國立臺灣師範大學資訊教育研究所

碩士論文

指導教授：林育慈 博士

程式設計迷思概念診斷與矯正

Programming instruction with misconception diagnosis and correction

研究生：許書毓 撰

中華民國 105 年 7 月

程式設計迷思概念診斷與矯正

許書毓

摘要

本研究希望利用迷思概念診斷與矯正來輔助學生建立正確的程式設計概念、或減少迷思概念的產生。我們研究發展了一個迷思概念診斷與矯正的程式設計輔助學習平台，學生可於平台上撰寫並編譯程式，平台並提供迷思概念診斷機制，並可於診斷出可能的迷思概念時，給予學生迷思概念的矯正回饋，學生亦必須在撰寫程式的過程中針對重要概念進行反思以避免迷思概念的產生。研究利用準實驗研究法檢驗所發展教學模式的效益，實驗參與者為 67 位高中二年級的學生（實驗組 32 人，控制組 31 人），實驗結果發現:透過本研究提出的迷思概念診斷與矯正的教學平台與策略，包含對程式執行的順序、變數、判斷式、迴圈、函式的概念，學生的程式設計成就顯著提升(程式實作、學習單、期中期末測驗)，此即由於學生經過迷思概念矯正回饋的引導與圖文概念的強調釐清，可針對容易產生迷思概念的程式結構，進行深度的思考與練習，以了解程式結構的概念與應用方式，進而促進程式設計能力。

【關鍵字】程式設計迷思概念、教學回饋、教學平台

Programming instruction with misconception diagnosis and correction

Shu-Yu Hsu

Abstract

This study intends to study the design and development of programming instruction with misconception diagnosis and correction, and its effectiveness on students' programming learning. We have implemented a programming learning platform with misconception diagnosis and correction functions, by which students can write and compile programs, and get misconception correction feedbacks once their programs are diagnosed with possible misconception. During writing programs, students are also required to carry out reflection for critical programming concepts to avoid misconception. A quasi-experimental study was conducted to examine the effectiveness of the proposed instructional methodology. The participants are 67 senior high school students. The experiment results show that the proposed instructioanl strategies and programming learning platform based on misconception diagnosis and correction, students' programming achievement can be significantly enhanced. With the guidance of misconception correction feedbacks provided by the learning platform and reflective learing activities, students can easily carry out in-depth thinking in various programming conception and constructs during practicing program implementation; thus, contributing to programming conception development and programming ability.

Keywords：programming misconception, feedback, learning platform.

誌謝

大學畢業，剛考上資訊教育研究所時，是選擇了電腦科學教育組，由於對電腦相關知識與程式設計有一些熱誠，以及有修過教育學程，最後在暑假的時候決定了要加入的實驗室，「精進計算學習實驗室- TELiC Lab」。

在這兩年內，吸收了教育領域與資訊領域的知識，對於教育方面的議題、程式設計的觀點、相關業界動向、研究生活的感受，都有一定的認知與經歷，也奠定了未來人生的方向與目標。

當然我一定要感謝的就是我的指導教授林育慈。在這兩年的時間，教授不但要引導我思考研究的方向，修改我的文筆與給予相關建議，還時時刻刻回饋給我們正面的能量與想法，還有生活、工作中的進退應對與待人處世的道理，相信未來會有更深刻的體會。

還有感謝 TELiC 的所有學長學姊、同輩、學妹們，感謝聖哲學長的大力支持，不管是在開會、伺服器建置、研究問題討論、資料分享方面都常受到幫助。也感謝文揚、粲絜一起討論研究、一起通宵趕工的時光，名璨、聖融的默默幫忙。

還有教學實驗的凌倩老師，分享了很多授課經驗與適時的幫忙。以及 CSE Lab 的各位，時常的開會討論互相交流，連結兩實驗室的論點與研究。最後還要感謝資教所辦公室的各位，在這過程中大家都是不可缺少的環節，感恩各位。

許書毓　謹誌

中華民國一〇五年七月

目錄

第一章　緒論 1
第一節　研究背景與動機 1
第二節　研究目的 3
第二節　名詞釋義 4
第二章　文獻探討 7
第一節　迷思概念 7
第二節　程式設計迷思概念 10
第三節　教學回饋 15
第三章　研究方法 19
第一節　研究設計與架構 19
第二節　研究實驗參與者 20
第三節　研究程序 21
第四節　研究工具 23
第五節　JSLA 輔助教學工具 29
第六節　資料蒐集與分析方法 36
第四章　分析結果與討論 38
第一節　分析結果 38
第二節　討論 51
第五章　結論與建議 66
第一節　結論 66
第二節　建議 67
參考文獻 68
附錄一 程式設計迷思概念蒐集表格 72
附錄二 運算思維測驗題目 79
附錄三 每週課堂學習單 95
附錄四 期中期末測驗題目 106

表目錄

表 3-1 實驗參與者之實驗組與控制組人數 20
表 3-1 每週教學主題暨課程內容 22
表 3-3 態度量表 24
表 3-4 為半結構式訪談兩班之題目 25
表 3-5 錯誤的語法知識編碼表 27
表 3-6 迷思概念編碼表 27
表 4-1 教學實驗前程式設計測驗 38
表 4-2 實驗後所整理的迷思概念列表 39
表 4-3 迷思概念列表所對應的回饋資料 40
表 4-4 實驗組與控制組學習成就之描述性統計與 ANCOVA 分析 43
表 4-5 實驗組與控制組學習單之描述性統計與 ANCOVA 分析 43
表 4-6 實驗組與控制組線上程式練習之描述性統計與 ANCOVA 分析 44
表 4-7 實驗組與控制組專題之描述性統計與 ANCOVA 分析 44
表 4-8 實驗組與控制組期中程式設計概念測驗之描述性統計與 ANCOVA 分析 45
表 4-9 實驗組與控制組期末程式設計概念測驗之描述性統計資料與 ANCOVA 分析 45
表 4-10 為控制組與實驗組兩班的錯誤的語法知識依時間順序列表 47
表 4-11 為控制組的迷思概念依時間順序列表 48
表 4-12 為實驗組的迷思概念依時間順序列表 48
表 4-13 實驗組與控制組態度後測之描述性統計與同質性檢定 49
表 4-14 為兩班態度前後測成對樣本 T 檢定 49
表 4-15 為態度量表個面向的信度分析結果 50
表 4-16 為課程中 M1 迷思概念依照時間排序 54
表 4-17 為課程中 M2-1、M2-2、M2-3 迷思概念依照時間排序 58
表 4-18 為課程中 M3-1、M3-2、M4-1、M4-2 迷思概念依照時間排序 60
表 4-19 為課程中 M6 迷思概念依照時間排序 62
表 4-20 為系統回饋次數與學生高、中、低學習成就進行 ANOVA 分析 62

