

馬達與感測器教具平台

運算思維與程式設計 演算法及流程圖

公版教材

2019.09.27高師大自造者基地編輯

大綱(I)

1、邏輯及程式語言簡介

- 1.1 學習程式語言的目的
- 1.2 情境任務 - 以Google地圖為範例
- 1.3 人類的邏輯
- 1.4 電腦的程式邏輯

2、微課程教學目標及流程簡介

- 2.1 情境流程圖簡介
- 2.2 程式流程圖簡介
- 2.3 流程圖表達方式簡介
- 2.4 演算法步驟簡介
- 2.5 演算法表達方式簡介

3、流程圖符號(I) (II)

4、程式流程三種結構

4.1循序結構 Sequence

- 4.1.1 情境任務：RGB LED閃爍 為例
- 4.1.2 情境流程圖 vs 演算法步驟
- 4.1.3 演算法步驟 vs 程式流程圖
- 4.1.4 程式流程圖 vs 積木程式堆疊
- 4.1.5 情境延伸題：平交道閃燈警示器

4.2重覆結構 Iteration

- 4.2.1 Do While...Loop結構(先判斷後執行)
- 4.2.2 Do... Loop While結構(先執行後判斷)
- 4.2.3 情境任務：呼吸燈
- 4.2.4 情境流程圖 vs 演算法步驟
- 4.2.5 演算法步驟 vs 程式流程圖
- 4.2.6 程式流程圖 vs 積木程式堆疊

4.3選擇結構 Seleccction

- 4.3.1 二元選擇結構 (基本結構)
- 4.3.2 多重選擇結構 (二元選擇結構變化結構)
- 4.3.3 情境任務：安全風扇
- 4.3.4 演算法步驟
- 4.3.5 演算法步驟 vs 程式流程圖
- 4.3.6 程式流程圖 vs 積木程式堆疊
- 4.3.7 以「副程式」來呈現

大綱(II)

5、練習

5.1情境任務：直流抽排風扇

5.1.1演算法步驟(學生填空用)

5.1.2演算法步驟(教師用)

5.1.3程式流程圖(學生填空用)

5.1.4程式流程圖(教師用)

5.1.5積木程式堆疊(教師用)

5.2情境任務：鐵捲門

5.2.1演算法步驟(學生填空用)

5.2.2演算法步驟(教師用)

5.2.3程式流程圖(學生填空用)

5.2.4程式流程圖(教師用)

5.2.5積木程式堆疊(教師用)

6、流程圖繪製軟體

6.1 draw.io

6.2 fChart

邏輯及程式語言簡介

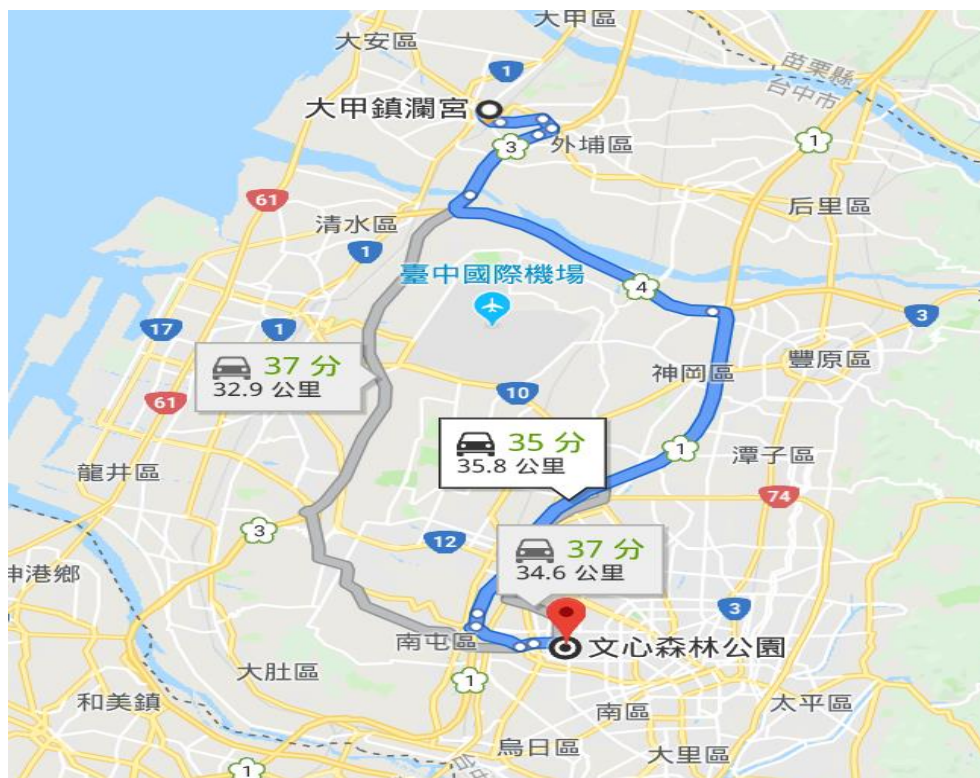
學習程式語言的目的

我們學習程式語言的主要目的，是要撰寫程式碼來建立應用程式，所以需要使用電腦的程式邏輯 (program logic) 來寫出程式碼，如此電腦才能執行程式碼來解決我們的問題。

邏輯及程式語言簡介

情境任務 - 以GOOGLE地圖為範例

例如：開車從大甲鎮瀾宮至文心森林公園，在Google地圖顯示如下：



邏輯及程式語言簡介

人類的邏輯

對於人類來說，我們只需檢視地圖，即可輕鬆寫下開車從大甲到文心森林公園的步驟，如下所示：

Step 1：從水源路直走上國道3號往南開。

Step 2：經國道4號。

Step 3：經國道1號南下，由南屯交流道下高速公路。

Step 4：接著走136縣道，直行即可到目的地。

上述資訊（使用人類的邏輯），已經足夠讓我們開車到達指定的目的地。

邏輯及程式語言簡介

電腦的程式邏輯

如果將上述步驟告訴電腦，電腦一定完全沒有頭緒，不知道如何開車到達目的地。

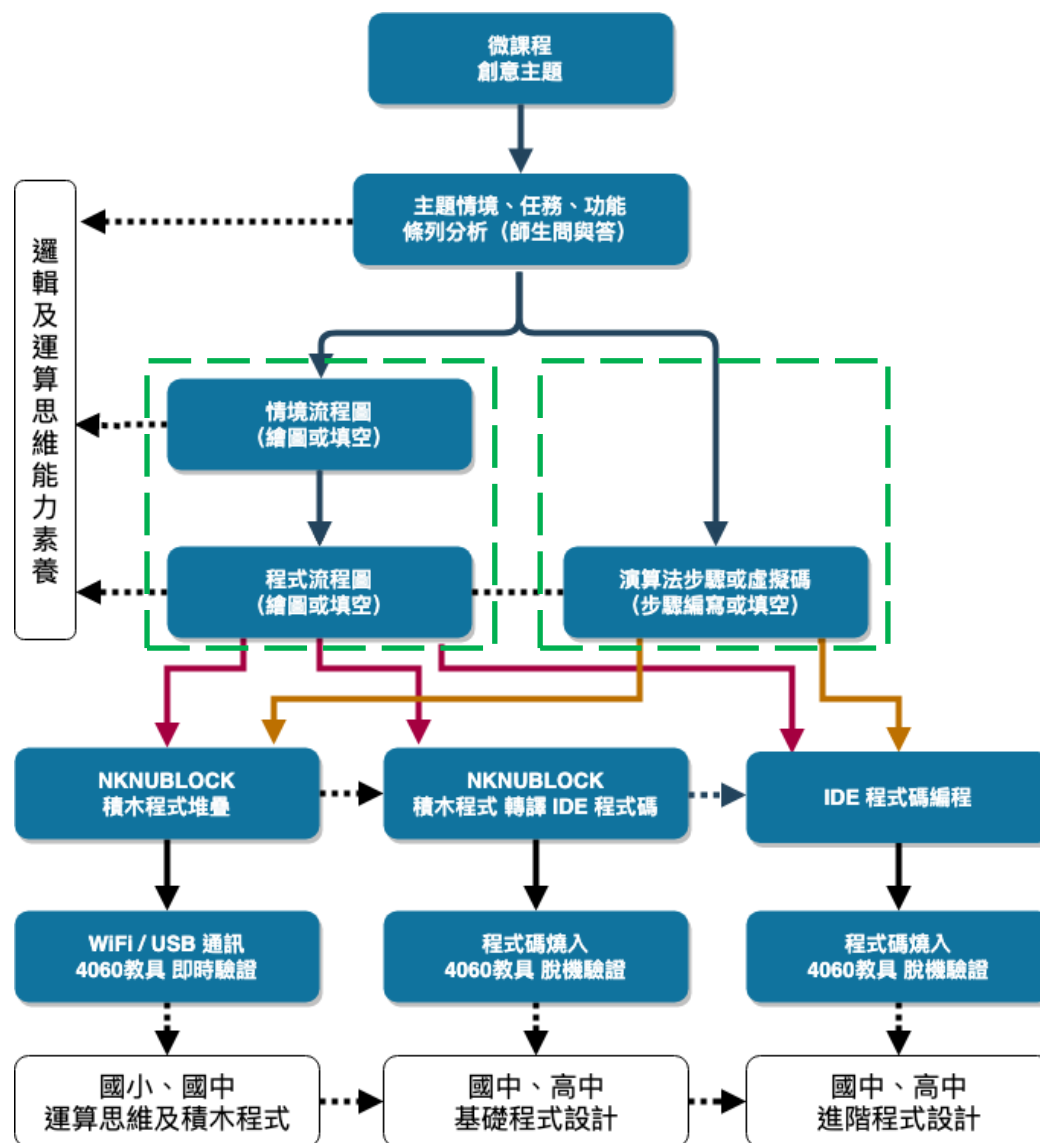
因為電腦一點都不聰明，這些步驟的描述太不明確，我們需要提供更多資訊給電腦（請改用電腦的程式邏輯來思考），才能讓電腦開車到達目的地，如下列之情境分析以條列方式表示：

- (1) 從哪裡開始開車 (起點) ?
- (2) 上國道3號高速公路需向南開幾公里？ 才會到達國道4號？
- (3) 又如何接國道4號？ 往東還是往西？
- (4) 開幾公里？ 才會到國道1號？
- (5) 又如何接國道1號？ 要往南還是往北開？
- (6) 如何分辨已經到了南屯交流道？ 如何從交流道下來？
- (7) 直行186縣道幾公里？ 可以看到紅綠燈？ 左轉或右轉？
- (8) 要開多少公里？ 才可以看到文心森林公園？
- (9) 最後要在那停車？ 如何停車(終點)？

微課程教學目標及流程簡介

微課程的教學目標是要透過一個**情境任務**，培養學生**邏輯及運算思維**的素養及能力。

依據微課程主題之情境任務，以下兩種教學流程選擇其中之一，依序完成教學步驟。



微課程教學目標及流程簡介

情境流程圖簡介

- **情境流程圖**是老師與學生間「**問與答**」中，確認學生對微課程的主題情境的邏輯程序的了解，是一種運算思維的訓練。
- 「**情境流程圖**」之目的是以不插電的方式，經由分析、設計、與**繪製（填空）**等步驟的執行，培養學生的「**邏輯及運算思維**」的素養及能力。

微課程教學目標及流程簡介

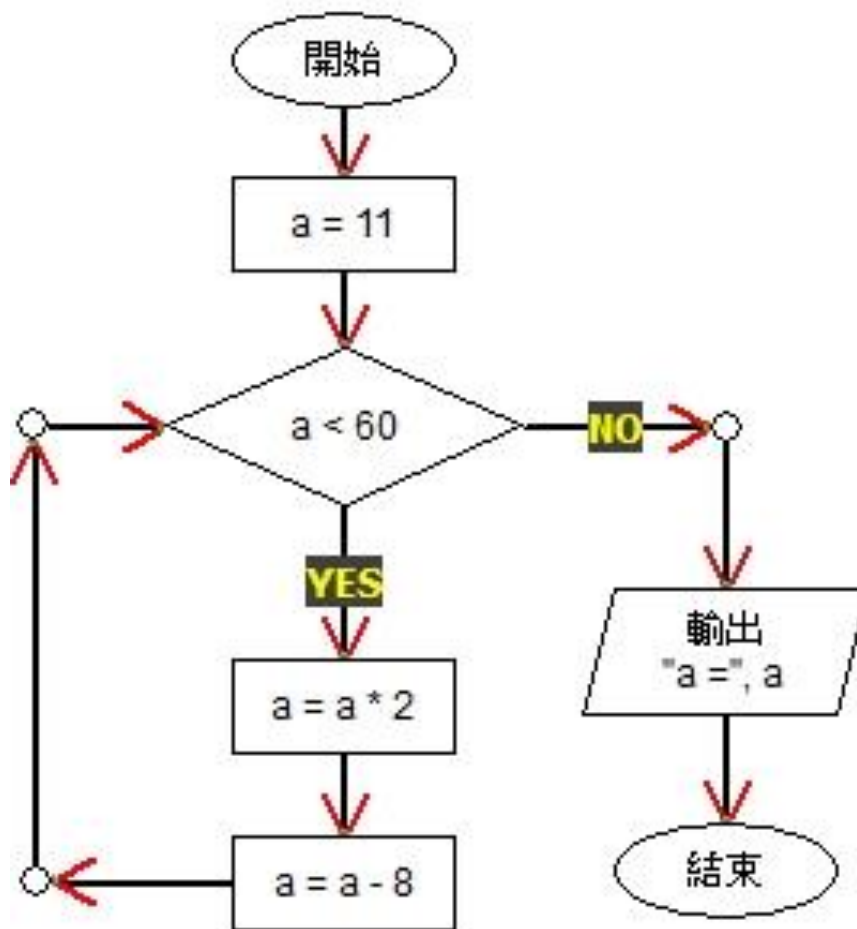
程式流程圖簡介

- 「程式流程圖」則是依據「情境流程圖」，導入積木程式堆疊的「邏輯程序細節設計」，每個人都可有不同創意的程式設計或編程手法，只要確認符合「程式流程圖」，依據流程圖進行程式堆疊，程式堆疊或編程沒有唯一的標準，程式只有更好，沒有最好。

