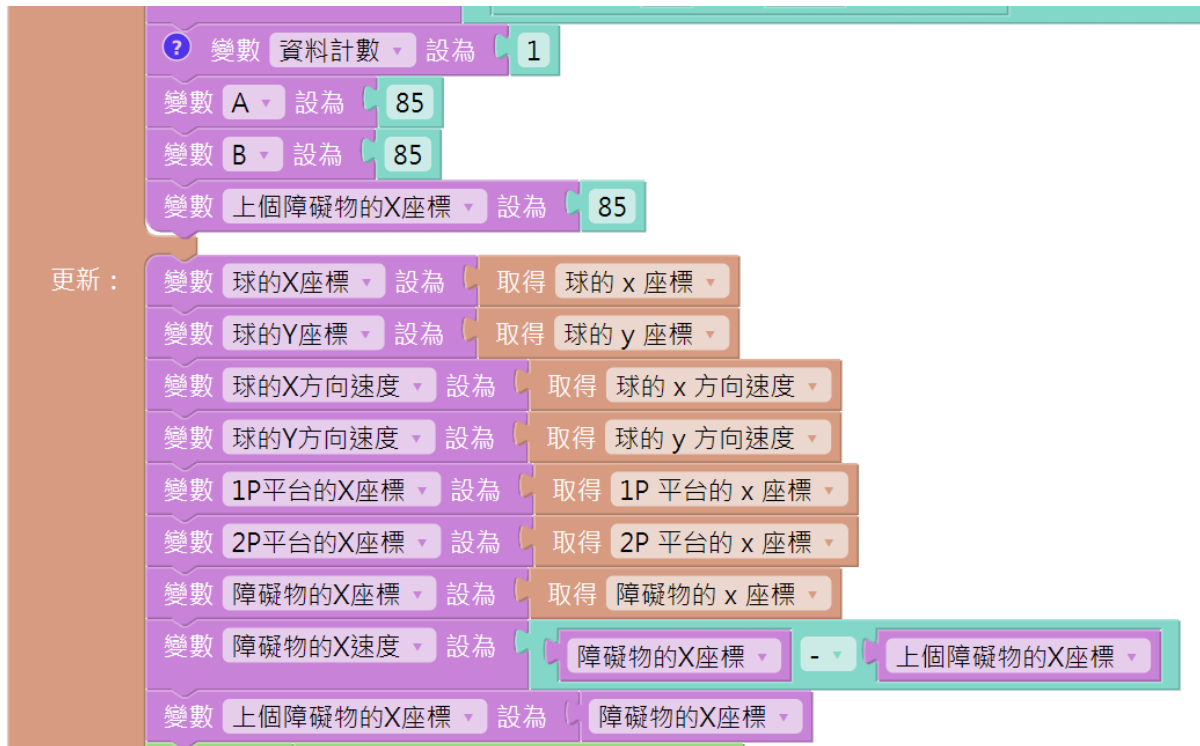


※ 1P auto.xml

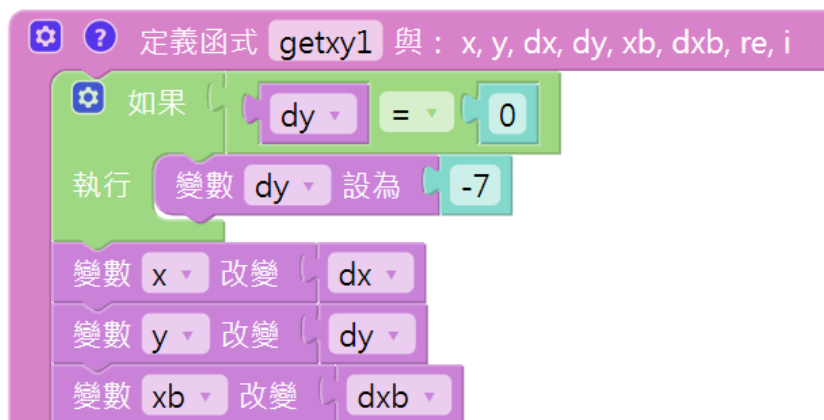
Step1.初始化



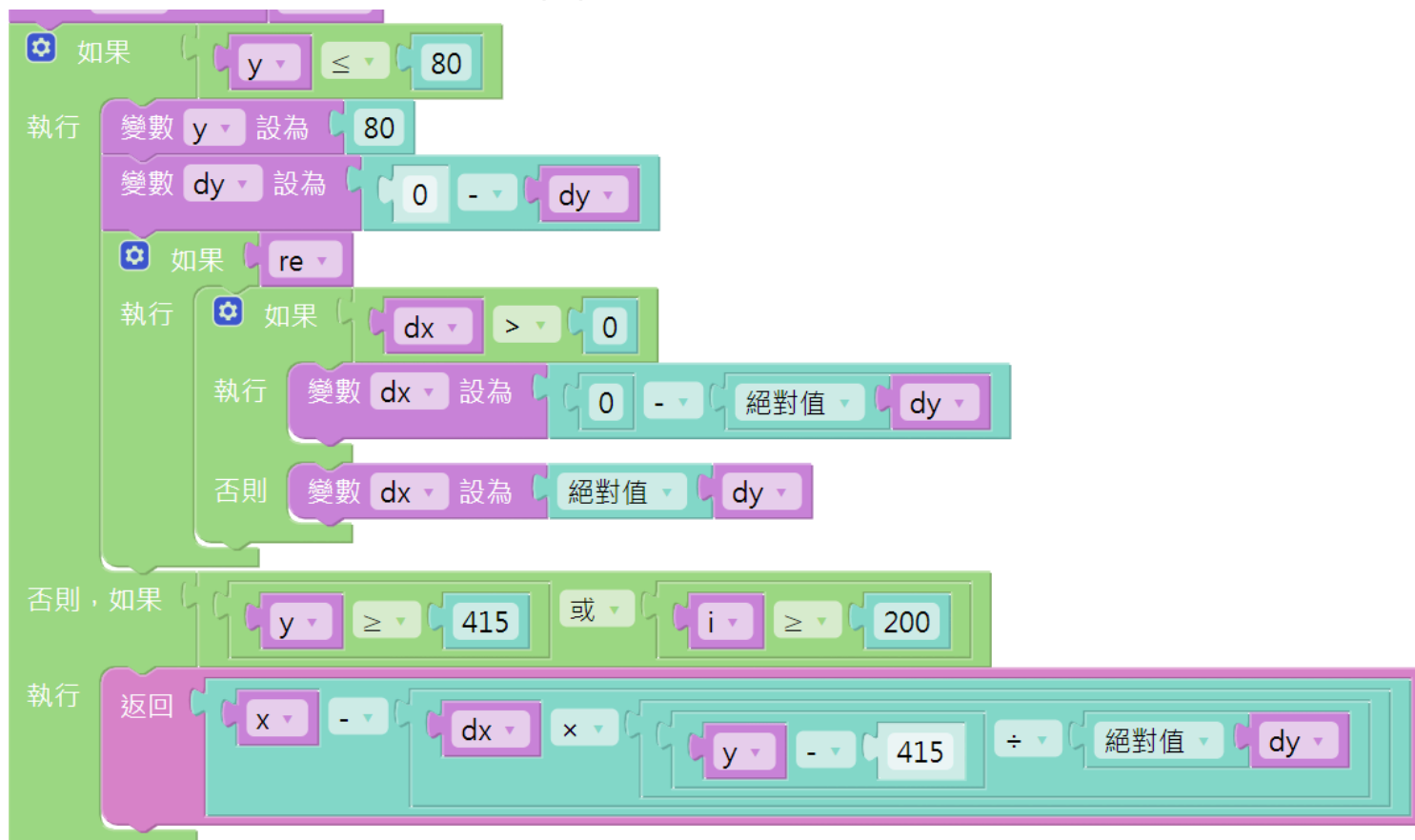