圖目錄

圖 3-1 研究流程圖.....22
圖 3-2 為教學系統登入帳戶後的畫面.....29
圖 3-3 為教學系統教學課程的畫面.....30
圖 3-4 為教學系統實作討論區的畫面.....31
圖 3-5 為教學系統線上練習的畫面.....32
圖 3-6 為教學系統線上練習區的區域功能.....32
圖 3-7 為線上練習區編譯區風格設定功能.....33
圖 3-8 更直觀的顯示方式或圖形化輸出.....34
圖 3-9 文字敘述的回饋.....35
圖 3-10 圖形與動畫的回饋.....35
圖 4-1 IF、ELSE IF、ELSE 回饋內容.....41
圖 4-2 FOR 執行順序的動畫截圖.....42
圖 4-3 巢狀 FOR 執行順序的動畫截圖.....42
圖 4-4 兩班期中測驗錯誤的語法知識與迷思概念的次數.....46
圖 4-5 兩班期末測驗錯誤的語法知識與迷思概念的次數.....46
圖 4-6 為 JAVASCRIPT 程式碼與輸出解果.....53
圖 4-7 為 JAVASCRIPT 資料型態例子與說明.....53
圖 4-8 為「=」誤用之範例.....53
圖 4-9 為 M2-1 迷思概念範例.....55
圖 4-10 IF、ELSE IF、ELSE 執行順序之回饋內容.....55
圖 4-11 M2-2 迷思概念狀況.....56
圖 4-12 M2-3 迷思概念狀況.....57
圖 4-13 M2-3 迷思概念圖文回饋.....58
圖 4-14 M3-1 迷思概念例題.....59
圖 4-15 M3-2 迷思概念例題.....60
圖 4-16 M3-2 迷思概念回饋資料.....60
圖 4-17 M6 迷思概念範例 S.....61

第一章　緒論

本章共分為三節，第一節說明本研究之背景以及動機；第二節為提出具體的研究目的，以及想要探討研究的內容；第三節就研究中所提及的重要名詞作簡要釋義。

第一節　研究背景與動機

自從 1970 年來，迷思概念之議題就被大家所重視，日後學者也陸續研究各方面的迷思概念議題(Hancock, 1940)。在程式設計的教學方面也有許多研究指出其對正確概念建立的重要性，學生在建立程式設計之概念時往往了解單獨的語法、語意，但綜合這些獨立的程式碼卻有困難與錯誤的概念。例如:學生在單獨迴圈任務是有機會解決的，但結合判斷式與其他變數就無法清楚了解整體結構與功能(Lahtinen, Ala-Mutka, & Järvinen, 2005; Sekiya & Yamaguchi, 2013)。而在慢慢建構程式設計的能力時會因為這些錯誤的概念造成往後學習的困難，如教學者可以清楚掌握學生學習程式設計時的經歷與所遇到的迷思概念，那就可以提供更適當的教材與教學模式並提升教學效率與成果(Danielsiek, Paul, & Vahrenhold, 2012; 陳建偉, 2008)。

程式設計課程的教學目標，不僅是要讓學習者瞭解程式的基本結構與概念，更希望學習者能夠自行運用程式設計解決問題，所以在初學程式的相關概念與邏輯就顯得相當重要。許多研究指出程式設計相關的迷思概念，記憶體的使用、物件導向與程序型程式、資料分派、判斷式、程式執行、迴圈問題、運算邏輯、函式結構等議題(C.-L. Chen, Cheng, & Lin, 2012; Eckerdal & Thuné, 2005; Lahtinen et al., 2005; Ragonis & Ben-Ari, 2005; Sanders & Thomas, 2007)，在實驗中顯示了不同種的狀況對應的迷思概念情形。為了讓學生達到更精熟的程式概念，教學者了解學生在學習程式設計時面臨錯誤概念的經歷是有正面影響的。而且大多的迷思概念都是非常基礎的，在第一次學習程式設計的課程中所提到的概念往往是後續進階課程裡必備且重要的知識，所以需要更精熟的了解。

在矯正迷思念方面，首先要找出學生迷思概念再針對其概念做矯正，相關研究中也指出放聲思考與開放式問題可助於學生了解自己的思考歷程，並讓老師更清楚的知道學生有困難的概念(Danielsiek et al., 2012)。而在學習程式設計之前有上過數理相關的課程或邏輯方面的知識是有助於程式結構、物件導向之概念的建立(0zdener, 2008)。也有研究利用 Visual program simulation (VPS)互動式的程式設計教學系統，來探究學生的迷思概念與學習歷程並藉此結果來評估上課時教學者要把重點放在哪些迷思概念上，其系統還提供了程式的結構模擬且可以自行操弄，在旁邊還會有額外的相關資料引導學生自行解決問題(Sirkiä & Sorva, 2012)。研究程式設計教學模式與教學工具方面，也發現問題解決能力會因為這兩個面向而有顯著差異，教學模式方面學生在程序導向教學與問題導向教學中更傾向於後者，結果顯示此教學模式是一種可以有效培養初學者對程式設計之學習理解與問題解決能力；在程式設計教學工具方面，初學者相較物件式程式設計工具更青睞於程序式工具，不過在學習者更為熟練後前者可以更有效的幫助學習理解複雜的概念與問題解決情境(M.-P. Chen, 2007)。