微課程教學目標及流程簡介

流程圖表達方法簡介

- 流程圖（flow chart）：
使用標準圖示符號來描述執行過程，以各種不同形狀的圖示表示不同的操作，箭頭線標示流程執行的方向。



微課程教學目標及流程簡介

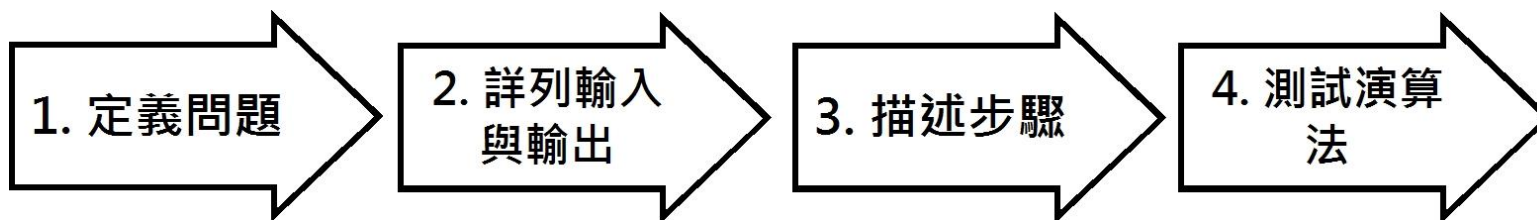
演算法步驟簡介

Step 1. 定義問題：使用明確和簡潔的詞彙來描述欲解決的問題。

Step 2. 詳列輸入與輸出：列出欲解決問題的資料 (input)，和經過演算法運算後，需要產生的結果 (output)。

Step 3. 描述步驟：描述從輸入資料轉換成輸出資訊的步驟。

Step 4. 測試演算法：使用測試資料來驗證演算法是否正確。



微課程教學目標及流程簡介

演算法表達方法簡介

因為演算法的表達方法是在描述解決問題的步驟，所以並沒有固定方法，常用的表達方法，如下所示：

- 演算法步驟：直接使用一般語言文字，條列式描述來說明執行步驟。
- 虛擬碼 (pseudo code)：一種趨近程式語言的描述方法，並沒有固定語法，每一行約可轉換成一行程式碼，如下所示：

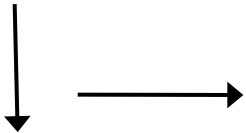
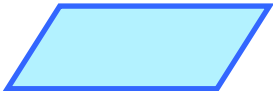
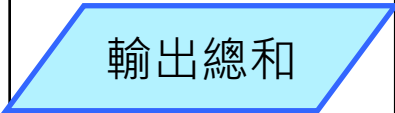
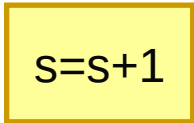
演算法步驟

1. 初始LED亮度變數 $i=0$ (全暗)
2. LED亮度增減變量值 $x=5$
3. 判斷迴圈計算是否 $i=255$
 - 3-1. 若否，LED做PWM輸出
 - 3-1-1. 計算 $i=i+x$ (亮度漸亮)
 - 3-1-2. 延遲30毫秒後，繼續迴圈步驟3
 - 3-2. 若是，LED全亮，結束迴圈至步驟4

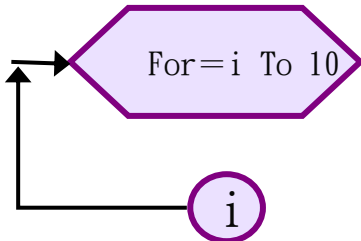
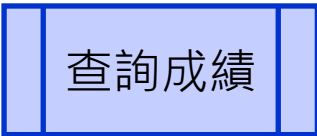
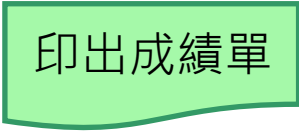
虛擬碼

```
/* 計算1加到10 */  
Let counter = 1  
Let sum = 0  
while counter <= 10  
    sum = sum + counter  
    Add 1 to counter  
Output the sum    /* 顯示結果 */
```

流程图符号(1)

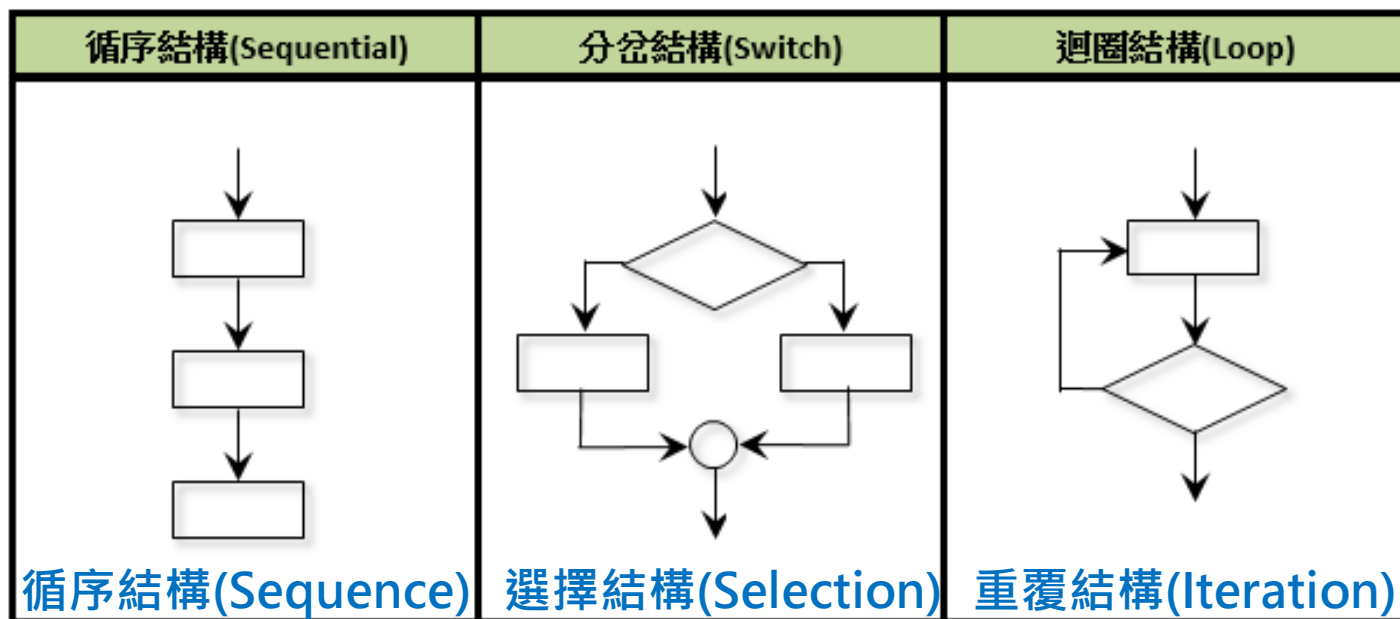
符號	名 稱	意 義	範 例
	開始 (Start) 終止 (End)	表示程式的開始 或結束	
	路徑(Path)	表示流程進行的 方向	
	輸入(Input) 輸出(Output)	表示資料的輸入 或結果的輸出	
	處理(Process)	表示執行或處理 某一項工作	

流程图符号(2)

符號	名 稱	意 義	範 例
	決策判斷(Decision)	針對某一條件進行判斷	
	迴圈 (Loop)	表示迴圈控制變數的初始值及終值	
	副程式(Subroutine)	用以表示一群已經定義流程的組合	
	報表(Document)	指列印出的報表文件	

程式流程三種結構

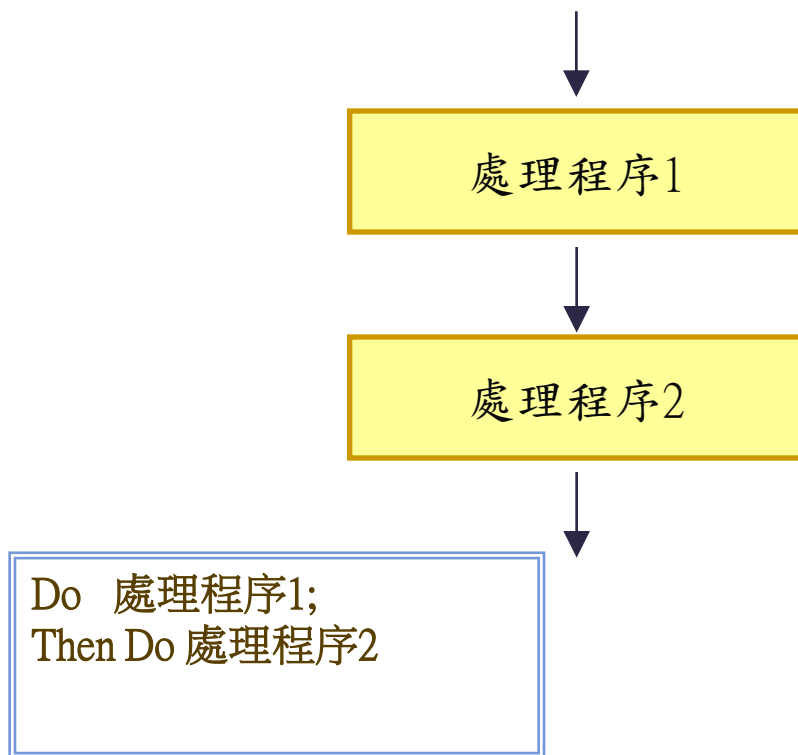
程式流程控制的三種結構



程式流程三種結構

循序結構 SEQUENCE

- 循序結構 (Sequence)
- 意義：處理程序循序進行。



循序結構 SEQUENCE

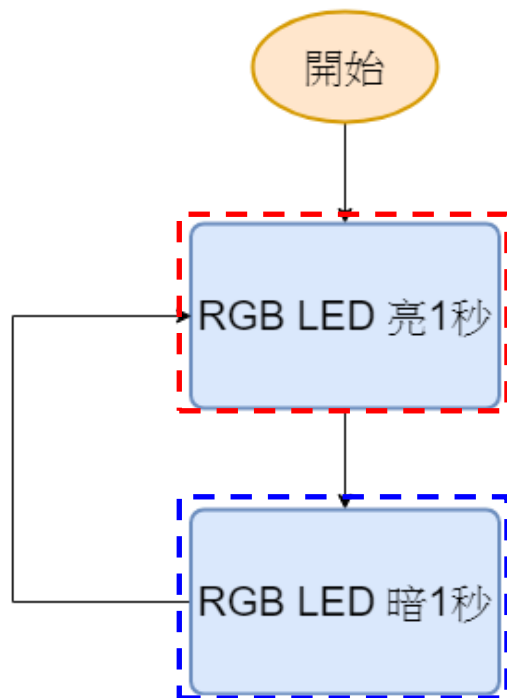
情境主題：RGB LED閃爍 為例

- 情境主題: RGB LED閃爍
- 情境目的: 讓一個三色RGB LED燈亮1秒，暗1秒，無限循環閃爍。
- 情境流程圖
- 演算法步驟
- 程式流程圖
- 積木程式堆疊
- 4060教具即時驗證

循序結構 SEQUENCE

情境流程圖 vs 演算法步驟

● 情境流程圖



● 演算法步驟

1.LED數位輸出高電位(全亮)

2.亮1秒

3.LED數位輸出低電位(全暗)