Step2.設定1p函式 及 AB點計算



Step3. 變數每次迴圈都要前進，dy不能為 0

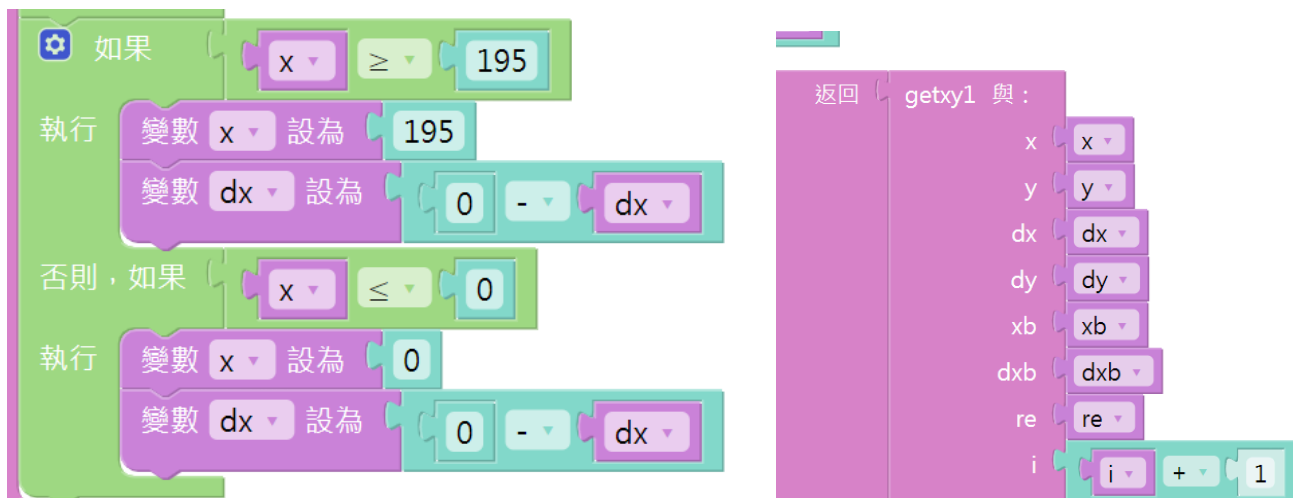


Step4.處理1P y 方向一般碰撞函數返回值，y大於 415就返回計算好的x碰撞點，要考慮對手接球時是否有切球(re)