為了改善教學的方式與效率，本研究會利用回饋式的程式設計教學平台來教學，學生在送出程式碼時會判斷出他們程式碼有迷思概念和語法錯誤的地方，並針對此錯誤的概念回饋出相關的教材與建議。這樣可以達到針對各個學生所犯的迷思概念做對映的教學內容，不但可以讓學生自行了解程式碼錯誤的問題與自我省思歷程，也可以增加課程的教學進度。

第二節　研究目的

一、　探究程式設計初學者常有的迷思概念。

二、　設計程式設計輔助教學平台與教學策略以矯正常見的程式設計迷思概念。

三、　評估所設計之程式設計輔助教學平台與教學策略之效能。

1. 對學習者程式設計學習成就之影響。
2. 對矯正學習者程式設計迷思概念之影響。
3. 對學習者程式設計學習態度之影響。

第三節　名詞釋義

本研究希望利用迷思概念診斷與矯正來輔助學生建立正確的程式設計概念、或減少迷思概念的產生。其過程中會探討診斷學生學習時可能有的迷思概念，以及如何以適當的回饋形式進行矯正與避免，最後再進行深入的量化與質性分析。本節會先針對本論文中所提及的迷思概念與回饋加以定義與討論。

一、　程式設計迷思概念

迷思概念(misconception)的意思是指在學習過程中，學習者對某種概念有特定的想法與認知。而這些想法與認知通常都跟專家有所差異，進而產生相關問題的錯誤。

程式設計迷思概念是指學習程式設計的歷程中，所發現學習者的誤解或錯誤應用。程式設計概念包含邏輯判斷、程式結構的理解、程式內部運行方式、記憶體分配、物件導向等。本研究中程式設計迷思概念定義為對該程式設計課程中所提的程式設計概念有不同的想法，不同於課堂中所學的或該方面專家的通用觀念，且會影響後續的相關課程概念與成果。有該方面迷思概念的學習者會認為某個概念因為某些事情而造成，或是覺得某些狀況是要用某方法實施。本研究的程式設計迷思概念僅討論學生學習 JavaScript 語言時，對判斷式、迴圈、程式執行順序…相關的概念。迷思概念的種類將彙整文獻探討與迷思概念調查測驗而得(詳見第四章第一節)，表 4-6。

根據所得之迷思概念種類，本研究將於以下幾種學生的程式設計練習或測驗中診斷其迷思概念:

1. 學習單:該程式設計課程中會有學習單的配合教學，學生在填寫學習單時會因某些迷思概念造成一些錯誤，而這些迷思概念會記錄次數、內容供後續的探討。
2. 線上程式練習:課堂中會配合線上輔助平台來進行程式編寫與執行，而在此發現的迷思概念也會記錄下來。

期中與期末測驗:整個課程中會有兩次測驗，期中考與期末測驗。兩次測驗學生的答題狀況與迷思概念的次數也會被記錄下來。

二、 迷思概念診斷

本研究迷思概念的診斷將會有人為與系統兩種，會根據學生迷思概念發生的狀況，利用迷思概念編碼表進行分類。人為診斷部分包含了學習單、期中與期末測驗、以及部分線上程式練習，系統診斷部分是針對線上程式練習之題目輸出結果，進行系統後端的程式碼錯誤診斷。

三、 迷思概念矯正

在本研究中會診斷學生迷思概念類型，並利用適當回饋矯正其迷思概念。迷思概念矯正是指針對學生某些迷思概念的狀況，給予正確的思維或釐清學生對其概念的誤解。本研究透過老師或教學系統所給予不同模式的回饋，來矯正學生的程式設計迷思概念，進而提高學習者的學習成就。

四、 教學回饋

本研究中會針對迷思概念的狀況給與適當的回饋，以達到矯正或避免的效果。回饋在教學過程中扮演著很重要的角色，且回饋通常會伴隨著反思，不同的教學過程與內容會有不同的回饋促使學生的反思，並矯正該程式設計迷思概念。此課程的回饋包含了文字、圖片、動畫等相關概念的回饋，而研究中設計回饋的部分如下:

1. 學習單:程式設計課程中的學習單，會在每次老師批改完後，根據學生迷思概念的狀況，給予概念的釐清或補充相關的教材，不但可了解錯誤的原因，也可以了解其他相關的知識、促進自我反思與概念內化。
2. 線上程式練習:課堂中線上輔助平台會有兩個階段的回饋，首先是系統針對該題目批改後的狀況給予的回饋，且系統會記錄下每個學生回饋的次數與內容，第二階段是由老師檢閱再給予回饋。
3. 期中與期末測驗:課程中的兩次測驗，也會依照學習單的回饋形式進行，即測驗完後會按照學生錯誤的情形給予回饋。

五、 程式設計學習成就

本研究中的程式設計學習成就，會根據參與者在課程中所填寫的課堂學習單、線上程式練習、專題作業、期中與期末測驗的成績為基礎，再依照比例加權並總和。而在實驗前會測驗運算思維能力，以此知道控制組與實驗組是否有先備知識上的差異。本研究會依據準實驗研究法進行，其 ANCOVA 的共變數為運算思維前測分數，固定因素為班級，獨立變數為程式設計學習成就。

六、 程式設計學習態度

本研究中的學習態度分為量化與質性的分析，量化的程式設計學習態度，會依據所設計的態度量表(五點量表)所測得的分數進行分析。該態度量表中會有實用性、程式設計之自我體會、動機意願、自信、問題解決能力，五個面向。其中會有反向的題目，而將全部的面向進行加總，可得程式設計學習態度的分數。

而質性的程式設計學習態度，會於課程後抽樣幾位學習成就高、中、低的參與者，進行結構性訪談。訪談內容會包含對整個程式設計課程與教學系統的個人感受、與學習心得、建議。

第二章　文獻探討

本章將針對研究主題相關之理論及概念，進行文獻整理與探討。共分為三節，第一節為迷思概念，說明迷思概念的定義與特質，以及如何矯正迷思概念；第二節為程式設計迷思概念，說明程式設計的困難處，和常見的程式設計方面迷思概念，以及如何矯正程式設計的迷思概念；第三節為教學回饋，說明教學回饋的種類與教學回饋的效益，再探討迷思概念方面的回饋。