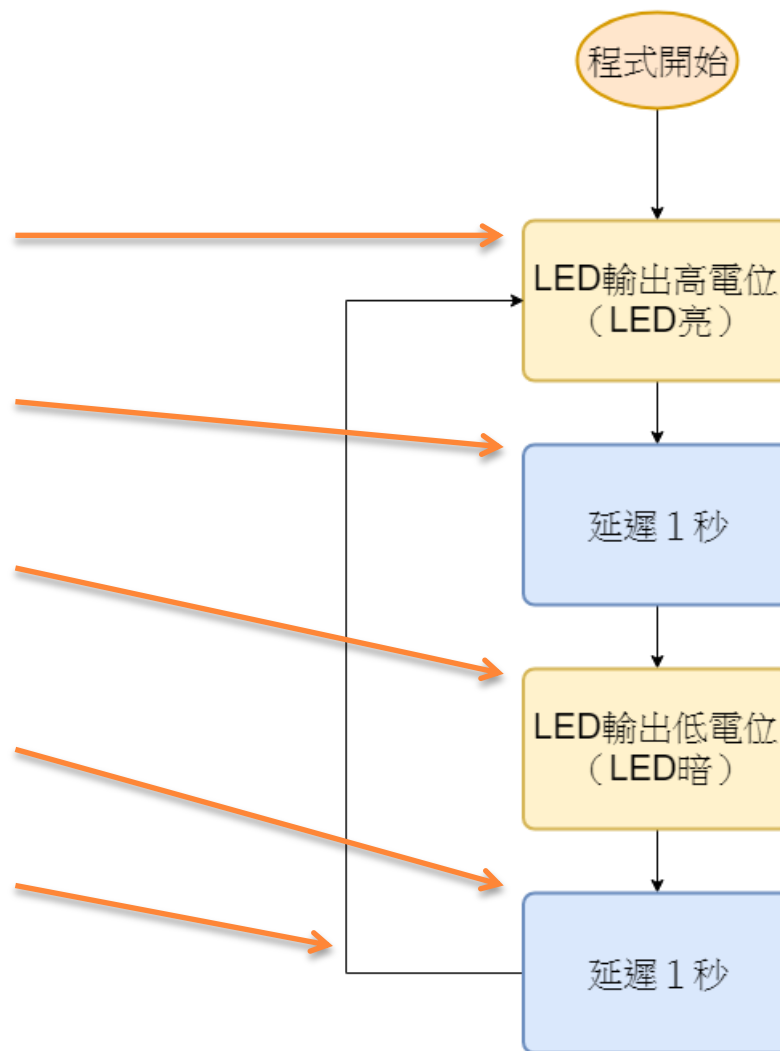
4.暗1秒

5.跳至步驟1

循序結構 SEQUENCE

演算法步驟 vs 程式流程圖

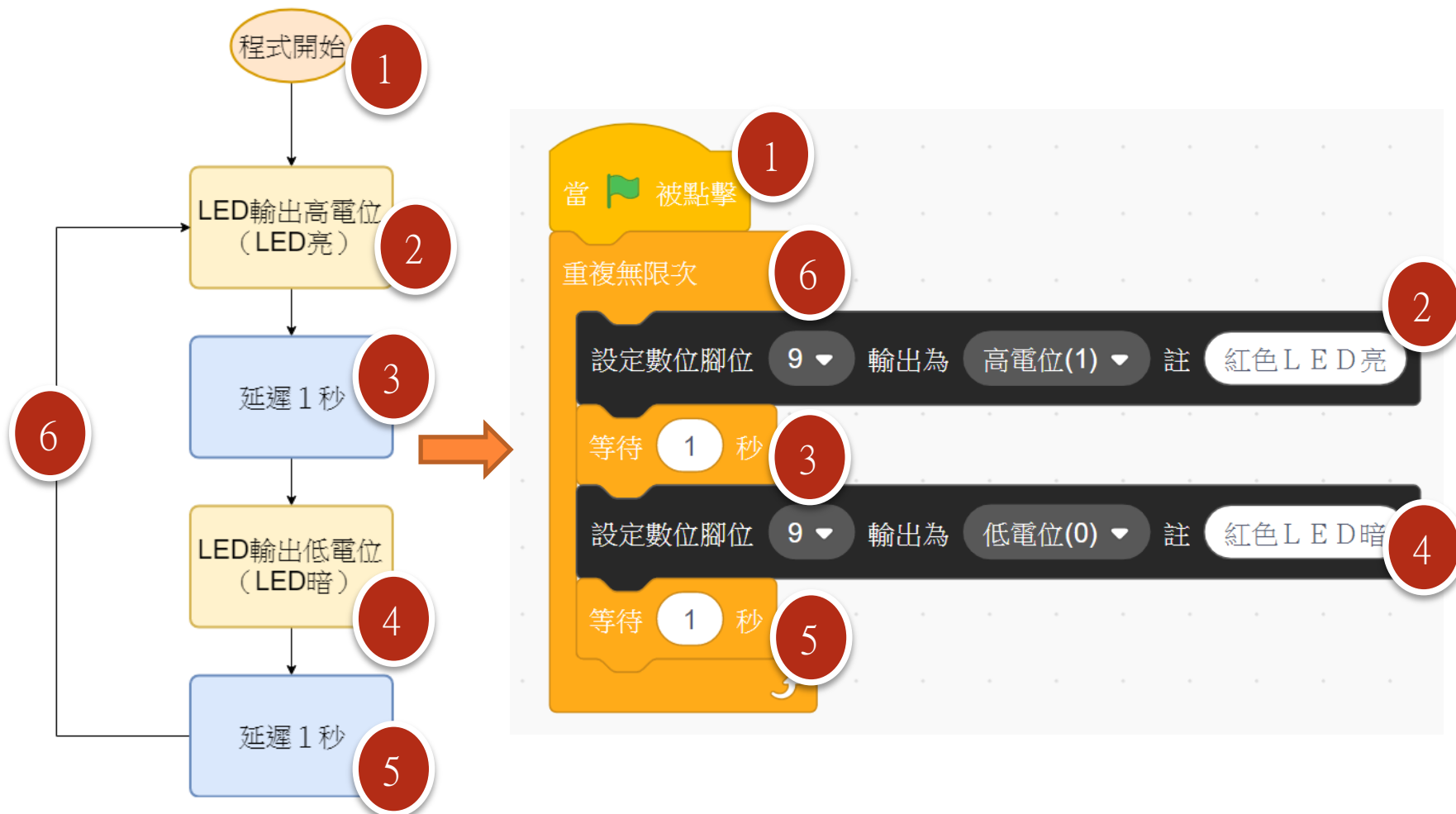
- 1.LED數位輸出高電位(全亮)
- 2.亮1秒
- 3.LED數位輸出低電位(全暗)
- 4.暗1秒
- 5.跳至步驟1



循序結構 SEQUENCE

程式流程圖 vs 積木程式堆疊

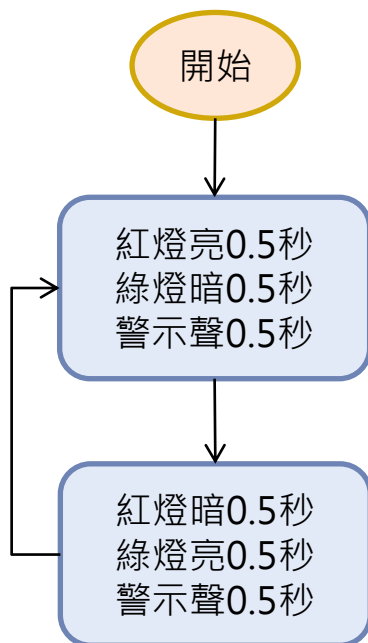
- 依據「程式流程圖」執行「積木程式堆疊」



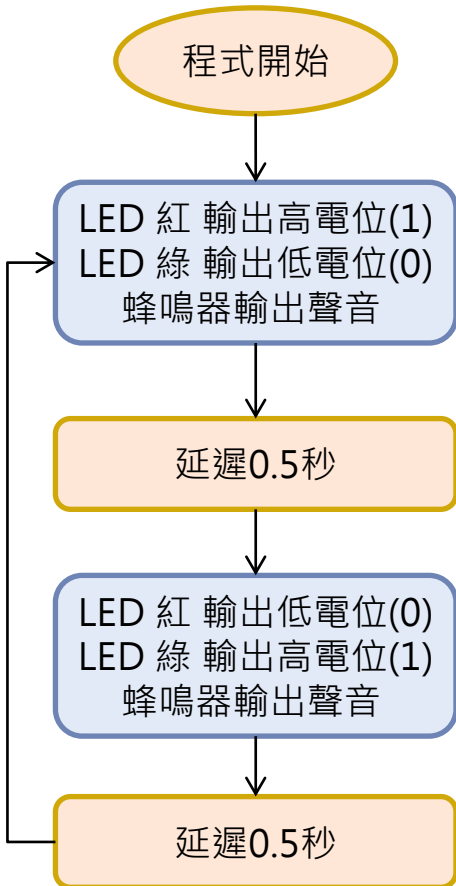
循序結構 SEQUENCE

情境延伸題：平交道閃燈警示器

情境流程圖



程式流程圖



積木程式堆疊

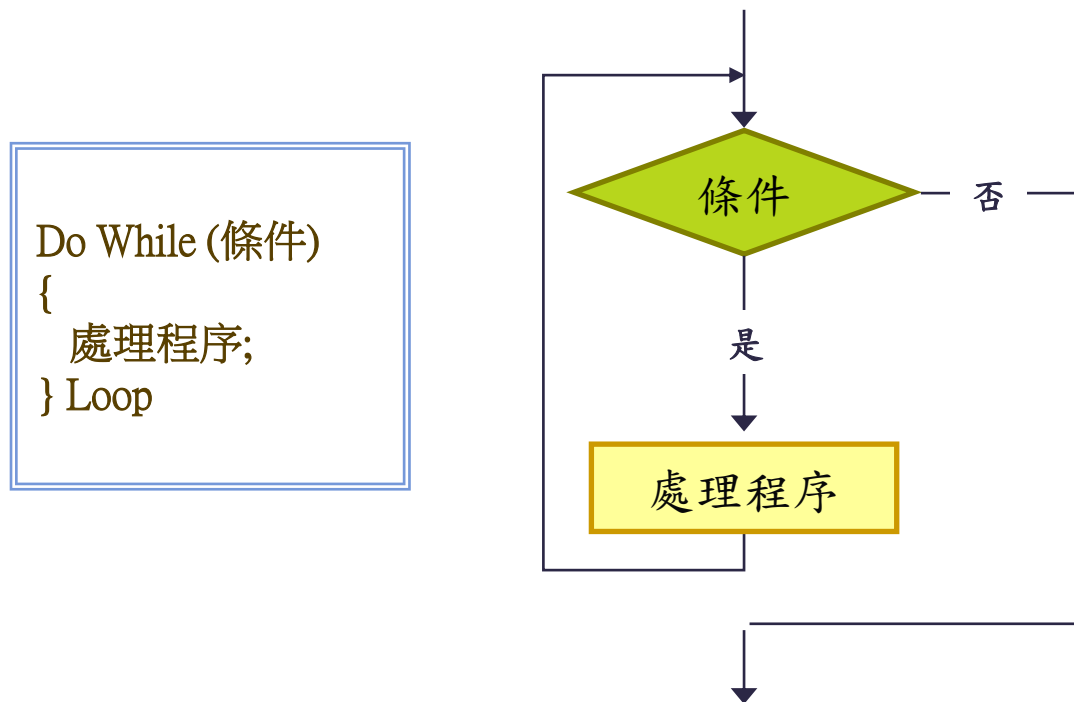


程式流程三種結構

重覆結構 ITERATION

Do WHILE...LOOP結構(先判斷後執行)

意義：依據條件是否成立，決定執行的情況。當條件成立時，就不斷重複執行處理程序，直到執行的條件不成立後，即離開重覆結構，進行下一個流程。



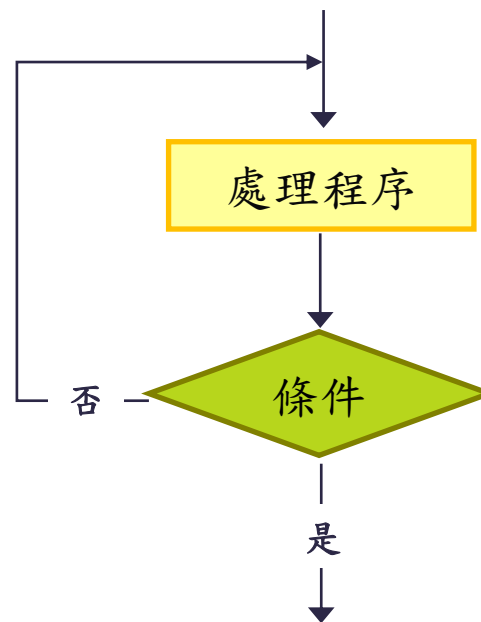
程式流程三種結構

重覆結構 INTERATION

Do... LOOP WHILE結構(先執行後判斷)

意義：重覆執行處理程序，直到條件變成真（true）為止。

```
do
{
    處理程序;
}
Loop While (條件);
```



重覆結構 ITERATION

情境主題：呼吸燈

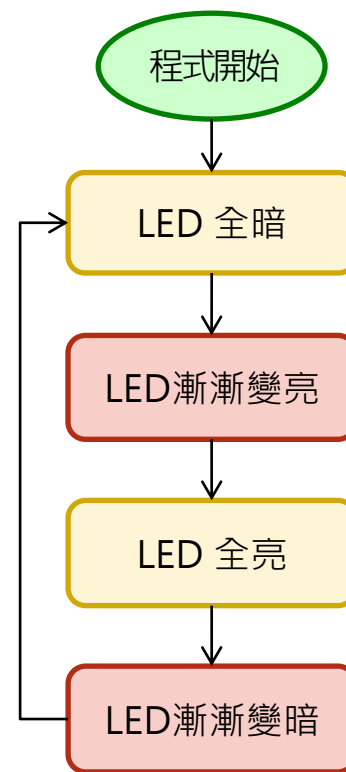
(1) 情境主題：呼吸燈

(2) 情境目的：調整LED燈的亮度，從全暗漸漸增亮到全亮，再由全亮漸漸減亮到全暗，反覆執行。

(3) 情境分析：

- 1.LED全暗→LED漸漸增亮(連續性漸增)
- 2.LED全亮→LED漸漸減亮(連續性漸減)
- 3.非數位輸出，而是PWM輸出(類比輸出)
- 4.PWM輸出值範圍0(全暗)～255(全亮)

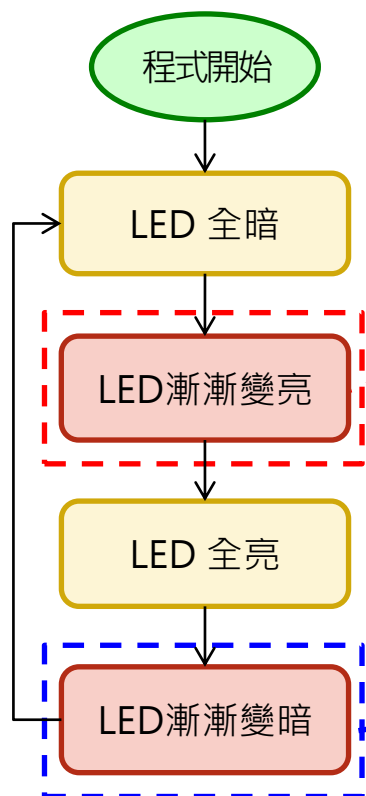
(4) 情境流程圖：



重覆結構 (呼吸燈)

情境流程圖 vs 演算法步驟

情境流程圖



演算法步驟

1. 初始LED亮度變數 $i=0$ (全暗)

2. LED亮度增減變量值 $x=5$

3. 判斷迴圈執行條件是否 $i=255$ (全亮)

3-1. 若否，LED做PWM輸出

3-1-1. 計算 $i=i+x$ (亮度漸亮)

3-1-2. 延遲30毫秒後，繼續迴圈步驟3

3-2. 若是，LED全亮，結束迴圈至步驟4

4. 判斷迴圈執行條件是否 $i=0$ (全暗)

4-1. 若否，LED做PWM輸出

4-1-1. 計算 $i=i-x$ (亮度漸暗)

4-1-2. 延遲30毫秒後，繼續迴圈步驟4

4-2. 若是，LED全暗，結束迴圈至步驟3

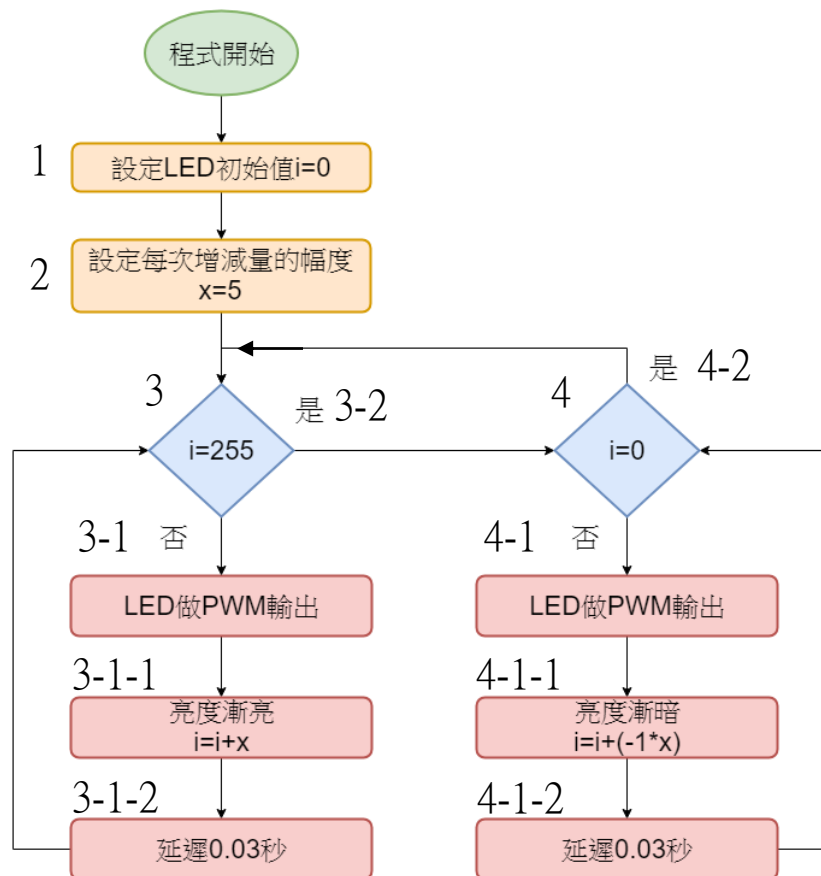
重覆結構 (呼吸燈)

