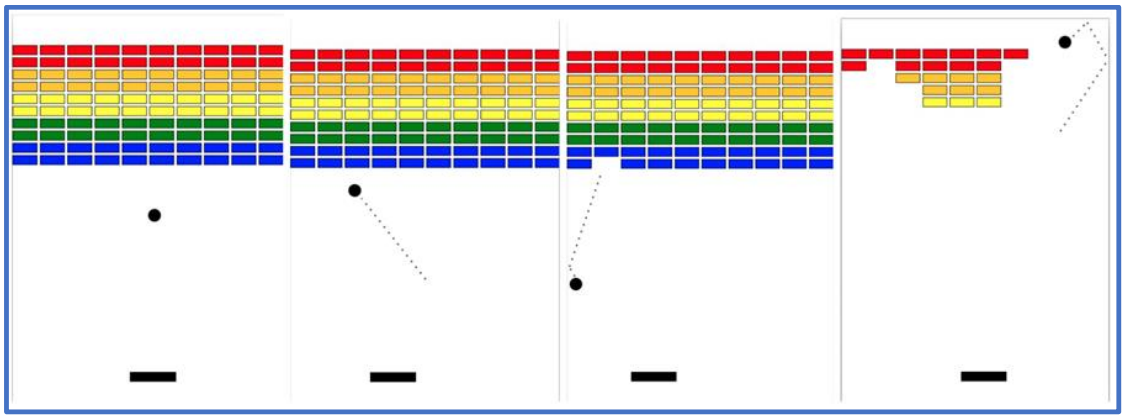
stanCode

標準程式教育機構

Assignment 5

This assignment is based on the Assignment 5 of CS106AP at Stanford University



作業檔案下載

歡迎各位同學來到打磚塊！這份作業將使用電腦科學中最困難的概念之一 (Class and Object) 來寫出一個 Python 遊戲。完成這份作業之後，您將可以非常驕傲地跟朋友、家人、甚至面試官展示作品，相信不管是誰看到都會非常驚艷的！

本次作業的繳交時間為 **12/24 (四) 23:59**

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，也會因此讓其他學生的程式碼與你/妳的極度相似，讓防抄襲軟體認定有抄襲嫌疑!

The Breakout Game - 遊戲介紹

如**作業開頭的圖片**所示，遊戲視窗上方有許多磚塊 bricks（紅色、橘色、黃色、綠色、藍色）遊戲視窗下方有一個隨著滑鼠移動的板子 paddle（黑色）

遊戲進行方式為利用 paddle 反彈球（入射角等於反射角），消滅所有磚塊。圖片中，球的後方黑線是我們後製加上去的，實際遊戲裡並不存在，黑線的目的只是讓大家更了解球反彈的軌跡

遊戲終止的條件有兩個：玩家「消滅所有磚塊」或是當「球超過視窗下方三次」。換句話說，玩家有三次機會消滅所有磚塊

The starter files

本次作業的資料夾中包含了兩份檔案。第一份，**breakout.py**，包含了 main( ) 處理主要邏輯並讓遊戲動畫順利進行。第二份，**breakoutgraphics.py**，定義一個 class 叫做 **BreakoutGraphics** 去處理所有重要的圖像元件！兩份檔案我們都寫好了一些起始程式碼，裡面包含：

- 所有需要 import 以及您會用到的 classes
- 定義遊戲畫面元件位置、大小的常數。您的程式務必使用我們幫各位定義的常數，讓遊戲可以隨著改變常數數值而改變。詳細常數 constants 說明如下

Constants 介紹

```
BRICK_SPACING = 5      # Space between bricks (in pixels). This space is used for horizontal and vertical spacing.
BRICK_WIDTH = 40       # Width of a brick (in pixels).
BRICK_HEIGHT = 15      # Height of a brick (in pixels).
BRICK_ROWS = 10        # Number of rows of bricks.
BRICK_COLS = 10        # Number of columns of bricks.
BRICK_OFFSET = 50      # Vertical offset of the topmost brick from the window top (in pixels).
BALL_RADIUS = 10       # Radius of the ball (in pixels).
PADDLE_WIDTH = 75      # Width of the paddle (in pixels).
PADDLE_HEIGHT = 15     # Height of the paddle (in pixels).
PADDLE_OFFSET = 50     # Vertical offset of the paddle from the window bottom (in pixels).
INITIAL_Y_SPEED = 7.0  # Initial vertical speed for the ball.
MAX_X_SPEED = 5      # Maximum initial horizontal speed for the ball.
```

