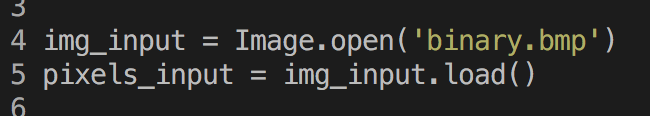
**CV hw7 / 電機所R06921082 陳與賢**

**Description:**

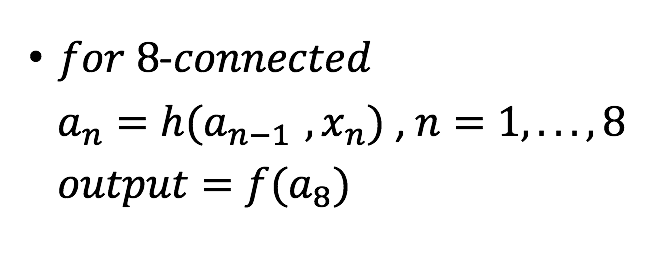
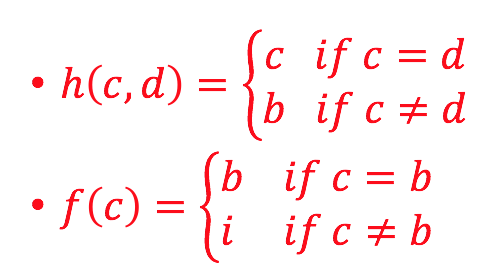
利用python來處理bmp檔，進行**Thinning**

首先讀入hw2的binary圖檔，code如下：



**Algorithm:**

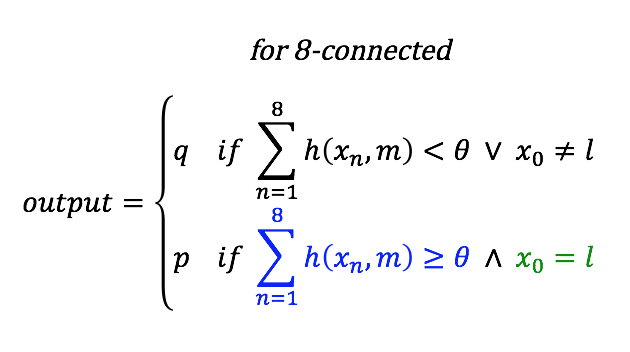
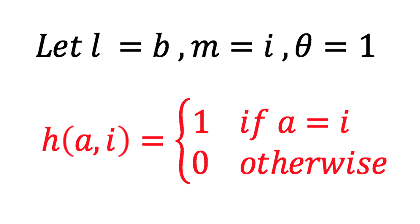
首先要先做marked-interior/border-pixel operator，依照投影片8-connectivity的作法標出是”b” （我記為1000）還是”i"（我記為2000），如下圖所示：



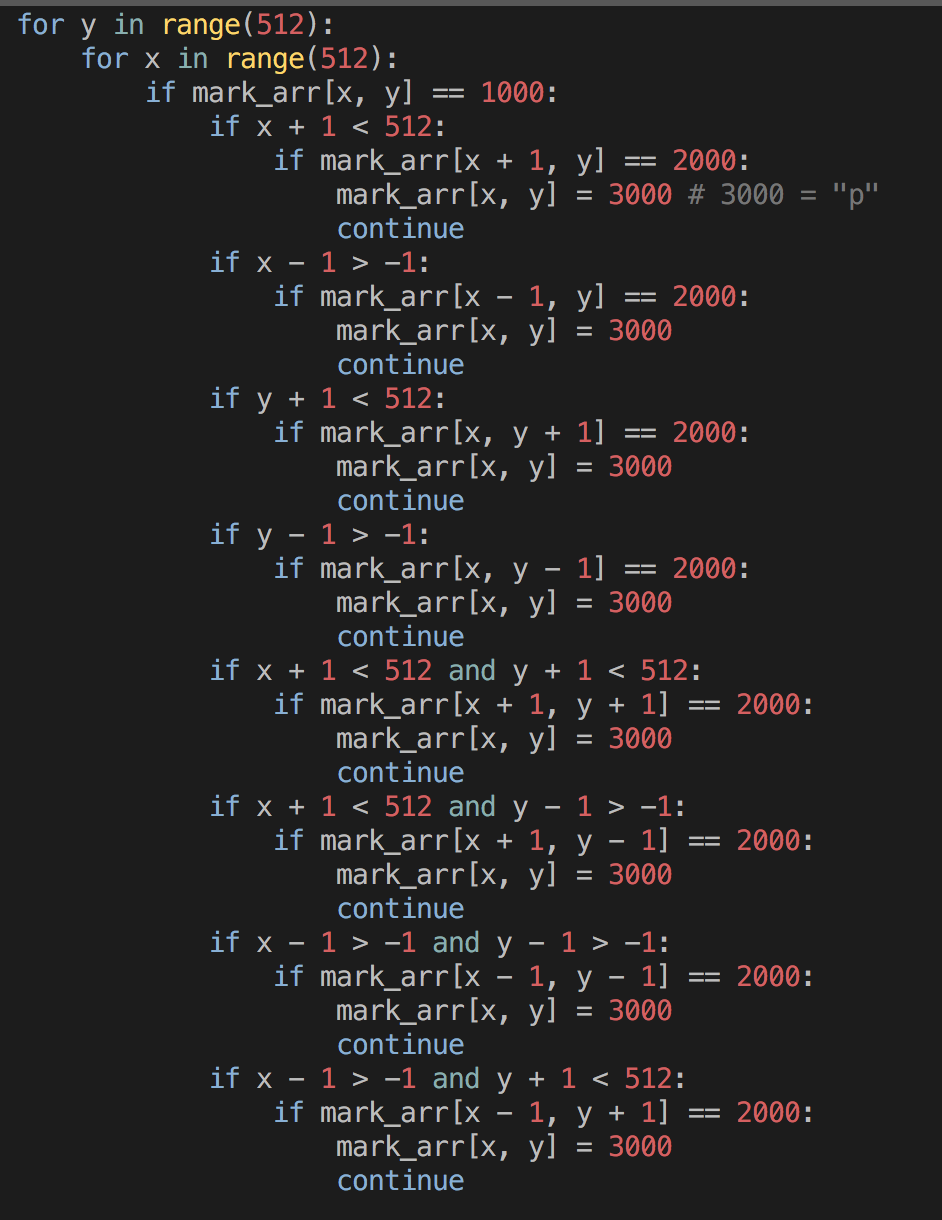
因為投影片沒有特別說明針對boundary的pixel要如何處理，因此我假設在boundary之pixels皆會被當作border，也就是我一開始會給超出boundary的pixels給值100（因為binary的輸入圖只會有255跟0，所以掃到100的話，當前檢查的值就會被標成”b”），code如下：