演算法步驟 vs 程式流程圖

演算法步驟

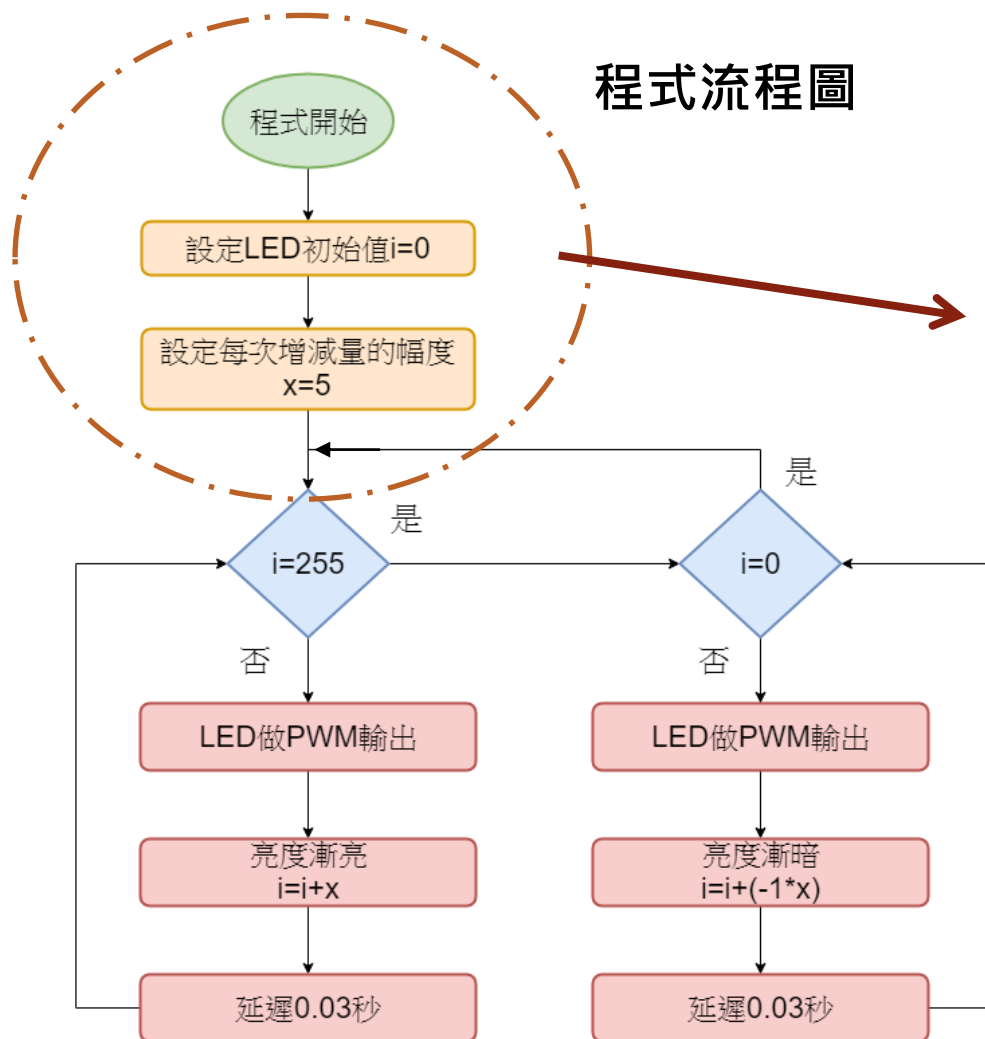
1. 初始LED亮度變數 $i=0$ (全暗)
2. LED亮度增減變量值 $x=5$
3. 判斷迴圈執行條件是否 $i=255$
 - 3-1. 若否，LED做PWM輸出
 - 3-1-1. 計算 $i=i+x$ (亮度漸亮)
 - 3-1-2. 延遲0.03秒後，繼續迴圈步驟3
 - 3-2. 若是，LED全亮，結束迴圈至步驟4
4. 判斷迴圈執行條件是否 $i=0$
 - 4-1. 若否，LED做PWM輸出
 - 4-1-1. 計算 $i=i-x$ (亮度漸暗)
 - 4-1-2. 延遲0.03秒後，繼續迴圈步驟4
 - 4-2. 若是，LED全暗，結束迴圈至步驟3

程式流程圖



重覆結構 (呼吸燈)

程式流程圖 vs 積木程式堆疊

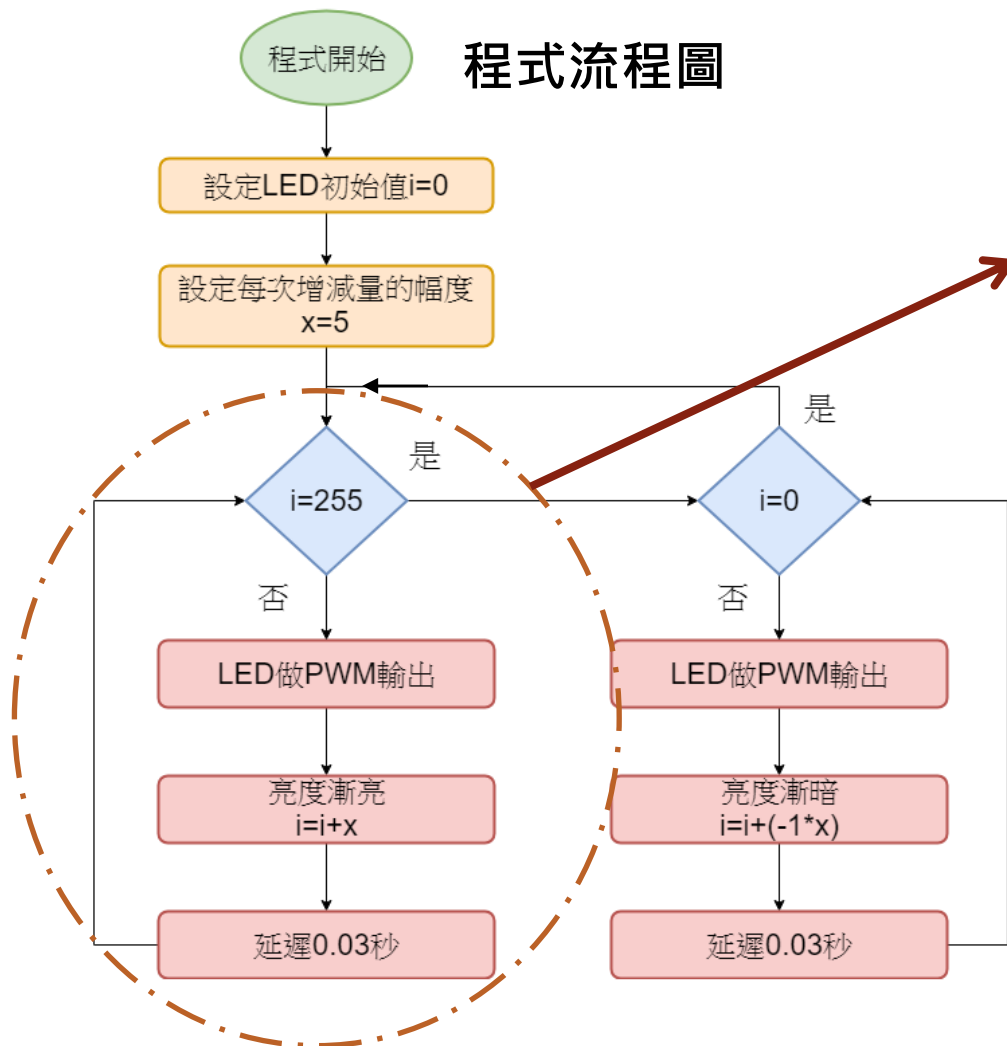


程式 1

重覆結構 (呼吸燈)

程式流程圖 vs 積木程式堆疊

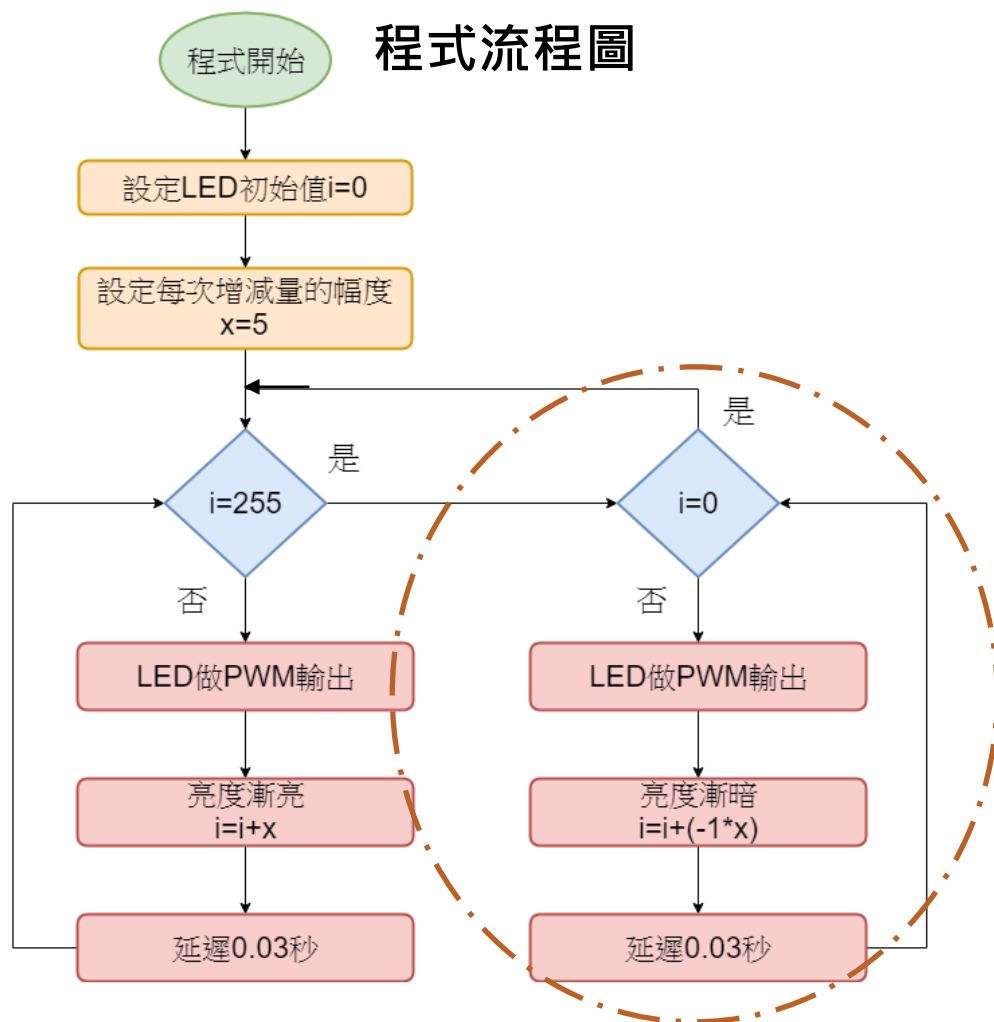
程式流程圖



程式 2

重覆結構 (呼吸燈)

程式流程圖 vs 積木程式堆疊

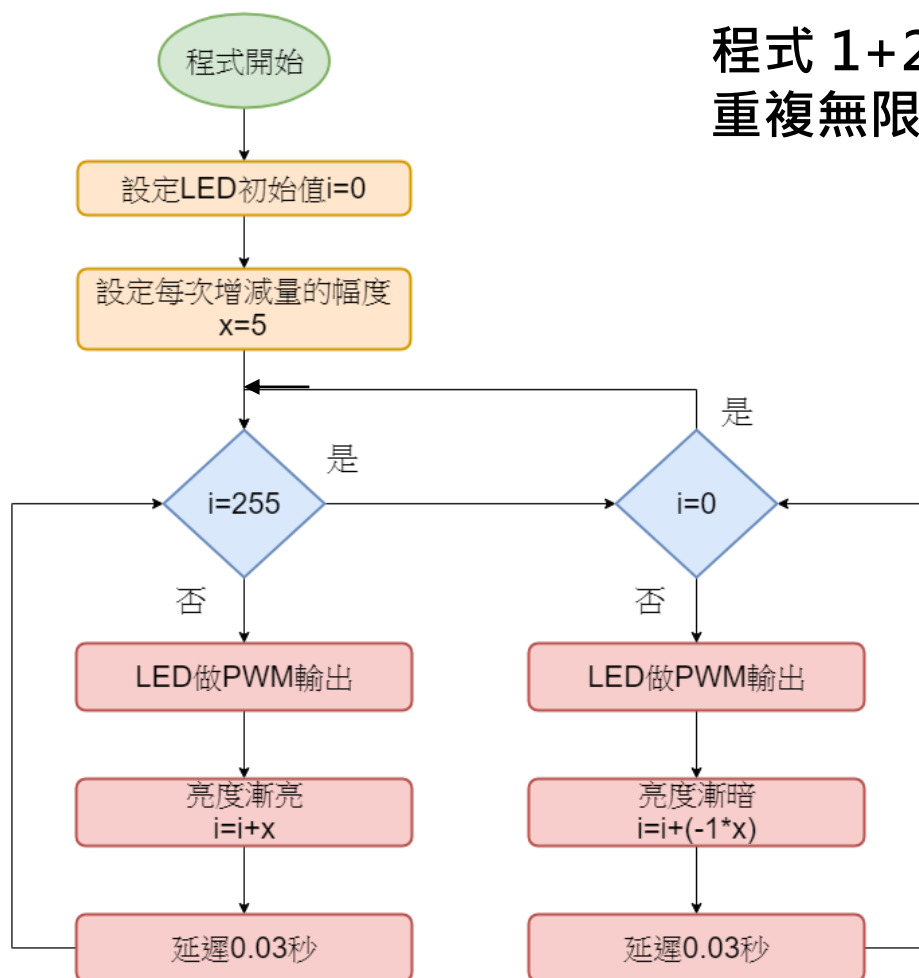


程式 3

重覆結構 (呼吸燈)