1. BRICK_SPACING 為磚塊與磚塊之間（左右、上下）的小空隙
2. BRICK_WIDTH 為一個磚塊的寬
3. BRICK_HEIGHT 為一個磚塊的高
4. BRICK_ROWS 為總共有幾列磚塊
5. BRICK_COLS 為總共有幾行磚塊
6. BRICK_OFFSET 為第一列磚塊頂部與視窗頂端之距離
7. BALL_RADIUS 為球的半徑
8. PADDLE_WIDTH 為板子的寬
9. PADDLE_HEIGHT 為板子的高
10. PADDLE_OFFSET 為板子與視窗底部之距離
11. INITIAL_Y_SPEED 為初始球在 y 方向移動的速度
12. MAX_X_SPEED 為球在 x 方向移動的最大速度

Milestone 1 - BreakoutGraphics Constructor (breakoutgraphics.py)

這是一份相對複雜的作業，因此我們將重點分割，讓同學可以用里程碑 - **milestone** - 來安排作業進度。請同學務必按照 **milestone** 的順序完成本次作業

首先，請到 **breakoutgraphics.py** 檔案裡將 **BreakoutGraphics** 這個 class 的 constructor 完成（請同學不要改變已經寫好的 constructor keyword arguments）

如下圖（一）程式碼所示，視窗 GWindow 的部分已經完成，並以 **self.window** 儲存 GWindow 的 instance。如此一來，使用者在編輯動畫時就可以呼叫此視窗並改變它上面的一切圖像。您的工作將接續完成板子 (paddle)、球 (ball)、球的速度 (dx, dy)、滑鼠功能啟動 (onmousemoved(...), onmouseclicked(...))、並在最後畫上所有的磚塊 (bricks)。**Constructor** 的目標就是完成所有靜止的遊戲基本圖像。請先不必擔心動畫的部分，我們會在後面的 milestone 再跟同學介紹

```
class BreakoutGraphics:

    def __init__(self, ball_radius=BALL_RADIUS, paddle_width=PADDLE_WIDTH,
                 paddle_height=PADDLE_HEIGHT, paddle_offset=PADDLE_OFFSET,
                 brick_rows=BRICK_ROWS, brick_cols=BRICK_COLS,
                 brick_width=BRICK_WIDTH, brick_height=BRICK_HEIGHT,
                 brick_offset=BRICK_OFFSET, brick_spacing=BRICK_SPACING,
                 title='Breakout'):

        # Create a graphical window, with some extra space.
        window_width = brick_cols * (brick_width + brick_spacing) - brick_spacing
        window_height = brick_offset + 3 * (brick_rows * (brick_height + brick_spacing) - brick_spacing)
        self.window = GWindow(width=window_width, height=window_height, title=title)

        # Create a paddle.
        # Center a filled ball in the graphical window.
        # Default initial velocity for the ball.
        # Initialize our mouse listeners.
        # Draw bricks.
```