第一節　迷思概念

一、 迷思概念的定義與特質

迷思概念(misconception)的意思是指在學習過程中，學習者對某種概念有特定的想法與認知。而這些特定的錯誤概念會造成在學習過程中負面的效果與影響後續的概念的建立。通常每一項科目的各主題都有一些特定的迷思概念，而這些就是在各主題下大部分的學生經常會有錯誤觀念的地方，在此狀況下學生們會產生差不多的錯誤或。

迷思概念(misconception)之詞最早出現在「An evaluation of certain popular science misconceptions」(Hancock, 1940)。迷思概念在一開始的研究中都是以科學現象為主，包含小尺度之日常生活的物理化學現象、到大尺度之宇宙地球的自然現象。這些錯誤的想法不但阻礙了學習，也對於人們的行為模式有一定的影響力。而現今被廣泛的解釋成在這領域概念中是有別於多數專家們的思維，進而推倒出錯誤的結果或是在學習過程中因為某些觀點不同所造成阻礙(Treagust, 1988)。

「概念」在數學相關研究中經常在探討程序型與知識型得知識，程序性知識可定義為怎麼辦到某件事情的；概念性知識可解釋成是何種原因此任務會適合該操作程序。雖然程序型與概念型知識常常是分開討論的，但是在數學這方面其實無法完全的分開(Morales, 2014; Rittle-Johnson & Schneider, 2015)。像在代數的課題中，往往是需要約一年的時間來教導代數相關的概念，但是在往後的數學課程裡，這將會變成程序上的計算過程，所以其實是混和再一起學習的。而在數學的代數課程中也被提到有迷思概念的現象，以下(1)學生知道如何使用「=」，但是不知其真正的意義，(2)學生知道「-」

的使用方式，但不知道何時可以在等號兩邊互相乘負號，(3)學生知道如何使用變數，卻會因為方程式相關的專有術語與變數而影響計算流程(Booth, 2011; Revival, 2014)。

在國小五年級力與運動的概念中，也存在著一些令他們困惑的迷思概念，例如動摩擦力、靜摩擦力、最大靜摩擦力相互關係的概念、物體加速度的影響、地心引力(超距力)對物體的影響、為什麼有重量(F=m*a)與質量的概念。在國小的階段大部分的學生會以之前對此現象或概念的認知做解釋，但因為有些現象是相較抽像的，所以學生難以接受並延伸應用，在一些有預設理想狀況的題目中，學生往往還是把摩擦力列入計算，這也是因為他們在直覺感官上認為一定會有摩擦力，無法理解「理想狀況」的意義(彭泰源 & 張惠博, 1999)。

像是在國、高中生的理化課程中，經常出現的迷思概念有光學原理、成像過程、浮力、原子結構、化學式平衡、平衡常數等等。例如:學生在液體現象之迷思概念約分為表面張力、內聚力與附著力、毛細作用等等。有些是因為專有名詞上所造成的概念混淆、科學方面概念連結有所缺失(例如:熱漲冷縮)、基本定義、錯誤的直覺.....等，所造成學生有錯誤的想法(陳建偉, 2008)。而一些研究中也指出在國中階段所建立的概念與思考、判斷模式會隨著年紀有所改變，在學習過程中概念與相關知識上會有所增長與矯正，漸漸的思考判斷模式會趨向於「科學知識或概念推論而形成的假設」，取代「直接觀察而形成的假設」(廖焜熙, 2001)。

在其他科學研究方面也指出，海洋科學概念上有部分的迷思概念會因為科學專有詞彙上所造成的，會讓學生因為字面上的意義認為是其它意思或自行理解成其它概念。也有些迷思概念是明顯受到媒體的影響導致有錯誤的相關概念，因為某些專有的詞彙每天都會在各種傳播媒體上不斷的被灌輸錯誤的用法與概念(Lwo, Chang, Tung, & Yang, 2013)。

然而科學的迷思概念通常有以下的特性：(1)他們有別於同領域專家的想法；(2)單一或少數的迷思概念具有普遍性（即很多不同個體都擁有之）；(3)許多迷思概念是難以改變的；(4)迷思概念有時涉及另有信仰系統(alternative belief system)，這些系統的想法會導致錯誤的概念連結；(5)學生所具有的迷思概念可能來自於自己學過的先輩知識。

二、 迷思概念的矯正

在學習任何一個學科時，都會有些迷思概念產生，而這些迷思概念往往會影響到學習的過程。學生如有一些先備知識是跟正要學的概念吻合，或是先備知識支持一些因果關係的推論，那他們在這方面的學習會是更容易、有效率的；但如果他們的這些先備知識是與現在要學習的事物有所衝突或矛盾，那將會阻礙他們建立這些新的觀念。而有些迷思概念是會因團體互相影響而產生的，所以要先釐清學生的迷思概念與學習時有困難的地方，再針對這方面做教學上的調整(Akbaş & Gençtürk, 2011)。

在科學方面的迷思概念研究中，如果利用應用***概念圖命題***的模式以及開放式的問答題可以更有效的了解學生對此概念的理解程度，以便於後續的分析與矯正學生的認知(Lwo et al., 2013)。利用概念式的方法配合學生自行體會新的觀點，且了解整個問題的前因後果一步一步的思考該歷程，並與其他背景知識與相關方面問題融會貫通延伸應用。

當確定學生有哪些迷思概念，可以利用***直述的方式***來告知並釐清正確的觀念與迷思的原因，這是最簡單也是最直接的方式，也可以使用***視覺化的圖片或影音媒體***來輔助釐清過程，學生可以更具體的了解該現象或問題的成因(Brinko, 2008)。

在數學之代數課程概念裡，如何釐清錯誤的運算思維或建立一個穩固的概念。可以在教數學時利用範例式的教學來引導學生的想法。在範例的演練時可以讓學生了解在類似的狀況下是如何解決的，並最後自己實際操作與思考要使用哪些程序才可以有更佳的效率(Booth, 2011)。

