

高等教育程式設計課程教學之多元化發展與變革

許育嘉

國立臺灣體育運動大學運動資訊與傳播學系副教授

一、前言

程式設計的能力，就如同過去所謂數學為科學之母般，是發展創新行動 App 開發、新型態的網路社群行銷、大數據分析、物聯網、人工智慧等於各領域應用之未來發展重要關鍵。在順應世界程式教育向下扎根的發展潮流之下，並深植發展人工智慧的基礎實力，教育部於 108 學年度十二年國民基本教育課程綱要，開始將「程式設計」納入科技領域成為必修課程。而在高等教育部分，則於 106 年度起開始推動的「教學創新試辦計畫」、「技專校院教學創新先導計畫」，以及後續「高等教育深耕計畫」的目標中，推動讓大學程式設計教育普及化。以經費挹注各校開設程式設計相關課程，使各大專校院修讀過程式設計相關課程的學生比例，由 104 年度計畫推動前的 15%，逐年提升至 106 年度 31.6% 及 107 年度 40.99%，而於 108 年度能達到超過半數的大學生（50%）曾修讀過程式設計課程之目標（馮靖惠，2019）。而這個目標於 109 年 6 月統計已達到了百分之六十（教育部電子報，2020），顯示出在高教政策大力推動之下，程式設計擴展到非資訊相關科系學生修讀之普及化已略見成效，但實質上由於各大學推動的作法不一，未必達到了均質化。

這個百分之六十之量化數字，為高教深耕計畫績效指標及衡量方式之一，學士班學生只要有修讀過邏輯思考與運算（程式設計）相關課程一門以上即屬有修過。但哪些課屬程式設計課程，各校並無一致的規範（謝青龍，2018）。有的認為課名必須有「程式」兩字；有的則認為部分課程單元與程式有關即可列計。而積極推動的學校，會善用高教深耕計畫補助之經費新增開程式課程、增聘教師、編製統一教材、研發程式教學工具，貫徹落實政策。而部分學校則保守地考量到學生程度、資訊教師員額、課程時數排擠等諸多內部因素，僅於既有之通識必修入門資訊課程，如計算機概論、資訊素養等，加入數週之程式設計單元，將這些原本必修課之學生數計入即能輕鬆達標。然而高教深耕計畫經費的挹注，是否真的有助於學生學習成效的提升，或僅是各校運用各種計算「技巧」來達到政策績效指標要求（葉裕民、謝青龍、陳靖瑋，2018），這也是百分之六十的數字背後，後續值得探討的。

「程式設計」對於學生而言，是一門挑戰性極高的課程。目前臺灣的大學入學新生，極大多數都是屬於程式設計的初學者，普遍都是在進入大學後，才開始接觸到程式語言。儘管有少部分學校在 12 年國教階段已開始導入運算思維、創客自造、積木式程式設計、機器人等基礎程式設計教學，然而學生在不同的國、高中所接受到的程式運算、邏輯訓練、問題解決等課程差異極大，呈現出顯著的

數位落差（高嘉峻、蔡銘修，2018），使得高等教育程式設計的教學充滿著挑戰。

在教育政策之主導與獎勵下，為了能讓高等教育中不論是否為資訊相關科系的學生都能學好程式設計，各大學教師無不絞盡腦汁，嘗試著各種創新的程式設計教學模式，相關經驗可供其他教師教學參考。

二、高等教育程式設計課程之發展現況

臺灣於高等教育大規模推動程式教育，始於 106 年之高教深耕計畫，其中一個重要指標即是大學生不分科系要有百分之五十的學生數修過程式設計的課程。而這也開始讓各個大學進一步思考程式設計教學上的挑戰，如何讓非資訊相關科系的學生提升學習動機，如何解決這些學生所遭遇到的學習困難，如何提升學習的成效。各校採取之因應做法，呈現出以下特徵：

（一）廣泛於通識領域中開設

過去僅於資訊相關科系開設的程式設計，開始被納入到通識領域中廣泛開設，甚至規定為不分系的大一必修課程，如東華大學、東海大學、逢甲大學等校。

（二）不同程度的涉入與能力要求

這些開在通識領域的程式設計課程，有不同的型態與深度。為了讓非主修資訊的學生，初次學習程式設計時不致遭遇太大的學習障礙，許多學校的課程設計是以先培養運算思維，進一步再與程式設計結合，採取由淺入深、循序漸進的策略。開設的課程有極簡單入門程度的運算思維與程式設計，有的則直接於現有計算機概論的課綱中加入一定時數的程式設計單元，讓學生認識程式設計。另有一些學校則是培養學生能寫程式，開設初階的基礎程式設計、APP 程式設計，甚至也提供更進階的程式課程供選修，以因應不同大專校院之屬性與學生特質。

