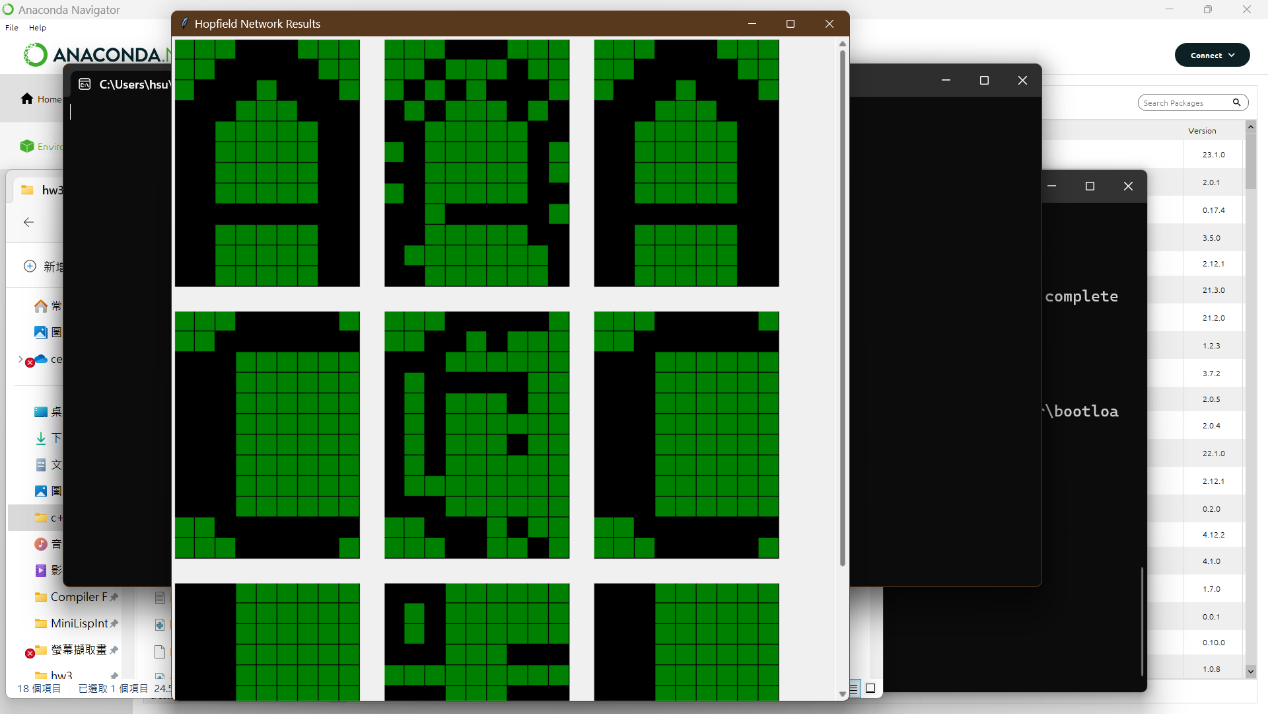
**類神經網路作業三**

**110502516許尚軒**

1. **執行說明**

執行.exe檔後就會跑出結果(exe檔要與data放在同一層資料夾)

