stanCode

標準程式教育機構

Assignment 6

This assignment is based on the Assignment 4 and 5 of CS106B at Stanford University
Image Credit:Algodaily.com

歡迎來到 **SC101** 的最後一份作業！改編各大公司軟體工程師面試題目，包含四大科技龍頭 **FAGA** – Facebook, Apple, Google, Amazon...

遞迴 (recursion) 的觀念在 **LeetCode** 上佔了 1/3 以上的篇幅！因此我們會透過最後一份作業再讓大家熟悉這個技能，也讓大家挑戰人生中第一道 **LeetCode** 的 **Hard** 題 - **LC212**。除此之外，教學團隊也引入最底層的程式邏輯 - 連結串列 (LinkedList) 面試題目，希望讓各位 SC101 結業的同學都具備基本刷題能力

作業檔案下載

作業預計時間：**15 小時**

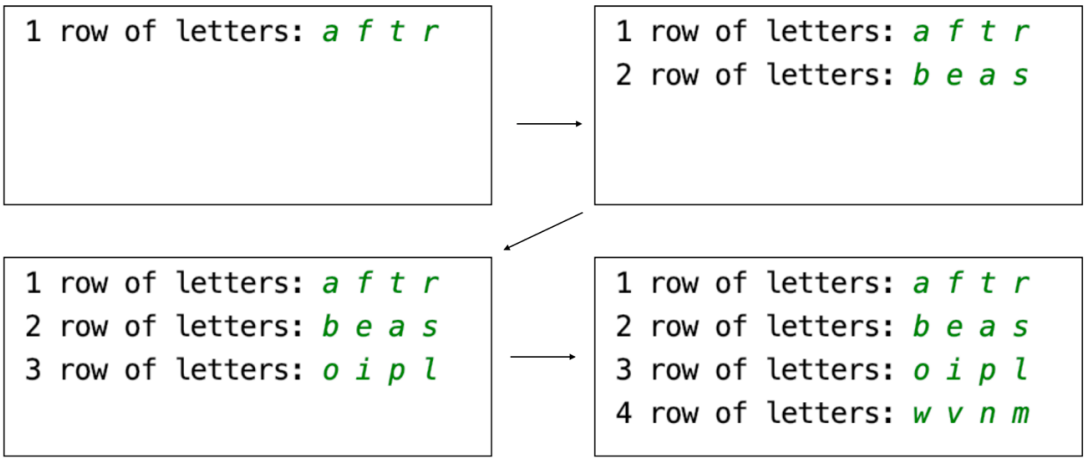
本份作業請勿使用global variables

如果作業卡關歡迎各位到社團提問，也非常鼓勵同學們互相討論作業之概念，但請勿把 code 給任何人看（也不要把程式碼貼在社團裡）分享妳/你的 code 會剝奪其他學生獨立思考的機會，也會讓其他學生的程式碼與你/妳的極度相似，使防抄襲軟體認定有抄襲嫌疑

LC212 – Word Search II – boggle.py

Boggle 是美國家喻戶曉的桌遊！遊戲一開始會先排出一個 4 x 4 的方形字母拼盤，接著玩家開始串連在字母盤上相連的字母，去找出存在於這個 4 x 4 的方形字母拼盤的所有英文單字。沒錯，SC101 作業的最後一題您將使用 backtracking 完成這個遊戲搜尋的程式，來為課程劃下一個完美的句點

在串連字母的過程，我們只能串連字母塊的「鄰居」– 上、下、左、右、左上、左下、右上、右下 – 且一個字母塊只能經過一次



遊戲一開始玩家會被要求輸入 4 排以空白隔開的 4 個英文單字（如下圖所示）

```
1 row of letters: a b v c
2 row of letters: aa n u i
Illegal input
```

若使用者的輸入不符合規定，程式就會終止執行。換句話說，我們強迫使用者輸入 16 個英文字母且每一排的字母間一定要用空白隔開

```
1 row of letters: f y c l
2 row of letters: i o m g
3 row of letters: o r i l
4 row of letters: h j h u
```

