LabVIEW 會選擇無正負符號位元者，而不用有正負符號位元者。
- 要改變數值物件的格式，請在物件上按滑鼠右鍵，再從捷徑選單中選擇 **Representation**。選擇最適合呈現資料的資料類型。
- 當LabVIEW 將雙倍精密度浮點數值轉換為整數時，它會簡化為最接近的整數。LabVIEW 將x.5 簡化為最接近的偶整數。舉例來說，LabVIEW 將2.5 約化為2，將3.5 約化為4。



數值轉換 (Numeric Conversion)

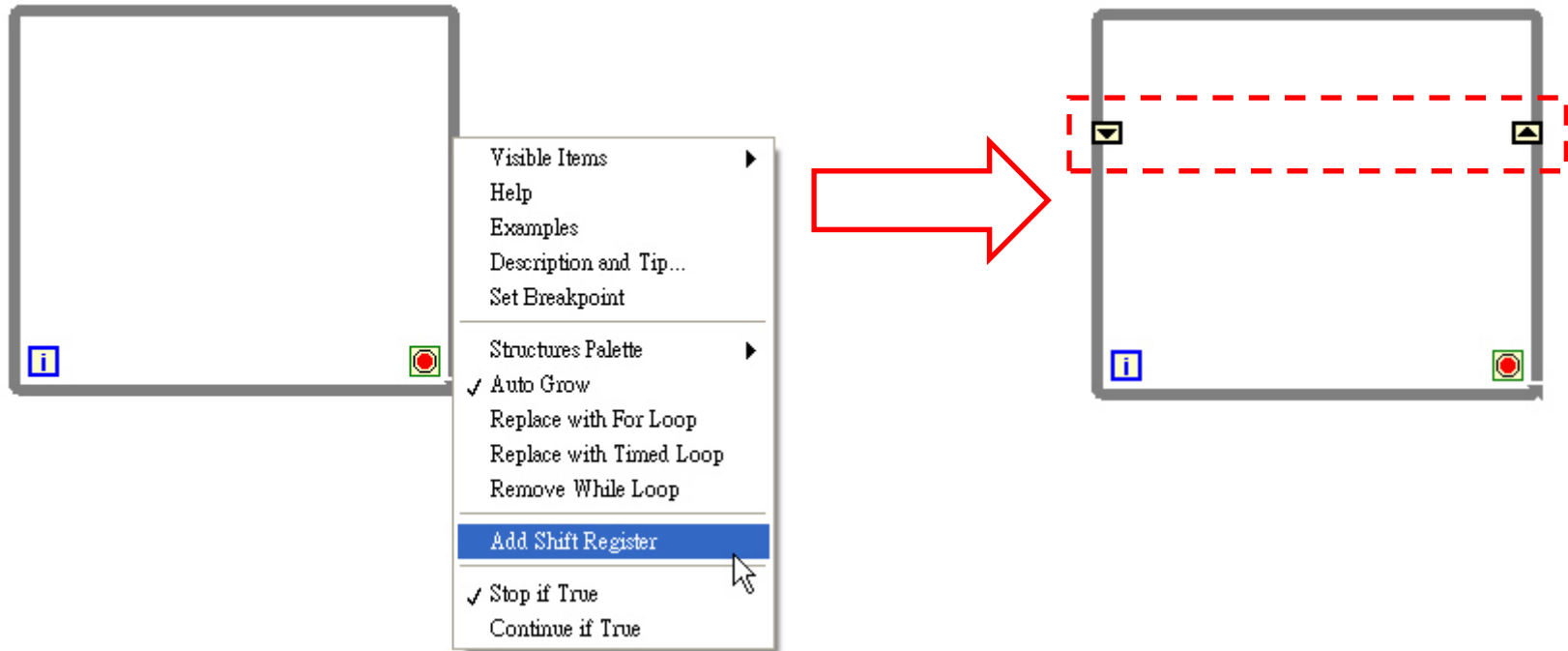
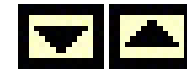


- 如果給定不相符的資料格式，那麼LabVIEW將自動轉換，但會消耗電腦資源。所以盡量避免。
- 若對不熟析，可以使用「按滑鼠右鍵»Create Control」，此時就會產生相符資料格式的control或indicator