其中左側為訓練資料中間為測試資料右側為回想結果，最右側有scrollbar可以往下拉



**basic.exe**: basic data

**bonus2.exe**: bonus data

**bonus.exe**: bonus data but bias is 0

**noice\_basic.exe**: basic data making noice

**noice\_nonus.exe**: bonus data making noice

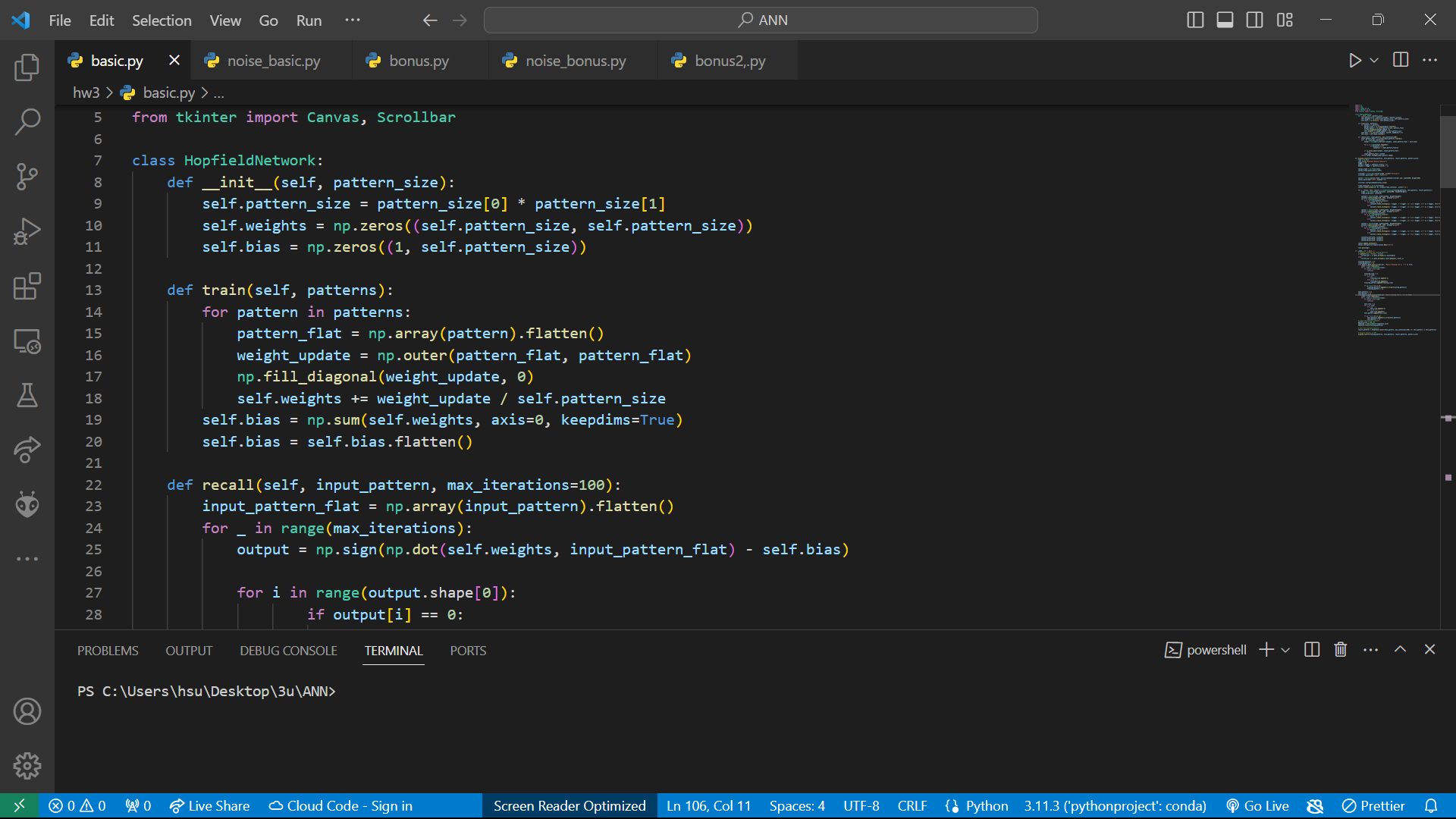
1. **程式簡介:**

