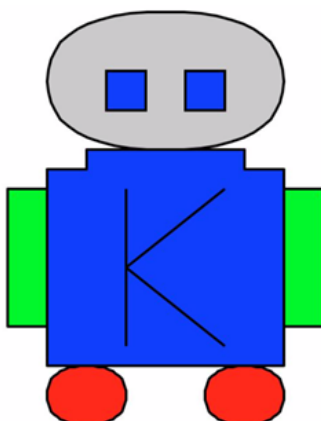


stanCode

標準程式教育機構

Assignment 1

Based on the Assignment 1 of CS106A at Stanford University



作業檔案下載

這份作業將訓練各位同學寫程式的基本技能，包括：寫出容易閱讀的格式 (Style)、清楚的使用說明 (Comment)、正確的空格 (Indentation)，以及寫程式最有趣的 – 除錯 (Debug) !

運用上課所教的概念就可以完成本次作業的全部內容 (估計需要時間為 13 小時)

注意：請勿使用未在第一堂課討論過的 Python 指令

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

請同學「**按照題目順序**」完成本次作業。本次作業需要大量的思考時間，請大家運用decomposition 概念，可以先將 algorithm 寫在紙上，再填進程式裡。

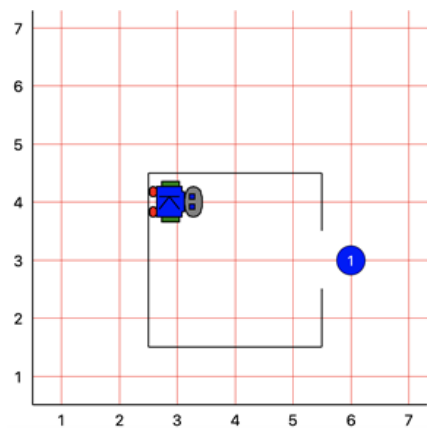
注意：每寫一個 function 就馬上測試一次！千萬不要等程式全部建造完畢才測試

Problem 1 - CollectNewsPaperKarel

開始撰寫程式碼之前，請先寫一段屬於自己的程式使用說明 – 註解(Comments)

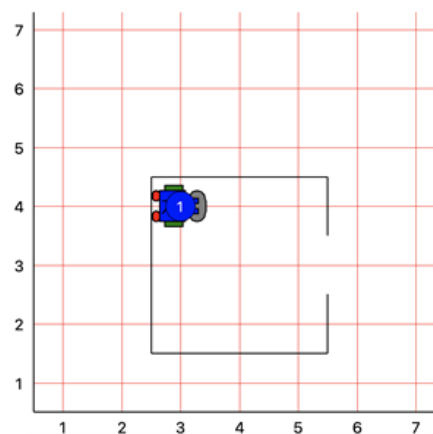
請同學將 `def main()` 裡面的 `TODO:` 刪去，再重新用自己的想法撰寫這個程式的概要 (盡量英文)。之後的每一提及所有作業也請同學記得做這件事！

Pre-condition



Karel一開始位在家的西北方。請同學協助 Karel 到家門外拿報紙 (也就是我們的 Beeper，位置永遠在 Street 3, Avenue 6)。拿完報紙後，請讓 Karel 回到原位、面向東方、放下報紙。

Post-condition



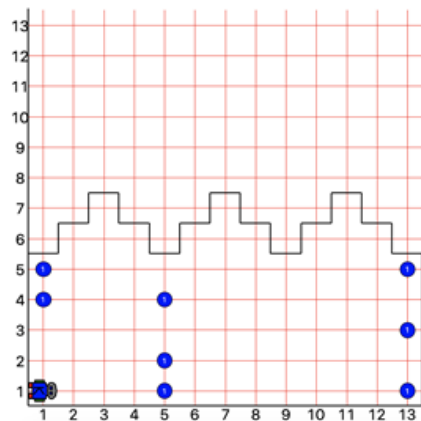
以下 三件事情 請同學注意：

1. 請寫兩個Functions 練習 “decomposition”
 - *Function (1) – 移動到報紙上
 - *Function (2) – 將報紙帶回家閱讀
2. 每一個function 都需要寫上它的「function comments」
3. 這題就是這麼簡單！千萬不要想太多！