練習2.4 – 每秒讀取溫度一次，持續30秒

1. 在程式區點選「select vi...」，拉進「<CD>\Ch2\Thermometer\Thermometer.vi」
2. 建立一個for loop，把for loop執行的次數設定為30次
3. 建立delay的函數，Delay時間為1000ms
4. 在「Iteration接頭」按滑鼠右鍵，選擇「Create»Indicator」，執行後就可以知道for loop跑了幾次
5. 把溫度的資料接到for loop的邊框，形成結構通道，在經過30秒後可以顯示之前這30秒所讀取的所有溫度值

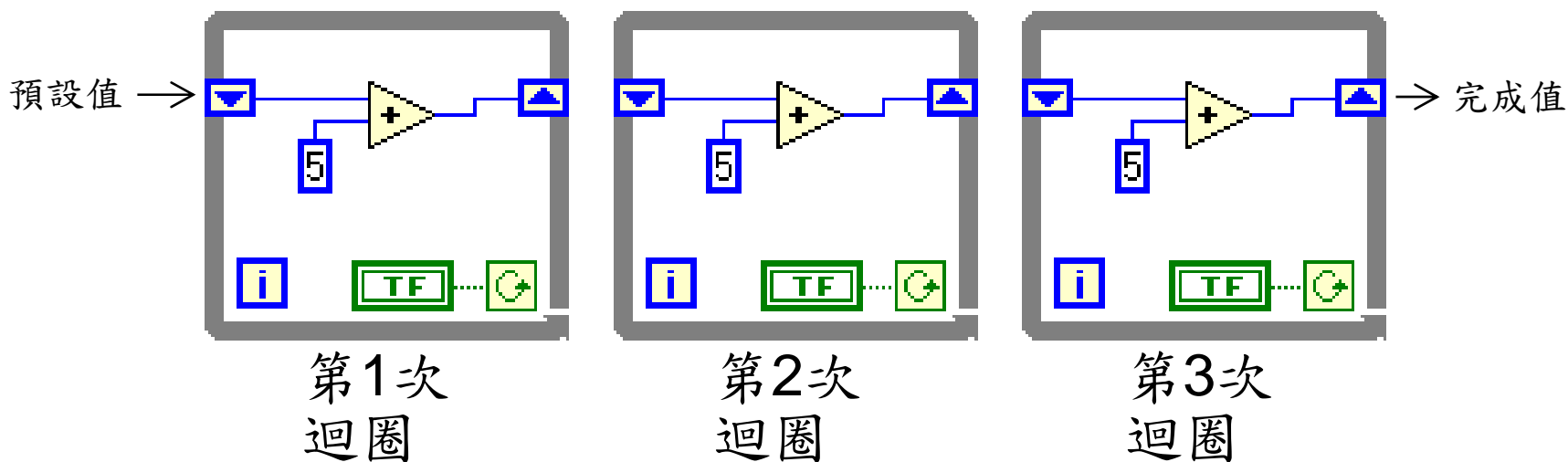
移位暫存器 (Shift Register)



- 迴圈連續執行時，若要取得上一次迴圈的執行結果，則要使用移位暫存器(shift register) 以或回饋節點(Feedback Node)
- 在While Loop按滑鼠右鍵，選擇「Add Shift Register」