**本次作業我是採用hopfield類神經網路架構再將結果用tkinter顯示出來**

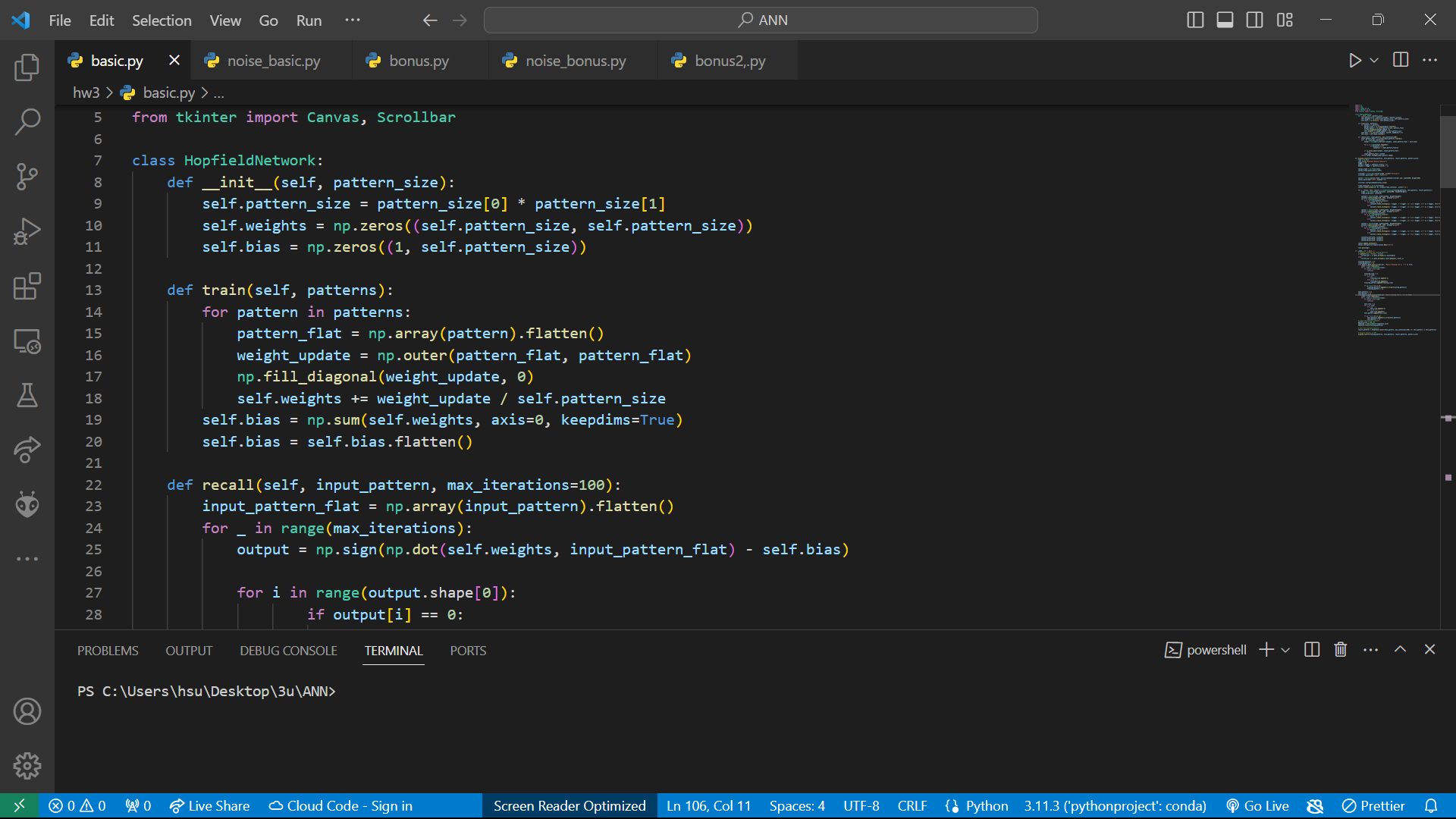
**Hopfield:**



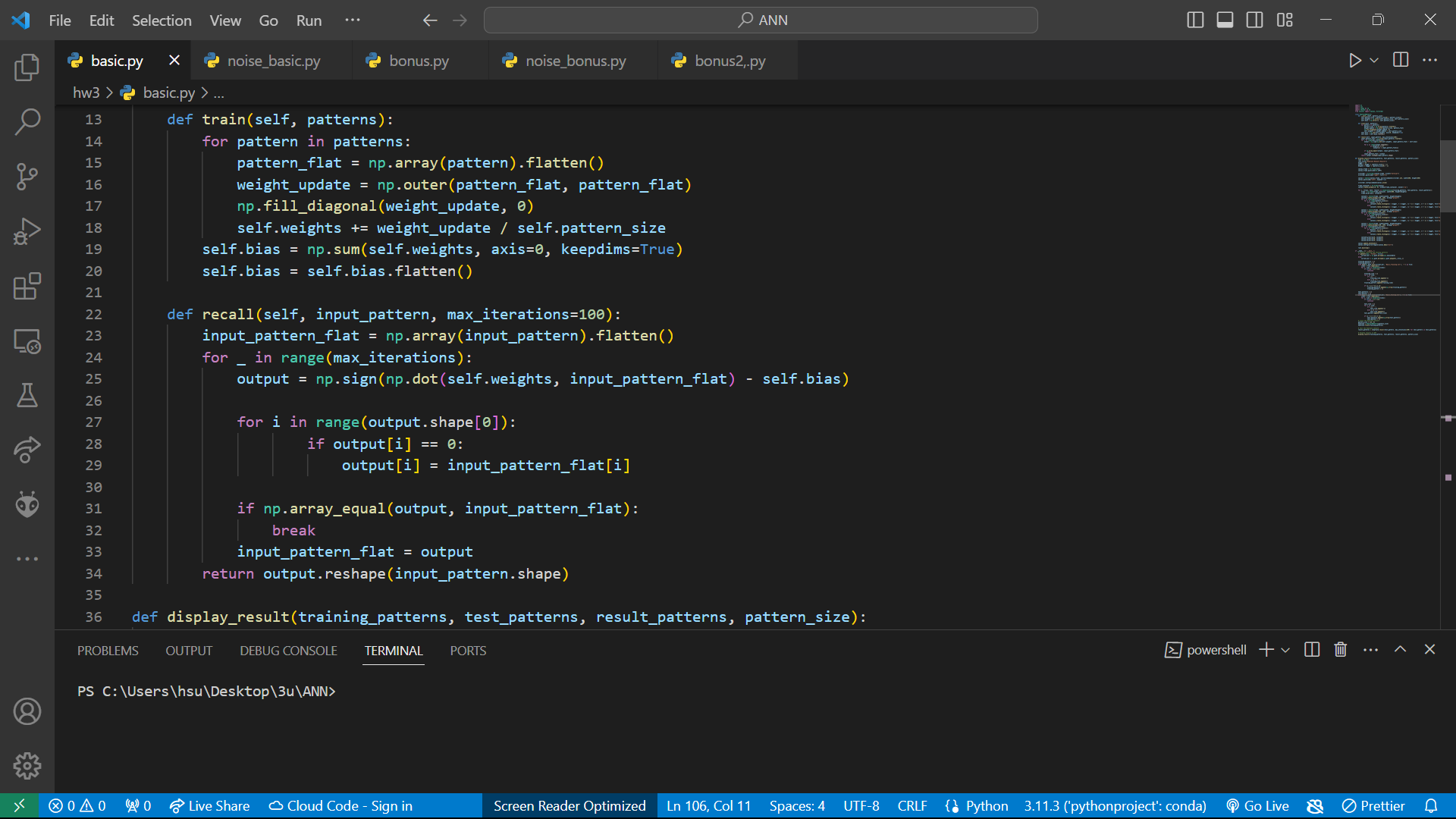
一開始先定義測資格式及初始化鍵結值和閥值為0



訓練時依照公式 改變閥值和鍵結值

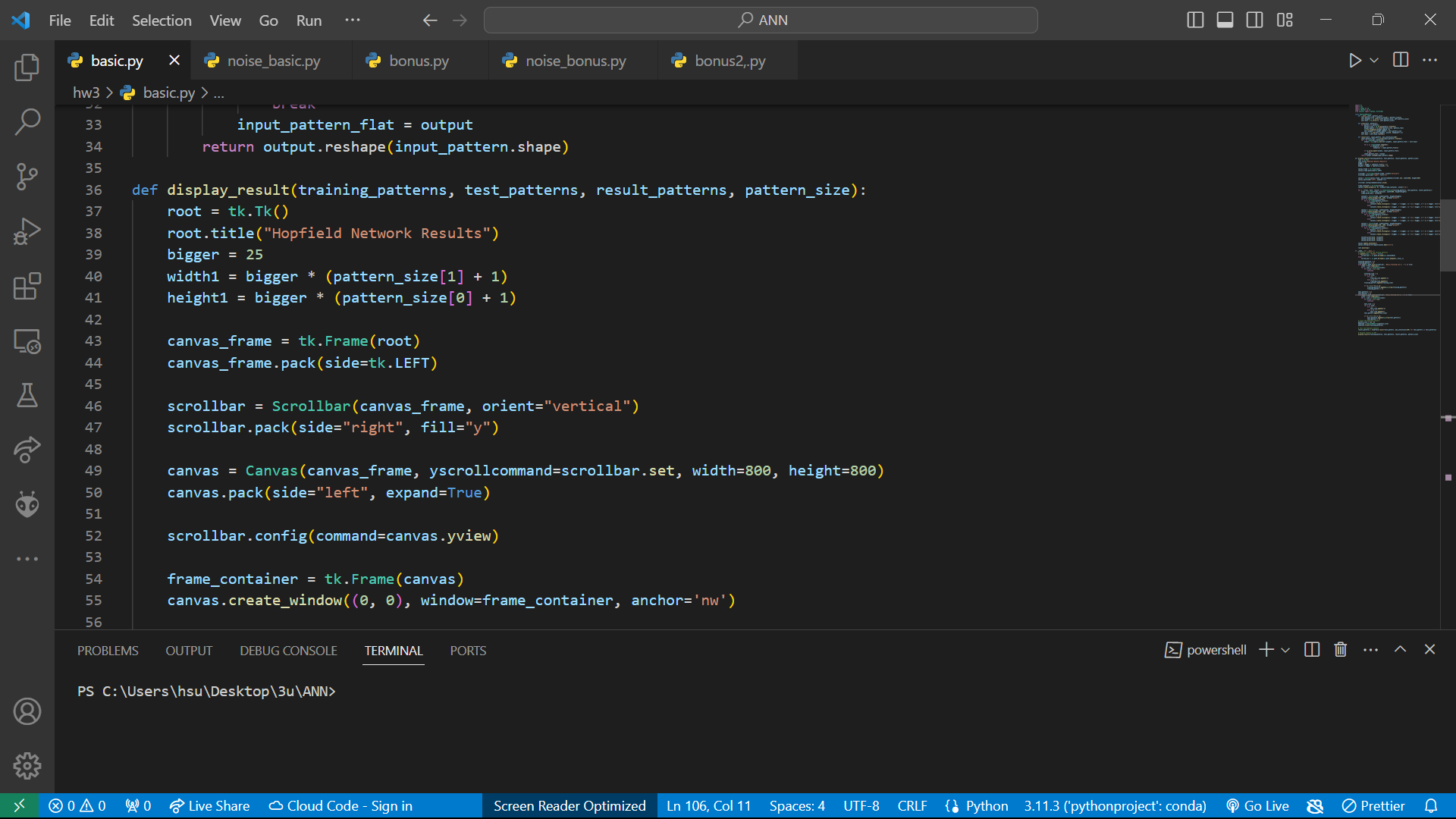


回想時根據鍵結值和閥值一步步回想直到達到最大回想次數(在此設為100)或是達到穩定狀態

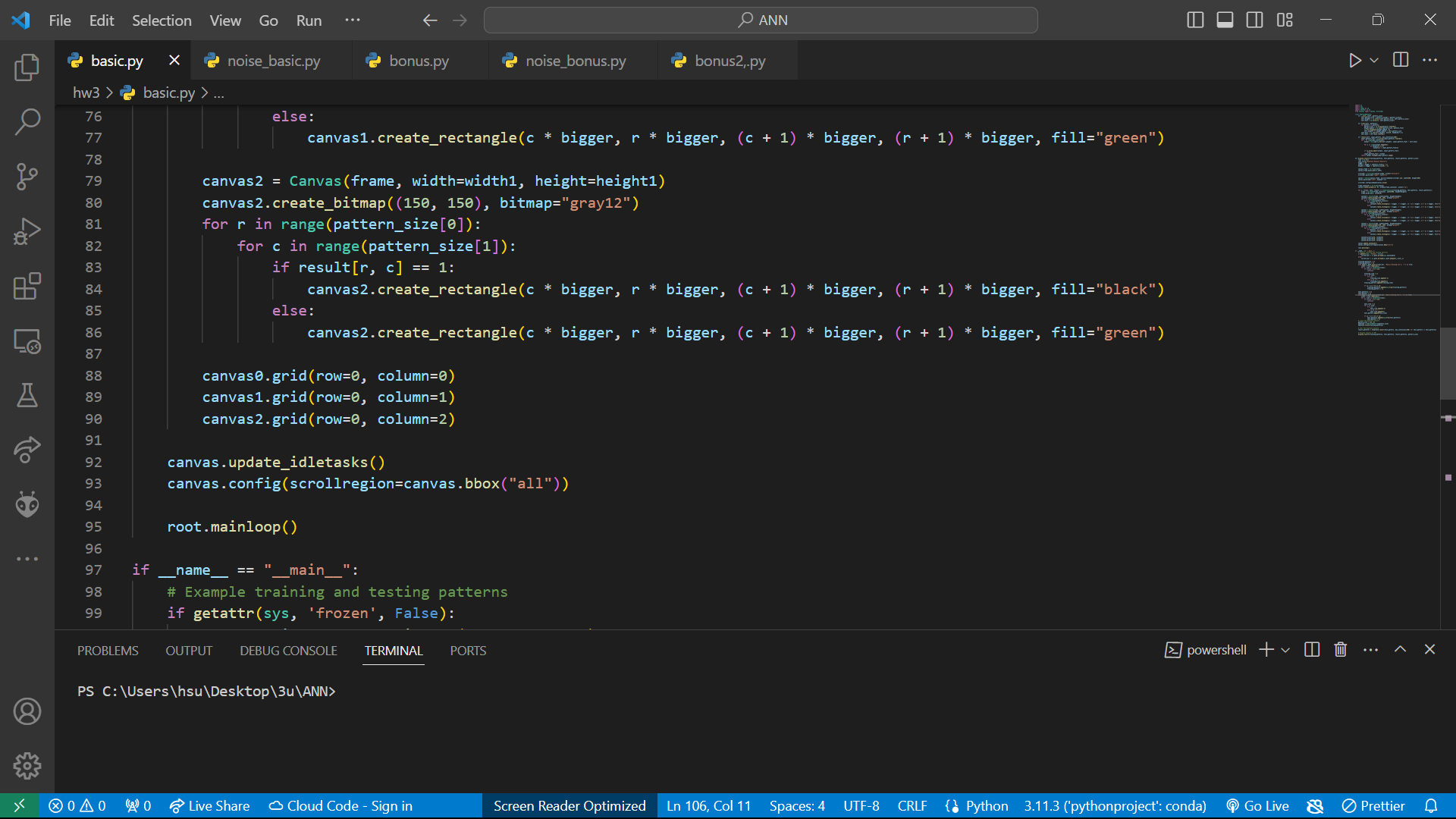