假設使用者正確輸入了下圖所示的 16 個字母：

這時您的程式就會開始搜尋這個 4x4 字母盤上所有可以被串連起來的英文單字

若您的程式撰寫正確應該能在 Console 完美重現下圖中的每一行文字：

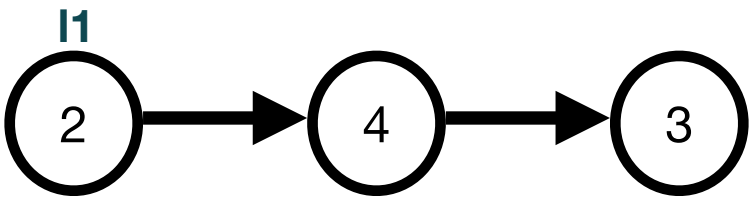
```
1 row of letters: f y c l
2 row of letters: i o m g
3 row of letters: o r i l
4 row of letters: h j h u
Found "firm"
Found "form"
Found "foil"
Found "coif"
Found "coir"
Found "corm"
Found "coil"
Found "moor"
Found "moil"
Found "miri"
Found "giro"
Found "glim"
Found "roil"
Found "roof"
Found "room"
Found "roomy"
Found "rimy"
Found "iglu"
Found "limy"
Found "limo"
Found "hoof"
There are 21 words in total.
```

以下八個重點提醒：

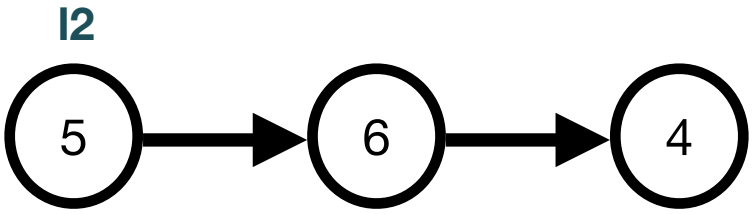
1. 起始字母可以任意選擇。然而，一旦選擇了，請遵守上述串連法則
2. 程式應該為 Case insensitive
3. 程式只搜尋長度大於等於 4 的英文單字
4. 編輯名為 **def read_dictionary( )** 的函式，讀進整本英文字典至程式
5. 編輯名為 **def has_prefix(sub_s)** 的函式，檢查是否存在以 sub_s 為開頭的英文字
6. 當您撰寫的遞迴程式找到一個單字時，請印出 **Found** 並緊接著印出 **“找到的文字”**
7. 這題最困難的地方莫過於上圖中 **“room”** 與 **“roomy” -** 該如何在找到一個解答後，繼續要求我們的遞迴程式往下搜尋更長的單字呢？
8. 最後，請印出在該 4 x 4 字母盤上找到了幾個單字的總數

LC2 – Add Two Numbers – add2.py

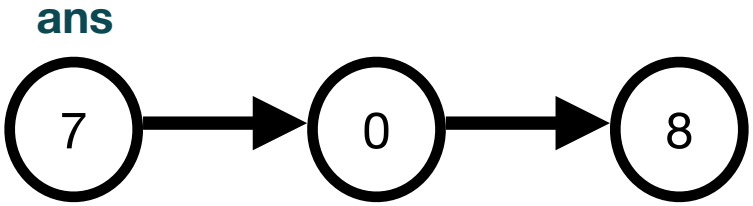
兩條不是 **None** 的 linked lists 代表了兩個正整數。然而，數值是「反向儲存」。舉例來說，圖一的 2 -> 4 -> 3 所代表的正整數是 342 ; 而圖二的 5 -> 6 -> 4 所代表的正整數是 465。請將那兩個正整數數值總和變成一條新的 linked list! 舉例來說，您要 return 的 linked list 在圖三（7 -> 0 -> 8 代表圖一的正整數 342 與圖二的正整數 465 總和為 807）