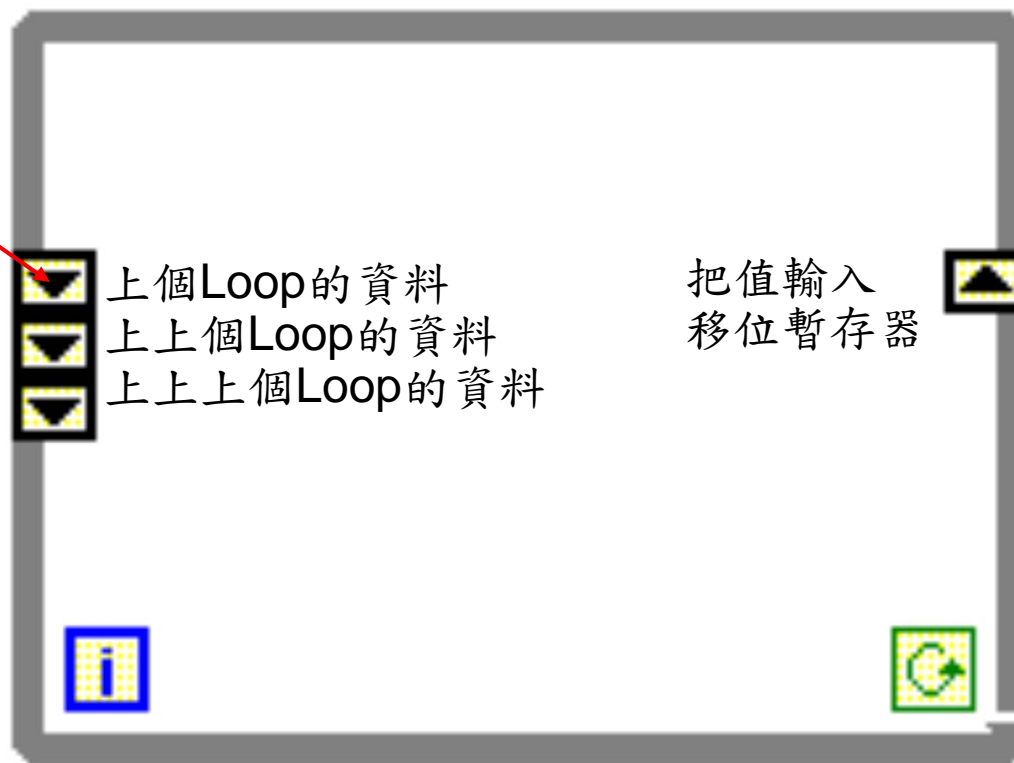
移位暫存器 (Shift Register)

- 移位暫存器可以接收任何資料型態
- 資料從移位暫存器的左邊傳送到右邊，迴圈結束後，資料再被傳遞到左邊



堆疊移位暫存器 (Stacked Shift Register)

用拖曳方式
新增堆疊位
移暫存器



⚠ 堆疊移位暫存器只能發生在迴圈的左側

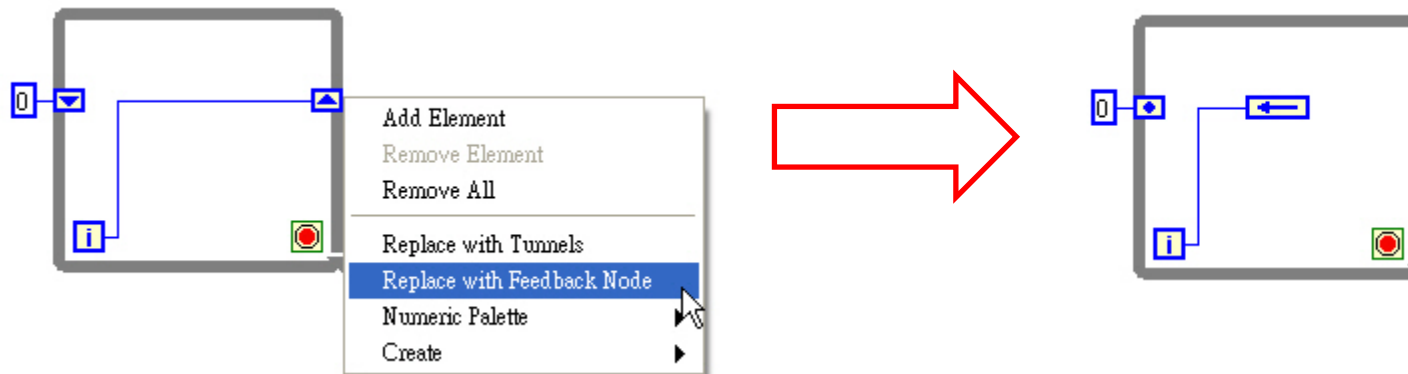
練習2.5 – 算出6!