在理化方面的迷思概念，教學上可以改進為，(1)將***相似的概念一同教學***並強調其差異性與定義。 (2)在教導某概念時可以***把相關的知識連結起來***，讓學生有正確的概念連結。(3)教學時可以***利用文字與圖表***來加強所需說明的科學概念(陳建偉, 2008)。多利用圖像解釋給學生了解以及動手做相關的實驗活動，讓學生自主測試該現象會有更深刻的體會。老師也可以利用分組的討論互動互相發現彼此迷思概念並一齊釐清與矯正(彭泰源 & 張惠博, 1999)。

第二節　程式設計之迷思概念

一、 程式設計學習困難

學生在學習基礎的程式設計時一直都有困難之處。在一些情況下是知道問題的方向且了解要如何解決，但是在不熟悉此程式語言的規則、語法與相關指令下，難以在裡想的處理想法下順利解決問題。一些則是在教學上已學會了一些基礎的觀念與相關程式語法、語意，但可能因為所了解的知識是片斷、不清楚的或無連續、相關性，所以產生了一些錯誤的概念導致在程式設計上會有所困難。而大部分對於具體的語意轉化成抽象的語法上有很大的困難，像是在「物件」的概念中有些人是因為根本不了解所以沒有產生迷思概念，這些都是教學上值得關注的議題(Kaczmarczyk, Petrick, East, & Herman, 2010)。

學生會對程式設計感到困難不單單是因為抽象的概念所造成，也因為程式結構的問題所疑惑。例如: 動態程式部分學生通常不是因為單一概念所造成困難，而是伴隨著多種問題所造成且需要將困難劃分為多個子問題在逐步克服(Bonar & Soloway, 1985; Danielsiek et al., 2012; Du Boulay, 1986)。所以教學模式與教材設計就相當重要，要有效的提升學生程式設計概念與程式技術能力並具備自我練習、思考的機會。 最難學習的概念往往是整合全部程式碼的題目，而非單一的概念。需要了解整體的運作與細節關係才能完全的了解。 還有一部分是學生了解該程式設計概念，但是當要做相關應用時就會感到困難。這要配合相關教材的多次自我練習才能建立穩固的概念與應用能力(Lahtinen et al., 2005)。

新手程式設計者有大量「脆弱知識」的現象包括「不完全的」、「難以存取的」、「時常誤用」的狀況。「脆弱知識」(fragile knowledge)分為四種:部分知識(Partial knowledge)是一個很直接的狀況，就是學生可能不記得或是根本沒有學過那樣的知識以至於陷入一個僵局之中。惰性知識(Inert knowledge)此情況就是學生沒辦法擷取他們的指令知識，在腦中是有的映象的，只是拿不出來。錯位知識(Misplaced knowledge)是指學生把某些指令結構寫在不該出現的程式碼中，所以會導致程式錯誤。混合知識(Conglomerated knowledge)是指學生寫出的程式碼中包含了多種語意和語法的結構，為了達到題目的要求但卻因不恰當的混雜程式碼而導致錯誤(D. Perkins

& Martin, 1986; Spohrer & Soloway, 1986)。

二、 程式設計常見迷思概念

程式設計迷思概念是指程式設計該科目，學習者所發生的概念誤解或錯誤應用，進而導致程式的錯誤狀況。在程式設計的課程中也是有很多的迷思概念，物件導向、變數狀態、迴圈、判斷式等觀念都是學習者常常有困難的地方(Kaczmarczyk et al., 2010; D. N. Perkins & Simmons, 1988; Sekiya & Yamaguchi, 2013)。

許多的研究顯示在程式設計的學習中要提高學習成效都要正確的觀念與做法。而在這過程中學生如有一些迷思概念存在著，往往會降低生產力或寫出很多有問題的程式。尋找學生在學習程式設計時的迷思概念，大多都是無法直接的發現，這些問題要在親自去實作或程式執行時才會被發覺。例如:在時間複雜度的演算法裡程式碼越短、變數越少、指令越少就會有較高的執行效率之迷思概念，事實上只要兩程式碼邏輯程序上是一樣的，所執行的時間就會是相同的(Ozdener, 2008; Lahtinen et al., 2005; Ragonis & Ben-Ari, 2005; Sirkiä & Sorva, 2012)。

有研究指出在基礎的 BASIC Programming 課程中也發現很多迷思概念的現象:

(1)input

學生會對於資料怎導入記憶體的過程不了解。例如:電腦如何等待使用者輸入資料，取得資料後如何儲存於某記憶體位置再執行接下來的程式。

(2)read

學生對於讀取的過程不清楚，不確定要讀取哪個記憶體且取哪個值。

(3)判斷式

學生對於判斷式的執行程序上有所迷思，不知道當判斷式的條件不成立後程式該如何執行。

(4)Let

學生不能區分變數的定義與變數之間的運算之「=」的差異。

(5)print

學生不了解 printC 與 print「C」之間的差異、與這背後程式的意義。

在此研究中也強調，很多電腦內部指令的意義是要更清楚的解釋與釐清一些概念，這樣較可以有效的讓學生理解(Bayman & Mayer, 1983; Putnam, Sleeman, Baxter, & Kuspa, 1986)。

在Java程式語言的實驗中，以紙筆測驗與程式實作測驗歸納出物件導向之迷思概念。分類為:

(1)不清楚某些敘述的電腦內部運作情形，例如迴圈條件式和方法呼叫之寫法及控制流程。

(2)不了解某些物件導向概念，例如繼承（inheritance）、多型（polymorphism）、及方法重載（overloading）。

(3)雖能說出某些概念的定義，但並不了解該定義之真正意涵，以致無法將其應用於實際解題。

(4)無法自行查閱並應用Java標準類別庫的功能。

(5)未能釐清各個類別在程式專案整體架構中所扮演的角色。

(6)無法理解編譯器的錯誤訊息並進行除錯(C.-L. Chen et al., 2012; Holland et al., 1997)。

在其他的研究中也歸納了學習程式設計時的迷思概念與困難之處(Goldman et al., 2008; Kaczmarczyk et al., 2010)。

1. 記憶體、指標(Memory Model, References, and Pointers)
2. 資料型態(Primitive and Reference Type Variables)
3. 控制流程(Control Flow)
4. 迭代與迴圈(Iteration and Loops)
5. 條件判斷(Conditionals)
6. 指派(Assignment Statements)
7. 陣列(Arrays)
8. 運算子優先順序(Operator Precedence)

