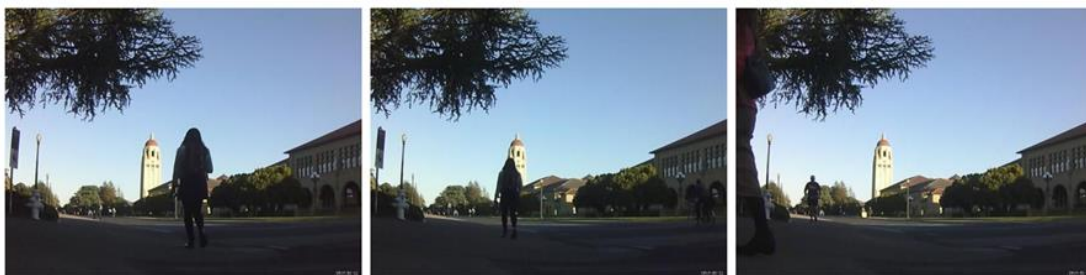


stanCode

標準程式教育機構

Assignment 6-I

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



作業檔案下載

這份作業的第一部份將帶領同學熟悉我們在 **Python** 習得的第一個資料結構 **list** 並挑戰真實軟體工程師的工作內容 - 接續他人撰寫的程式碼，完成一份完整的程式作品！

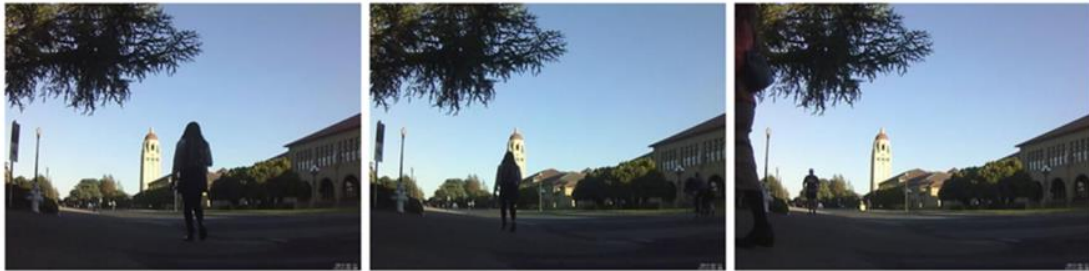
本次作業的繳交時間為 **12/31 (四) 23:59**

(估計需要時間為 **4 小時**)

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（也不要將程式碼貼在社團裡）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

stanCodoshop.py

stanCode 推出一個全新 APP - **stanCodoshop** 讓使用者可以將美麗風景照裡的路人消失！（觀光景點超多路人一直出現真的很困擾啊～～）



若使用者輸入上圖所示的 3 張「位置相同，但都有路人出沒的史丹佛大學風景照」到神奇 APP **stanCodoshop.py**，將得到意想不到的美圖（如下圖）：



Algorithm

假設使用者輸入了 N 張照片（路人分別出現在不同位置）我們可以比對每一張照片的**每一個 pixels**，選擇沒有路人的 pixel 並将它複製到空白檔案

然而，我們要如何在 N 張照片挑選出特定位置 (x, y) 且沒有路人的 pixel 呢？

假設使用者輸入的照片數量 $N = 4$ 。這些照片在特定位置 (x, y) 的 (r, g, b) 數值分別為： $(1, 1, 2)$, $(1, 1, 1)$, $(1, 2, 2)$, $(28, 27, 29)$ ，如此我們就可以很明顯地發現： $(28, 27, 29)$ 異常突兀！（代表 `image4.get_pixel(x, y)` 很有可能是路人！）

Color Distance

雖然我們可以很輕易地挑出突兀之 pixel 並棄之而不顧，但我們又該怎樣選出最好的 pixel 呢？因此我們要借助一個名為 **Color Distance** 的演算法來達成目標。試想：若我們將圖片一在 (x, y) 這點的 R, G, B 數值分別填入我們熟知的直角坐標

系 x, y, z 座標就會成為三維空間中的一個點。如果我們進一步將剩下 3 張照片在 (x, y) 的 R, G, B 數值填入 x, y, z 座標，三維空間中就會再多出 3 個點（如下圖所示）。而點與點之間的距離就稱作 **Color Distance**

