stanCode

標準程式教育機構

Assignment 3

This assignment is based on the Assignment 2 of CS106AP
and the Assignment 3 of CS106A at Stanford University

作業檔案下載

這份作業將帶領同學熟悉並運用 CS 領域中最重要的資料類型：**string**
透過 string manipulation 的技巧，各位同學會解決實際運用於生物科技的問題、破解密碼學解密問題，並創造深受大人小孩喜愛的「吊死鬼」單字遊戲。

請注意：為了讓程式更容易被理解，請利用自己建造的 function(s) 來設計並達成 decomposition 的架構

本份作業估計需要時間為 **12 小時**

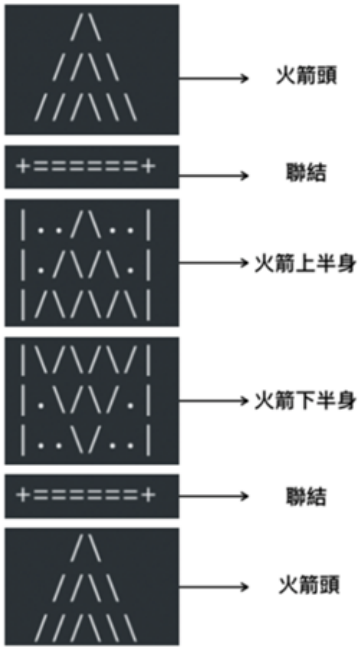
如果作業卡關 **歡迎與助教討論**，stanCode也非常鼓勵同學們互相討論作業的概念，**但請不要把自己的code給任何人看**，分享您的code會剝奪其他學生獨立思考的機會，同時會導致其他學生的程式碼與您的code極度相似，使得防抄襲軟體認定有抄襲之嫌疑

Problem 1 - rocket.py

這一題需要各位同學幫忙，使用 **print & double for loop** 來打造台灣的新型火箭！（如下圖所示）

做出一架火箭需要專業的分工，因此 **整個火箭可以分成六個部分**（如下圖）：

- 火箭頭 (head)
- 聯結 (belt)
- 火箭上半身 (upper)
- 火箭下半身 (lower)
- 聯結 (belt)
- 火箭頭 (head)



火箭的每一個部分都是一個function，且每一個 function 裡都至少需要一個 for loop(s)

更具體地說，**def main( ) 裡面不應該出現任何的 print()**，而是包含許多的 functions 例如 head()、belt()、upper()、lower()……等，就好像是 Karel 作業時大家非常熟練的 **decomposition** ！ 請用一個 function name 把一部分的程式碼賦予全新的意義。

此外，各位台灣未來的工程師們請注意，在各個 function 的 **for loop 裡面一次只能print 出一個火箭元件**（" ", "/", "\\", "=", "+", “|"）。也就是說，不能一次把一行火箭元素印出來，例如 print ("+=======+") 是違法的，這樣會被長官記過處分喔！

眼尖的同學應該有注意到 "\\" 雖然包含兩個倒斜線 (backslash) 但print出來後卻只有一個會被印在Console。原因是 backslash 對電腦來說是特殊符號，因此需要特殊處理（如 Assignment 2的Problem 2 為了印出 "Weather Master 4.0" 必須在引號裡使用 \"）

除了完成上述火箭建造的重要工作之外，您的程式必須能夠使用定義在作業最上方的 **Constant（SIZE=3）**讓政府可以藉由改變 SIZE 後面的數字而建造不同大小的火箭。舉例來說，當國防部長改變 SIZE 後面的數字為1, 2, 5, 10 您的程式應該能輸出下列圖片，由左至右

請注意：請勿使用上課未討論過的 Python 指令（例如：文字的乘法）

Problem 2 - complement.py

Problem 2 和 3 將帶領同學了解 coding 在生物科技上的應用 - DNA 序列工程

DNA 是由四種含氮鹼基A (Adenine), T (Thymine), C (Cytosine), G (Guanine) 所構成。A 與 T 為互補鹼基序列、C 與 G 為互補鹼基序列（互補檢基序列：能量最低、最穩定的鹼基對結構）。如下圖所示：

請編輯 **complement.py** 讓使用者可以輸入一個 DNA 片段並得到該片段之最低能量互補片段。您所寫出的程式應該要能完美重現下圖中的所有文字：

```
Please give me a DNA strand and I'll find the complement: ATGCAAG
The complement of ATGCAAG is TACGTTC
```