1. $6! = 1 * 2 * 3 * 4 * 5 * 6$
2. 用迴圈方式計算6!
3. 那麼，算出6!後，請計算10!
4. 你的程式可以算出100!嗎? 為什麼不能呢?
5. 存檔

回饋節點 (Feedback Mode)



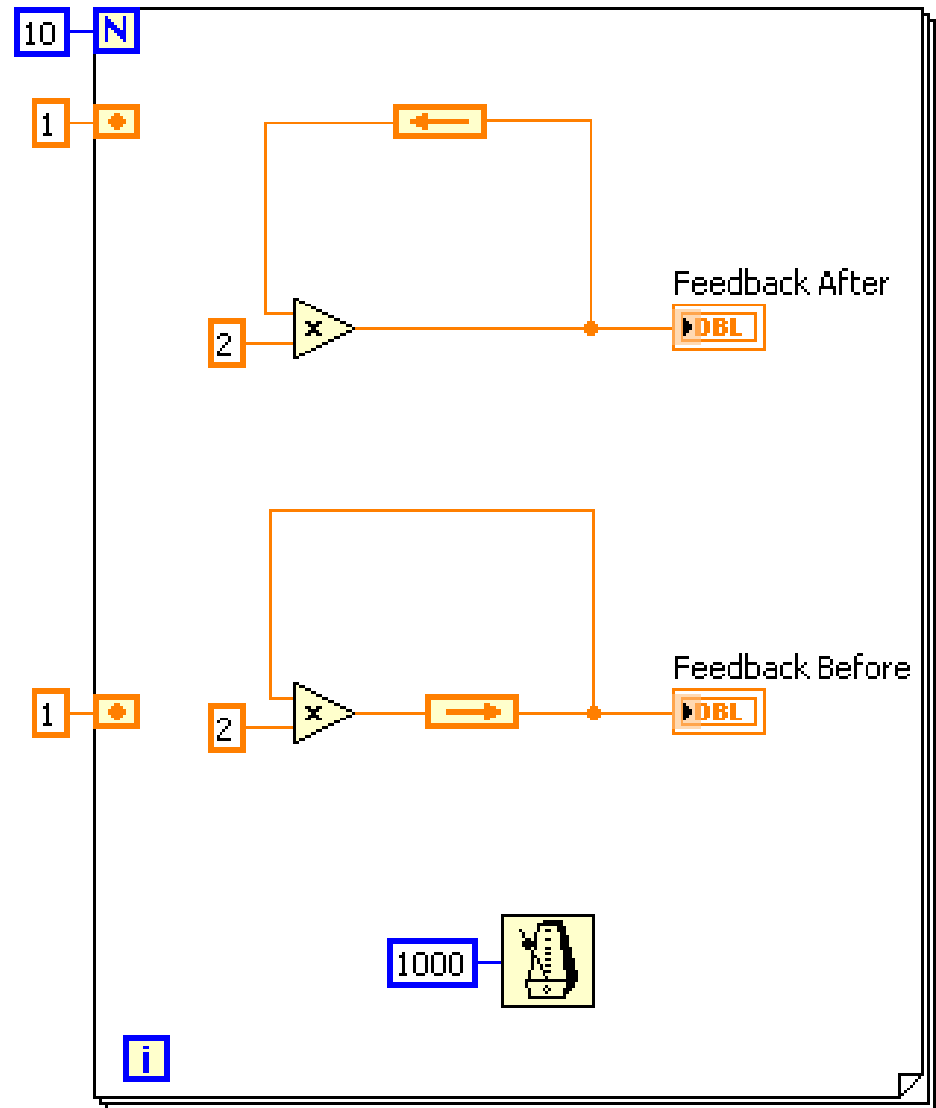
- Feedback Node的功能與Shift Register的功能相同，但可以避免接線過於複雜的缺點
- Feedback Node只能使用於While Loop與For Loop中



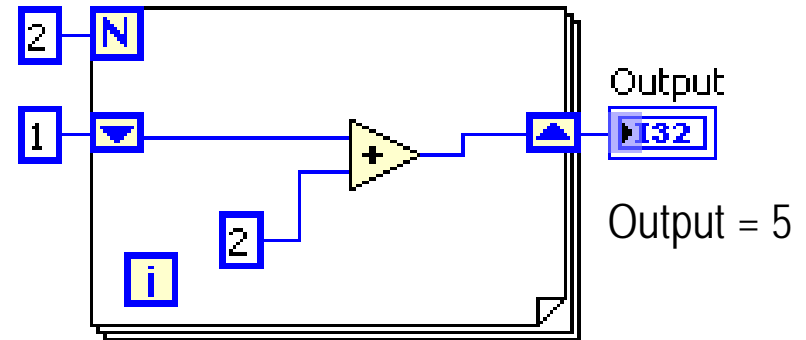
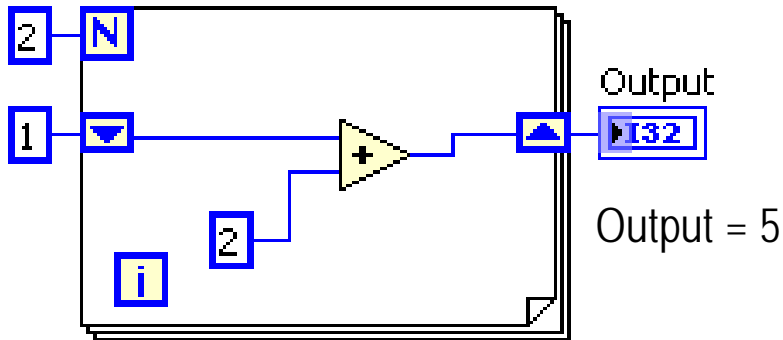
Feedback Node