程式流程圖 vs 積木程式堆疊

程式流程圖



主程式

程式 1+2+3
重複無限次

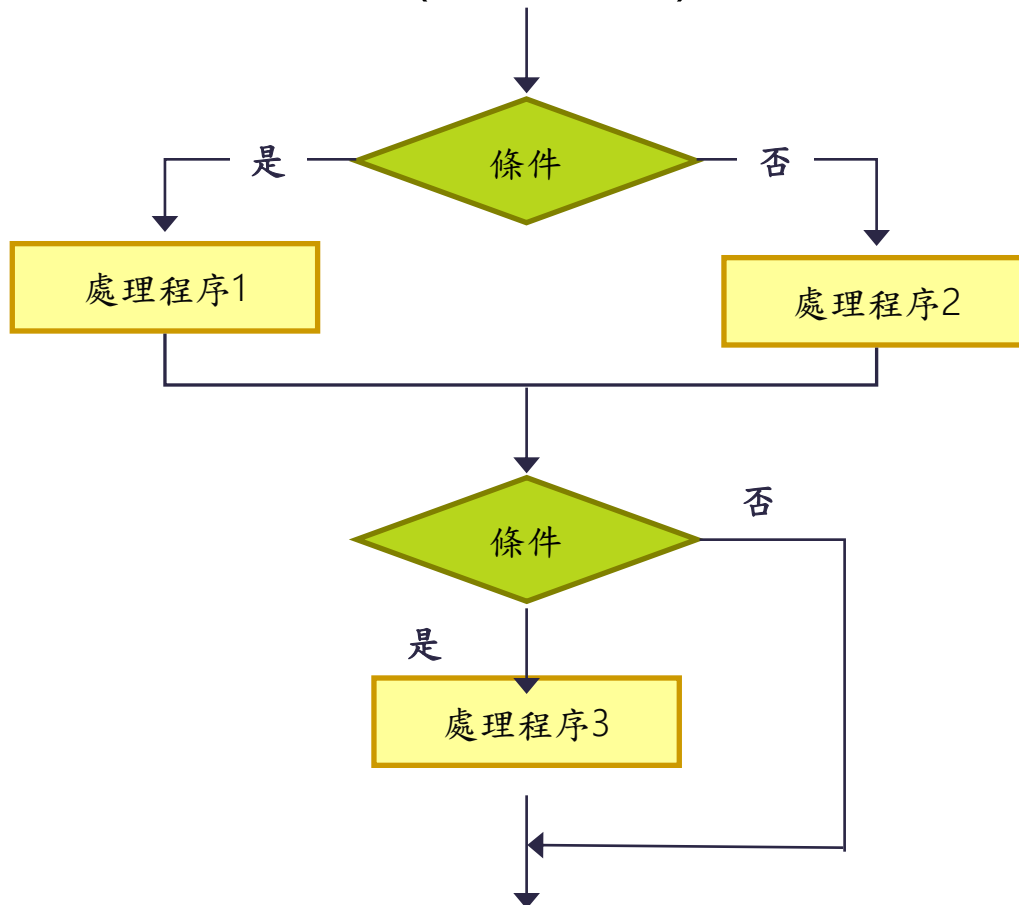


程式流程三種結構

選擇結構SELECTION

二元選擇結構

- 二元選擇結構（基本結構）



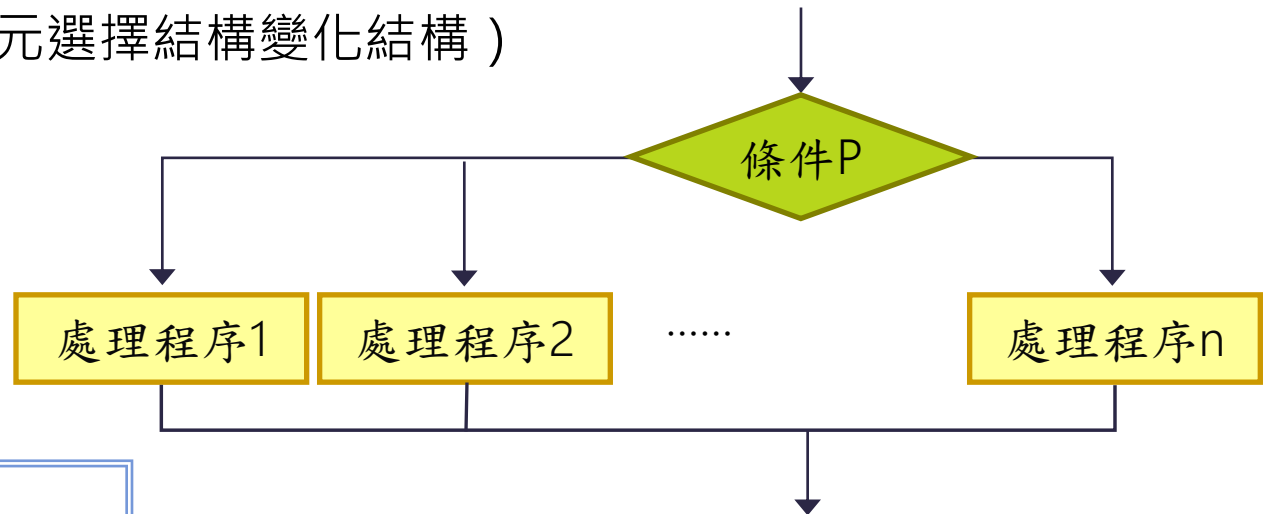
```
If (條件) {  
    處理程序1;  
}  
else {  
    處理程序2;  
}  
If (條件) {  
    處理程序3;  
}
```


程式流程三種結構

選擇結構SELECTION

多重選擇結構

- 多重選擇結構（二元選擇結構變化結構）



```
SELECT (條件p) {  
  case =1:  
    處理程序1;  
  case p=2:  
    處理程序2;  
  ...  
  case p=n:  
    處理程序n;  
}
```

選擇結構 SELECTION

情境主題：安全風扇

- 情境主題：安全風扇
- 情境目的：結合超音波感測器與風扇的運轉，利用超音波感測器感測到有人員接近時，可以快速將風扇立即停止，以保護人員的安全。
- 情境分析：
 1. 設定超音波感測器與人員之間的距離為 H ，
 2. 如果距離30公分以上，風扇高速運轉，綠燈亮(安全區)
 3. 介於10~30公分 之間，風扇低速運轉，藍燈亮且發出嗶嗶 聲響(警戒區)
 4. 低於10公分以內，風扇立即停止且發出急促嗶嗶聲響，紅燈亮(危險區)。

選擇結構 (安全風扇)

演算法步驟

1. 讀取超音波感測器量測距離H
2. 判斷 $H > 30$
 - 2-1. 成立：風扇高速運轉，綠燈亮，執行後再回至步驟1
3. 否則：
 - 3-1. 不成立：再判斷 $10 < H < 30$
 - 3-1-1. 成立：風扇低速運轉，藍燈亮，發出嗶嗶聲響，執行後再回至步驟1
 - 3-2. 否則：
 - 3-2-1. 不成立：再判斷 $H > 0(*)$
 - 3-2-1-1. 成立：風扇停止，紅燈亮，發出急促嗶嗶聲響，執行後再回至步驟1

*(超音波感測器測不出距離時會傳回0，所以要把0濾除掉)

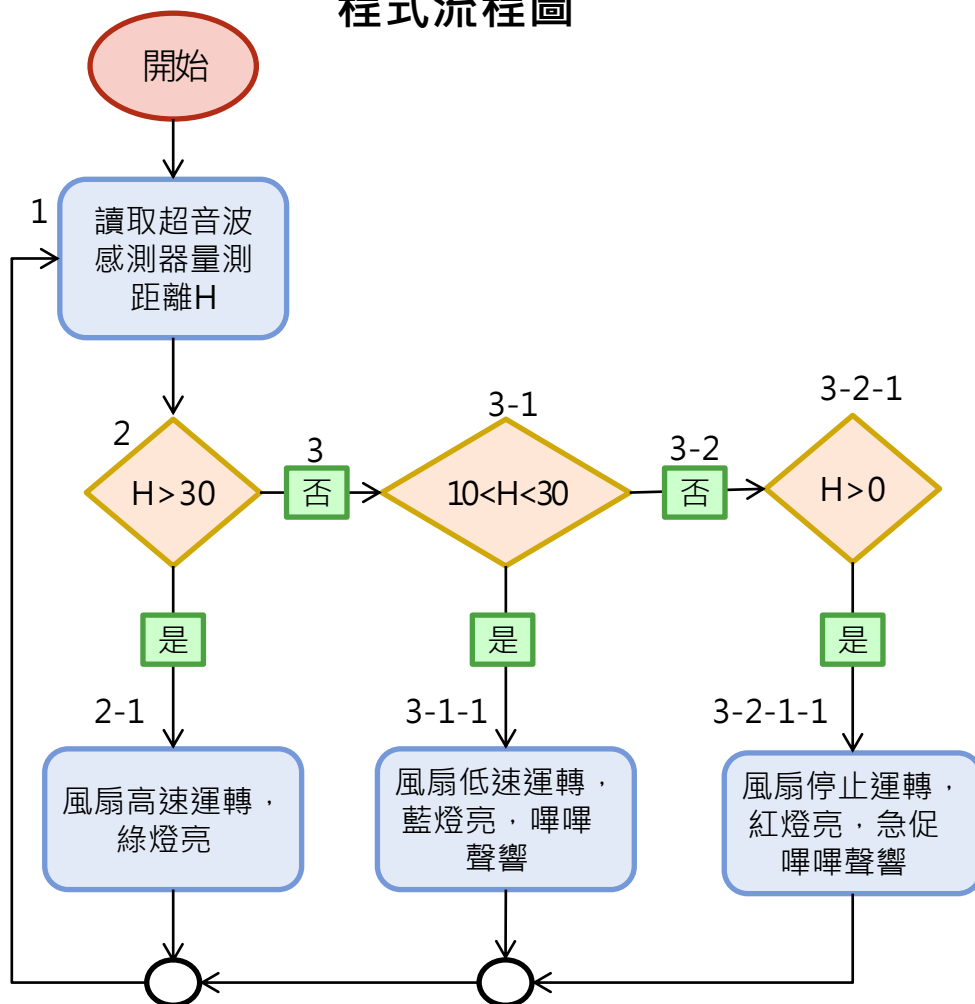
選擇結構 (安全風扇)

演算法步驟 vs 程式流程圖

演算法步驟

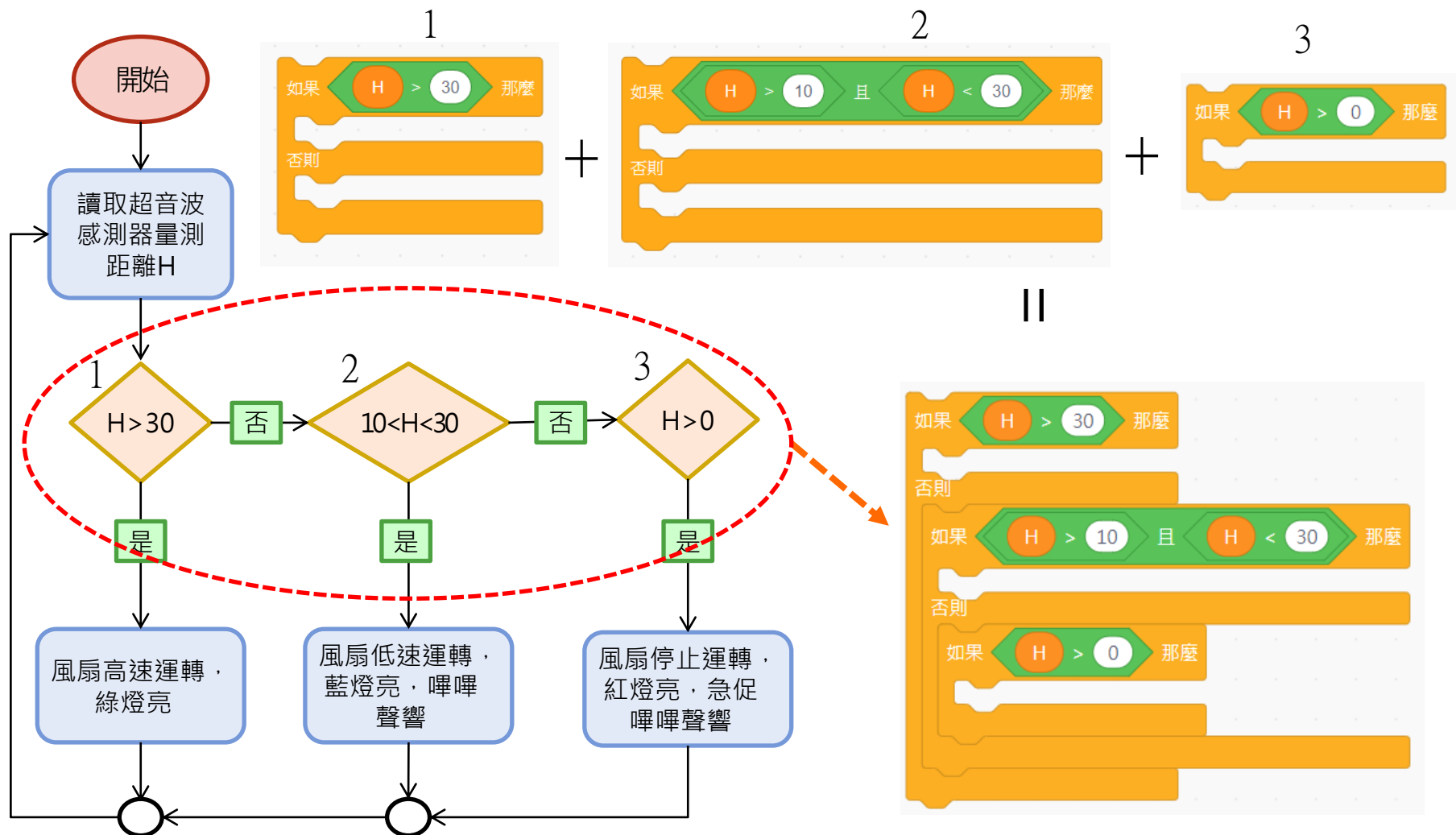
1. 讀取超音波感測器量測距離H
2. 判斷 $H > 30$
 - 2-1. 成立：風扇高速運轉，綠燈亮，執行後再回至步驟1
3. 否則：
 - 3-1. 不成立：再判斷 $10 < H < 30$
 - 3-1-1. 成立：風扇低速運轉，藍燈亮，發出嗶嗶聲響，執行後再回至步驟1
 - 3-2. 否則：
 - 3-2-1. 不成立：再判斷 $H > 0$
 - 3-2-1-1. 成立：風扇停止運轉，紅燈亮，發出急促嗶嗶聲響，執行後再回至步驟1