在課堂中學會 function, parameters 與 return value 的您，現在應該能夠妥善地使用 **decomposition** 的概念使程式碼架構更加清晰。因此，**請在 complement.py 裡建造一個名為 build_complement( ) 的 function** ，並讓它可以 return 出輸入其中的互補序列。舉例來說，build_complement('ATC') 應該要 return 出 'TAG'

請注意，build_complement 應該是 **case-insensitive**，也就是使用者輸入的文字大小寫不會影響最後得到的答案。換句話說，不論是 build_complement('atC') 或 build_complement('Atc') 都會得到相同的結果 'TAG'

（您可以假設使用者輸入的 DNA 片段裡只會包含 'a', 'A', 't', 'T', 'c', 'C', 'g', 'G'）

Problem 3 - caesar.py

加密（Cipher）是國際上非常普遍的保密手段，甚至有許多數學家或是電腦科學家費盡畢生潛心研究密碼學（Cryptography)這門學問。同時，在許多電影中也能看到密碼學的應用，像是著名電影「模仿遊戲」(The Imitation Game）的劇情便是講述天才數學家、密碼學家與電腦科學家 – 圖靈 (Turing) 如何解密（decipher）由英國軍隊所攔截的德國海軍情報。因此，本題將帶領同學了解簡易的密碼學，並讓同學們更加熟悉 string manipulation 的應用！

密碼學這門學問非常有趣，最早的源頭竟然可以追朔至西元前70年的羅馬帝國！當時，羅馬共和國獨裁者 - 凱薩 (Caesar) 為了保護他的軍事機密不會讓敵軍知道所以發明了最早的加密系統 “凱薩密碼” (Caesar Cipher) 。在這一題中，就讓我們一起來看看這套加密系統的奧秘吧！

大家熟知的英文字母排列順序如下圖所示（這一串**未平移**的英文字母串我們稱之為 **ALPHABET**）

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

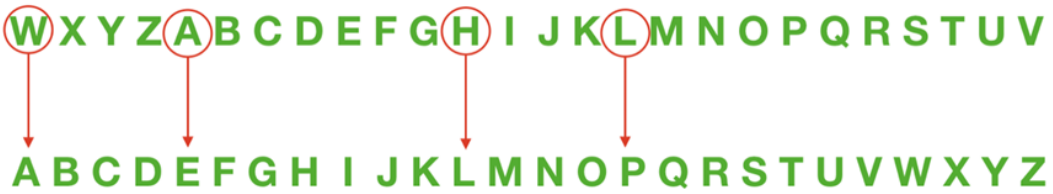
如果我們把 **ALPHABET** 所有英文字母 **向右平移 4 格**，V會成為第26個字母，然後WXYZ會被放到字母串最前面（wrap around）。如此一來我們就可以得到一個全新的英文字母串**new_alphabet**：

W X Y Z A B C D E F G H I J K L M N O P Q R S T U V

但到底要怎麼使用 new_alphabet 呢？舉個例子，今天凱薩如果想要傳一個機密訊息 “APPLE“ 給 Jerry，生性多疑的凱薩當然不可能直接傳 “APPLE” (因為如果這個內容被其他人看到，這個機密不就外洩了嗎！！) 。所以凱薩會將其要傳送的文字 “APPLE” ，根據 **ALPHABET** 中每個字母的位置，依序找到**new_alphabet**中對應到的字母，寫下加密文字 “WLLHA” ，加密的過程如下圖所示：

(A 字母對應到新字串的 W // P 對應到 L // L 對應到 H // E 對應到 A)

但收到 “WLLHA” 的 Jerry 要如何知道凱薩在說什麼呢？別擔心，凱薩傳來的信封裡，除了寫著 “WLLHA” 的白紙之外，抖一抖，裡面還藏著一張寫著 “4” 的小紙條，告訴 Jerry 請把我們所熟知的英文字母串 **ALPHABET** 向右平移 **4** 格，就能得到 **new_alphabet** 。所以只要依序將 “WLLHA” 的每個字母對應回沒有平移的英文字母串 **ALPHABET**，就可以得到機密訊息 “APPLE”了！解密的過程如下圖所示：



現在，就請同學完成名為 caesar.py 的檔案，並完成解密（decipher）的過程。當您完成時，您的程式可以完美重現下圖中每一行文字與數字

```
Secret number: 4
What's the ciphered string?WLLHA
The deciphered string is: APPLE
```