三、 程式設計迷思概念診斷

要如何檢測評量出學生的那些錯誤是因為哪幾種迷思概念所造成的呢?在許多研究中是以問卷的形式來了解學生的迷思概念，有些是開放式解釋程式碼的問卷、是非題問卷、選擇題問卷。在開放式的問卷中是最能了解學生對此概念的理解程度，而在是非題與選擇題方面，是非題較不能準確的知道學生的學習狀況，當然在題目的闡述中也要能清楚的表達問題。這樣才能更精準的診斷出學生的迷思概念，以及降低題意不清所造成的結果(Britos, Rey, Rodríguez, & García-martínez, 2008; Taylor & Kowalski, 2012)。

在 BASIC 程式語言的迷思概念診斷中，是特別針對程序思維設計了相關考題，將程式碼分段的讓學生編寫，且利用引導的方式讓學生了解下每一步驟的功能與用意，再自行轉換成程式碼，最後再依照學生作答的狀況，了解是哪一步驟出了問題，並診斷是何種迷思概念(Bayman & Mayer, 1983)。且有許多研究是會先讓學生進行相關的測驗，再配合後續系統化的訪談結果，進行診斷分析學生可能有的迷思概念(Goldman et al., 2008; Treagust, 1988)。

有自動診斷方面的研究，透過線上作答系統進行測驗，系統會顯示題目的程式碼，再依照此程式碼給予程式設計的問題。作答者填寫答案與附註說明原因，系統會自動批改所填寫的答案是否正確，再由人工的方式去診斷觀察所填寫的原因，以推論是甚麼迷思概念導致這樣的答案(Chick & Baker, 2005; Sekiya & Yamaguchi, 2013)。在一些程式碼除錯的研究中，會利用後端的方式進行程式碼每行的斷字分析與語法核對檢查，最後再分析那些關鍵字出現的次數與程式碼錯誤的機率與原因，來判斷該程式系統的運作效率或穩定性(Baker, 1992; Moons & De Backer, 2013; Rutar, Almazan, & Foster, 2004)。

四、 程式設計迷思概念矯正與避免

學生在學習程式設計時有所疑惑或是過程中有發生困難，可以利用階段性的引導來促使他們獨力解決。如在初學者學習程式設計的實驗中有發現，如學生一開始有程式起步編寫上有所困難會先利用高階的提示來進行引導初步的解決方法，但也提到這方法大多都是較沒有效果的。所以在下一個階段如還是有所困難或完全沒進展，就會利用暗示的方法來提供較直接的問題解決策略，在此階段的教學下相較會更有成效。如最後還是無法順利的解決此問題，那就會將其問題的正確答案與做法告訴學生，在多種狀況下往往會有些許的學生需要此方式來教學(D. Perkins & Martin, 1986)。

當確定學生會有可能發生某錯誤概念，那就可以避免迷思概念的產生:(1)在相關範例的說明可以更加明確且詳細與更多元的狀況實例。(2)用適當的例子來針對該迷思概念當作解釋，可以有效避免其錯誤想法。(3)大部分的問題可以經由良好且適當的練習來避免與矯正(Eckerdal & Thuné, 2005)。

在物件導向概念方面的建議:(1)在重要的核心概念部分需要用完整的範例來解釋說明，沒有特別的捷徑。例如:建構子在物件導向裡是非常重要的，這一定要花時間做好扎實的基礎。(2)在學習類別與物件易混淆的概念時，一再強調兩者之間的關係與差異。(3)在該主題適當的時機教導較困難的核心概念，有利於後續的課程理解與內化(Ragonis & Ben-Ari, 2005)。

在教學方面(1) 教程式設計時，內容盡量可以去減低脆弱知識的問題。例如:教迴圈概念時要解釋為可以重覆任何東西、任何數字、任何次數，且輸入值後並可以自動計算。(2)教程式語言時要維持該語言的探索性。例如:某些既定的理論如有所疑惑，就會開始一步一步的思考每個過程，而這個歷程可以激勵學生建立更完善的概念。(3)鼓勵學生使用基本的問題解決策略。例如:直接性的提示可以讓學生有效的了解現在所面臨的問題，並可引導學生使用基礎、簡易的策略解決任務目標(Eckerdal & Thuné, 2005)。

第三節　教學回饋

一、 教學回饋的種類

教學回饋是根據學習者某方面的成效或理解成果，給予的概念型資訊。可以是老師、系統或是學生間的回饋。回饋在教學成就上是非常有影響力的，根據不同類型、方式的回饋會對應到正或反向的最終成果。

回饋可分為四個層級的運作 任務面-引響任務的成功與否、或對錯。為直接或正確的資料回饋。 執行面-針對程序方向的回饋，大多為影響整個任務的解決方向與整體的概念。 自律面-引響學習者後續的應用。學習者因為了解了某概念後，可以自行利用該知識進行連結性的應用。 自我面-針對個人的正向肯定式回饋。例如:" 你表現得非常棒"，此種回饋可以建立學生正向的自我肯定，藉此正向影響學習動機與成效(Hattie & Timperley, 2007)。

在操作技能或行為學習方面也有相關回饋之研究，口頭贊同式的回饋教學可以應用在學習樂器的領域。大致可分為以下幾種，口頭表達學術資訊(approval academic information):口頭的表達某個地方是對的，表現不錯。口頭肯定進步(student improvement):口頭表示比之前進步的更多了。無依據式肯定(norm-referenced feedback):口頭表示學生表現的比起以往的學生都還要好。正向讚賞(person praise):口頭表示學生有該方面的天分。個人化認同(personal approval):針對某科目核心重點或觀念給予認同。行為技巧肯定(approval-control):口頭認同學生的操作模式與技巧。而不同的教學模式配合學生的個性特質與學習習慣，會交互影響有不同的結果。例如:個性內向且理性判斷的學生，配合學術資訊式的回饋有明顯的效果(Running, Ligon, & Miskioglu, 2016)。