**GUI:**

一些基本設定

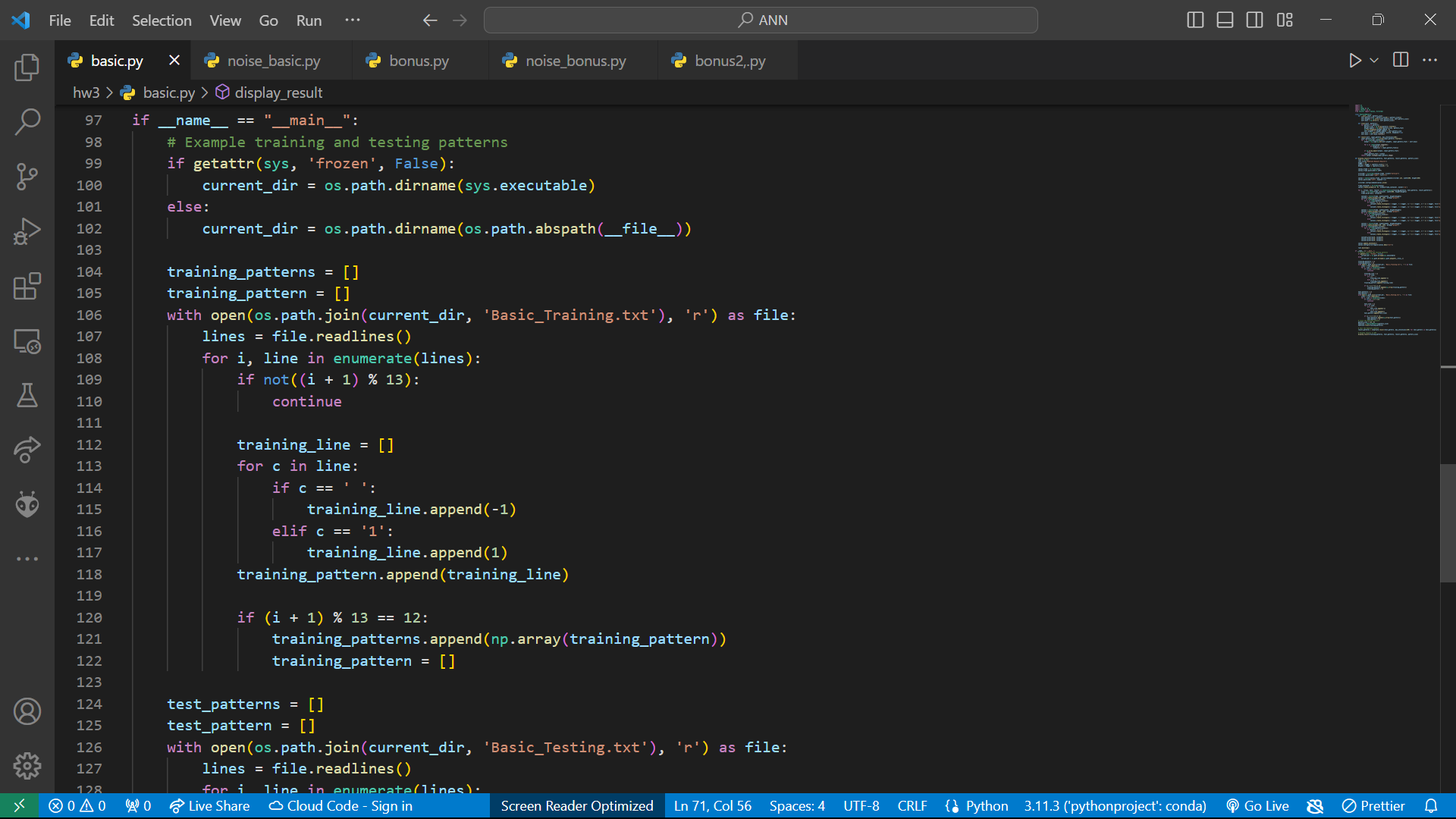


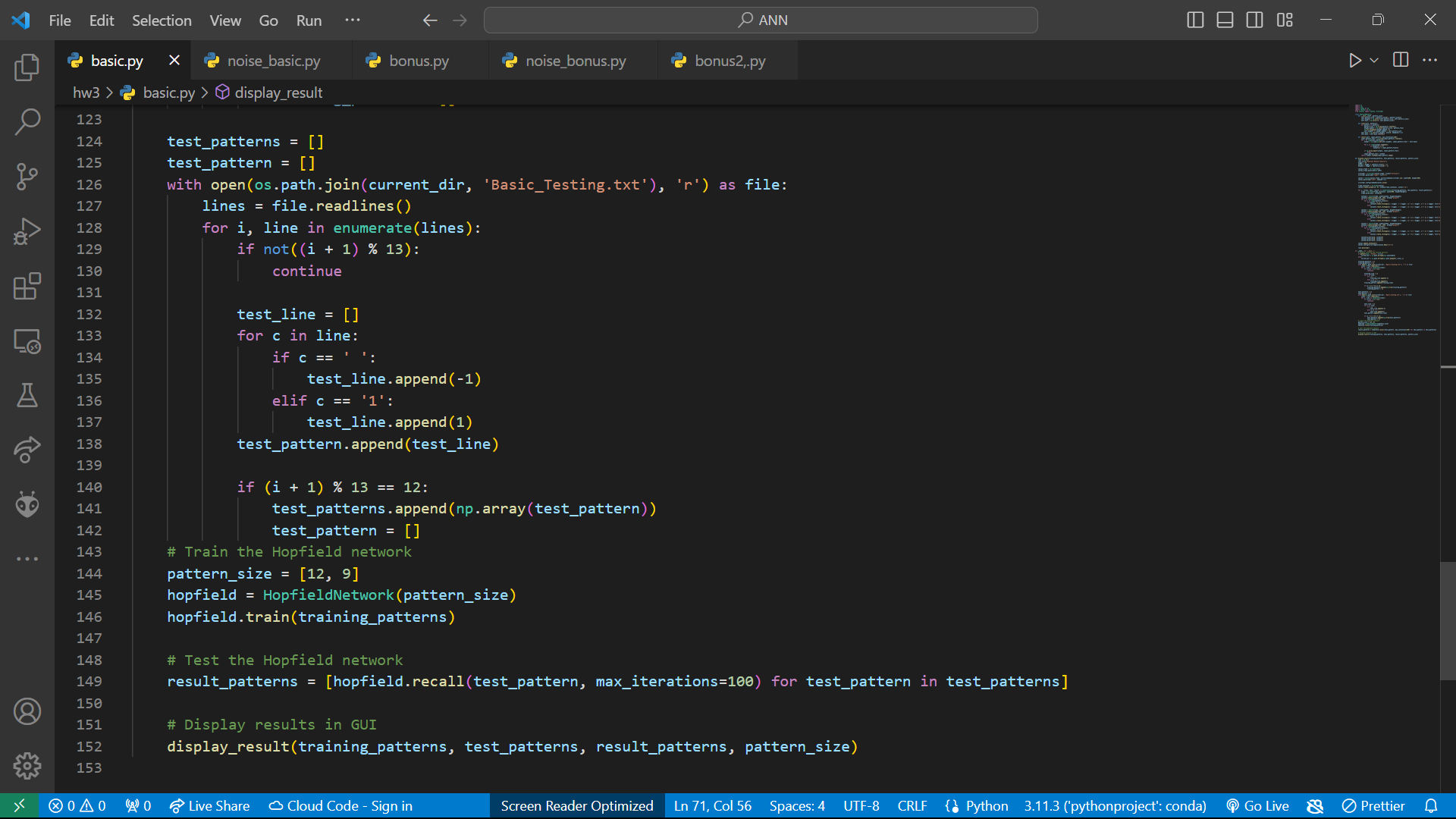
將每一筆輸入顯示，出來包含train data(canva0), test data(canva1), recall(canva2)



**主程式:**

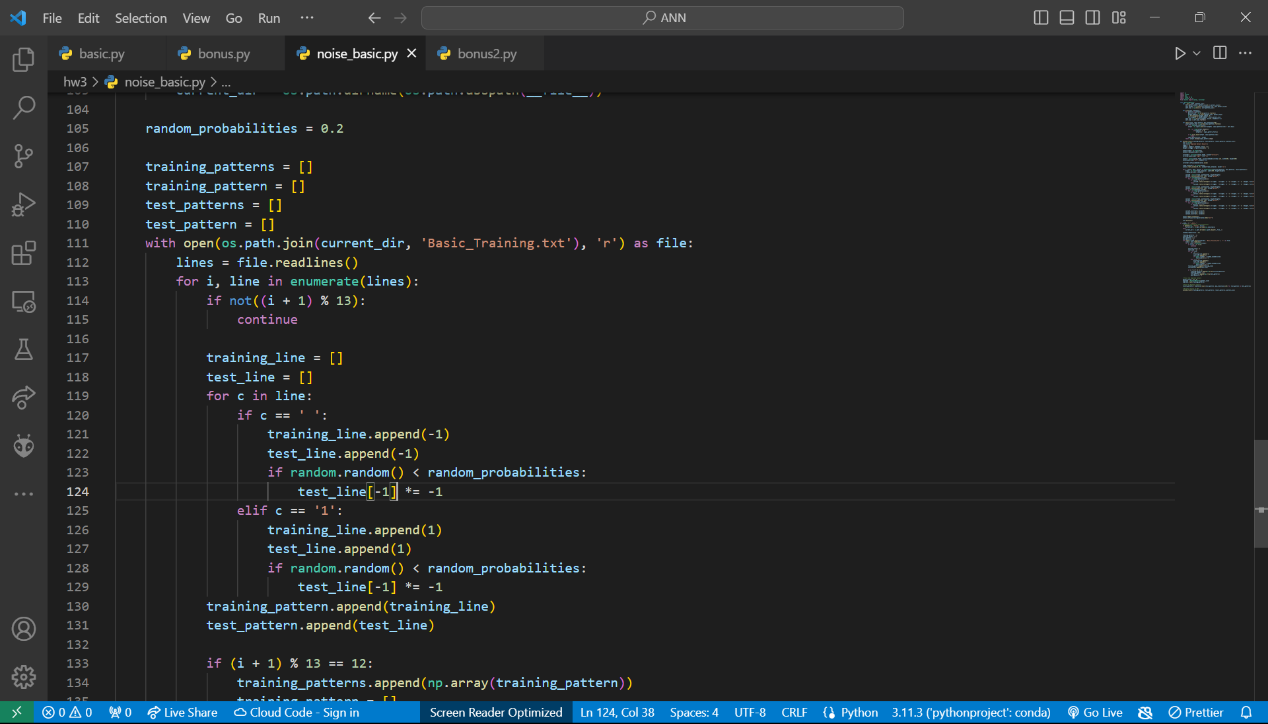
獲取basic trainning data及basic test data並將1當成1，空格當成-1，然後將上述資料變成nparray的list送去訓練及回想然後顯示