（三）各校自由發揮從差異中建立特色

程式設計課程受重視的程度，也讓許多學校把握契機發展辦學特色，例如元智大學的「新雙語政策」，將基礎程式設計課程列為大學部的必修；東海大學發展 PBL¹議題導向的課程，並開發輔助學習程式設計的流程圖工具；輔仁大學推展程式設計融入現有專業課程當中，達到提高動機與跨域學習的目的；國立屏東

¹ PBL 指的是「問題導向學習」，英文全稱為 Problem-Based Learning。

大學透過制度設計與鼓勵機制，依循 ADDIE²為教學方案系統化的設計模式流程，由院校系三個層次全面推動程式設計教育；致理科技大學則是設計精彩的磨課師課程，以提升學生的學習成效（教育部即時新聞，2020）。

（四）結合教學實踐研究與教師多元升等發揮進一步推升力道

在高等教育全面的改革下，教育部在推動程式設計的同時期，也積極推動大學教師多元升等方案，鼓勵教師以教學實務做為研究之主題，並以之做為多元升等的管道之一。在 107 學年起推動「大專校院教學實踐研究計畫」補助之獎勵，使得許多教授程式相關課程的教師更積極投入了教學創新的研究。在異於傳統教科書講述或刻板的填鴨式程式解題教學外，我國的程式設計課程引入了一些參酌其他課程領域中之嶄新教學方法進行革新，產生教學上的變革。

三、程式設計課程教學遭遇之問題與挑戰

程式設計課程大都存在著通過率低、重修人數比例高、學習動機與學習成就低落等問題。這些問題與挑戰，不分是在國內或國外，也不分學生之科系屬性，所遭遇到的情況大致相同，在許多教學研究的文獻中經常被提出探討。

在探討程式設計教學的文獻中，有幾篇是採用了系統文獻回顧法（systematic review and meta-analysis），整理出了近幾年相關研究之結果進行統整分析。有學者以文獻回顧法篩選出 100 篇相關文獻進行回顧與探討（Medeiros, Ramalho, & Falcão, 2019），來解答三個研究問題：程式初學者需具備哪些關鍵學前技能與背景知識？初學程式的學生遇到那些學習困難？教師課堂上的挑戰為何？研究結果指出，要學好程式設計需具備的相關基礎技能，依序為：問題解決能力、數學能力、學過程式設計運算思維、摘要能力。而學生的學習困難可歸納成：問題的界定與形成、解決方案表達、解決方案的執行與評估、學習行為四大類。在這四類中最常見的困難各為：問題解答、程式語言語法、除錯技巧、動機與投入。而教師最常面臨的挑戰，依序為：教學方法與工具、維持學生的動機投入和堅持、師生的溝通與回饋。

也有學者針對程式設計教學所採用之教學介入的形式進行文獻系統化的回顧（Vihavainen, Airaksinen, & Watson, 2014）。從 32 篇相關研究中歸納出有下列十種常見的教學介入活動形式，依頻率最常見的依序為：教材變革、同儕支持、協同合作、情境化、配分比重、媒體運算、遊戲主題、協助支援、先修課程、團

² ADDIE 為代表「分析（analysis）」、「設計（design）」、「開發（development）」、「實施（implementation）」、「評鑑（evaluation）」五個階段。

隊合作學習。而其中有些是採取多種教學介入之混合方式，從教材建構、工具的使用、評分比重的調整、成對編程、團隊專案製作等，若以這些形式的教學介入，來對傳統課程的教學模式進行改革，可提高學生課程的通過率。

這些回顧的文獻整理出了各類的問題與挑戰，但對於國內的大學各個系所的教學現場，雖可能都在這些分類中，但每個遭遇到的情況都不一樣。需要各任課教師針對自己的課程個別深入探討，參考上述文獻的分類方式先釐清自己需要聚焦的問題是什麼，可能僅是某類問題特別嚴重，也可能是全面性的問題，之後再參考這些可行的解決做法，嘗試來克服教學上的挑戰。