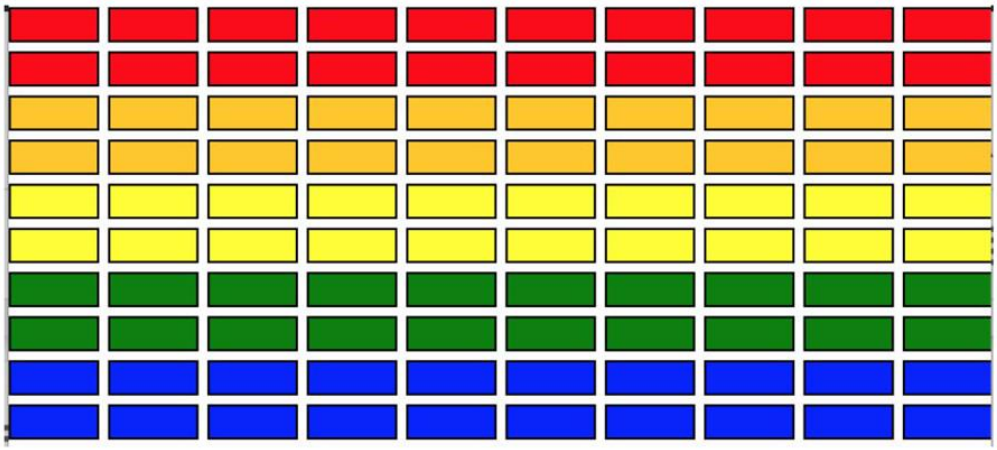
圖（一）breakoutgraphics.py 檔案內容

您的第一步應該先：

- 製造 paddle 並將它加在 window 底部中間 (paddle 的視窗底部的距離定義在常數 PADDLE_OFFSET)
- 製造 ball 並將它加在 window 的正中間
- 單行註解提到的 **Default initial velocity for the ball** 請先忽略，球的起始速度數值 - **dx**, **dy** - 我們會在 **Milestone 2** 詳細與大家說明
- 啟動兩個滑鼠程式 - **onmouseclicked** 以及 **onmousemoved** - 但放入括弧裡的程式先不需要處理

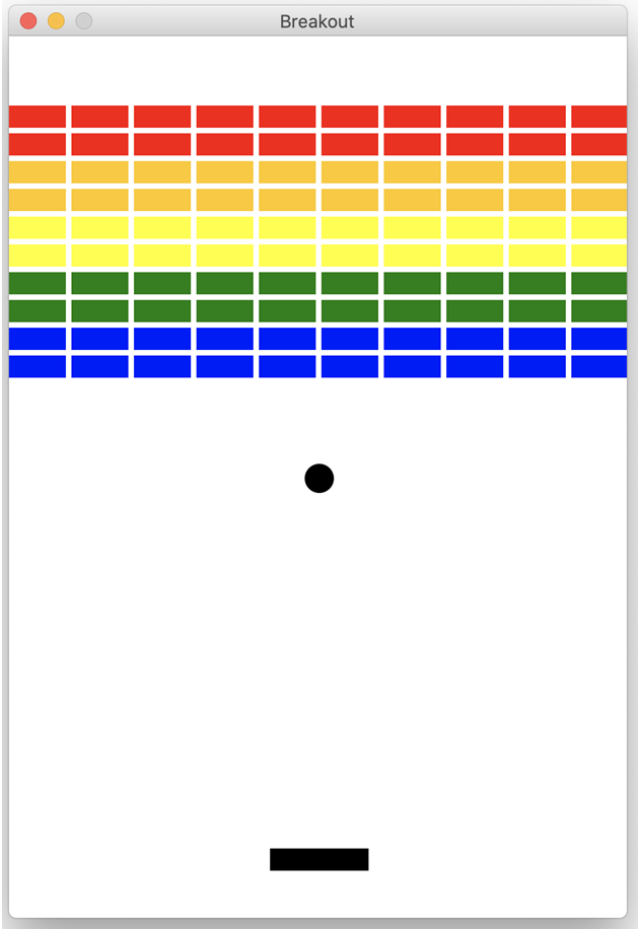
再來會是 milestone 1 最複雜的部分 - 將磚塊放上去。如下圖（二）所示，每一排的第一個磚塊 x 座標都是 0；最後一個磚塊的右邊會緊鄰右側視窗。請同學使用我們定義的常數
BRICK_SPACING, BRICK_WIDTH, BRICK_HEIGHT, BRICK_ROWS, BRICK_COLS, BRICK_OFFSET 來完成與圖（二）一模一樣的磚塊配置。

請注意：若使用者更動常數 BRICK_ROWS, BRICK_COLS 數值，您的磚塊數目應該也可以隨之變動



圖（二）所有磚塊完成示意圖

若完成上述所有項目，恭喜！Milestone 1 就結束、打磚塊的雛型也完成了！若您現在執行 **breakout.py**，彈出的視窗會與下圖（三）相同



圖（三）打磚塊起始畫面。此時畫面都是靜止的

Milestone 2 - Even-driven && Animation

里程碑 2 將開始建造動畫。首先，第一步請同學在 **breakoutgraphics.py** 將滑鼠與 paddle 連結起來！當電腦偵測到滑鼠移動，程式中 **onmousemoved(...)** 括弧內的方程式就會被啟動。這部分需要注意的地方有兩點：