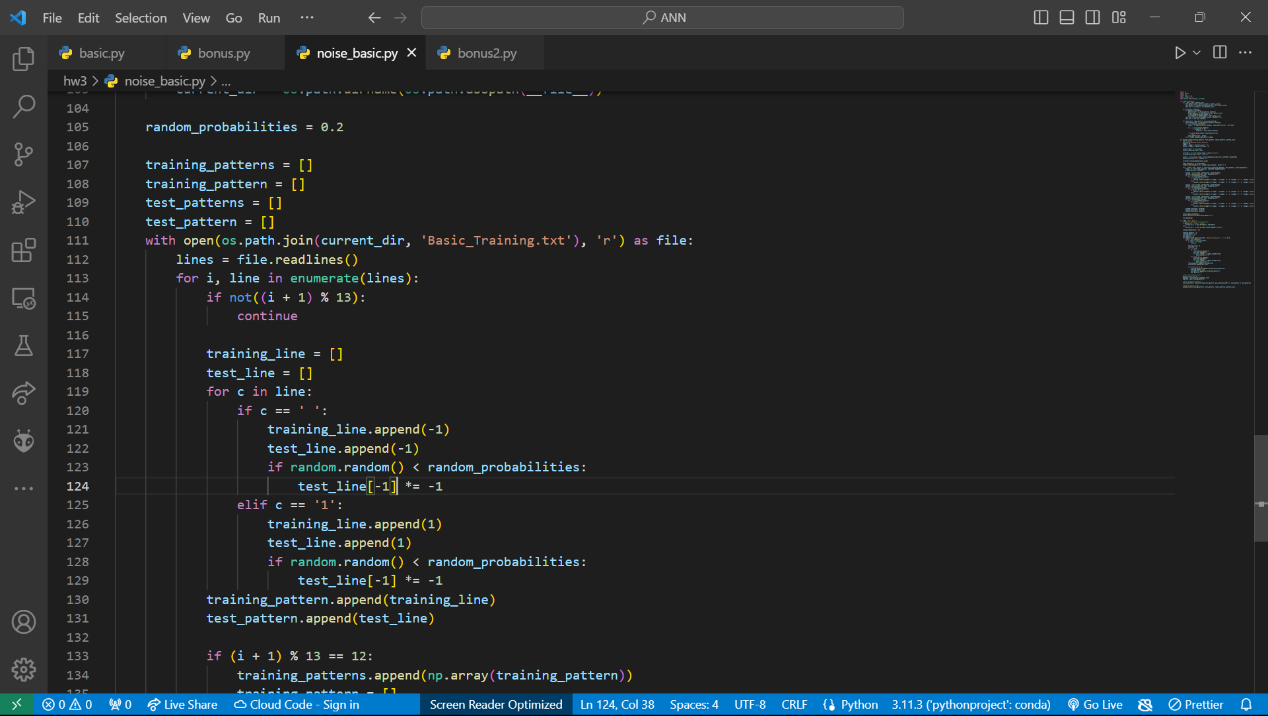




**Noice:**

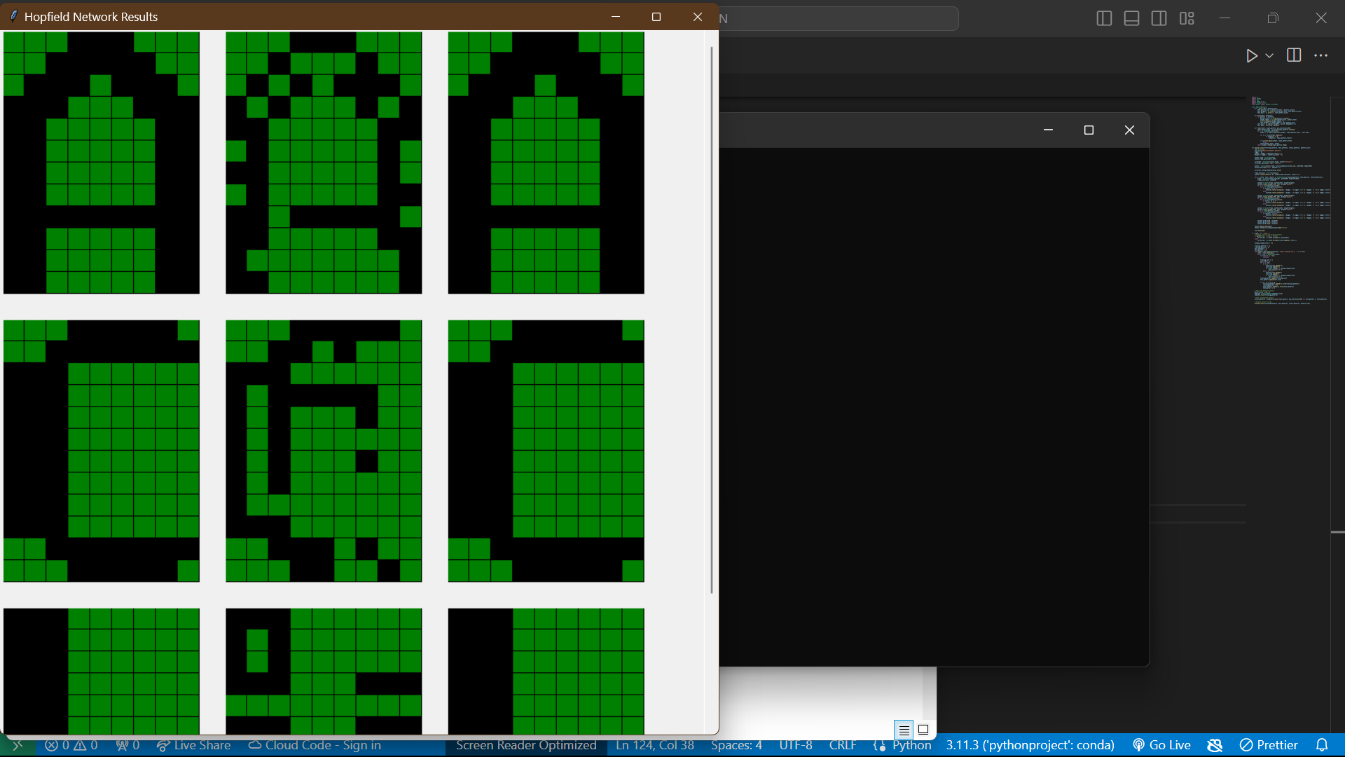
以0.2的機率隨機產生noice

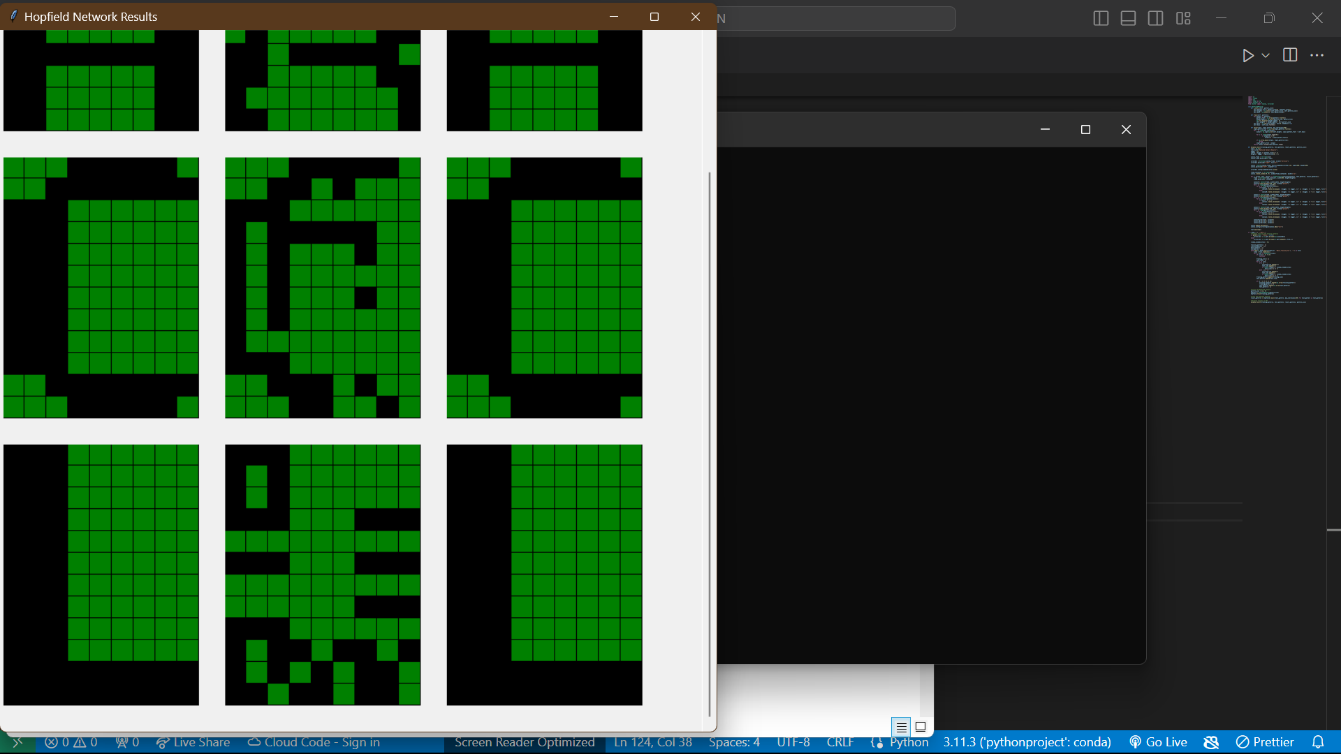




1. **實驗結果與分析及討論**

**Basic:**





**結果:**

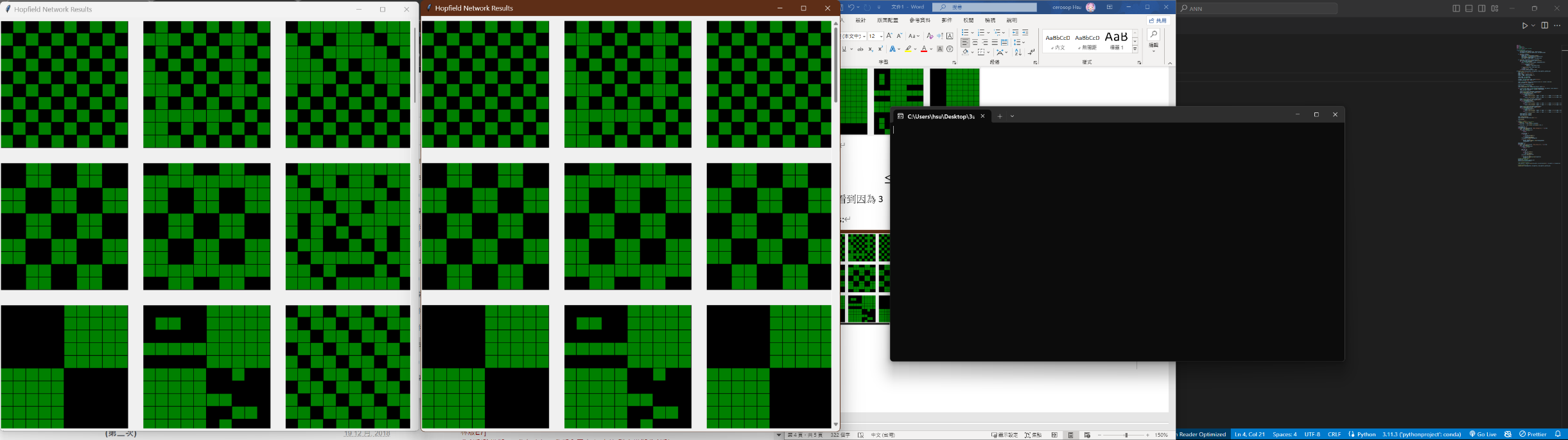