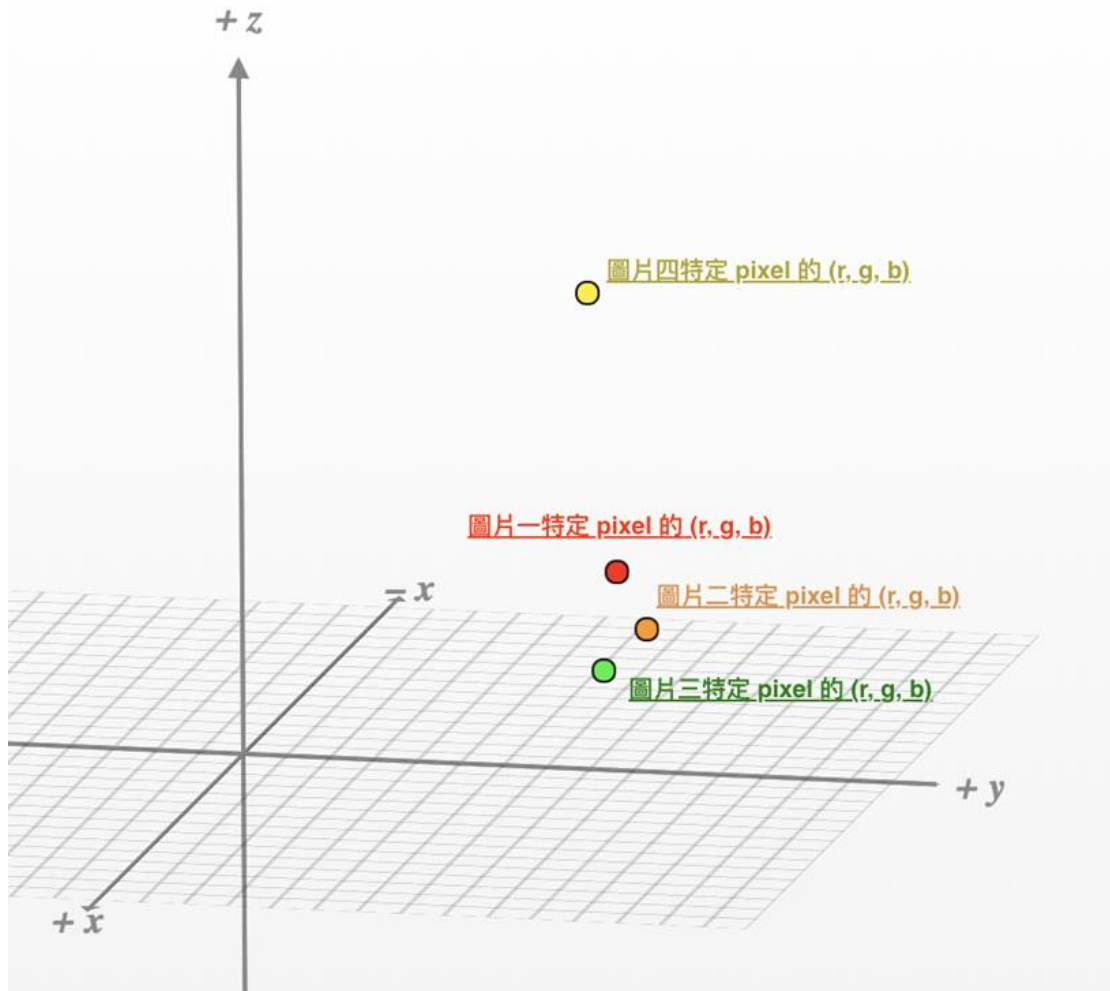


Fig. 1 - 四張圖片在 (row, col) 位置的 pixel 之 r, g, b 座標點示意圖

假設 N 個點之平均 RGB 數值分別為 **red_avg**, **green_avg**, **blue_avg** 並在空間中佔據一個點 (以下簡稱 **avg_point**)，我們可以使用下列公式計算某 **pixel** 與 **avg_point** 之 **Color Distance**：

$$color_distance = \sqrt{(red_avg - pixel.red)^2 + (green_avg - pixel.green)^2 + (blue_avg - pixel.blue)^2}$$

Milestone 1 - get_pixel_dist(pixel, red, green, blue)

請完成 `stanCodoshop.py` 檔案裡名為 `get_pixel_dist(pixel, red, green, blue)` 之 function 並 return 「pixel 與 avg 之 color distance」。red, green, blue 分別為 N 張圖片 已經算好的平均 R, G, B 數值