當同學順利解出一個單字後，請思考一下：如果今天傳來的機密內容是一個完整的句子，我們該怎麼解呢？（您的程式最終應能完美重現下圖每一行文字與數字）

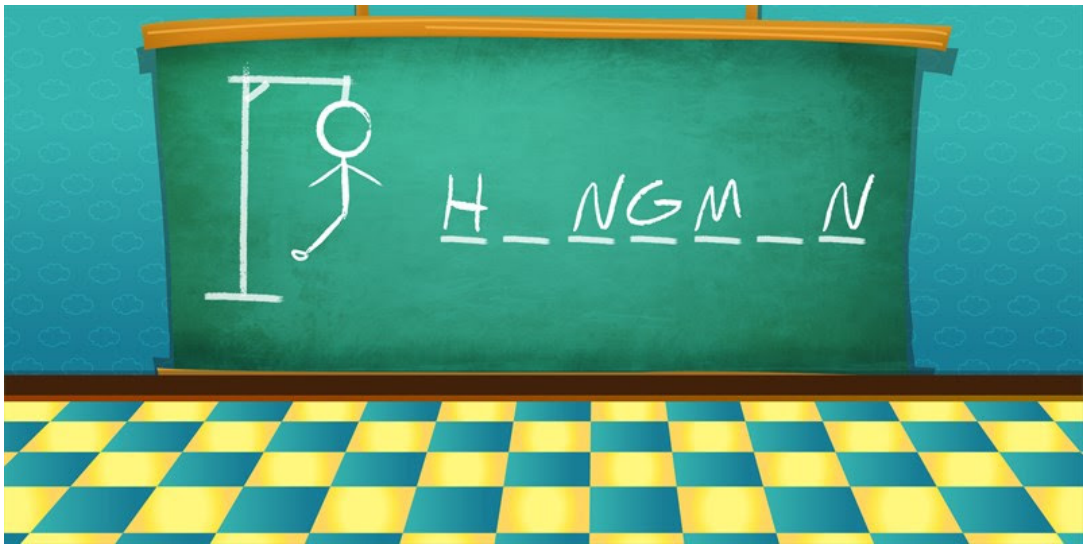
```
Secret number: 7
What's the ciphered string?RHN TKX MAX UXLM!
The deciphered string is: YOU ARE THE BEST!
```

請注意：您的程式應該是 **case-insensitive**。也就是使用者輸入的文字大小寫不會影響最後得到的答案（如下圖所示）

```
Secret number: 7
What's the ciphered string?rhn TKx Max UXLm!
The deciphered string is: YOU ARE THE BEST!
```

Problem 4 - hangman.py

最後一題將請同學使用 Console 來創造經典猜字遊戲 Hangman (吊死鬼)！



程式的一開始會從字庫裡隨機選擇一個英文單字 (以下簡稱為 **answer**)，並將每一個字母用橫槓遮住（以下簡稱為 **dashed**），因此遊戲開始時，玩家會先得到一列與**answer**長度一樣的橫槓。而後每一輪中，玩家會輸入一個大寫或小寫的字母（以下簡稱為 **input_ch**），如果 **input_ch 存在於 answer 之中**，程式就會更新 dashed 並**把所有 input_ch 所在的位置展示出來**。但**如果 input_ch 並不存在於 answer 之中，玩家就會損失一條命**。若七條命都被扣完但玩家還沒猜出來 answer，玩家則挑戰失敗。

真正的 Hangman 遊戲會在玩家猜錯時更新吊死鬼的圖樣（如下圖所示），當吊死鬼的頭部、身體、左手、右手、左腳、右腳、臉部都被呈現出來時，玩家挑戰失敗

然而，**我們在這題中並不需要做到圖樣的部分！**只要完成 Console 版的即可（但想挑戰的同學，或許可以將圖樣版Hangman當做自己的 Extensions 喔 ^^ ）