圖一、開頭為 l1 的第一條 linkedlist，代表正整數 342

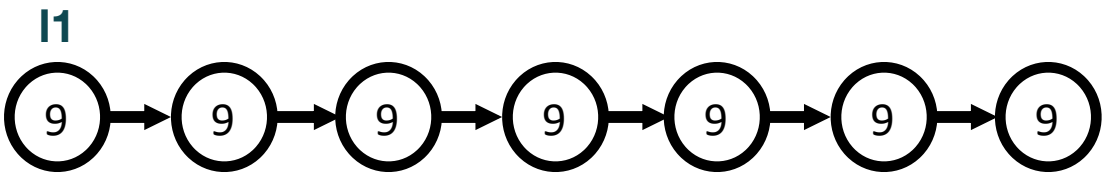


圖二、開頭為 l2 的第二條 linkedlist，代表正整數 465

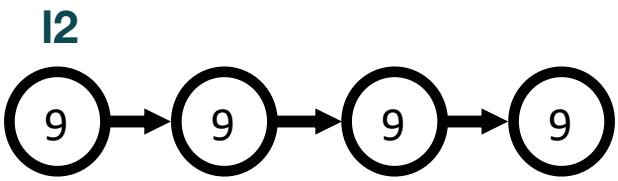


圖三、為第一條 linkedlist 與第二條 linkedlist 的總和，代表正整數 807

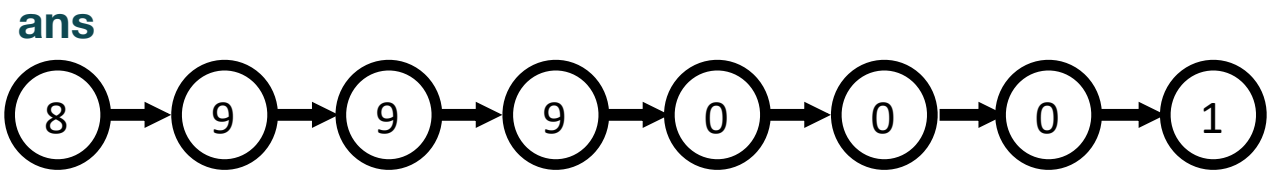
將兩條 linked lists 相加時，每一條 linkedlist 的長度可以不一樣！如下圖四的 9->9->9->9->9->9->9 所代表的正整數是 9999999 ; 圖五的 9->9->9->9 所代表的正整數是 9999 ; 而您要 return 的 linked list 在圖六 (8->9->9->9->0->0->0->1代表圖一的正整數 9999999 與圖二的正整數 9999 總和 10009998)



圖四、開頭為 l1 的第一條 linkedlist，代表正整數 9999999

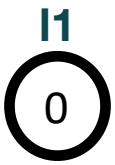


圖五、開頭為 l2 的第二條 linkedlist，代表正整數 9999



圖六、為第一條 linkedlist 與第二條 linkedlist 的總和，代表正整數 10009998

您可以假設 node 的數量一定介於 1 顆到 100 顆、node 的值一定介於 0-9、且數值一定不用有多餘的零在開頭（舉例來說， 0 -> 1 -> 2 是不會出現的）但 l1、l2 可以都只存 0！如下圖七所代表的正整數是 0 ; 圖八所代表的正整數也是 0； 而您要 return 的 linked list 如圖九所示也是 0



圖七、開頭為 l1 的第一條 linkedlist，代表正整數 0

圖八、開頭為 l2 的第二條 linkedlist，代表正整數 0

圖九、為第一條 linkedlist 與第二條 linkedlist 的總和，代表正整數 0

請編輯名為 **def add_2_numbers(l1: ListNode, l2: ListNode) -> ListNode:** 的函式。請注意這種新的 Python 註解寫法：**l1**, **l2** 是變數名稱，冒號後面的 **ListNode** 是 data type ; 而最後的 **-> ListNode** 代表 return type 是一個 **ListNode**。若您撰寫正確，當您在 **Terminal/cmd** 執行 **python3 add2.py test1/py add2.py test1** 您將會看到與下圖一樣的輸出：

```
jerryliao@Jerrys-MacBook-Pro SC101_Assignment6 % python3 add2.py test1
---------test1---------
l1: 2->4->3
l2: 5->6->4
ans: 7->0->8
-----------------------
```

若您撰寫正確，當您在 **Terminal/cmd** 執行 **python3 add2.py test2/py add2.py test2** 您將會看到與下圖一樣的輸出：

```
jerryliao@Jerrys-MacBook-Pro SC101_Assignment6 % python3 add2.py test2
---------test2---------
l1: 9->9->9->9->9->9->9
l2: 9->9->9->9
ans: 8->9->9->9->0->0->0->1
-----------------------
```

若您撰寫正確，當您在 **Terminal/cmd** 執行 **python3 add2.py test3/py add2.py test3** 您將會看到與下圖一樣的輸出：

```
jerryliao@Jerrys-MacBook-Pro SC101_Assignment6 % python3 add2.py test3
---------test3---------
l1: 0
l2: 0
ans: 0
-----------------------
```

評分標準

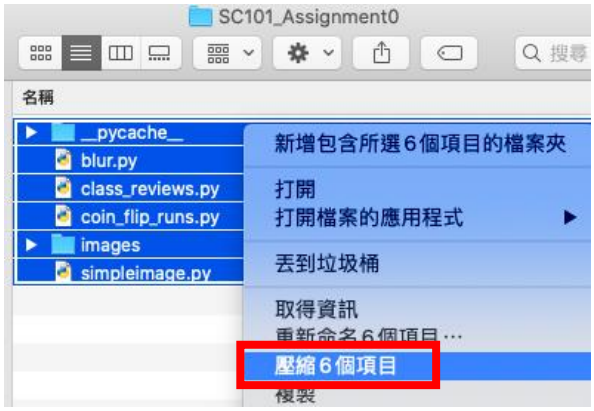
Functionality - 程式是否有通過我們的基本要求？程式必須沒有 bug 、能順利完成指定的任務、並確保程式沒有卡在任何的無限環圈（Infinite loop）之中