四、應用於程式設計課程之新興教學模式與工具

有別於傳統的講述教學法，一些新興的教學模式在歷經教學的實踐與研究驗證後，也開始在國內大學的程式設計課程中被採用。這些教學模式、教學方法、教學策略通常被結合在一起應用，在一些教學研究中常見的有：(1)合作學習；(2)專題導向式學習（Project-Based Learning）；(3)問題導向式學習（Problem-Based Learning）；(4)翻轉教室（Flipped Classroom）；(5)遊戲式學習（Game-based learning）；(6)做中學（Learning by doing）；(7)其他教學模式：如 ADDIE 模型、ARCS³動機模式等。

在實施這些教學模式時，技術上常用的科技輔助工具，則有：

(一) 程式線上解題系統（online judge，OJ）

國外常見的有 repl.it、Microsoft Azure Notebook、UVa Online Judge、URI Online Judge、Code Marshal Online Judge、Spoz Online Judge 等平台，國內大學像是清華大學、交通大學等校也自行架設 OJ，提供學習者自我練習之環境，常用於程式解題加強及程式競賽之訓練。一些平台進一步加入了適應式學習、遠端協作等工具，強化合作學習之溝通與互動。

(二) 程式碼自動產生工具

可由流程圖自動產出相對應的程式碼，降低學習者所需具備之程式語法要求。如東海大學就自行開發了 CT2Code，可依所畫的流程圖自動生成程式碼，用於全校通識必修的「運算思維與程式設計」課程中。

³ ARCS 指的是「注意(Attention)」、「相關(Relevance)」、「信心(Confidence)」、「滿足(Satisfaction)」四個要素。

(三) 積木式語言及其開發環境

如 Scratch、APP Inventor、Alice 等，以圖形化介面進行積木區塊的拖曳來組成程式，適合初學者。如逢甲大學使用圖性化之 App Inventor 於大一通識必修的「App 程式設計」課程中。

(四) 主控板及微控制器

如 Arduino、Raspberry Pi、micro:bit、機器人套件等，常被應用於結合硬體之程式教學，強調做中學與創客教育（maker），在許多大學的程式進階課程中被採用。

五、教學變革之實施困境與建議

儘管課程之變革與創新的教學模式在一些教學場域中已被驗證可改進學習的成效，然而最普遍被採用的仍是傳統的講述教學法，主要是這些創新的教學模式若要實施仍有其限制，採用時需考慮的因素探討如下：

(一) 符合學生特質因材施教

某一些教學法雖然可以提升學習動機，但不見得可以提升學習成效。以「主動學習」和「翻轉教室」為例，若大部分的學生均無法做到課前預習時，很難達到預期的學習成效。

(二) 程式語言的選擇

普遍用於教學的程式語言有 JavaScript（JAVA）、Python、C/C++/C#、VB，然而這些程式語言在語法上的要求不一，學習的難易度也不同，未來可能的應用方式也不盡相同。或是要採用不須太重視語法的積木式程式語言來教學，其學習難易度與修畢課程後之未來發展性需一併考慮。

(三) 目標與門檻的設定

若課程被設定具體的達成目標時，實施創新教學模式時需考慮是否能有效輔導學生達到預期的目標。如通過程式設計能力之畢業門檻、取得程式設計證照、程式競賽等。

(四) 能力培養與學習成就感

究竟學生學習程式設計，能力須達到何種程度？從最基本的了解運算思維與程式的流程、能依指示完成程式實作、能利用程式套件製作網頁或系統、能用程式撰寫系統解決問題，能力要求越高可能使得多數學生的挫折感越高，然而若能力要求較低雖能使得學生的學習成就感提升，但是否與傳統印象中有修過程式設計課程的學生已具備一定「程式設計」能力，值得存疑。

而在實施這些創新教學模式時，與傳統的講述法相較時，有時需有額外的資源支持方能實施，如下：

(一) 人力與空間資源

以實施做中學的教學模式為例，若有具備足夠能力的助教協助學生排除困難、從旁引導，較易實施；或是進行磨課師教學時，需有人力協助錄影及後製。有一些合作學習、強調群體互動討論的教學，較容易在專屬且配置經特別設計的教室中實施。