要檢查是否正確，請在 `def solve(images):` 裡面加入紅色框框裡的三行程式碼（如下圖所示）

```
def solve(images):
    """
    Given a list of image objects, compute and display a Ghost solution image
    based on these images. There will be at least 3 images and they will all
    be the same size.

    Input:
    images (List[SimpleImage]): list of images to be processed
    """
    width = images[0].width
    height = images[0].height
    result = SimpleImage.blank(width, height)
    ##### YOUR CODE STARTS HERE #####
    # Write code to populate image and create the 'ghost' effect
    green_im = SimpleImage.blank(20, 20, 'green')
    green_pixel = green_im.get_pixel(0, 0)
    print(get_pixel_dist(green_pixel, 5, 255, 10))
    ##### YOUR CODE ENDS HERE #####
    print("Displaying image!")
    result.show()
```

按下 PyCharm 下方的 Terminal

Mac 電腦輸入 "python3 stanCodoshop.py hoover"

windows 電腦輸入 "py stanCodoshop.py hoover"

看到下方字樣即完成 **Milestone 1**

```
Loading hoover/200-500.jpg
Loading hoover/158-500.jpg
Loading hoover/156-500.jpg
11.180339887498949
Displaying image!
```

Milestone 2 - get_average(pixels)

請完成 **stanCodoshop.py** 檔案裡名為 **get_average(pixels)** 之程式並

return 包含N張圖片 R, G, B 數值平均的 Python list - **[red, green, blue]**

pixels 為一個 **list**，裡面有N張圖片在某特定位置(**row, col**)的N個 **pixels**。return 出去的 **red, green, blue** 分別代表這N個點的平均紅色數值、綠色數值、與藍色數值

要檢查是否正確，請在 **def solve(images):** 裡面加入紅色框框裡的四行程式碼（如下圖所示）

```
def solve(images):  
    """  
    Given a list of image objects, compute and display a Ghost solution image  
    based on these images. There will be at least 3 images and they will all  
    be the same size.  
  
    Input:  
    | images (List[SimpleImage]): list of images to be processed  
    """  
    width = images[0].width  
    height = images[0].height  
    result = SimpleImage.blank(width, height)  
    ##### YOUR CODE STARTS HERE #####  
    # Write code to populate image and create the 'ghost' effect  
    green_pixel = SimpleImage.blank(20, 20, 'green').get_pixel(0, 0)  
    red_pixel = SimpleImage.blank(20, 20, 'red').get_pixel(0, 0)  
    blue_pixel = SimpleImage.blank(20, 20, 'blue').get_pixel(0, 0)  
    print(get_average([green_pixel, green_pixel, green_pixel, blue_pixel]))  
    ##### YOUR CODE ENDS HERE #####  
    print("Displaying image!")  
    result.show()
```

按下 PyCharm 下方的 Terminal

Mac 電腦輸入 "python3 stanCodoshop.py hoover"

windows 電腦輸入 "py stanCodoshop.py hoover"

看到下方字樣即完成 **Milestone 2**

```
Loading hoover/200-500.jpg  
Loading hoover/158-500.jpg  
Loading hoover/156-500.jpg  
[0, 191, 63]  
Displaying image!
```


Milestone 3 - get_best_pixel(pixels)

請使用 **Milestone 1** and **Milestone 2** 您撰寫的指令完成 **stanCodoshop.py** 檔案裡名為 **get_best_pixel(pixels)** 之程式並 **return** 「**pixels** 之中最好的一個 **pixel**」 (最好的 **pixel** 定義為「與平均 **RGB** 數值點距離最近之 **pixel**」)

要檢查是否正確，請在 **def solve(images)** 裡面加入紅色框框裡的五行程式碼 (如下圖所示)

```
def solve(images):
    """
    Given a list of image objects, compute and display a Ghost solution image
    based on these images. There will be at least 3 images and they will all
    be the same size.

    Input:
    | images (List[SimpleImage]): list of images to be processed
    """
    width = images[0].width
    height = images[0].height
    result = SimpleImage.blank(width, height)
    ##### YOUR CODE STARTS HERE #####
    # Write code to populate image and create the 'ghost' effect
    green_pixel = SimpleImage.blank(20, 20, 'green').get_pixel(0, 0)
    red_pixel = SimpleImage.blank(20, 20, 'red').get_pixel(0, 0)
    blue_pixel = SimpleImage.blank(20, 20, 'blue').get_pixel(0, 0)
    best1 = get_best_pixel([green_pixel, blue_pixel, blue_pixel])
    print(best1.red, best1.green, best1.blue)
    ##### YOUR CODE ENDS HERE #####
    print("Displaying image!")
    result.show()
```