您所完成的程式應該要可以完美呈現下圖的每一行文字與結果：

```
THe word looks like ------
You have 7 wrong guesses left.
Your guess: r
There is no R's in the word.
The word looks like ------
You have 6 wrong guesses left.
Your guess: r
There is no R's in the word.
The word looks like ------
You have 5 wrong guesses left.
Your guess: x
There is no X's in the word.
The word looks like ------
You have 4 wrong guesses left.
Your guess: y
There is no Y's in the word.
The word looks like ------
You have 3 wrong guesses left.
Your guess: a
There is no A's in the word.
The word looks like ------
You have 2 wrong guesses left.
Your guess: z
There is no Z's in the word.
The word looks like ------
You have 1 wrong guesses left.
Your guess: p
There is no P's in the word.
You are completely hung : (
The word was: BUNDLE
```

```
THe word looks like --------
You have 7 wrong guesses left.
Your guess: c
You are correct!
The word looks like C-------
You have 7 wrong guesses left.
Your guess: C
You are correct!
The word looks like C-------
You have 7 wrong guesses left.
Your guess: a
You are correct!
The word looks like CA------
You have 7 wrong guesses left.
Your guess: u
You are correct!
The word looks like CAU---U-
You have 7 wrong guesses left.
Your guess: t
You are correct!
The word looks like CAUT--U-
You have 7 wrong guesses left.
Your guess: I
You are correct!
The word looks like CAUTI-U-
You have 7 wrong guesses left.
Your guess: O
You are correct!
The word looks like CAUTIOU-
You have 7 wrong guesses left.
Your guess: s
You are correct!
You win!!
The word was: CAUTIOUS
```

以下五項重點提醒

1. 請使用我們已經寫好的 **random_word( )** 來得到一個隨機的英文單字。同學不用了解此 function 的每一行 code，只要知道 random_word( ) 每一次會隨機 return 一個英文單字出來即可
2. 使用者的輸入為 **case-insensitive**，因此不論輸入**upper case (大寫)** 或 **lower case (小寫)** 的字母都可以
3. 若輸入兩次一樣的錯誤答案，玩家還是會少一條命 (如上頁左圖的 r and r)
4. 若輸入兩次一樣的正確答案，玩家會得到一樣的結果 (如上頁右圖的 C and c)
5. 當使用者輸入的格式錯誤，例如輸入的內容不是英文字母(可以用 **str.isalpha( )** 來判斷)，或不只輸入一個字母，這時候程式應該要**印出 "illegal format."** 並重複要求使用者輸入，直到格式正確為止（如下圖所示）

```
THe word looks like ---------
You have 7 wrong guesses left.
Your guess: 2
Illegal format.
Your guess: aa
Illegal format.
Your guess: E
You are correct!
The word looks like -E-------
You have 7 wrong guesses left.
Your guess:
```

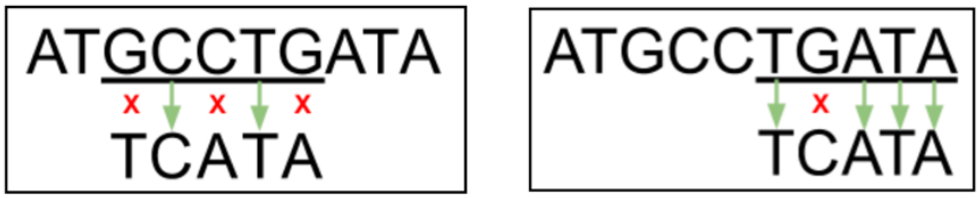
如果有想做本題作業要求外的extension，例如畫出hangman的圖，
請繳交兩份檔案：

1. **一份剛好達成作業要求 (hangman.py)**
2. **一份加入 extension 並將檔名改為 hangman_ext.py**

歡迎寫完作業的同學挑戰Extension! **完成下方兩題Extension就有機會得到++喔！**

Extension - similarity.py

coding 在生物產業界很常見的應用就是找出某一個較短的 DNA 片段在較長的 DNA 片段中配對率最高的區間，我們稱之 **homology**



相似率：40%　　相似率：80%

假設使用者輸入 “ATGCCTGATA” (以下稱之為 long_sequence)的DNA序列、並輸入待配對的DNA片段“TCATA”(以下稱之為short_sequence)，我們要在long_sequence裡找尋跟short_sequence最相似的片段。因此，我們會在long_sequence裡找跟short_sequence一樣長的片段，並比對有幾個含氮鹼基（A、T、C、G）一樣，如上圖所示，左圖僅有2個一樣，因此相似率為40%，而右圖有4個一樣，因此相似率為80%，所以右圖相似率80%的”TGATA”是我們的答案！

您所完成的程式應該要可以完美複製下圖中的所有文字：

```
Please give me a DNA sequence to search: ACTGACATTG
What DNA sequence would you like to match? TGCCA
The best match is TGACA
```

請注意：

1. 您的程式應該是 **case-insensitive**。也就是使用者輸入的文字大小寫不會影響最後得到的答案（如下圖所示）

```
Please give me a DNA sequence to search: ATcgAtCGatCgC
What DNA sequence would you like to match? tCgC
The best match is TCGC
```