- 請讓 **paddle 的中點**隨著滑鼠移動，且 paddle 的 y 座標永遠都固定在 PADDLE_OFFSET
- 請勿讓 paddle 超過我們視窗的左右兩側；也就是說，就算滑鼠移到視窗外，paddle 整體應該都還是停留在視窗中

再來，我們將讓球動起來！這個部分請同學先忽略磚塊、板子的反彈，以及球跑到視窗下方死掉的情況（上述的所有我們都會在下一個 milestone 完成）

為了讓球動起來，我們必須先定義球的水平速度 dx 以及垂直速度 dy。然而，dx, dy 是打磚塊遊戲中最重要的變數（我們不希望使用者隨便更動這個遊戲的靈魂！）因此，我們要把 dx, dy 定義成 **__dx**, **__dy**。

完成後，請同學回到 **breakoutgraphics.py** 的 constructor 將 __dy 設為我們定義的常數 INITIAL_Y_SPEED。然而，為了讓遊戲有趣，身為貼心遊戲設計者，我們要盡量避免「球垂直落下」與「每次都往相同方向」的窘境。因此，請同學使用 random **從 1 到 MAX_X_SPEED** 隨機選一個**整數**，再使用下方程式碼隨機變動方向

```
if (random.random( ) > 0.5):
    __dx = -__dx
```

一旦您定義好速度，接下來的挑戰將是讓球可以在視窗的上、下、左、右牆壁反彈！

請同學換到 **breakout.py** 檔案，讓球可以先忽略磚塊與板子而只與視窗反彈。

然而，這邊同學會遇到無法得到 __dx 與 __dy 的困境。因此，請同學在 **breakoutgraphics.py** 定義兩個 **getters**。

製作遊戲動畫最重要的莫過於 pause。因此，請同學務必在操控動畫的 while loop 裡使用我們定義的常數，放上 **pause(FRAME_RATE)**

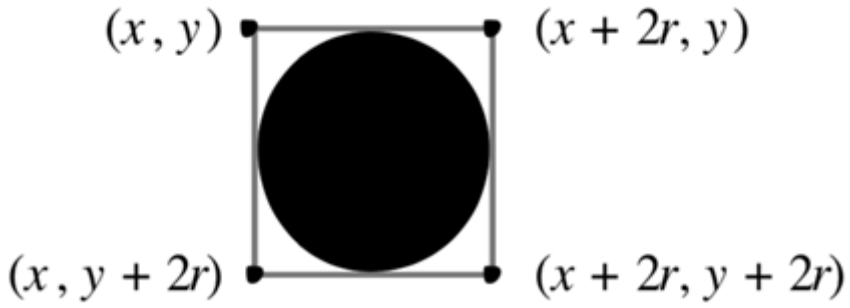
最後，Milestone 2 最困難的部分在於：我們該怎麼在使用者點按滑鼠後開始動畫，並**不會受到重複點按影響**？這邊我們將讓您思考：）

HINT：onmouseclicked(...)括弧裡面的程式設計必須偵測「遊戲是否已經開始？」

Milestone 3 - Check for collisions

這份作業最有趣也最具挑戰性的部分莫過於反彈條件。再來，我們將讓球碰到磚塊與板子會反彈，並讓「球超過視窗底部」代表遊戲結束。因此，在您製作動畫的 while loop 裡，每一圈都要檢查一次是否需要改變方向、結束遊戲、抑或是移除物件。因為球碰到物件（板子、磚塊）反彈與牆壁反彈不同，我們要製作一個 method 特別處理與物件的碰撞。為了製作此 method ，我們須要思考：「球是否碰到物件」這件事該怎麼定義？

球的四個座標如下圖所示：



若我們分別將上圖四個頂點丟入

window.get_object_at(..., ...)