100%正確回想

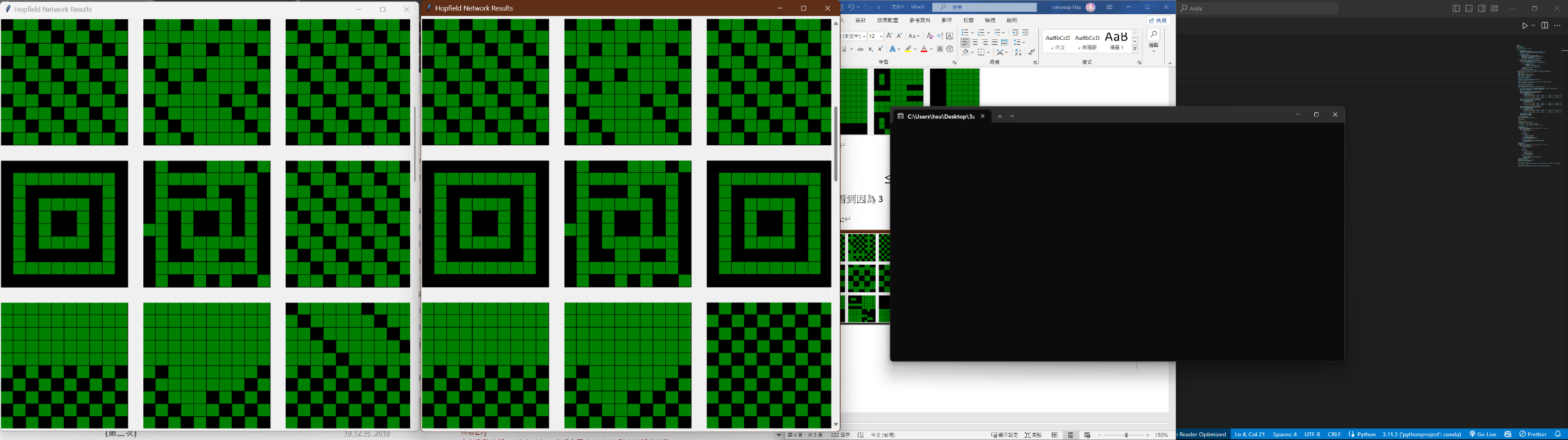
**分析:**

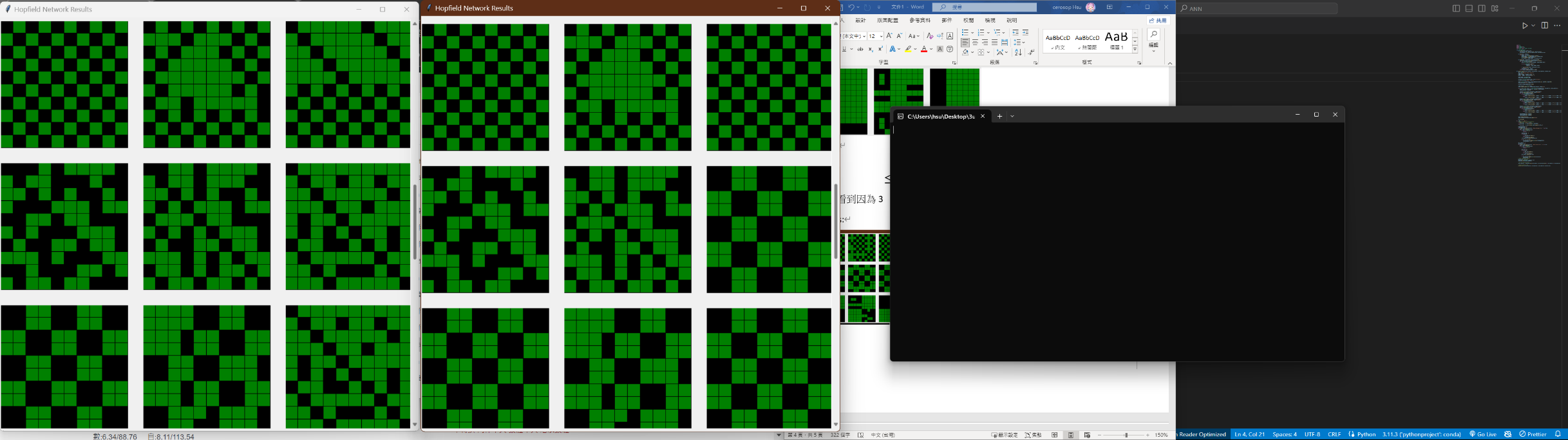
可以看到因為3  = 5.8，因此三個測資都可以正常回想。

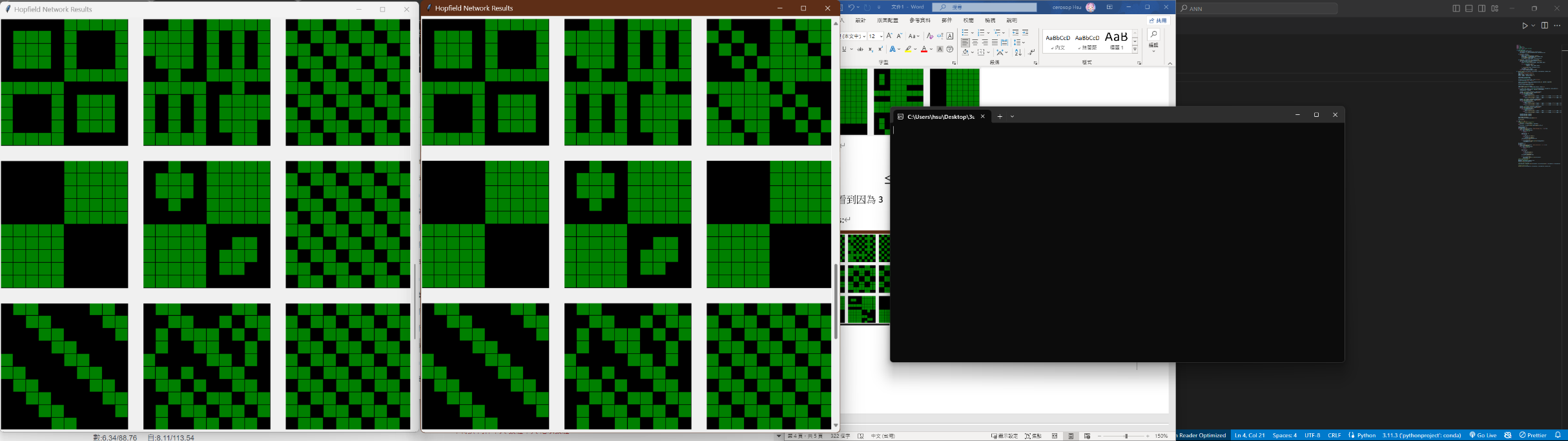
**Bonus:**

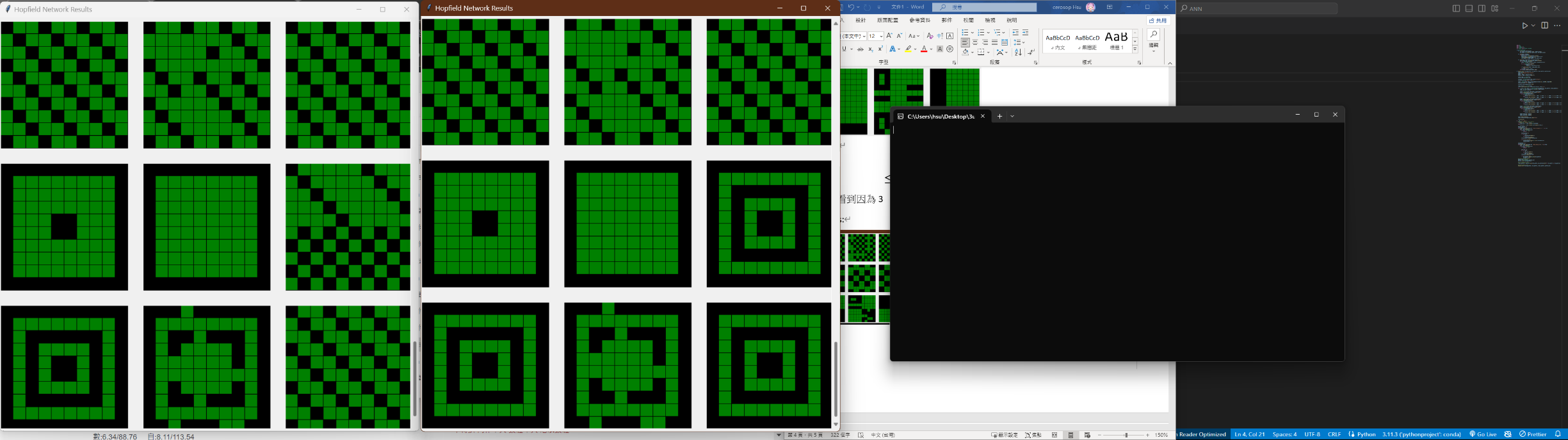