Step5.處理x 方向一般碰撞

Step6.處理返回值

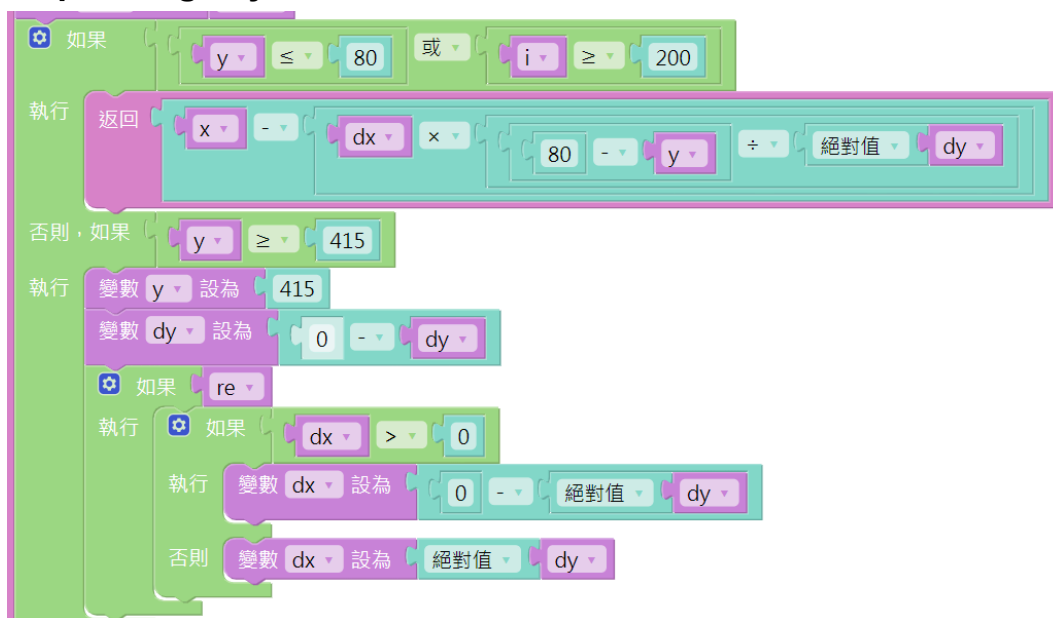


※ 2P

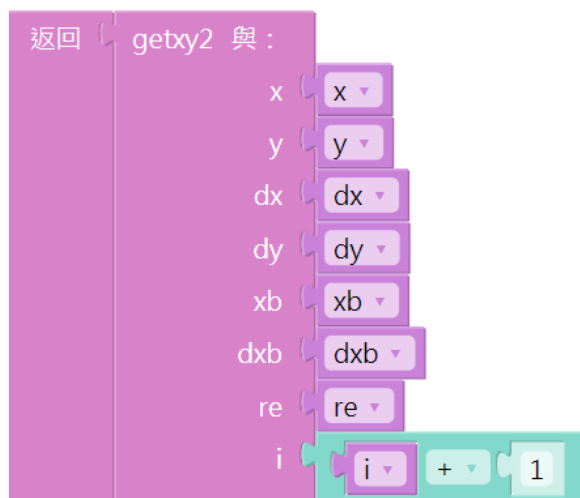
Step1 複製 getxy1 得到 getxy2，寫好 2P預判落點



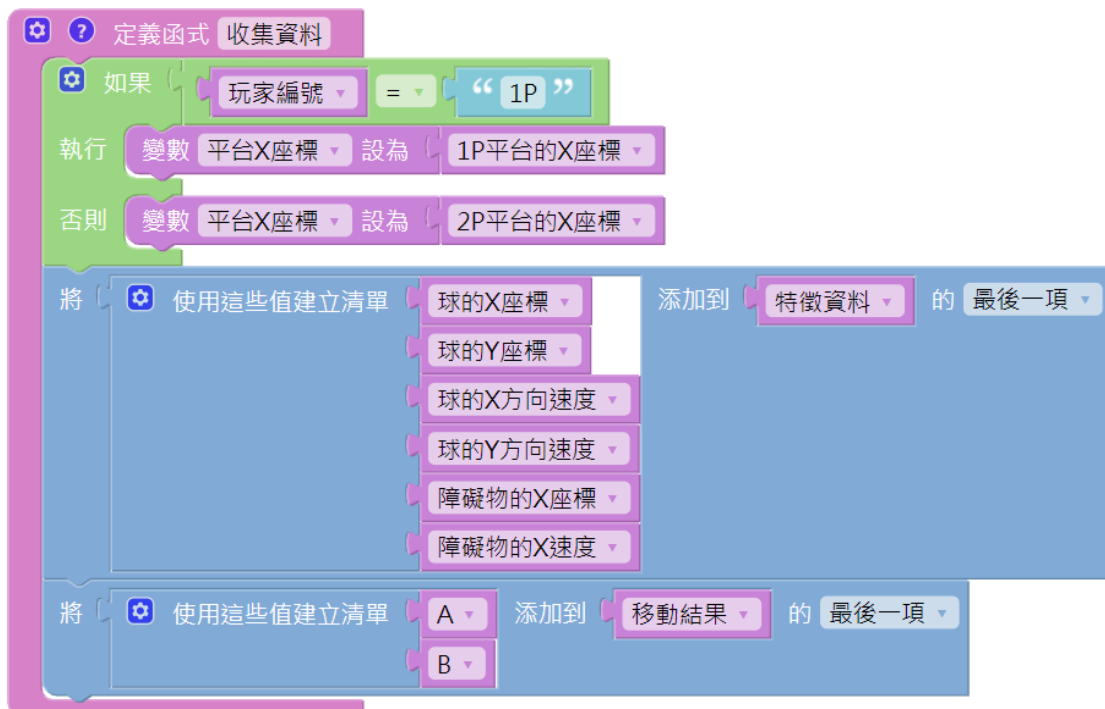
Step2.修改getxy2 邊界對調，415時反彈,80時返回，公式的部分也要調整好