程式流程圖



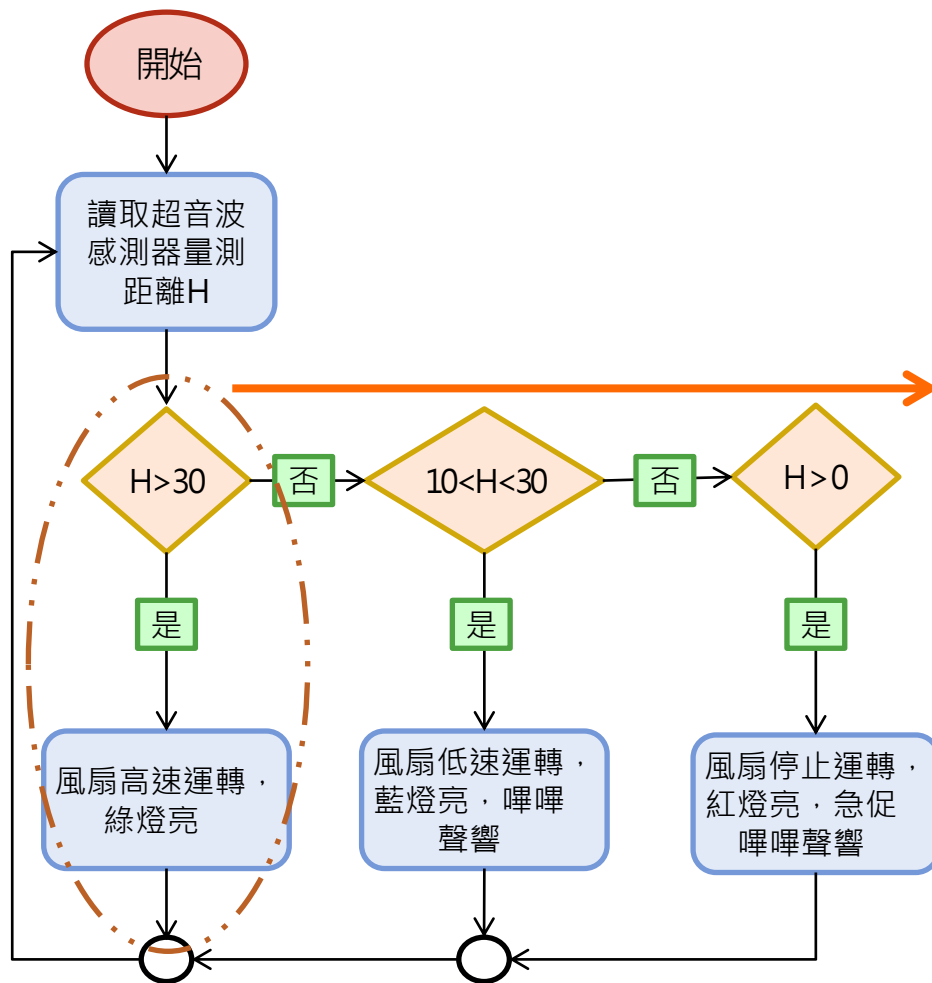
選擇結構 (安全風扇)

程式流程圖 vs 積木程式堆疊



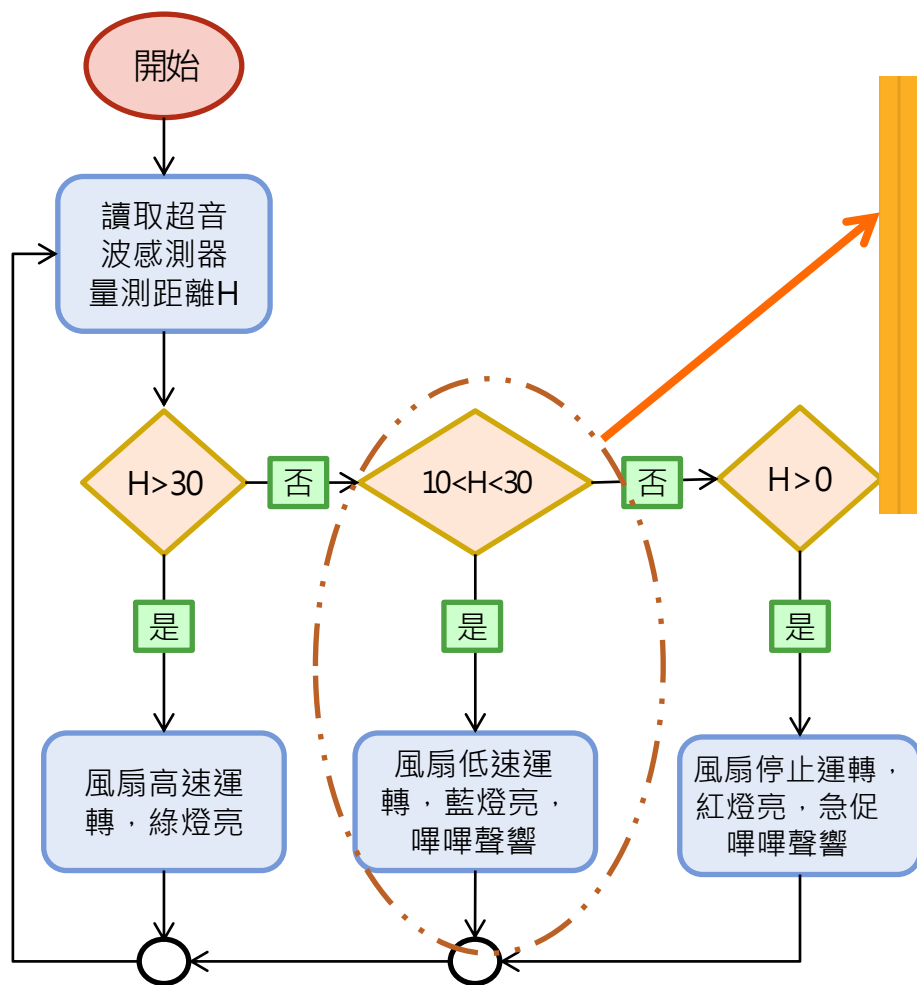
選擇結構 (安全風扇)

程式流程圖 vs 積木程式堆疊



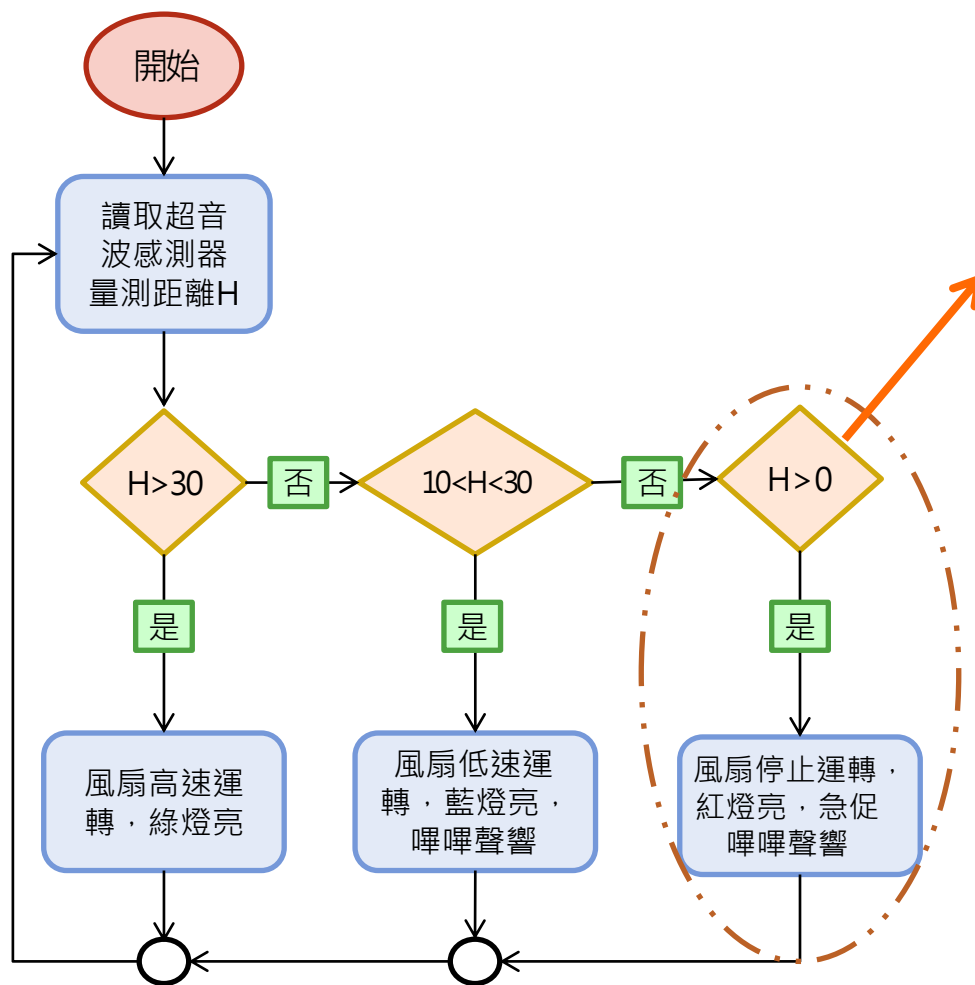
選擇結構 (安全風扇)

程式流程圖 vs 積木程式堆疊

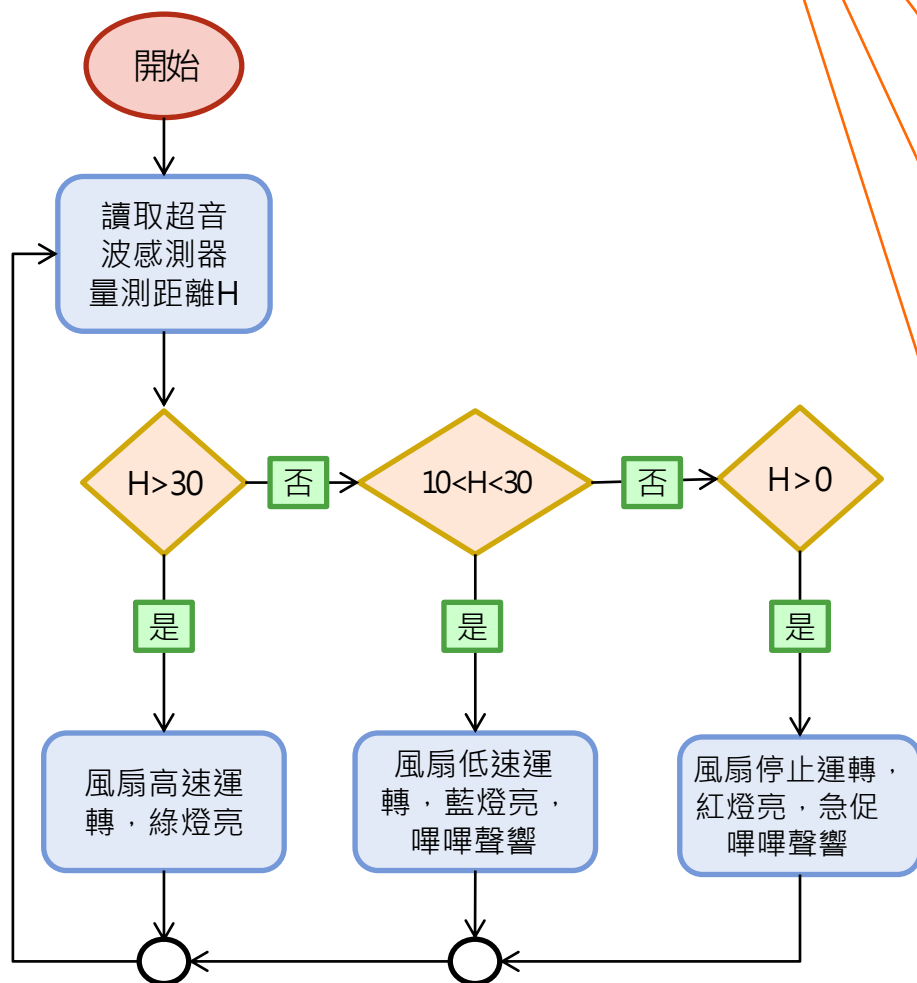


選擇結構 (安全風扇)

程式流程圖 vs 積木程式堆疊



選擇結構 (安全風扇) 以「副程式」來呈現



選擇結構 (安全風扇) 以「副程式」來呈現



練習

情境主題：直流抽排風機

- 情境主題：直流抽排風機
- 情境目的：結合搖桿與風扇的運轉，利用搖桿左移、右移及按壓鍵模擬排風、抽風及停止功能。
- 情境分析：
 - 1.讀取搖桿方向值，
 - 2.如果搖桿左移(X軸)，且X值 < 50 ，則馬達正轉(排風)，綠燈亮；
 - 3.如果搖桿右移(X軸)，且X值 > 800 ，則馬達逆轉(抽風)，藍燈亮；
 - 4.按壓搖桿按鍵，則馬達停止運轉，LED燈滅。

練習--情境主題：直流抽排風機

演算法步驟(學生填空用)

1.讀取搖桿方向值及按鍵值X、Y、Button

2.判斷

2-1. 成立：

3.否則：

3-1. 不成立：

3-1-1. 成立：

3-2. 否則：

3-2-1. 不成立：

3-2-1-1. 成立：

3-2-2.否則：

3-2-2-1. 不成立，

練習--情境主題：直流抽排風機 演算法步驟(教師解答)