神奇的事情是，那四個點都不在球上！因此，我們如果用那**四的頂點當作探針**，並在 while loop 裡的每一圈輪流使用四個頂點探測是否有碰到 object，而不是球本身，我們就可以進而將探測到的 object 移除或是改變球的移動方向。換句話說，球的四個頂點要輪流做的事情有四項：

1. 使用 get_object_at( ) 探測該點是否有物件
2. 如果得到的物件**不是 None**，就可以將此物件 return 出您定義的 method
3. 如果某一頂點探測結果**是 None**，請往下檢查另一個頂點
4. 如果四個頂點**都得到 None**，那麼我們就可以確定沒有碰撞發生

當您成功完成上述要求，此 milestone 的最後一步就是判斷碰撞的物件到底是 paddle 還是 bricks？如果是 paddle，我們只需改變球的方向；然而，如果是 bricks，我們不僅要改變球的方向，還要移除該物件!

Finishing up - 完成

當您抵達這個部分時，代表作業大致上已經完成（Yeah!）然而以下幾點常見問題請同學留意：

- 當球超過視窗下方不能讓它反彈
- 遊戲終止條件為「消滅所有磚塊」或是「球碰到視窗下方的次數等於 NUM_LIVES」
- 最常見的 bug 是「當球快要通過板子時，快速移動板子讓球反彈」。您的程式是否會讓球好像「黏」在板子上呢？如果有，想想看到底是什麼地方出了問題

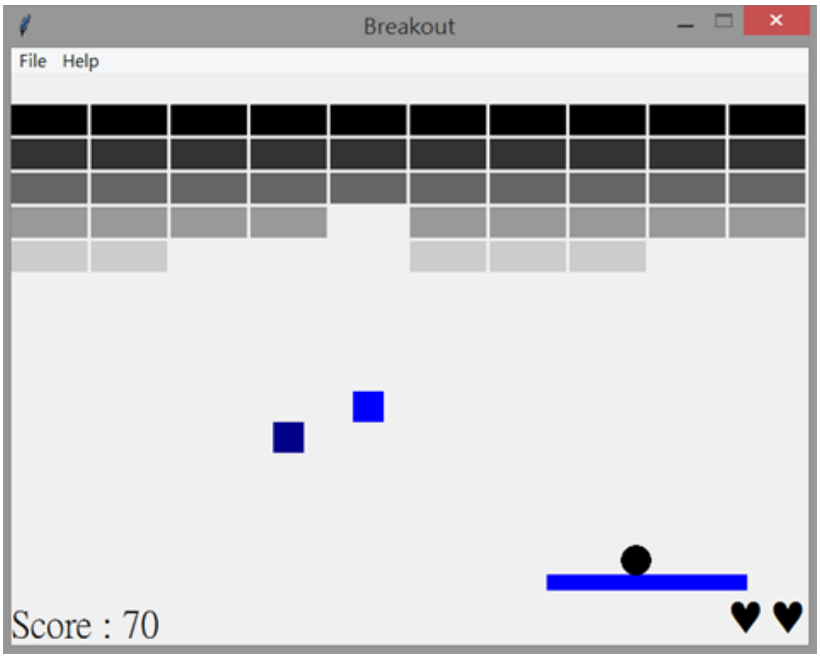
為了讓教學團隊驚艷，請在完成上述基本要求之外加入有趣的**延伸（Extensions）**。若決定要加入 Extensions 的同學，請**另外開一個檔案**，以方便我們測試您打磚塊遊戲基本的要求是否通過。以下提供一些延伸的想法給各位參考：