回饋與反思在臨床醫學是個很重要的教學方式，整個醫學教育核心就是回饋，面臨不同的教學過程會有不同的回饋促使自我的反思。 反思可以讓自己完整的思考並重整概念、技能、知識結構，進而提升自我成效。 如果經常有適當的回饋，也會帶動良好的反思，最終提高學習效果，反之亦然。 回饋大致分為三種: (1)簡要式回饋-老師給予直接且明確的建議，具有高度可靠、高效率，且最有幫助的一種回饋。(2)正式的回饋-正式的回饋通常是以小組進行臨床教學，透過一些實際的例子或症狀來教學，再

循序進行解釋與討論。其回饋主張通用型的概念，但不排斥例外想法與狀況。老師也可以利用引導式的方法，促使學習者自行給予反饋，思考整個歷程的概念並想方法解決。(3)主要式的回饋-依照小組的模式，實際的進行臨床的觀察，透過實際所觀察到的狀況、行為、問題等等，結合先備知識經驗進行事後的回饋。這回饋通常約持續15~30 分鐘，討論相關的概念與想法，且這階段學習者可以充分的表現與反思(Branch & Paranjape, 2002)。

教學系統回饋的種類可以分為(1)直述型-直接的回饋給學習者哪裡有錯誤與正確的概念、可以是直接的說明或提供結果。(2)視覺化類型- 視覺化的呈現想要表達的事物，比起純文字可以直觀的理解與引起學生的興趣增強學習效率。(3)概念圖-利用概念的思考與建立流程，呈現出思考的經歷讓學生更好了解每個階段的思考方法(Brinko, 2008)。

在授課過程的任務裡，也可以給予一些提示。可以分為(1)高階的提示:在某些困難處提示所需要應用的概念或方法，但過程與明確的做法不會供應。(2)直接的暗示: 在困難處直接的指出錯誤的位置或原因，明確的讓學生知道哪裡的問題(D. Perkins & Martin, 1986)。

在合作式學習的程式設計課程中，不但會讓老師也會允許學生對該課程之作業的程式碼進行主觀的想法、心得或相關意見回覆。這樣的做法也是一種教學回饋，學生可以看到老師與同儕間給予的回饋，並根據這多元的回饋來修正或啟發新的想法(Hwang, Shadiev, Wang, & Huang, 2012)。

二、 教學回饋的效益

有些研究(D. Perkins & Martin, 1986)將提示分為直接性與高階型，直接性提示是根據學生發生的狀況給予具體、明確的答案，讓學生直接的得知錯誤的概念或地方，高階型提示是針對學生目前有問題的地方，給予大致的方向或相關概念讓學生自行的體會與嘗試。而直接性的提示通常都可以解決一般的問題，且因為學生在較困難的概念上較為鬆散，所以如給予太高階型的提示式相對沒有效果的，以至於直接性的提示都比高

階型的提示來的有效率、更明顯有成效。在合作式學習為主的研究中，也提到老師與同儕間給予的回饋是對學習有正向幫助的。例如:學生可以依照回饋進行自我反思與改進，同儕間的回饋不但可以互相得到學習上的幫助，也可以促進團體中的合作式學習與問題解決氣氛。在互相給予回饋的過程，會因為瀏覽其他人的課程作業並自我思考所要給對方的訊息，重新的整合並反思所學過的概念，也會增強學習的動機與學習成就(Hwang et al., 2012; Van Den Boom, Paas, Van Merriënboer, & Van Gog, 2004)。

回饋在醫學界是非常重要的環節，簡要的回饋會相較多數的；正式的回饋是較多老師使用的方式，也取決老師怎們跟患者互動並從中學習或教學。主要的回饋是學習者最讚賞的教學模式，但是學生需要主動的反思才會有明顯的效果(Branch & Paranjape, 2002; Brandt, 2008)。在數學方面的研究也顯示，學生如擁有良好的相關主題概念，是對於後續上的學習非常有幫助的。因為他們先前的概念知識會讓後面所學習到的知識有所連結，可以增進他們學習效率與意願，相關課程內容的概念誤用也可以有所釐清(Anseel, Lievens, & Schollaert, 2009; Booth, 2011)。

三、 迷思概念的回饋

要先釐清學生對該方面的知識是否正確，才可以進行基本觀念的建立。在科學教育中，學生在學習新的理論與自然現象之前就有了前先的概念與對此物的觀點。而這些概念如是錯誤的不但會影響到這一階段的學習也會延伸影響到其他的概念。而有些學生的迷思概念是要加以釐清的才能完全接受正確的想法。相較於傳統的口述教學，如以文本與概念圖配合著教學，某概念可以更有效斷絕迷思概念的建立與激勵學生有新的思維以及了解自己正確概念建立的過程(Akbaş & Gençtürk, 2011; Brinko, 2008)。

程式設計教學系統方面可以加入引導的輔助功能，此研究分為提問與解釋兩種引導方式回饋給學生。首先對學生的錯誤提出疑問並確定其觀念是否正確，再對其問題進行進一步的解釋說明。在(Java 程式設計初學者之迷失與迷思概念分析)研究中將初學 Java 的學生之錯誤分類為:(1)迷思概念-想法上的迷思與對於類似概念的混淆。(2)缺失的概念-不熟悉語法所造成或無法連結應用某概念(C.-L. Chen et al., 2012;

Holland et al., 1997)。在此研究是以問題填答的狀況來判斷並加入引導，而本篇的研究是以程式碼編寫的狀況進行後端的比對判斷。

在針對迷思概念的回饋方面，可以使用範例式的教學模式來矯正概念，教學者可以配合適當的範例引導學生有自我解釋的機會，這過程可以訓練學生利用已有的概念並結合、解釋新學會概念。有時候也可以利用有錯誤的題目，這樣可以解釋為何該題目是錯的，反而相較純解釋正確的題目更有所幫助(Booth, 2011; D. N. Perkins & Simmons, 1988)。