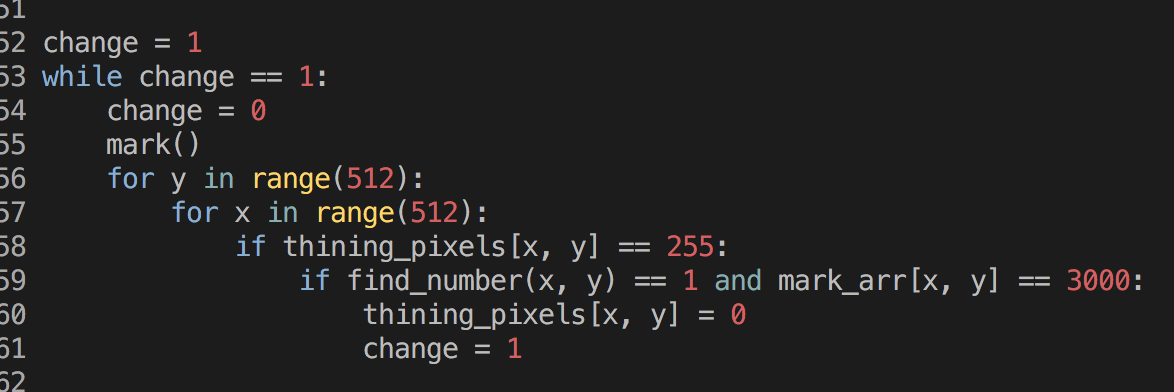
再來要做pair relationship operator，依據投影片8-connectivity的作法來relabel，如下圖所示：



這邊threshold的值因為作業沒有特別說明，所以我們設成跟投影片一樣θ = 1，白話一點也就是如果當前的pixel被標記為”b” （我記為1000），而其八連通的pixel有任一個被標記為”i” （我記為2000）的話，那當前的pixel被relabel為p（我記為3000），code如下：



再來進行最後的部分，也就是connected shrink operator，依據公式會發現其實就是yokoi number = 1的時候，因此我們從圖的左上到右下，依序對每個pixel值執行hw6的演算法找出yokoi number，若yokoi number = 1且相應的mark為”p”（我記為3000），則當前pixel值改成0。另外此演算法需不斷迭代直到沒有改變為止，code如下：

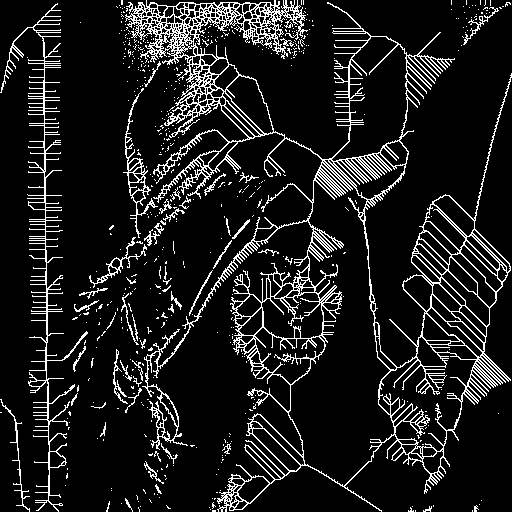


另外marked-interior/border-pixel operator跟pair relationship operator我都是用8-connectivity，yokoi number也用8-connectivity的話result跟助教給的解答差蠻多的…，所以也附上yokoi number用4-connectivity的result，跟助教給的解答較相近。

所以我code會產生兩個圖，一個yokoi number是4- connectivity，另一個是8-connectivity

（marked-interior/border-pixel operator跟pair relationship operator都是8-connectivity，因為我看投影片第78頁有標8-connextivity）

**Result:**

**Yokoi number 4-connectivity**

**Yokoi number 4-connectivity**

****