1. 加入計分板 GLabel！然而要注意的是，當球碰到計分板是否會反彈？

2. 真實的打磚塊遊戲會將球的速度隨著分數變高而變快，破關的難度瞬間提升！

3. 每一個磚塊的分數可以不一樣（例如紅色最高分）

4. 發揮你的想像力！期待看到有趣的延伸：）



breakout ++

評分標準

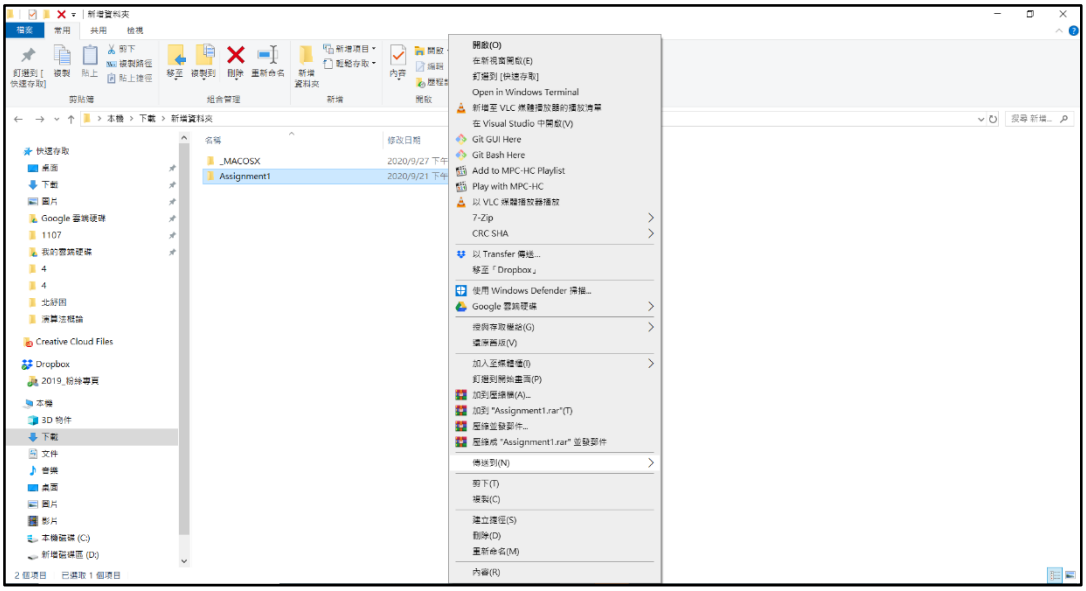
Functionality - 程式是否有通過我們的基本要求？程式必須沒有 bug 、能順利完成指定的任務、並確保程式沒有卡在任何的無限環圈（Infinite loop）之中.

Style - 如同我們在課堂上所說，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、method 敘述、Class 敘述、單行註解（簡單觀念不用說明）、method 名稱的命名