Style - 好的程式要有好的使用說明，也要讓人一目瞭然，這樣全世界的人才能使用各位的 code 去建造更多更巨大更有趣的程式。因此請大家寫**精簡扼要**的使用說明、function 敘述、單行註解

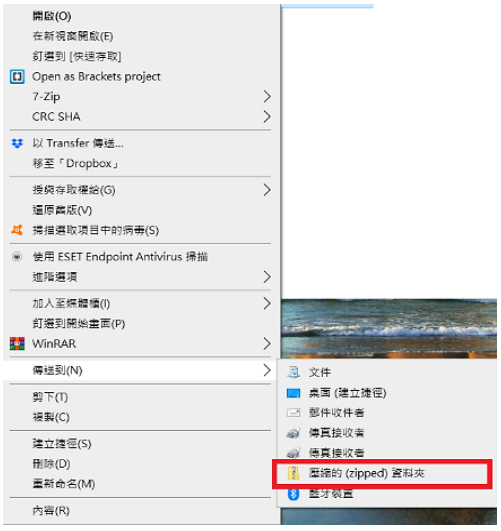
作業繳交

1. 以滑鼠「**全選**」作業資料夾內的所有檔案，並壓縮檔案。請見下圖說明。

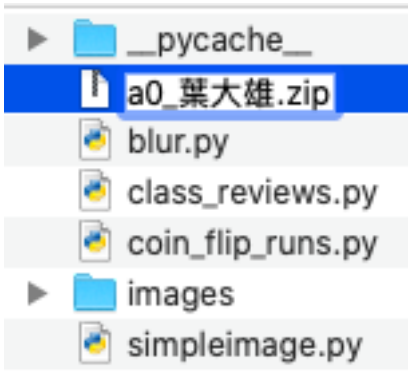
macOS：按右鍵選擇「**壓縮 n 個項目**」



Windows：按右鍵選擇「**傳送到**」→「**壓縮的(zipped)資料夾**」

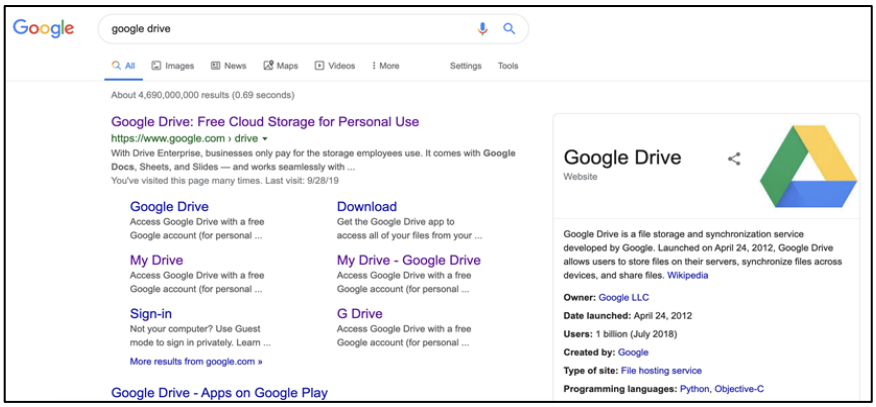


2. **將壓縮檔(.zip)重新命名為「a(n)_中文姓名」。如：**
   assignment 0 命名為 a0_中文姓名;
   assignment 1 命名為 a1_中文姓名; ...



3. **將命名好的壓縮檔(.zip)上傳至 Google Drive（或任何雲端空間）**

   1) 搜尋「google drive」



   2) 登入後，點選左上角「新增」→ 「檔案上傳」→ 選擇作業壓縮檔(.zip)

4. **開啟連結共用設定，並複製下載連結**

1) 對檔案按右鍵，點選「共用」

2) 點擊「**變更任何知道這個連結的使用者權限**」後，權限會變為「可檢視」

3) 點選「**複製連結**」

5. **待加入課程臉書社團後，將連結上傳至作業貼文提供的「作業提交表單」**

stanCode

Should you have any idea or questions, please feel free to contact.