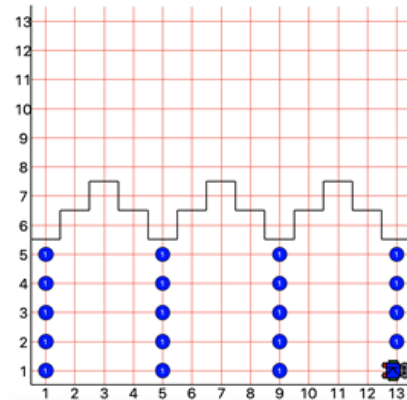
Problem 2 - StoneMasonKarel

台灣政府雇用 Karel 來維修因921大地震而坍塌的拱門，並提供 Karel 無限的 Beepers 原料來完成這項艱鉅的任務。請同學操控 Karel，讓 Karel 可以用 Beepers 將每一個拱門兩旁的柱子填滿，讓拱門變得更牢固！

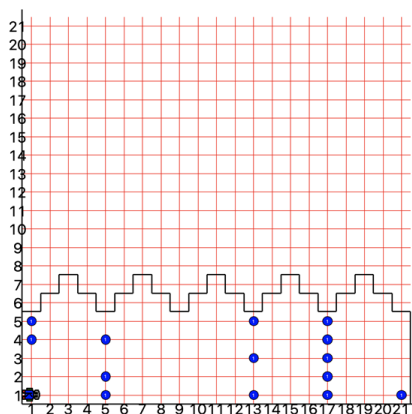
Pre-condition



Post-condition



當完成這份作業時，每一個拱門兩旁的柱子都會被 Beepers 填滿（如上方右圖所示），且 Karel 最後所在的位置會在 Street 1的最右邊，面對東邊。



請注意：每一個拱門的大小都一樣、每一根柱子與柱子之間的距離永遠都是差三格。然而，我們會用不同的 Worlds 來測試你們的程式。

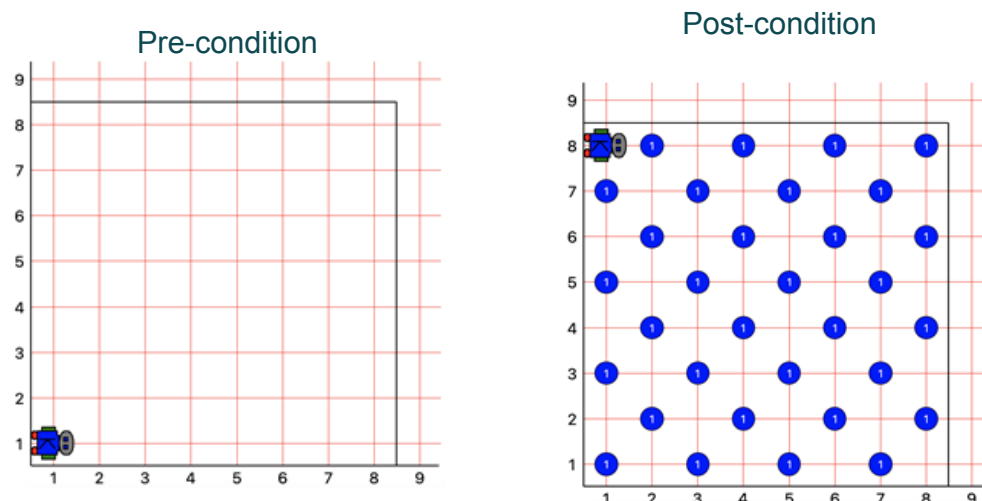
舉例來說，左圖的 World 共有 5 個拱門、6 根柱子，且每一根柱子跟柱子的中間都距離 3 格。