Step3. x 方向一般碰撞和1P一樣，別忘了尾部遞迴的返回函數要用getxy2



※收集資料



※訓練程式(截錄1P)

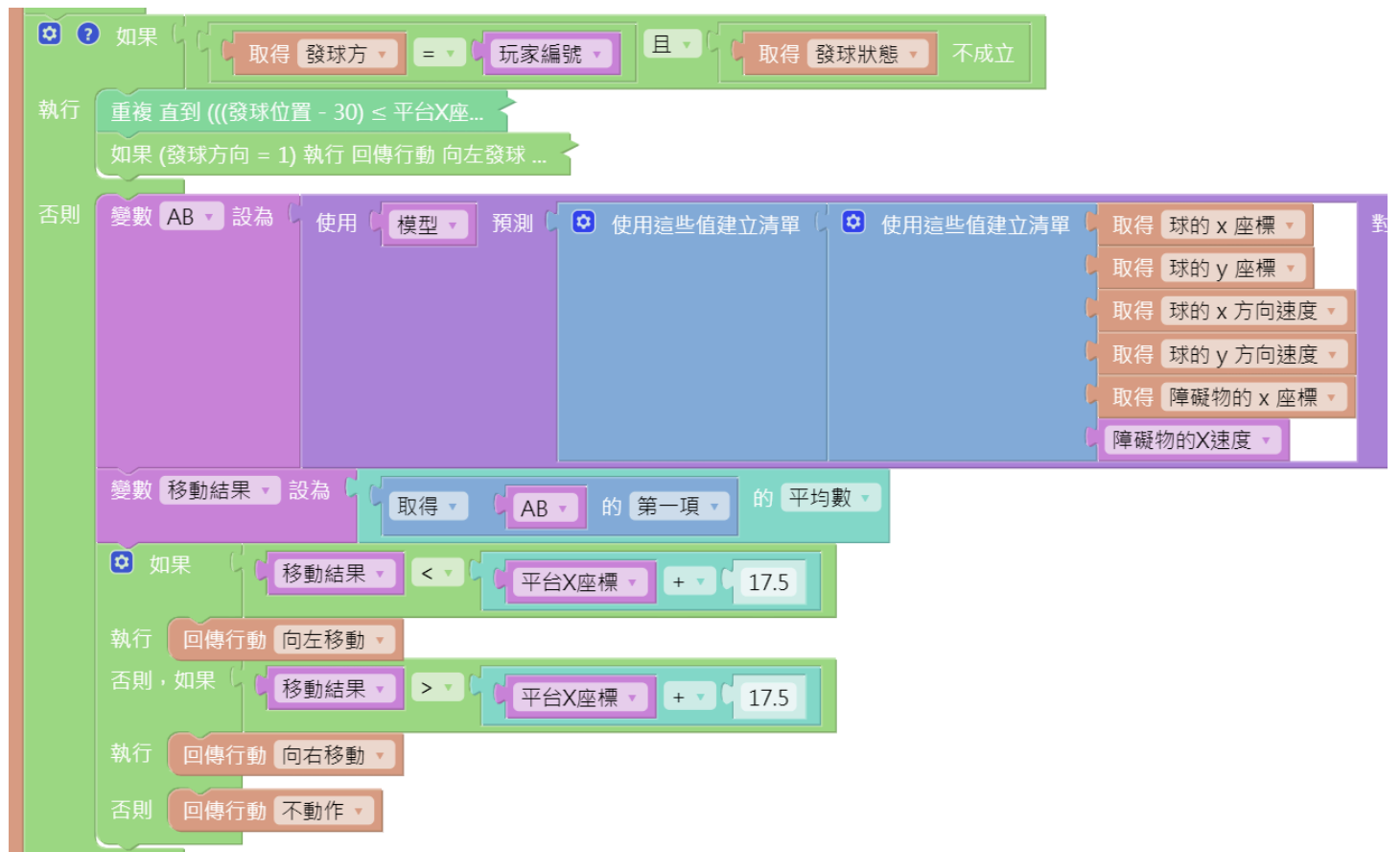


※ml_play.xml

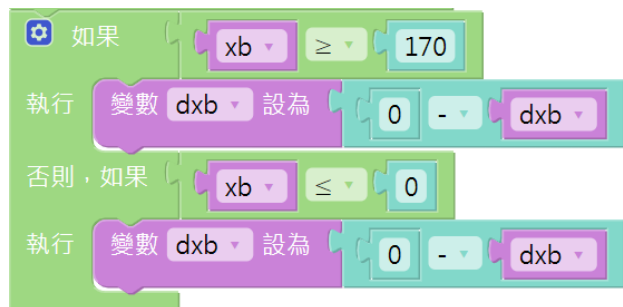
Step1.處理障礙物x速度及模型設定



Step2. 使用模型預測得到AB點再取平均變成預測的落點(移動結果)，確認回傳行動(球往哪裡動)



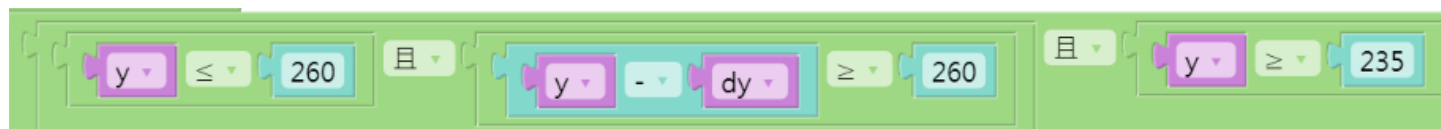
※障礙物碰左右壁



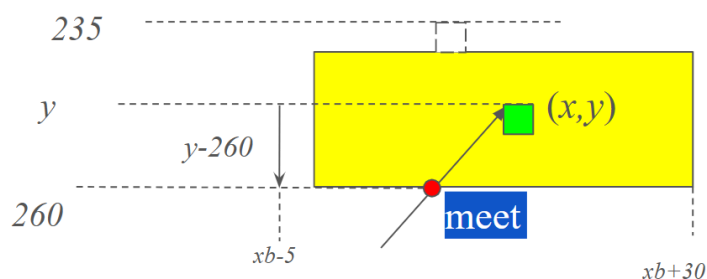