2. 如果long_sequence中有兩段片段與short_sequence的相似率一樣，最後The best match的答案呈現兩者中任何一個片段即可

Extension - name_sq.py

stanCode 這一題想請同學編輯name_sq.py，完成一個可以請使用者輸入名字，並在console上印出名字方形形狀的程式。舉例來說，如果使用者今天輸入的名字為Jenny，則在Console印出的方形 (如下圖所示) — 上邊會為名字 Jenny 本身，方形左邊由上而下也會是Jenny，方形下邊則會為名字倒過來的 YNNEJ，而方形右邊從上而下也會是名字倒過來的 YNNEJ。

```
JENNY
E   N
N   N
N   E
YNNEJ
```

您所完成的程式應該要可以完美複製下圖中的所有文字：

```
This program prints a name in a square pattern!
Name: Jerry
Jerry
e   r
r   r
r   e
yrreJ
```

以下三項重點提醒

1. 首先，您的程式應該會印出 “This program prints a name in a square pattern!”，並請使用者輸入任何一個名字
2. 不管使用者輸入的名字是否全為英文，皆可以印出相對應的方形形狀(如下圖)
3. 您的程式應該是 **case-sensitive** ，也就是輸入的文字大小寫會影響最後的答案

```
Name: Jennifer***
Jennifer***
e         *
n         *
n         r
i         e
f         f
e         i
r         n
*         n
*         e
***refinneJ
```

```
Name: **Dennis!
**Dennis!
*       s
D       i
e       n
n       n
n       e
i       D
s       *
!sinneD**
```

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、並確保程式並沒有卡在任何的無限迴圈（infinite loop）之中。

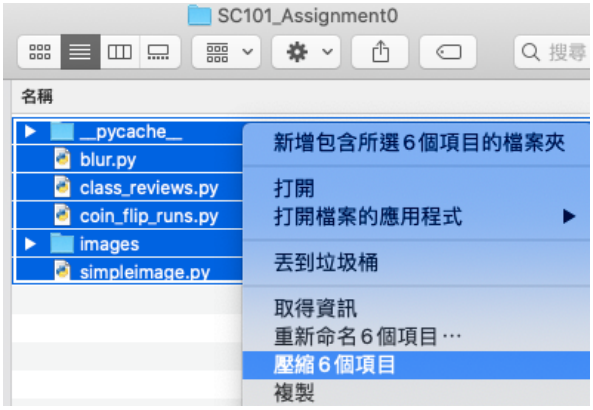
Style – 如同在課堂上所說，好的程式要有好的使用說明 (comment)，也要讓人一目瞭然，這樣全世界的人才能使用各位的 code 去建造更多更巨大更有趣的程式，因此請大家寫出**精簡扼要**的main( )程式概要、function comments和單行註解。

作業繳交

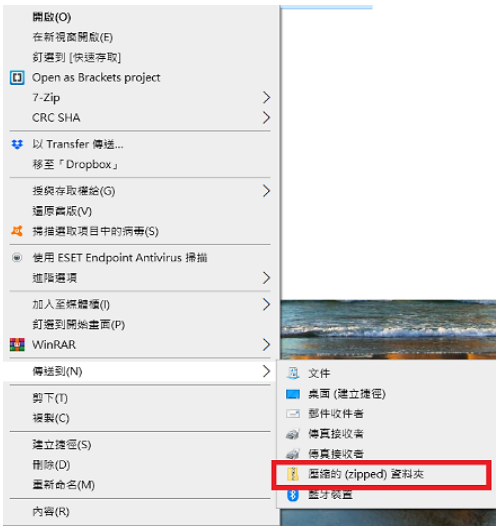
恭喜您完成Assignment3！請同學於**作業繳交期限前**，依照下圖將您完成的作業的**下載連結**上傳至社團提供的**作業繳交表單**。

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。

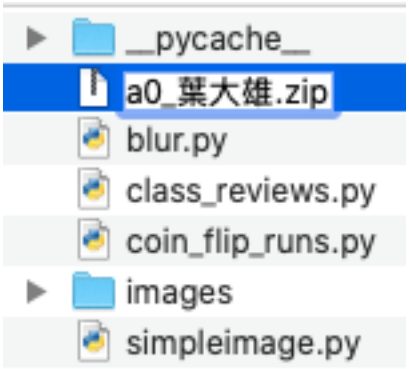
macOS：按右鍵選擇「**壓縮n個項目**」



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.