編譯世界的起點 高雄中學 312 張哲瑋

**START FROM SCRATCH**

**PROGRAMMING THE WORLD**

1. 程式設計的學習歷程
2. 學習成果總覽
3. 個別程式簡介

　開發動機

　開發中遇到的難題

　Debug的過程

　最終成果

1. 從程式設計中的學習中檢討

自傳：

　　成長背景：

　　個人特質：

學習動機：

程式設計的學習歷程：

學習的起點 – 社團的啟蒙：

開始學習編寫程式，是從高一上學期的社團開始。

因為從小就對電腦的一切事務感興趣，但因為高中以前沒有接受過程式設計的相關教育，對於編寫程式的能力僅限於Scratch，更多是對於電腦硬體的探索；但是從高一開始，我加入程式設計社，開始對“程式設計”這個一直遠觀，卻不曾接觸過的領域發動探索。

自學的經歷 – 用城市解決生活的困難：

一開始接觸的程式語言是C++，從zerojudge解題開始學起；但因為我個人對於程式開發的熱忱並不在這些程式數學題上，所以高二就離開程式設計社，開始自學Python。

選擇Python的原因是因為Python有內建大量的module可供調用，對於編寫以實用性為導向的程式而言更為方便；也比較容易入手。

學習成果總覽：

從高二開始到高三結束，除了一些測試和學習用的程式，我共完成了11個專案，其中有三個屬於大型專案，分別是：

1. [高雄中學體溫自動填報系統](https://hackmd.io/@KSHS-PyClub/body-temp)
2. 線上課程自動點名系統
3. Open CV 人機互動介面

四個為跨學科領域開發的小工具，分別是：

1. 排列組合計算機
2. DNA密碼子轉譯
3. 一元二次計算機
4. Physics Simulator and calculator

三個為個別開發的小工具，分別為：

1. Markdown to HTML
2. YouTube Downloader
3. Py-ENIGMA

最後一個是屬於Python的初學者教程：

1. [Python環境配置](https://hackmd.io/@KSHS-PyClub/PDE)

在這段時間也有3個開發失敗的計畫，分別是：

1. 解謎類遊戲-Luker
2. Discord Server Robot-KS Tube
3. 基於雄中物理輔教的解題模擬程式-Physics Simulator

　　第一個專案會失敗的原因是由於人員的招集與調動不當，只有會編寫程式的組員，卻沒有負責電繪、音效與劇情的編輯人員，導致最終計畫失敗；第二個的DC機器人失敗則是因為我對於Discord的API不熟悉導致的，雖然在測試過程中，半手動操作的情況下是可以運行的，但是因為是使用“模擬使用者”的方式來操作而不是原生的API，呈現出的效果遠遠不如預期，加上時間的關係導致失敗；第三個專案最初的想法是希望可以輔助老師教學時使用，但是因為輔教真的太厚了，綜合開發時間與利益的權衡之下暫時擱置，但是因為技術與實踐方式並不是問題，所以大學時間有計畫繼續開發此專案。

個別專案簡介：

高雄中學體溫自動填報系統：

　開發動機：因為學校要求每天上午10點前都要上傳體溫，但是這種事情相信大家都是認真的填個兩三天就開始嫌煩了。對於這種枯燥乏味、每天重複勞動的工作，加上開始接觸程式語言，便萌生出寫程式來填體溫的想法。同時也可以藉由這隻程式的不斷開發及完善，幫助更多同學完成每日填體溫的需求。

　開發中遇到的難題：這個體溫填報程式算是我在學習Python的第一支程式，也是開發時間最久的程式，所以不管是從開發中的時間、其中遇到的問題或是不斷完善的功能，都是十個專案裡面最長最多的，所以我會單獨做一份報告來記錄。

　　開發中遇到的問題大概有幾個：Selenium及Chrome Driver設置、錯誤迴圈的邏輯以及發送email的方式。

　Debug的過程：Selenium及Chrome Driver設置，是在電腦自動更新完Chrome之後出現的問題，因為不同的Chrome版本需要配合相應的Chrome Driver版本使用，一開始我並不知道發生了甚麼事情，明明昨天就順利運行的程式，今天怎麼都執行。多次重開電腦依然無法運行之後，我選擇用VS Code來開啟程式，一來是可以確認是否是檔案損壞所引起的，二來是察看Terminal回傳什麼錯誤，發現是Chrome Driver版本不同導致的錯誤。

　 錯誤迴圈的邏輯，這算是才中後期解決的問題，一開始的解決辦法是針對錯誤的填報名單手動填報；但是從高二下開始，學校的伺服器變得都很奇怪，常常出現not respond的情況，所以出錯的名單也從一開始的一個禮拜一兩個，變成一天五六個，所以不做出改變也是不行了。改善的方法很簡單，一開始的程式是屬於順序執行的，所以只要在填報的迴圈裡面出錯，後面就無法進行操作；我新增了一個try跟except來判斷是否順利執行，以及跳過出錯的步驟，搭配email每日回報系統，實現了無間段的填報工作。