※1P加入障礙物判斷

Step1.處理往上碰障礙物 (1P加入障礙物判斷)

前提： $dy < 0, 235 \leq y \leq 260$ 且 $y - dy \geq 235$ ($y - dy$ 即上個 y)



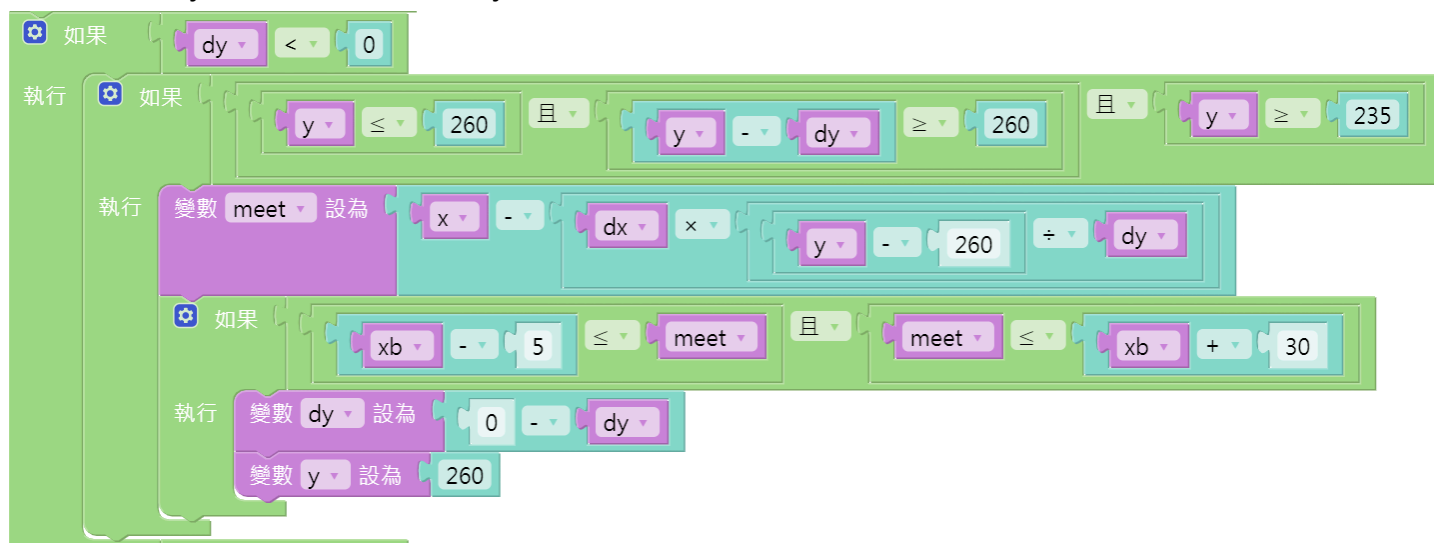
算出碰撞點 meet



$$meet = x - dx \frac{y - 260}{dy}$$

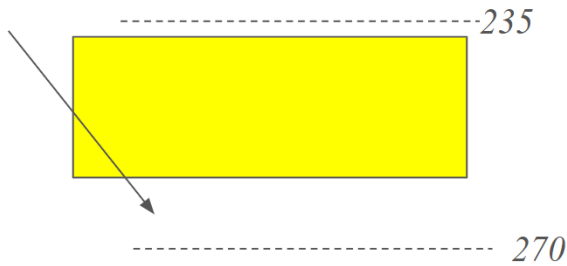
$xb - 5 \leq meet \leq xb + 30$ (障礙物長度為30, xb :障礙物左邊界)