(二) 「協助工具」的取得

需搭配資訊平台進行學習，如 MOOCs 或 Moodle 之學習平台。

(三) 教學法之進修與研習

大多數之大學教師未受過相關教學法之訓練，需有機會相參與相關研習。

六、從政策面解決實施困境的可行方式

目前高教政策是採取鼓勵機制以競爭型計畫引導各校自行規劃推動方案，並不強制規定以 KPI⁴來呈現（通識在線雜誌社，2019）有多少計畫補助經費的比例用在程式設計課程上。程式設計課程現階段雖已達普及化，但各校間的差異有待弭平，以確保學習成效達均質化。未來若規定各校須投入一定比例之專項經費用於推動程式設計，擇徹底落實之學校當典範，以實質用於相關課程之教師增聘數、新增開課數、獎勵創新教學數、新開發教材數、建置輔助學習工具數、教師研習成長數等，做為均質化指標管考之參考，以確保位在第一線面對學生的教師，有足夠觸發教學創新變革的原動力，在目前已達「量」的基礎上，繼續朝「質」

⁴ 全稱為 Key Performance Indicator，關鍵績效指標。

的方面提升。

七、結語

當程式設計課程普遍實施於大學各個科系時，如何在有限的教學時數和學生學習動機不足的限制下，引入新的教學方案、設計出符合學生特性的教材，來提升學生的學習成效，是對教師極大的挑戰；每一種教學法都有其適合與不適合的學生，在高等教育中具有充分教學自主的教師，面對高等教育程式設計課程教學之多元化，需要各任課教師針對所授課程個別深入探討，先釐清自己面對的問題是什麼，需要聚焦的問題是什麼，來克服教學上的挑戰。

本文認為，教師在現今充沛的各類開放網路教育資源中，教學目標上可從學生的學習動機以及學完程式設計後可以獲得什麼能力的角度來思考，是要成為專業的程式設計師？或是當遇到用程式能輕易解決問題時，具備使用程式解決問題的能力？或僅是達到理解程式運作的原理即可。教學方法上，若能運用新興教學模式，適切使用各種工具，輔導學生跳脫課程分數與學分的限制，自由選擇適合的課程或教材來主動學習，加強自我想要的程式設計能力，學生的學習動力和期望能力將會獲得提升。在學校方面，若學生能夠充分自由選課，且學校開課的數量達到一定經濟規模時，學生就能依教師的教學特色與風格，同時考量自己的期望與需求，自行選擇是要遊戲式學習、PBL、翻轉教室，或者選擇傳統講述教學法的學習方式，而不僅是消極地滿足畢業學分要求而已。

參考文獻

- 高嘉峻、蔡銘修（2018）。淺談十二年國教在國民中小學推行科技領域課綱資訊科技學習之障礙與策略。**臺灣教育評論月刊**，7(2)，80-84。
- 通識在線雜誌社（2019）。KPI的績效主義如何侵蝕臺灣的大學教育。**通識在線**，80，5-7。
- 教育部即時新聞（2020年6月28日）。**超過60%的大學生修過程式設計相關課程，教育部將於7月14日舉辦成果發表會**。取自 https://www.edu.tw/News_Content.aspx?n=9E7AC85F1954DDA8&s=03810B78E5BE8F8E
- 教育部電子報（2020年7月2日）。**六成大學生修過程式設計相關課程 教育部「推動大學程式設計教學計畫」見成效**。取自 https://epaper.edu.tw/print.aspx?print_type=topical&print_sn=1154&print_num=928

- 馮靖惠（2019年1月22日）。教育部推大專生修程式設計 108年目標為比例50%。聯合新聞網。取自 <https://udn.com/news/story/6928/3607389>

- 葉裕民、謝青龍、陳靖垵（2018）。教學資源挹注對於學生學習成效提升之成效評估－以南華大學通識課程為例。通識教育與跨域研究，18，71-93。

- 謝青龍（2017年8月2日）。謝青龍觀點：高教深耕計畫的程式設計之不合邏輯。風傳媒。取自 <https://www.storm.mg/article/308132>

- Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2019). A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education*, 62(2), 77-90.

- Vihavainen, A., Airaksinen, J., & Watson, C. (2014). A systematic review of approaches for teaching introductory programming and their influence on success. *Proceedings of the tenth annual conference on International computing education research*, 19-26.