■ 呼叫方式

- 在 Shift Register 按滑鼠右鍵，選「**Replace With Feedback Node**」
- 直接拖曳「**Structures»Feedback Node**」
- 在 While Loop 或 For Loop 中，把輸出端連線到輸入端，就會自動產生 Feedback Node



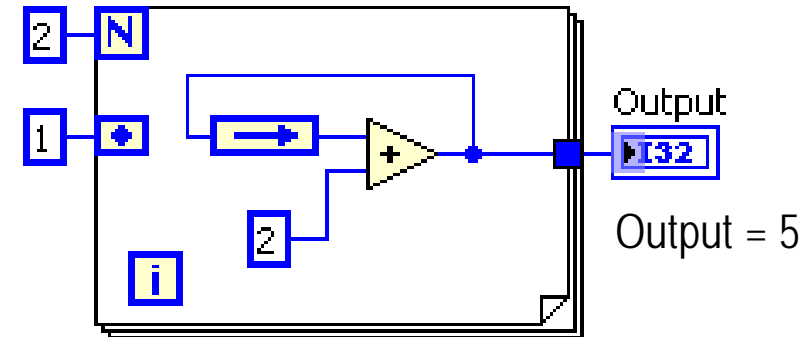
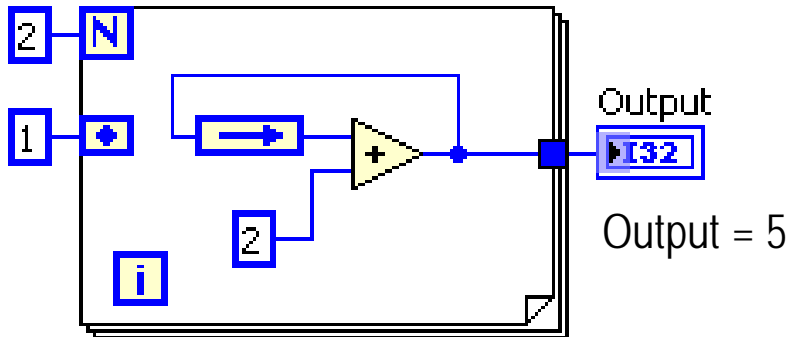
Shift Registers 與 Feedback Nodes 有初始化



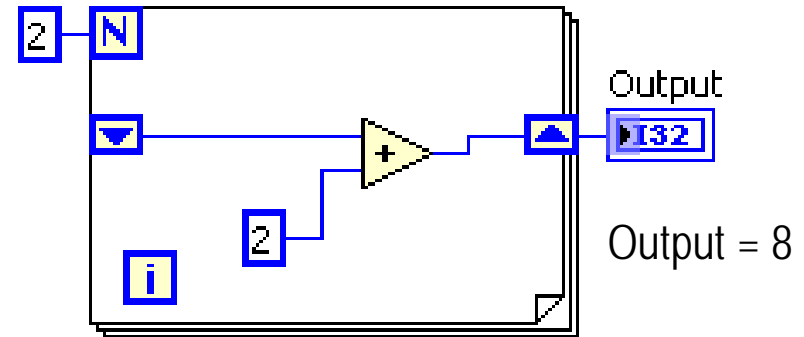
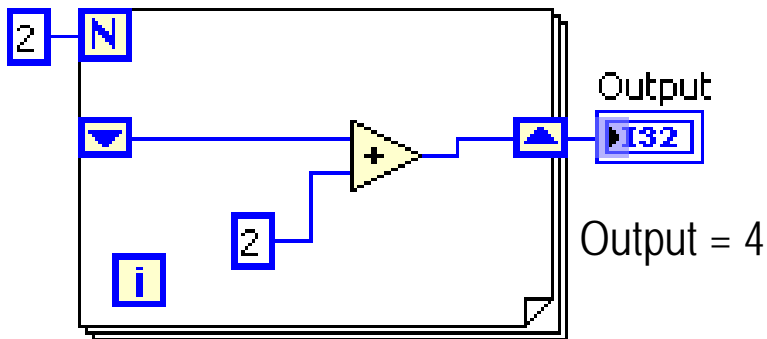
Run Once