按下 PyCharm 下方的 Terminal

Mac 電腦輸入 "python3 stanCodoshop.py hoover"

windows 電腦輸入 "py stanCodoshop.py hoover"

看到下方字樣即完成 **Milestone 3**

```
Loading hoover/200-500.jpg
Loading hoover/158-500.jpg
Loading hoover/156-500.jpg
0 0 255
```

Milestone 4 - solve(images)

最後，請同學將上方**所有紅色框框的測試程式刪除**，開始建造我們偉大的 APP!

請注意：**def solve(images)** 傳入的 **images** 為一個 list，儲存使用者在同一個景點拍攝的多張圖片。您的工作就是在 **images** 裡找到 (x, y) 這個位置最好的 pixel（以下簡稱 **best**）並將空白檔案 **result** 在 (x, y) 的 pixel 填入 **best** 之 RGB 數值

我們在 **def solve(images)** 的最後寫了 **result.show()**。因此若您成功完成這份作業，只有景點而沒有路人的照片就會在 30 秒內出現在您的電腦螢幕上

測試程式是否正確，我們提供了四套圖組（皆為史丹佛大學知名的景點！）請在 **Terminal** 輸入下方指令

- **Mac**
 1. `python3 stanCodoshop.py clock-tower`
 2. `python3 stanCodoshop.py hoover`
 3. `python3 stanCodoshop.py math-corner`
 4. `python3 stanCodoshop.py monster`
- **Windows**
 1. `py stanCodoshop.py clock-tower`
 2. `py stanCodoshop.py hoover`
 3. `py stanCodoshop.py math-corner`
 4. `py stanCodoshop.py monster`

評分標準

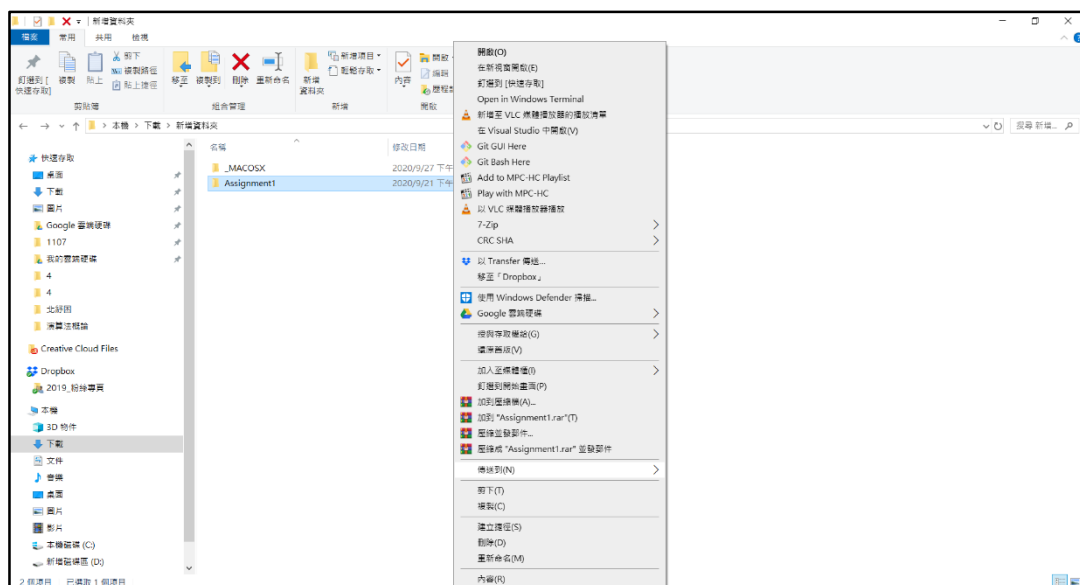
Functionality - 程式是否有通過我們的基本要求？程式必須沒有 **bug**、能順利完成指定的任務、並確保程式沒有卡在任何的無限環圈（**Infinite loop**）之中。