左側為bonus(bias = 0)、右側為bonus2(bias = wj)











**分析:**

Bonus: 13%正確回想

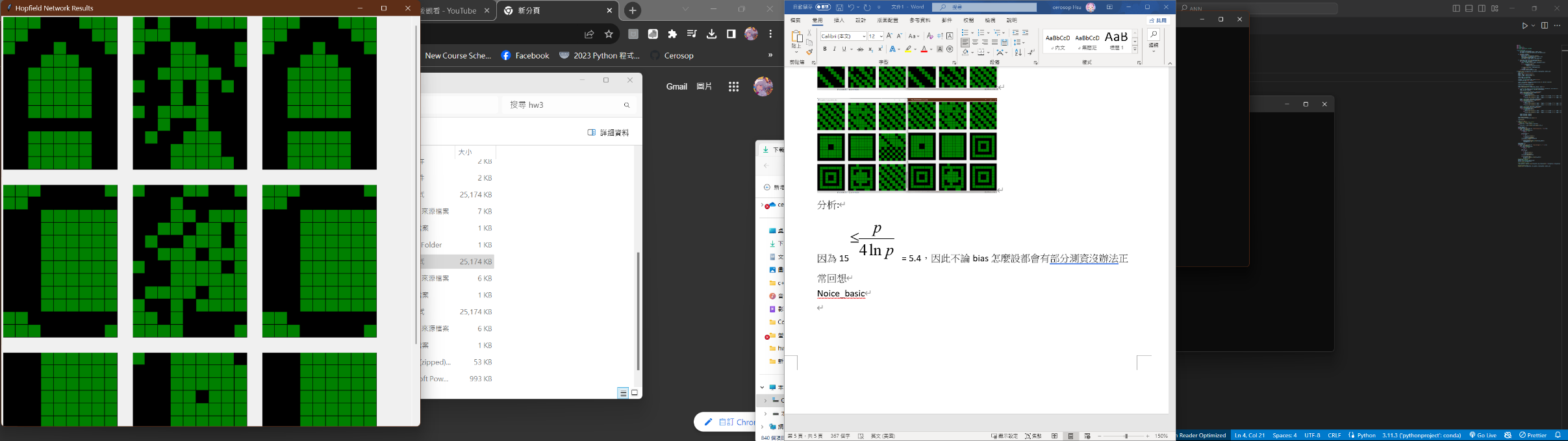
Bonus2: 66%正確回想

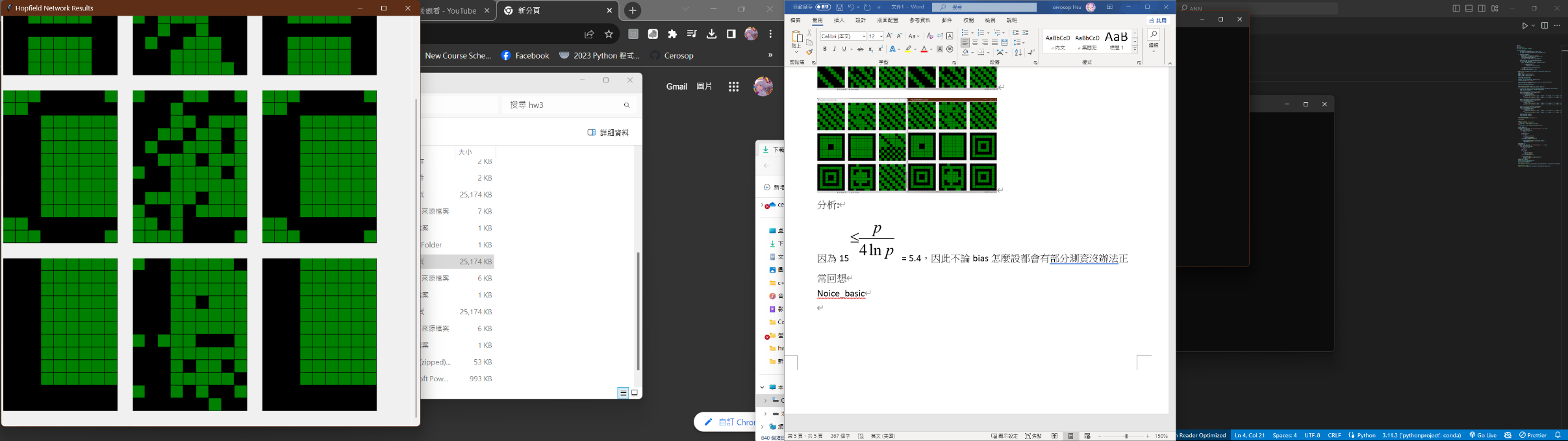
**討論:**

一開始測試bonus2時發現部分無法正常回想，就去測試bonus，但發現更糟糕，所以我認為是因為15  = 5.4，導致不論bias怎麼設都會有部分測資沒辦法正常回想。