撞到了的話 y 要向下推回 260 且 y 方向速度要反向

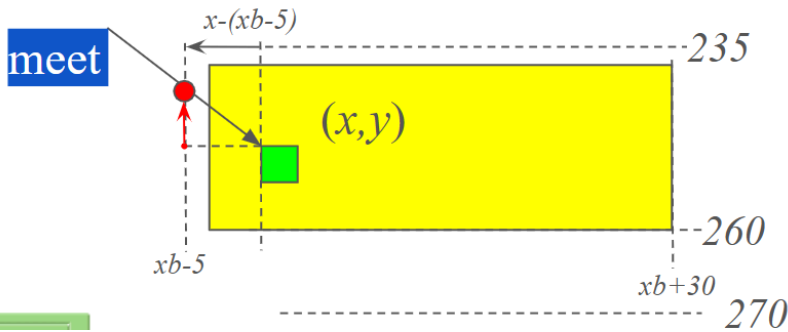


Step2.處理往右下碰障礙物 (1P加入障礙物判斷)

前提： $dy > 0, dx > 0, 260 \leq y \leq 270$ ， $x \geq xb-5$ 且 $x-dx \leq xb-5$ ($x-dx$ 即上個x)



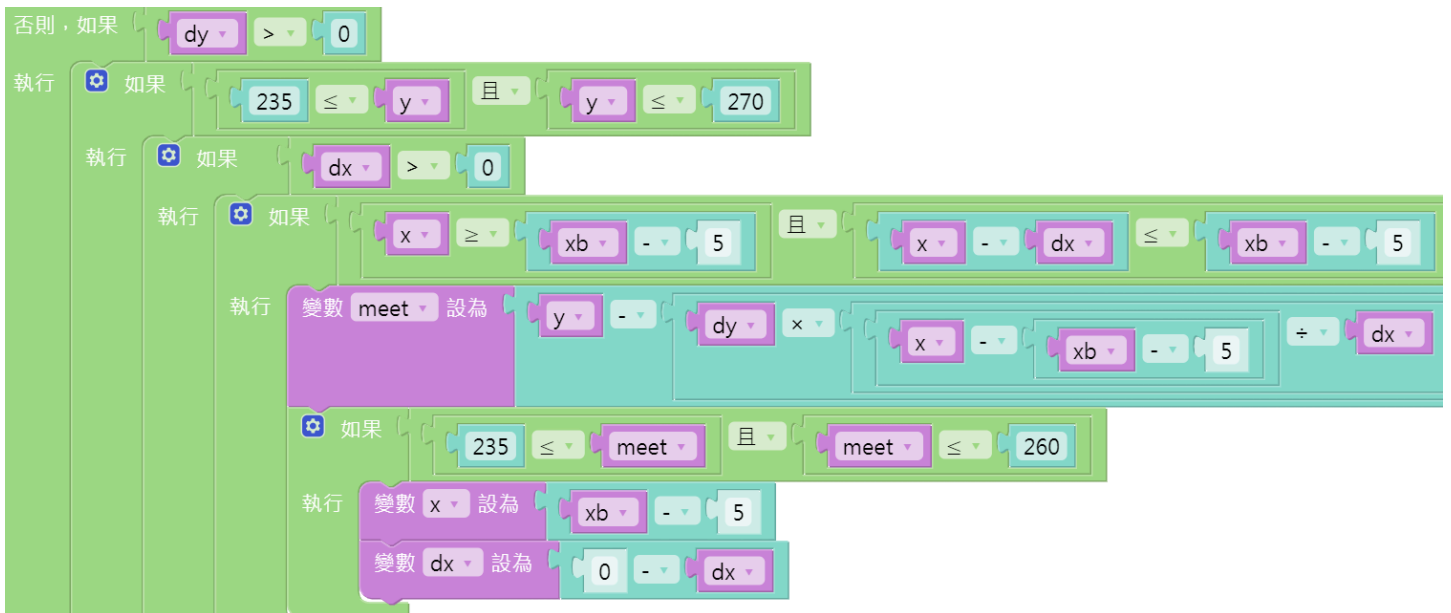
算出碰撞點 meet



$$meet = y - dy \frac{x - (xb - 5)}{dx}$$

$235 \leq meet \leq 260$

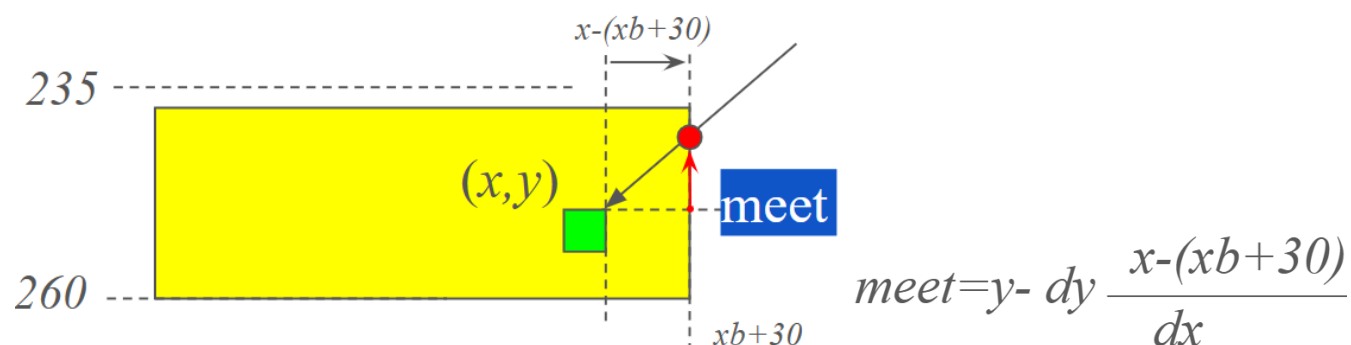