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、**function** 敘述、單行註解。

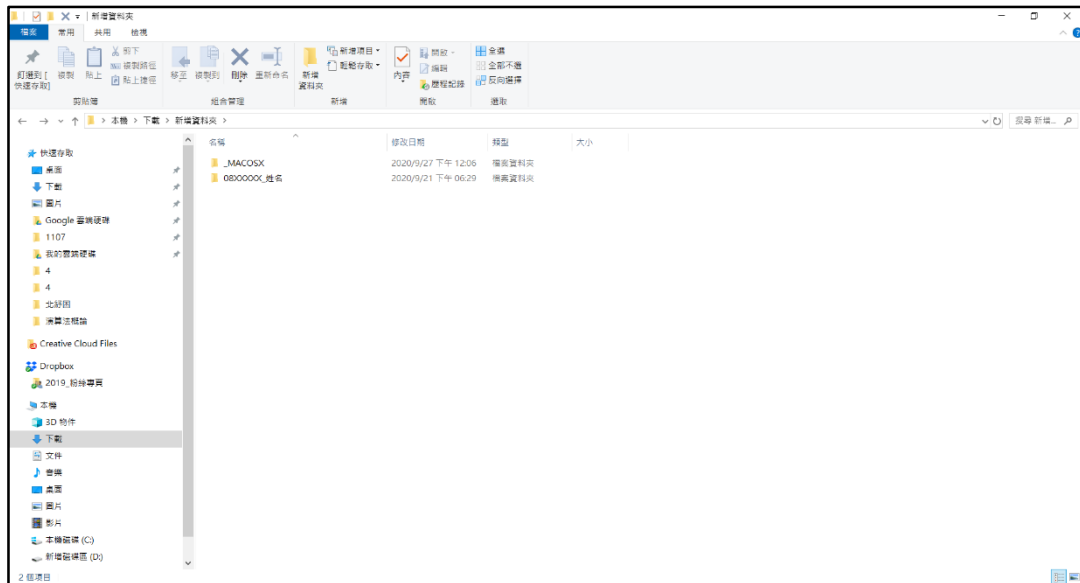
作業繳交

恭喜各位完成 **Assignment 6-I!** 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

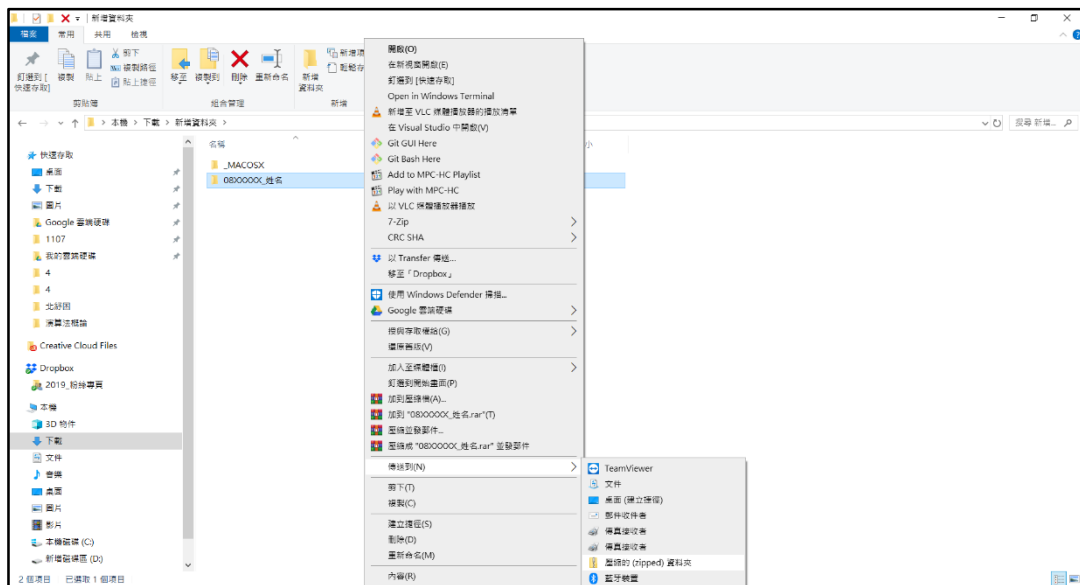
請同學於 **12/31(四) 23:59** 前依照下圖將作業上傳



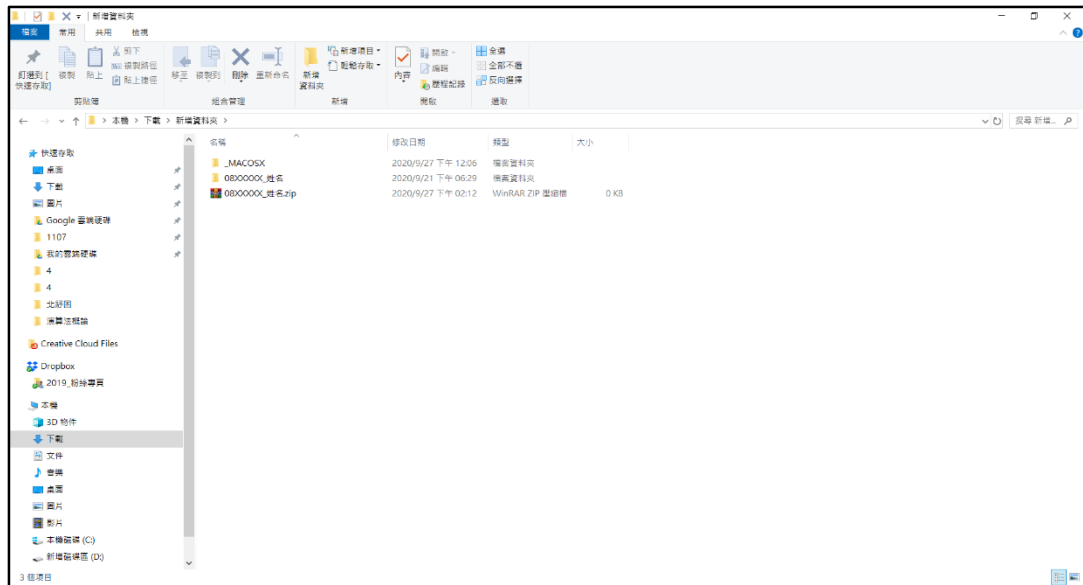
找到作業資料夾，按右鍵，選擇重新命名



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"



成品如上



點選「檢視」

國立交通大學 數位教學平台

課程資訊

- 課程綱要
- 成員
- 公告列表
- 我的郵件

內容

- 大綱列表
- 同步教室 (QC3)
- 教材列表
- 作業列表
- 討論區列表
- 試卷列表

評量

- 成績

工具

- 返回我的角色

【109上】1308計算機概論 Introduction to Computer Science

Assignment 1-Test

繳交狀態

作業繳交次數	這是第1次繳交
繳交狀態	沒有繳交作業
評分狀態	尚未評分
截止日期	2020年 09月 28日(Mon) 00:00
剩餘時間	6 日 5 小時
最後修改	-
作業加備註	評論 (0)

[繳交作業](#)

更改你所繳交的作業

按下「繳交作業」

國立交通大學 數位教學平台