1.讀取搖桿方向值及按鍵值X、Y、Button

2.判斷 $X < 50$

2-1. 成立：馬達正轉(排風)，綠燈亮

3.否則：

3-1. 不成立：再判斷 $X > 800$

3-1-1. 成立：馬達逆轉(抽風)，藍燈亮

3-2.否則：

3-2-1. 不成立：再判斷 $\text{Button} = 1$

3-2-1-1. 成立：風扇停止，LED燈滅

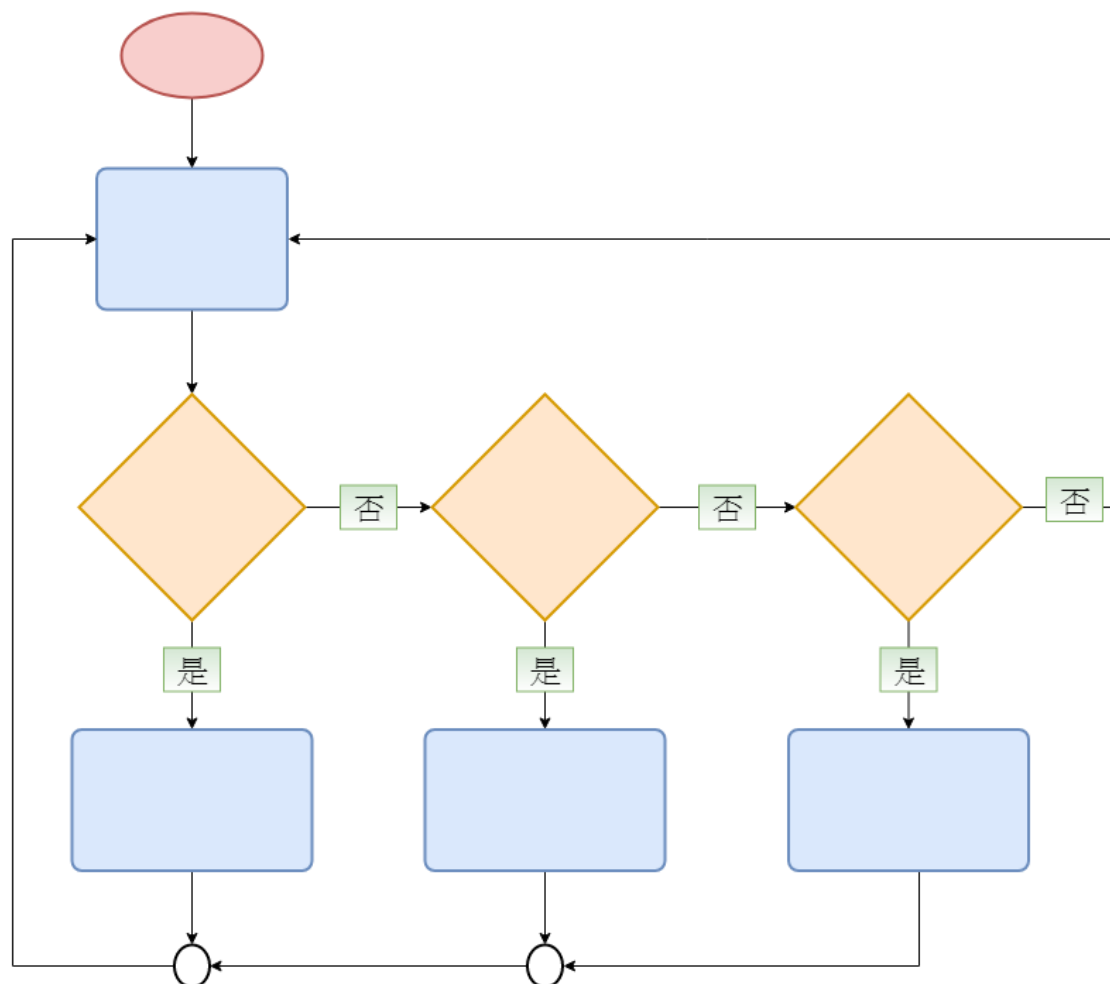
3-2-2.否則：

3-2-2-1. 不成立，回到步驟1

練習--情境主題：直流抽排風機

程式流程圖(學生填空用)

1. 讀取搖桿方向值及按鍵值X、Y、Button
2. 判斷 $X < 50$
 - 2-1. 成立：馬達正轉(排風)，綠燈亮
3. 否則：
 - 3-1. 不成立：再判斷 $X > 800$
 - 3-1-1. 成立：馬達逆轉(抽風)，藍燈亮
 - 3-2. 否則：
 - 3-2-1. 不成立：再判斷 $\text{Button} = 1$
 - 3-2-1-1. 成立：風扇停止，LED燈滅
 - 3-2-2. 否則：
 - 3-2-2-1. 不成立，回到步驟1



練習--情境主題：直流抽排風機

程式流程圖(教師解答)

1.讀取搖桿方向值及按鍵值X、Y、Button

2.判斷 $X < 50$

2-1. 成立：馬達正轉(排風)，綠燈亮

3.否則：

3-1. 不成立：再判斷 $X > 800$

3-1-1. 成立：馬達逆轉(抽風)，藍燈亮

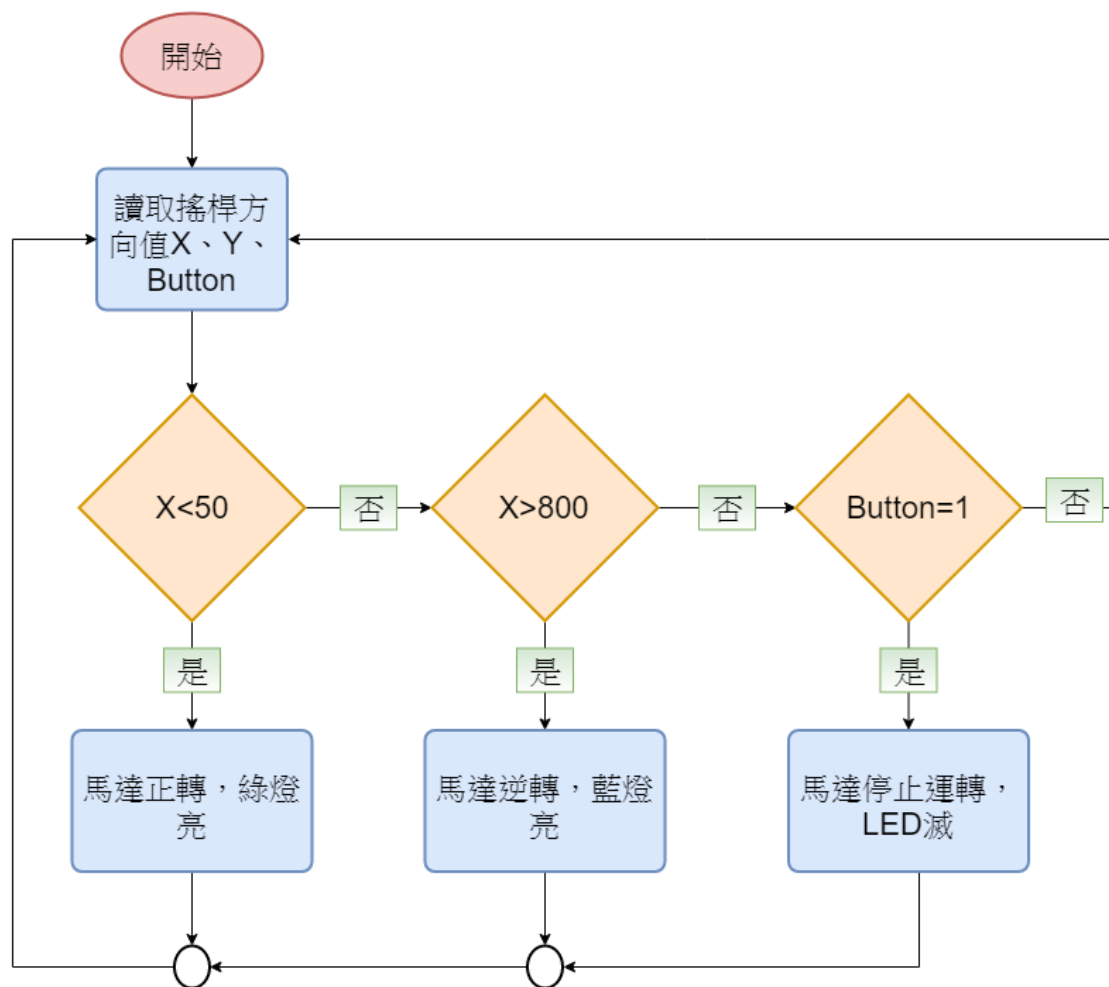
3-2.否則：

3-2-1. 不成立：再判斷 Button = 1

3-2-1-1. 成立：風扇停止，LED燈滅

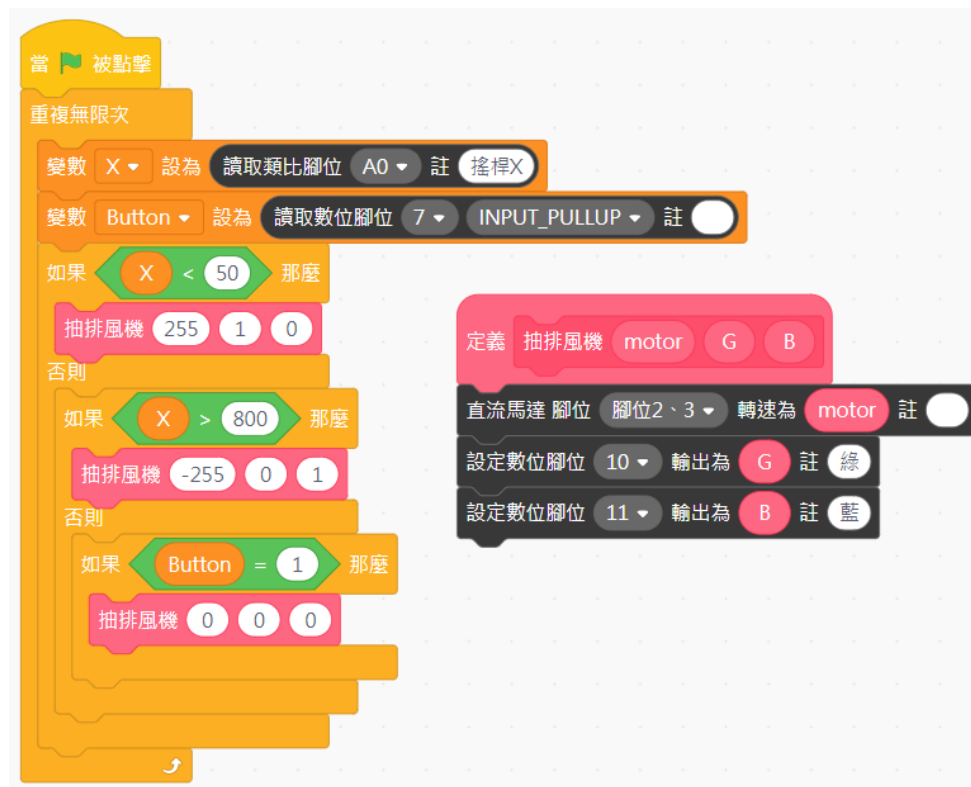
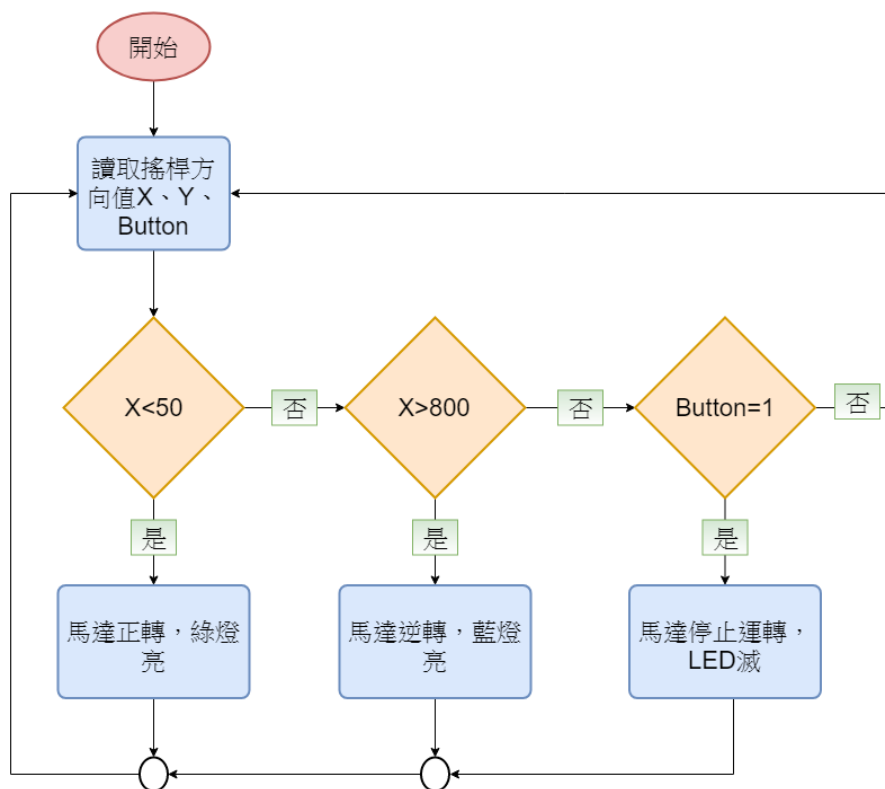
3-2-2. 否則：

3-2-2-1. 不成立，回到步驟1



練習--情境主題：直流抽排風機

積木程式堆疊(教師用)



練習

情境主題：鐵捲門

- 情境主題：鐵捲門
- 情境目的：結合搖桿與馬達的運轉，利用搖桿左移、右移及按壓鍵模擬鐵捲門上移、下移及停止功能。
- 情境分析：
 - 1.讀取搖桿方向與按鍵值；
 - 2.如果搖桿左移(X軸)，且X值 < 50 ，則馬達正轉(鐵捲門上移)，綠燈亮，此時將搖桿右移，仍需維持目前狀態，必須在馬達停止運轉後才可改變狀態；
 - 3.如果搖桿右移(X軸)，且X值 > 800 ，則馬達逆轉(鐵捲門下移)，藍燈亮，此時將搖桿左移，仍需維持目前狀態，必須在馬達停止運轉後才可改變狀態；
 - 4.按壓搖桿按鍵，則馬達停止運轉，LED燈滅。

練習--情境主題：鐵捲門

演算法步驟(學生填空用)

1.讀取搖桿方向值及按鍵值X、Y、Button

2.判斷 ($X < 50$)

2-1. 成立：馬達正轉(上移)，綠燈亮

3.否則：

3-1. 不成立：再判斷 ($X > 800$)