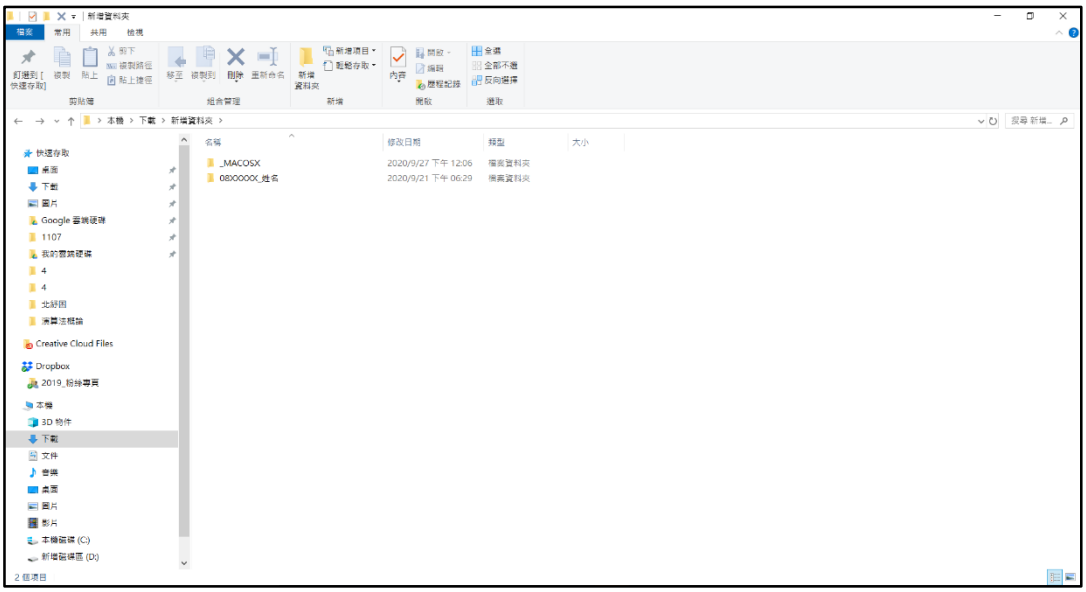
作業繳交

恭喜各位完成 Assignment 5! 大家應該要對自己的成就感到驕傲，因為這份作業跟史丹佛大學的學生作業非常相似，代表你們跟世界各國的菁英一樣厲害了

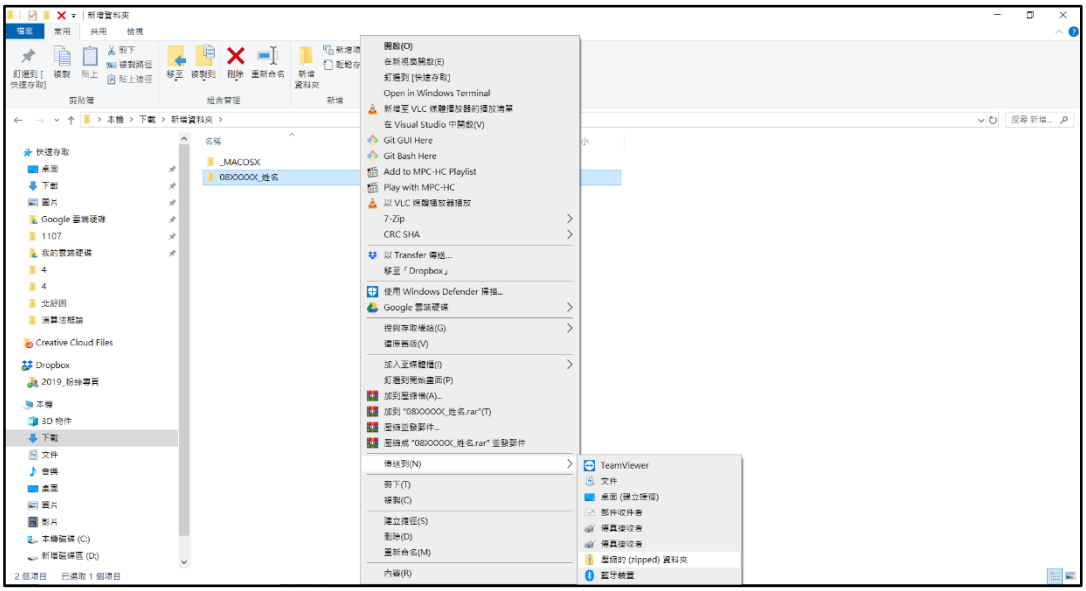
請同學於 **12/24(四) 23:59** 前依照下圖將作業上傳



找到作業資料夾，按右鍵，選擇重新命名

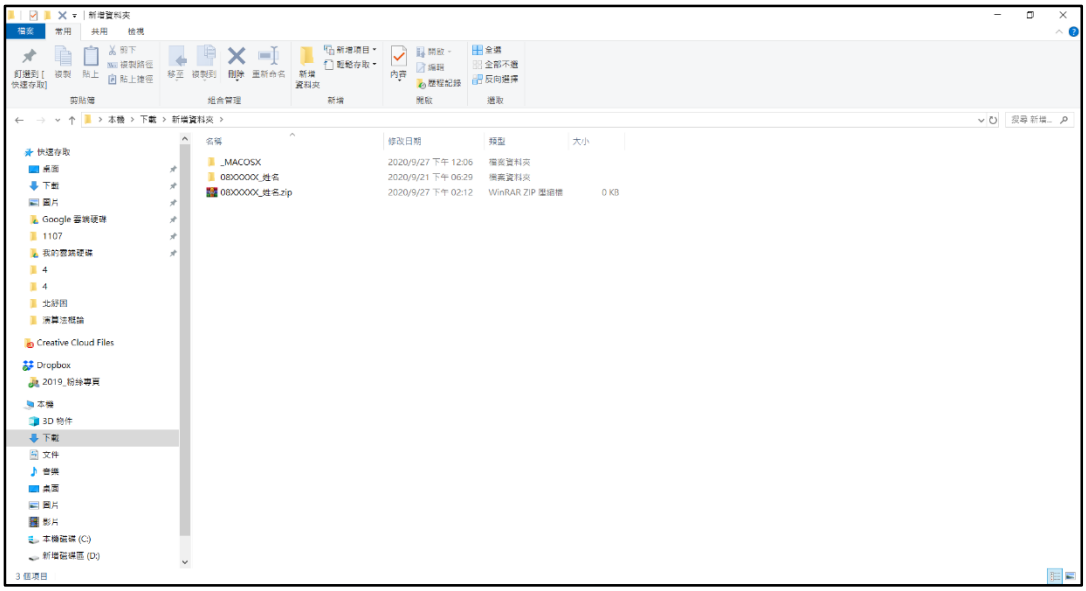


請命名成「學號_中文姓名」的格式



Windows 請點選「傳送到」->「壓縮的(zipped)資料夾」

Mac 請點選 Compress "學號_中文姓名"


新增資料夾
檔案
常用
共用
檢視
本機 › 下載 › 新增資料夾 ›
名稱
修改日期
類型
大小
_MACOSX
2020/9/27 下午 12:06
檔案資料夾
08X0000X_姓名
2020/9/21 下午 06:29