課程資訊

- 課程綱要
- 成員
- 公告列表
- 我的郵件

內容

- 大綱列表
- 同步教室 (QC3)
- 教材列表
- 作業列表
- 討論區列表
- 試卷列表

評量

- 成績

工具

- 返回我的角色

【109上】1308計算機概論 Introduction to Computer Science

Assignment 1-Test

上傳檔案

檔案

若要新增檔案，請將檔案拖放到這裡。

[儲存更改](#) [取消](#)

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」

國立交通大學 數位教學平台

課程資訊

- 課程綱要
- 成員
- 公告列表
- 我的郵件

內容

- 大綱列表
- 同步教室 (QC3)
- 教材列表
- 作業列表
- 討論區列表
- 試卷列表

評量

- 成績


工具

- 返回我的角色

【109上】1308計算機概論 Introduction to Computer Science

Assignment 1-Test

繳交狀態

作業繳交次數	這是第1次繳交
繳交狀態	已繳交
評分狀態	尚未評分
截止日期	2020年 09月 28日 (Mon) 00:00
剩餘時間	6 日 5 小時
最後修改	2020年 09月 21日 (Mon) 18:11
上傳檔案	 08xxxxx_姓名.zip
作業加備註	 評論 (0)

[修改已繳交的作業](#)

[更改你所繳交的作業](#)

出現「已繳交」確定成功

stanCode

stanCode - 標準程式教育機構

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