**Noice\_basic:**

0.2機率產生noice





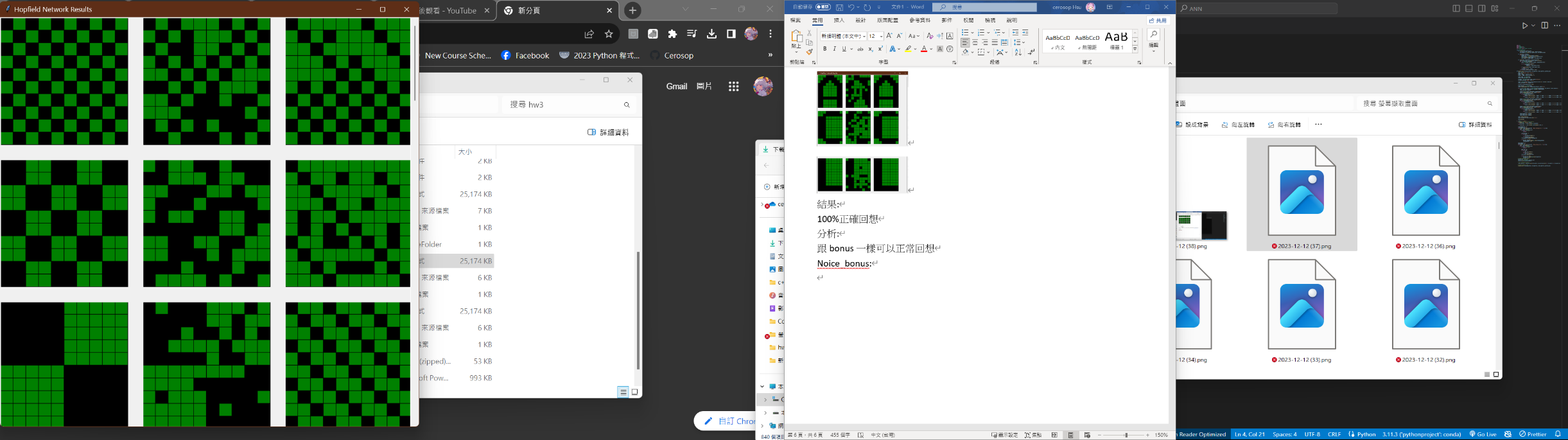
**結果:**

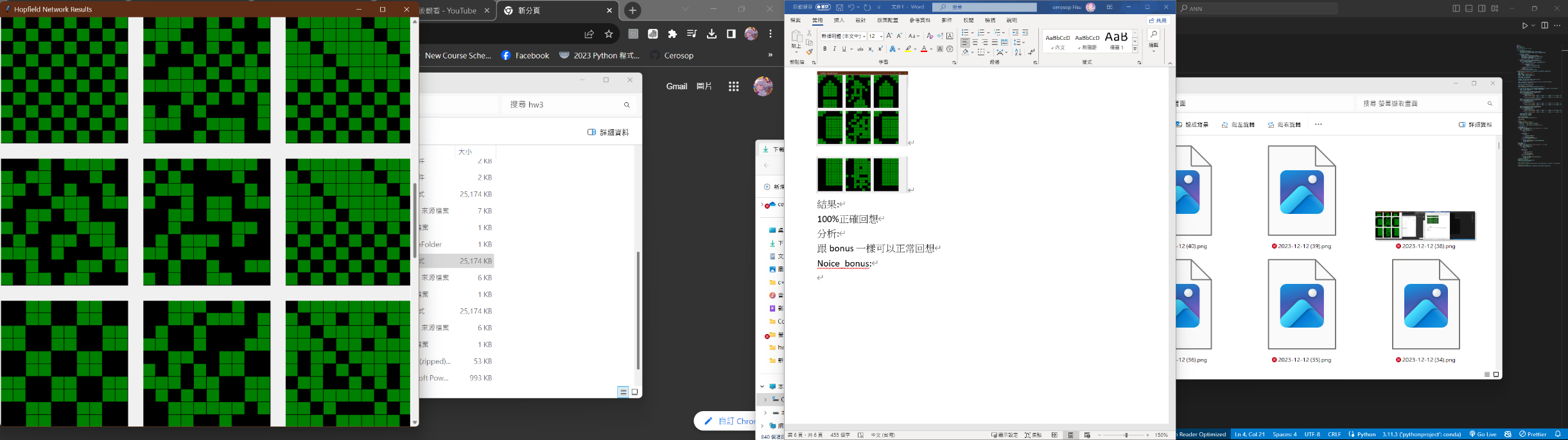
100%正確回想

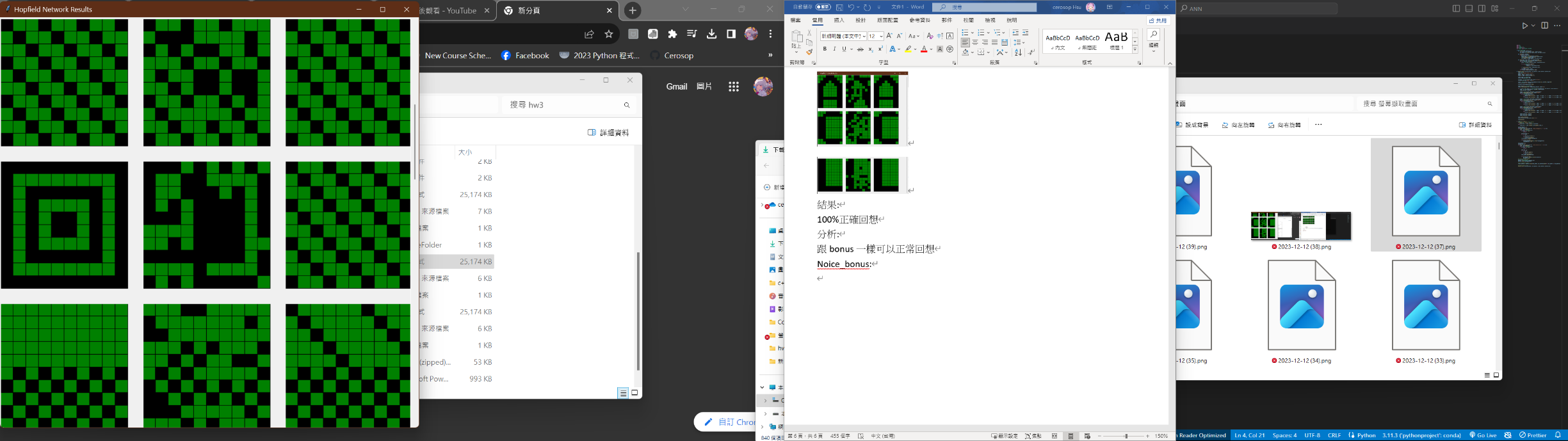
**分析:**

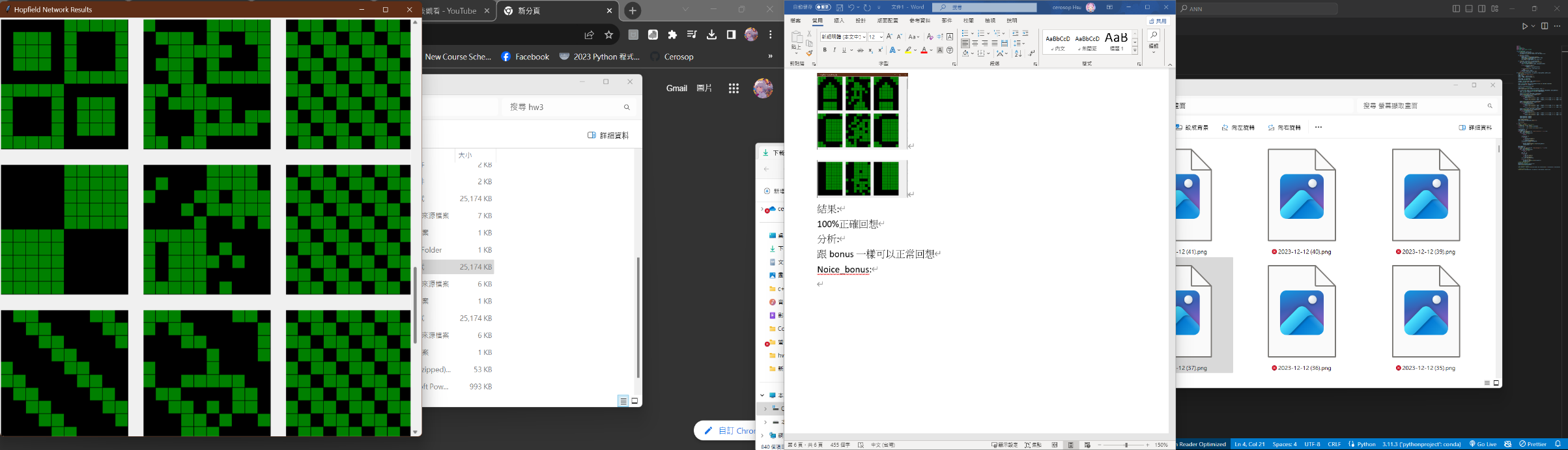
跟bonus一樣可以正常回想。

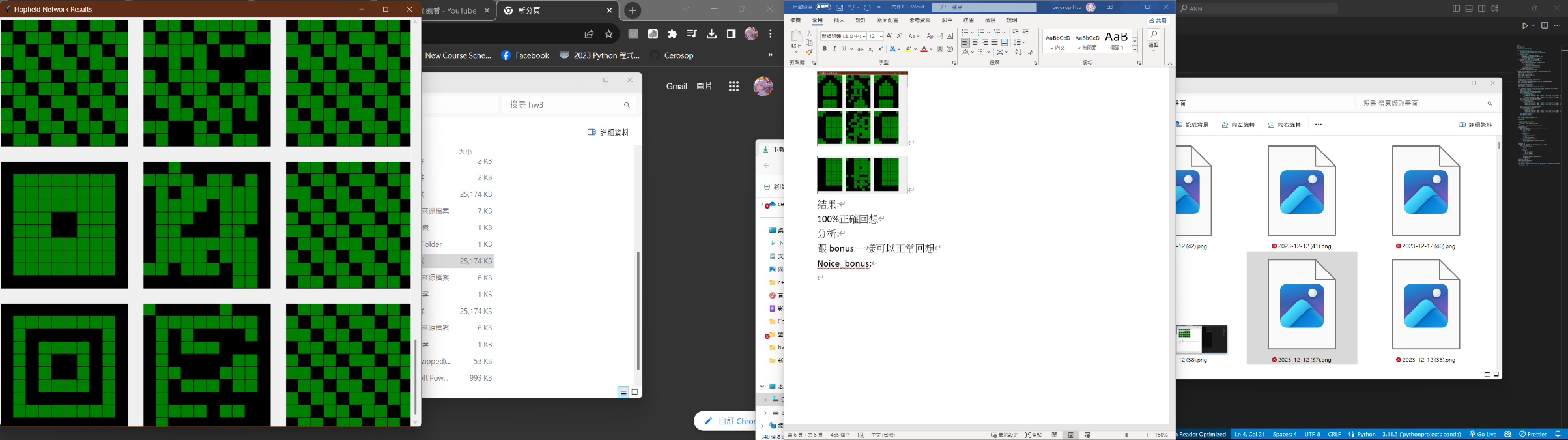
**Noice\_bonus:**











**結果:**

7%正確回想

**討論:**

可以發現正確回想率非常低，甚至大部分都回想成了相似的圖形，記憶量遠低於要記憶的數量。