發送email的方式，這個並不是大問題，主要是我一開始選用的mail server並沒有對外提供SMTP服務，又不想用gmail來發送郵件才產生的難題。解決方法是換用mail.ru的服務；起初我是用vk.com來嘗試，但是vk mail並不提供SMTP，在尋找資源的過程中我發現vk mail的供應商mail.ru可以提供SMTP服務，並且不像google一樣需要為應用程式設定特別的安全性登錄選項，便選擇改用mail.ru的服務。

　最終成果：我最終在這個系統上面實現了「多人同時填報」、「每日email回報」、「填報錯誤提醒」、「亂數生成體溫數據」以及「錯誤排除與無間斷執行」的功能。更多詳細資訊，請參考[高雄中學體溫自動填報系統說明](https://hackmd.io/@KSHS-PyClub/body-temp)。

線上課程自動點名系統：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

Open CV 人機互動介面：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

排列組合計算機：

　開發動機：因為數學課上到排列組合這個單元，對於像是C n取k這種P n取k再除以k階的運算，我認為可以開發一個計算機來解題；一方面是可以加快寫題目的速度，另一方面是我覺得這種簡單的計算機是很好練習GUI的範例。

　開發中遇到的難題：我將這次製作的GUI從tkinter換到ttkbootstrap，一來比較美觀二來是語法幾乎和tkinter一模一樣，所以開發過程遇到的問題不是程序上的問題，而是需要抉擇哪個主題比較好看。

　最終成果：這個計算機可以算C、P、U及n階乘，並且有簡單明瞭的GUI介面。更多詳細資訊，請參見[排列組合計算機](https://hackmd.io/@KSHS-PyClub/Permutation)。

DNA密碼子轉譯：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

一元二次計算機：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

Physics Simulator and Calculator：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

Markdown to HTML：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

YouTube Downloader：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

Py-ENIGMA：

　開發動機：

　開發中遇到的難題：

　Debug的過程：

　最終成果：

Python環境配置：

　編寫動機：對於入門一種程式語言，網路上面有十分充足的資源可以學習與運用，從簡單的基本語法到相對困難的人工智慧演算法都可以學習到。只要願意花時間去尋找，便可以尋找到資源；但是對於環境配置這件事，我自己也是花了蠻多時間去搞懂的，雖然說過程都大差不差的，但是有一些細節也是著實令我苦惱了一陣子，像是安裝新的module或是設置環境變數；因此我決定就我所學到的內容，整理成一分我認為適合初學者的教程。

最終成果：請參見[Python環境配置](https://hackmd.io/@KSHS-PyClub/PDE)

從程式設計中的學習與檢討：

從一開始學習C++刷題到自學Python專案開發，我對於寫程式的態度算是有很大的改變；從一開始刷題只求快速與AC（Zerojudge），到後來的穩定與功能性至上，對於編寫程式的能力可以說是有了一定的進步，也能透過寫程式抒發腦中的創意而不僅僅是為了完成作業而寫程式。

因為我對C++的學習大概就到高一下就暫停了，接下來的時間都花是自學Python上面，對於學習程式語言的歷程與成長大概都是針對Python。

我對於學習的心得與檢討主要分為三個部分：我學到了哪些東西、我改進了哪麼缺點以及我還有什麼需要改進。

我學到了哪些東西：最直觀的來說，我學習到了Python的基礎與法以及數個模組的使用方式；而隨著學習的時間拉長，我學到的不僅僅是編寫程式的知識，還有編寫程式與除錯的技巧。我個人認為後兩者是需要時間去累積的資產，甚至比編譯的知識還要重要，因為程式語言是一個非常大範圍的專業，也是不斷在更新的，掌握的了基本原則卻沒辦法百分之百精通任何基礎或特殊的用法，而且編譯的知識像是語法或是使用層面，都是可以透過書籍或網路去掌握的，所以只要去找就一定找的到；反之是後面兩個技巧，像是如何縮短程式碼來優化執行速度或是開啟程式後閃退要怎麼解決，面對問題的種種可能性可謂是多到數不清，而大多時候也是google沒辦法清楚去解答的，所以編譯的經驗累積出來的這兩個技巧我認為是最重要的，也會不斷去精進。

我改進了哪些缺點：我寫的程式碼最讓人詬病的問題在於「空格與換行」都沒有一個固定的格式，所以出了問題要去求助別人之前都得要整個程式碼重新縮排過，這無疑是大大拖慢了編譯的速度。所以我改進了編譯的習慣，不再過度換行以及頻繁的按空白鍵。

我還有什麼需要改進：可能是因為我開發的資歷還不夠深，我目前沒辦法一氣呵成的寫完整個程式，都是寫一段就要停下來查很久的資料，大概是我對於使用到的模組或是基礎語法的不熟悉導致的。我正在嘗試的改進方法是多加練習，並且在編譯中的程式中，以不影響執行效率、效果與大小的前提之下，更多的去運用一些常見的語法，減少使用特殊用法。