3-1-1. 成立：馬達逆轉(抽風)，藍燈亮

3-2.否則：

3-2-1. 不成立：再判斷 Button = 1

3-2-1-1. 成立：風扇停止，LED燈滅

3-2-2. 否則：

3-2-2-1. 不成立，回到步驟1

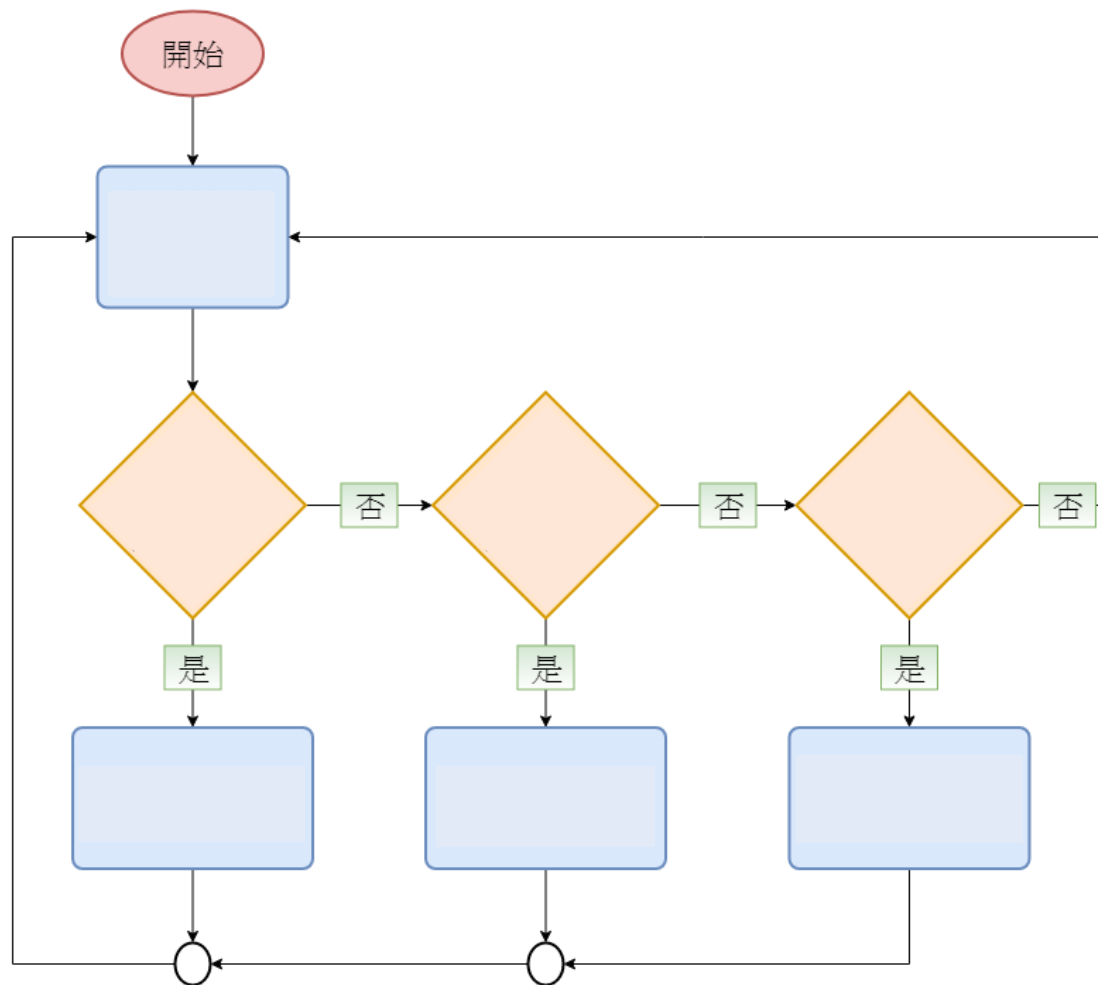
練習--情境主題：鐵捲門

演算法步驟(教師用)

1. 讀取搖桿方向值及按鍵值X、Y、Button
2. 判斷 ($X < 50$ 且 mode 不等於 2)
 - 2-1. 成立：馬達正轉(上移)，綠燈亮，設定mode=1
3. 否則：
 - 3-1. 不成立：再判斷 ($X > 800$ 且 mode 不等於 1)
 - 3-1-1. 成立：馬達逆轉(抽風)，藍燈亮，設定mode=2
 - 3-2. 否則：
 - 3-2-1. 不成立：再判斷 Button = 1
 - 3-2-1-1. 成立：風扇停止，LED燈滅，設定mode=3
 - 3-2-2. 否則：
 - 3-2-2-1. 不成立，回到步驟1

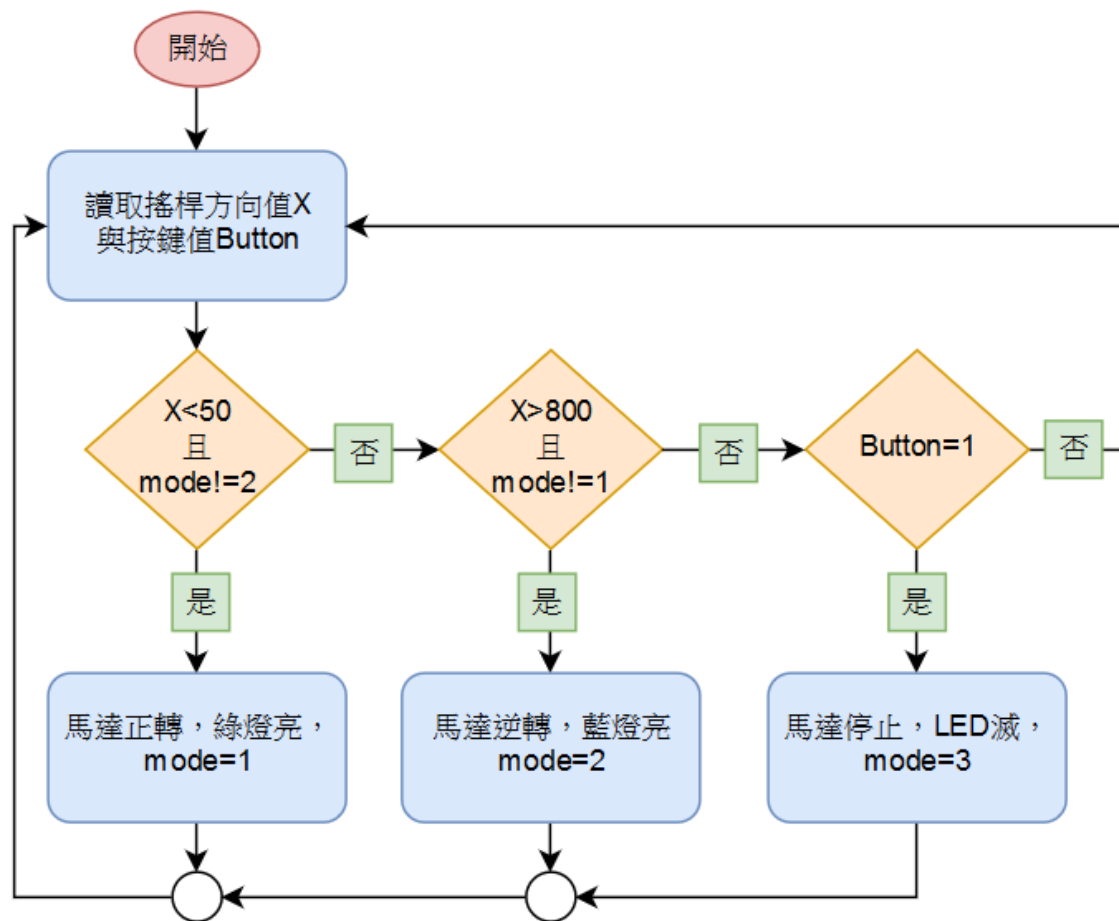
練習--情境主題：鐵捲門

程式流程圖(學生填空用)



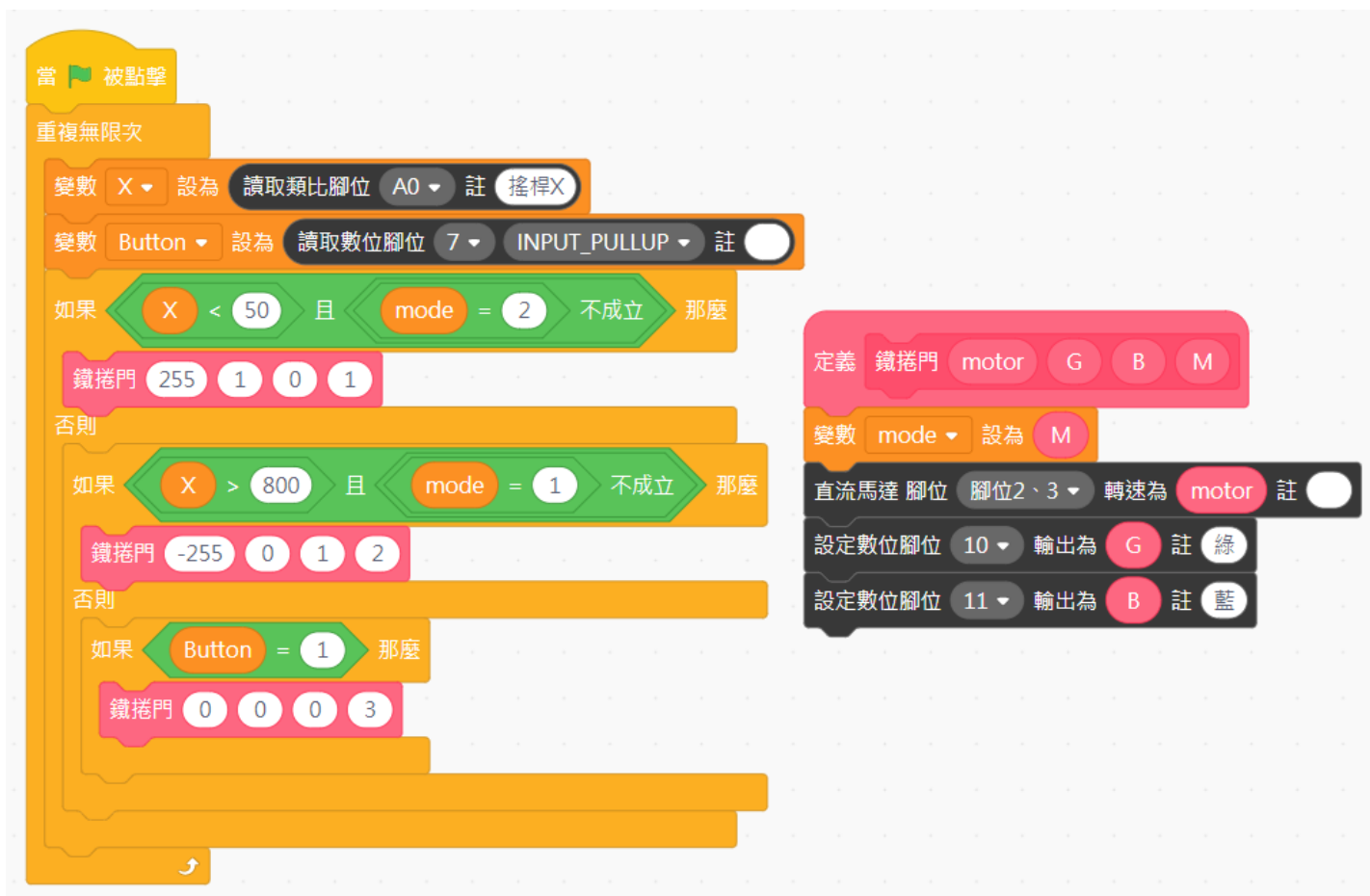
練習--情境主題：鐵捲門

程式流程圖(教師用)



練習--情境主題：鐵捲門

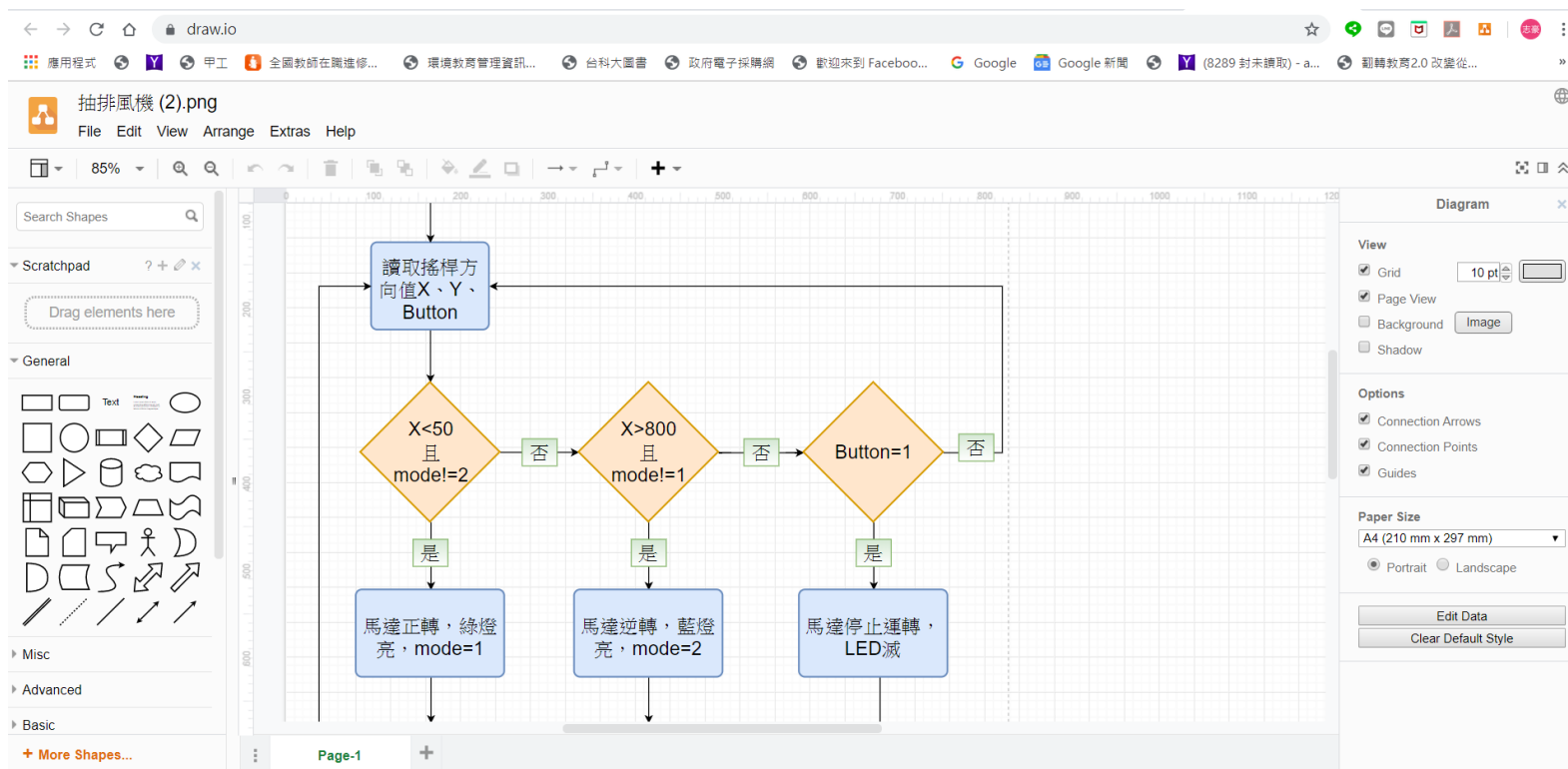
程式積木堆疊(教師用)



流程圖繪製軟體

draw.io

○ www.draw.io



流程圖繪製軟體

fChart

o fChart