檔案資料夾
08X0000X_姓名.zip
2020/9/27 下午 02:12
WinRAR ZIP 壓縮檔
0 KB
3 個項目

成品如上


國立交通大學 數位教學平台
【109上】1308計算機概論 Introduction to Computer Science
進行中作業
名稱
開始時間
結束時間
繳交狀態
功能
Assignment 1-Test
2020/09/21 00:00
2020/09/28 00:00
0 個已繳
55 個未繳
[檢視]
已繳作業
逾期未繳作業
預告作業

點選「檢視」

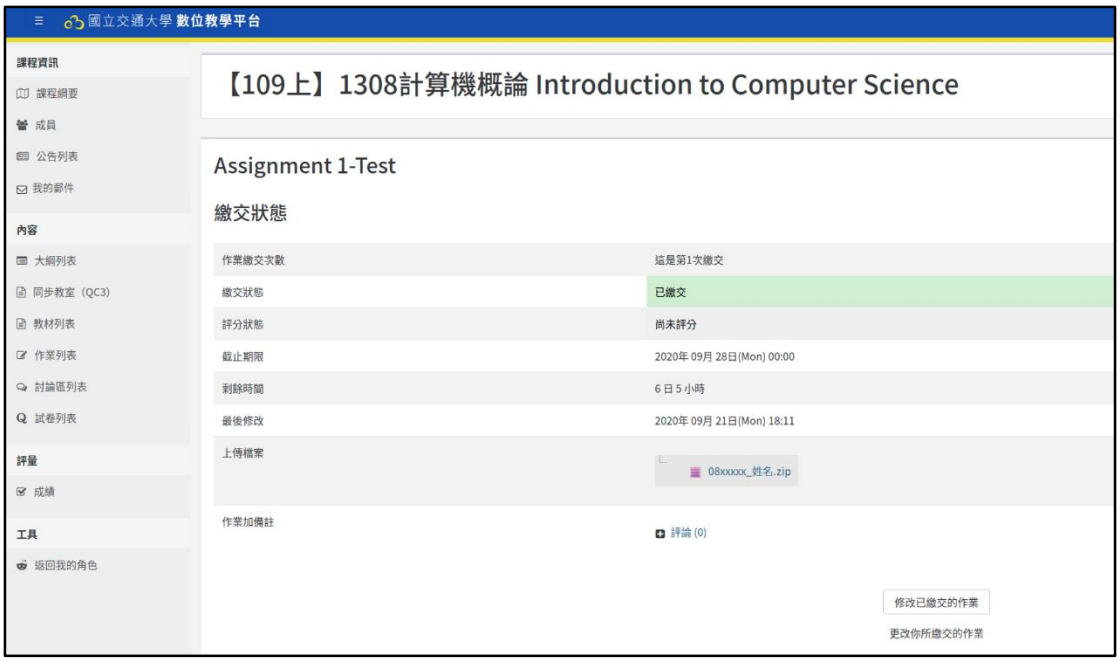
按下「繳交作業」

按下「檔案圖示」

選擇先前壓縮好的作業檔案

按下「上傳這一檔案」

按下「儲存更改」



出現「已繳交」確定成功

stanCode - 標準程式教育機構

Should you have any questions please feel free to contact us.