撞到了的話 x要往左推回 xb-5 且x方向速度要反向



Step3.處理往左下碰障礙物 (1P加入障礙物判斷)

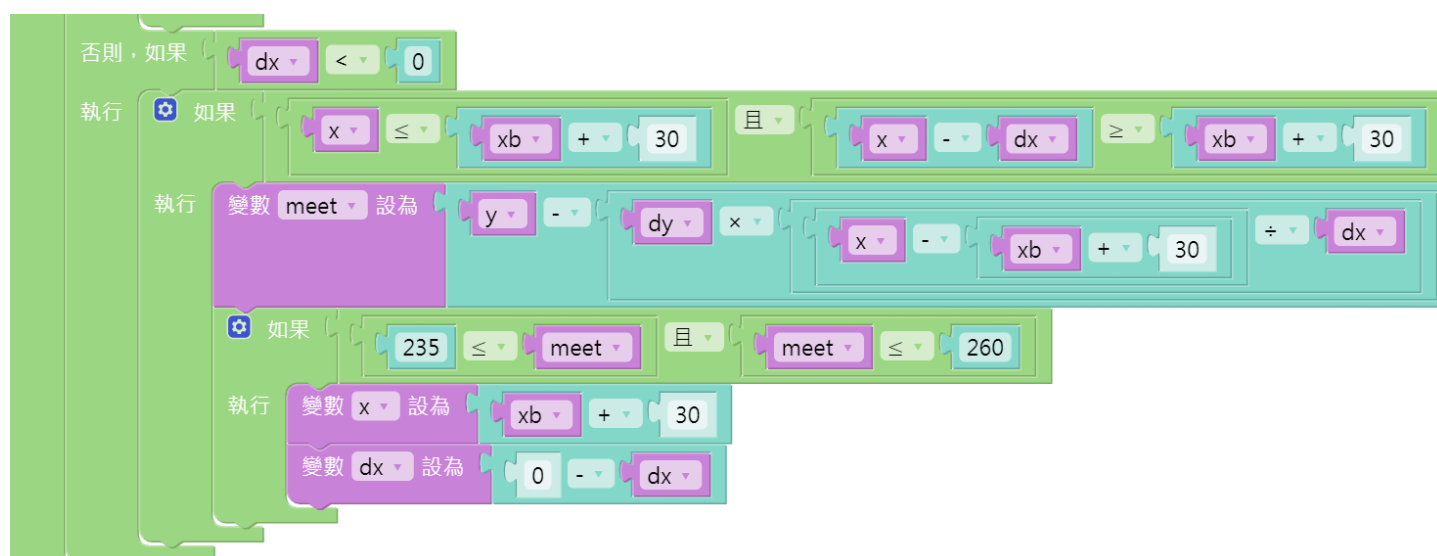
前提： $dy > 0, dx < 0, 260 \leq y \leq 270, x \leq xb + 30$ 且 $x - dx \geq xb + 30$ ($x - dx$ 即上個x)

算出碰撞點 meet



$235 \leq meet \leq 260$

撞到了的話 x要往左推回 xb-5 且x方向速度要反向

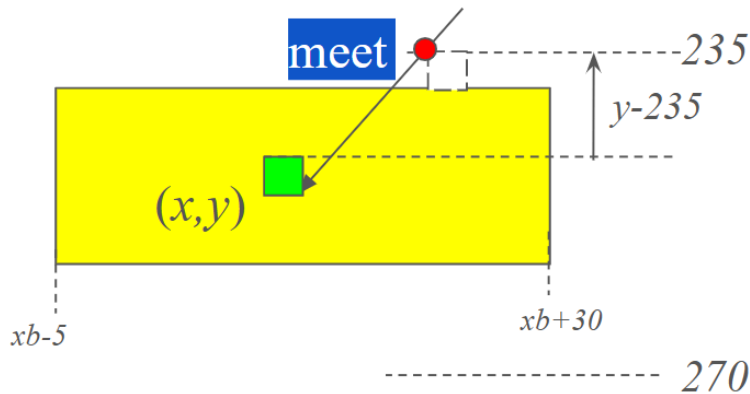


※2P加入障礙物判斷

Step1.處理往下碰障礙物 (2P加入障礙物判斷)