最後，**六項重點提醒**：

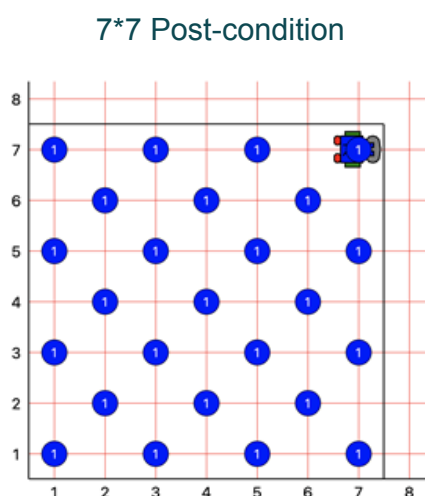
1. Karel 起始位置永遠是 (1, 1) 面向東方
2. 起始位置永遠有一根柱子（柱子左邊是牆壁）
3. 結束位置也都會有一根柱子（柱子右邊是牆壁）
4. 柱子缺的 Beeper 是隨機的，您的程式要能填滿任意柱子
5. 請勿在「已經有 Beeper」的地方放上 Beeper
6. Karel 不知道有幾根柱子要修理，但它知道最後一根柱子的右邊是牆壁

Problem 3 – CheckerboardKarel

Karel 是一名公共藝術家，他想用 Beepers 把它的世界彩繪成棋盤樣式！因此，你的任務是協助 Karel 在任何長方形、正方形的世界裡，用 Beepers 間格排列：一個有，下一個沒有 的方式排成棋盤狀(如下方右圖)。



Karel 一開始所在的位置為 (1, 1) – Street 1, Avenue 1 – 面向東邊 (facing east)。為了將世界彩繪成棋盤狀，karel 會先在 (1, 1) 放上 Beeper！然後，以 8*8 的世界為例，Karel 會在 Street 1 的 (1, 3) 與 (1, 5) 與 (1, 7) 放上 Beeper、Street 2 的 (2, 2) 與 (2, 4) 與 (2, 6) 與 (2, 8) 也都會被放上 Beeper。這份作業並 不要求 Karel 最後停在哪一格也 不要求 Karel 最後要面對哪個方向。



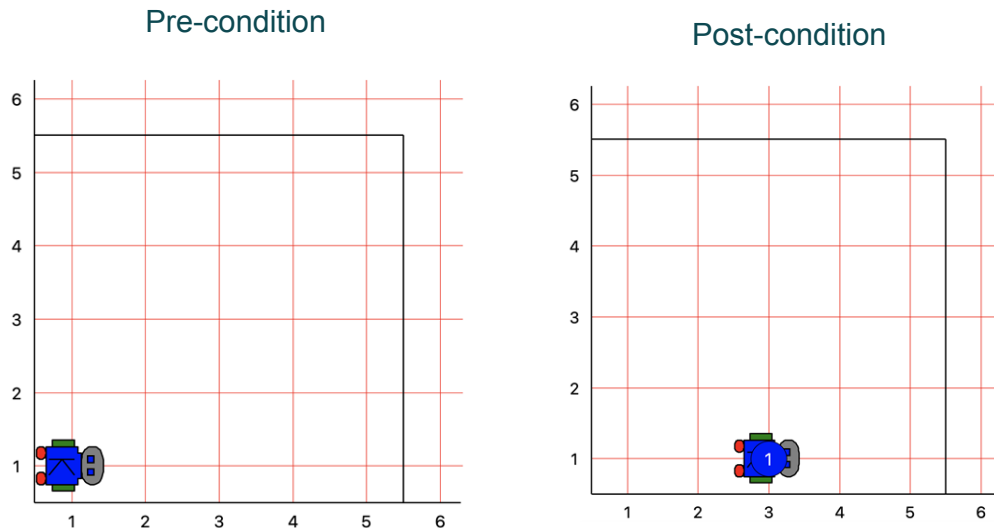
以下 四點建議 提供給各位同學：

1. 請先嘗試讓 Beepers 在 world 1*8 間隔排列
(把 code 寫成一個 function, 如 fill_one_line())
2. 嘗試 world 8*8
3. 嘗試 world 7*7 (如左圖所示)
4. 嘗試 world 8*1

這題困難之處並不在於程式本身，而是在想法。就算一位軟體工程師寫這份作業，可能也要花上至少一小時。因此，如果同學卡關，請不要感到灰心！建議大家先試著把想法寫在紙上，想想看行不行得通，之後再把 algorithm 寫成 code。