VI stops execution

Run Again



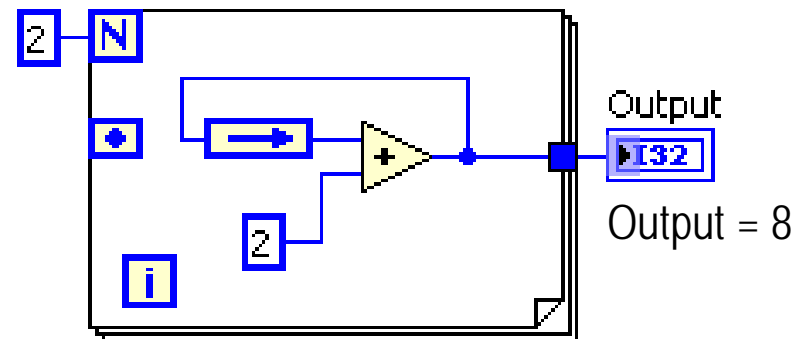
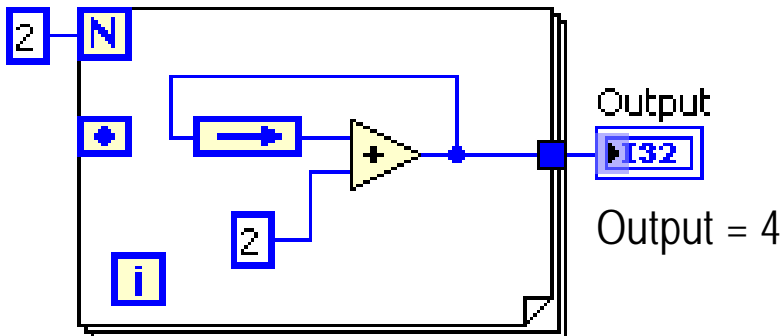
Shift Registers 與 Feedback Nodes 無初始化



Run Once

VI stops execution

Run Again



練習2.6 – 練習使用Feedback Node

1. 開啟Exercise2.4所存的檔案
2. 把Shift Registers置換成Feedback Node
3. 存檔

練習2.7 – 讀取溫度的移動平均值

■ 移動平均：

1. 以「 <CD>\Ch2\Thermometer\Thermometer.vi 」讀取溫度
2. 每一次的溫度讀值都要跟前兩次的溫度，共三次溫度值一起平均
3. 把有平均與沒有平均的數值畫在waveform chart上。

本章重點回顧

- 兩個用來重複執行的功能：While Loop and For Loop
- 時間的延遲可以使用：Wait Until Next ms Multiple function、Wait (ms) function或Time Delay Express VI.
- 當LabVIEW自動進行數值轉換時，接點處就會出現：簡約點。那是一個灰色的點，並會消耗系統資源。
- 我們可以使用Feedback nodes和Shift registers來把上一個迴圈的資料傳給下一個迴圈。
- 回饋節點在迴圈完成一次執行時，儲存其資料，將該資料傳送至下一次執行的迴圈，而且可以傳送任何資料類型。
- 使用回饋節點來避免不必要的長接線。

觀念

- 哪一種迴圈至少會跑一次? While Loop, For Loop
- 以下的情況，要用while loop還是for loop?
 - 在一秒鐘，每隔一秒讀取溫度一次，為期1分鐘
 - 讀取溫度，當溫度高於50度C就停止讀取
 - 持續讀取溫度和壓力，直到溫度與壓力都達到穩定狀態達3分鐘
 - 輸出電壓，從0伏特到5伏特，每秒升幅0.5伏特