前提：dy>0, 235≤y≤270 且 y-dy≤235 (y-dy即上個y)

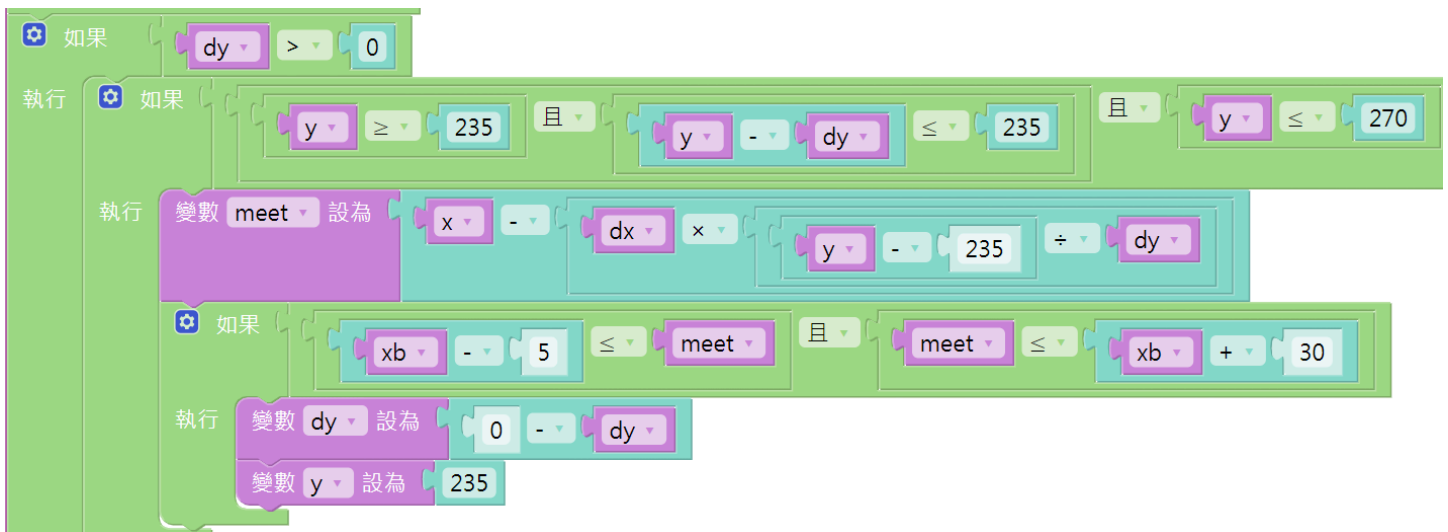
算出碰撞點 meet



$$meet = x - dx \frac{y - 235}{dy}$$

$xb - 5 \leq meet \leq xb + 30$ (障礙物長度為30,xb:障礙物左邊界)

撞到了的話 y要向上推回 235 且y方向速度要反向



Step2.處理往右上碰障礙物 (2P加入障礙物判斷)

Diagram illustrating the collision detection logic for moving up and to the right. The yellow rectangle represents the obstacle, and the green square represents the object at position (x, y) . The red dot indicates the point of collision at $(xb-5, 235)$. The formula for the collision point is:

$$meet = y - dy \frac{x - (xb - 5)}{dx}$$

The code block shows the implementation of this logic:

```

否則，如果
  執行 如果
    225 ≤ y 且 y ≤ 260
    執行 如果
      dx > 0
      執行 如果
        x ≥ xb - 5 且 x - dx ≤ xb - 5
        執行 變數 meet 設為
          (y - dy * (x - (xb - 5)) / dx)
        執行 如果
          235 ≤ meet 且 meet ≤ 260
          執行 變數 x 設為
            xb - 5
          執行 變數 dx 設為
            0 - dx
  
```

同1P

Step3.處理往左上碰障礙物 (2P加入障礙物判斷)

Diagram illustrating the collision detection logic for moving up and to the left. The yellow rectangle represents the obstacle, and the green square represents the object at position (x, y) . The red dot indicates the point of collision at $(xb+30, 235)$. The formula for the collision point is:

$$meet = y - dy \frac{x - (xb + 30)}{dx}$$

The code block shows the implementation of this logic:

```

否則，如果
  執行 如果
    dx < 0
    執行 如果
      x ≤ xb + 30 且 x - dx ≥ xb + 30
      執行 變數 meet 設為
        (y - dy * (x - (xb + 30)) / dx)
      執行 如果
        235 ≤ meet 且 meet ≤ 260
        執行 變數 x 設為
          xb + 30
        執行 變數 dx 設為
          0 - dx
  
```

同1P