Problem 4 - MidpointFindingKarel

最後一題是這份作業看起來最簡單卻也最有挑戰性的。我們將使用上課學過的指令操作 Karel 去找到 **Street 1** 的中點在哪裡。



Karel 一開始位於 (1, 1) 並面向東邊。假設我們的世界是 5*5，那你的程式最後要讓 Karel 停在 (1, 3)（面向哪邊都可以），並放置一個 Beeper（如上右圖所示）假設今天的世界是偶數的（例如：10*10的世界），那最後不管讓 Karel 以及 Beeper 停在 (1, 5) 或是 (1, 6) 的位置都可以。

下列**五點重點**提醒：

1. Karel 擁有無限多的 Beepers
2. 我們的世界一定是正方形的，且中間沒有任何阻礙
3. 不論今天Karel的世界大小如何，Karel都要可以找到中點
4. **請勿使用 Python 變數**
5. 過程可以放 Beepers 做記錄，但最後只有Karel停留的中間點有一個Beeper。

這題有很多不同的解法，希望同學們能發揮創意！建議同學先把任何可行的演算法寫在紙上，再一步一步把程式建造起來。

如果同學們有其他解法，歡迎使用我們為各位建立的extension的py檔案。除功課要求的一種寫法外，如果同學能寫出另外四種解法，就有機會獲得++喔！（注意：Extension py檔案的 parameters 一樣都是輸入 worlds/ MidpointKarel.kwld 喔！）

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。

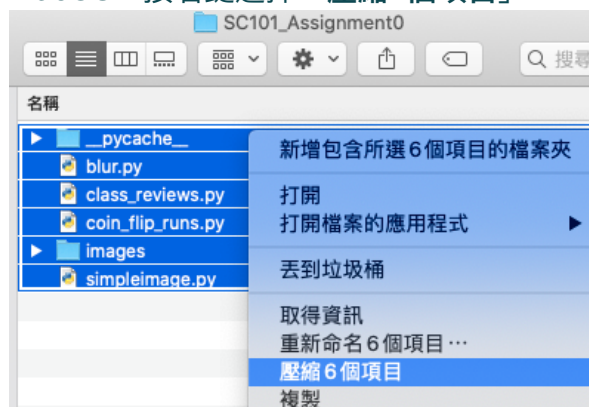
Style – 如同在課堂上所說，好的程式要有好的使用說明（comment），也要讓人一目瞭然，這樣全世界的人才能使用各位的 code 去建造更多更巨大更有趣的程式，因此請大家寫出**精簡扼要**的main()程式概要、function comments和單行註解。

作業繳交

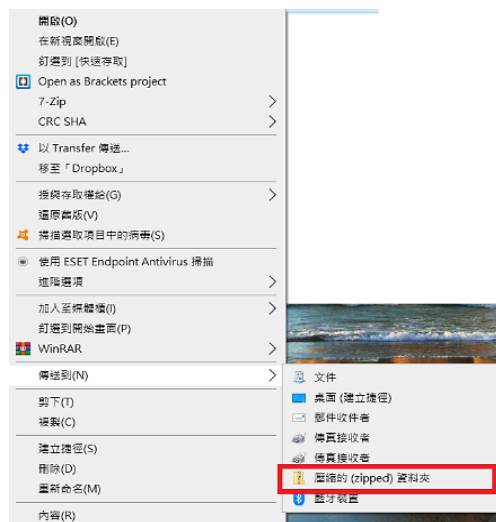
恭喜您完成 Assignment 1！請同學於**作業繳交期限前**，依照下圖將您完成的作業的**下載連結**上傳至社團提供的**作業繳交表單**。

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。

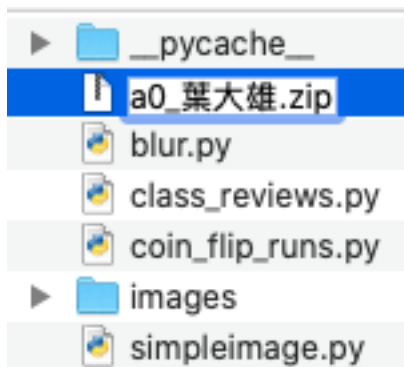
macOS：按右鍵選擇「壓縮n個項目」



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