第三章　研究方法

本章共分為六節，第一節說明本研究設計與架構；第二節說明研究對象的相關背景、能力；第三節描述本研究所需的教學平台及蒐集實驗資料的工具與方式；第四節說明實驗前、中、後的流程；第六節則敘述研究所蒐集的資料項目與如何做分析討論。

第一節　研究設計與架構

本研究希望利用迷思概念診斷與矯正來輔助學生建立正確的程式設計概念、或減少迷思概念的產生。故本研究會設計一套程式設計教學課程，內容依照課綱所指出的程式設計相關項目，會包含程式的邏輯概念、基本語法架構、判斷式、迴圈、函式，程式的執行順序。再配合自行開發的教學平台來輔助。根據學生的課堂狀況、學習單、專題、考試來診斷可能有的程式設計迷思概念，並設計適當的教學回饋來促使學生反思。最後再分析所蒐集的量化、質性資料做進一步的討論與分析此種教學模式的影響。

第二節　研究實驗參與者

本研究的參與者為新北市某公立高中高二(11 年級)兩個班級的學生作為實驗對象。課程內容為高二的資訊課程，實驗分為控制組 37 人(15 男、22 女)、實驗組 35 人(13 男、22 女)兩個班級都是社會組，且都是第一次學習程式語言，其中實驗組有一位學過 C 語言。

本研究在實驗過程因為有學生出國與多次相關成績沒有參與，所以最後列入後續實驗分析的學生為控制組 31 人(男、女)、實驗組 32 人(男、女)，且由於是社會組的班級所以女生較多。

表 3-1 實驗參與者之實驗組與控制組人數

	參與人數	男女人數	實際統計人數	男女人數
控制組	37	15 男 22 女	31	13 男 18 女
實驗組	35	13 男 22 女	32	11 男 21 女

第三節　研究程序

在整個研究的實驗前，先在學習過程式語言(C 語言)的學生中做紙筆與上機的測驗，先行測驗與找出與其他相關研究有關的錯誤類別。並配合之後要進行實驗的課程內容做適當的篩選與修改。

根據先前的結果設計出實驗時會遇到的迷思概念列表，並解針對這些迷思概念做適當的相關回饋教材。並發展該課程所需的教學平台，此教學平台可以在線上編譯程式碼(javascript)與顯示執行結果，且在課程中的程式練習題會結合圖形化的輸出來增加學生的興趣與更直觀化的了解程式功能。系統會有自動批改練習題的功能，包含針對學生編寫的程式碼做後端的偵錯，再依照類別回饋相關的教材，教材方面會包含直述性、視覺化或程式語言相關連結資料。

本研究的教學實驗課程持續時間共 8 週，控制組與實驗組每週各兩小時，實驗開始前與結束後亦各安排態度問卷填答，運算思維測驗只有在實驗前實施，而實驗後並未實施。教學過程會有學習單、線上教學平台實作練習、期中期末測驗、網頁專題，教學實驗程序階段完成後，則進入資料蒐集與統整的程序。

為分析學生的態度影響，以了解該教學模式或課程設計是否合適，資料蒐集會有量化與質性的資料，包含學生學習單、期中期末測驗、網頁專案、態度問卷、運算思維測驗、以及實驗後進行半結構式訪談的質性資料。在實驗數據分析階段會對學生的相關資料做迷思概念編碼，再進行後續相關的統計分析，研究流程圖如下圖 3-1 所示。

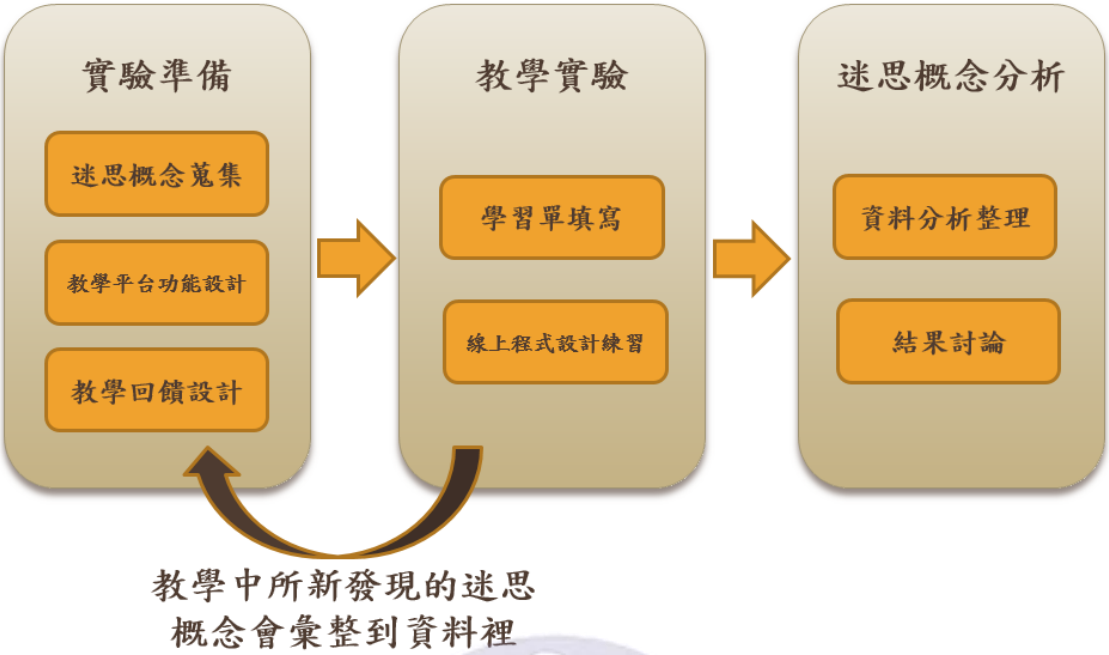


圖 3-1 研究流程圖，分為實驗準備、教學實驗、迷思概念分析三個階段，而實驗中的循環箭頭表示，迷思概念的蒐集與設計也會依照課程實驗中現場的學生狀況加以參考與修改。

每週依據課程進度有不同的主題概念與作業練習，針對課程主題每週學習單會有相關的練習，線上教學平台會有對應課程的作業練習。如表 3-2 所示，每週教學主題暨課程內容。在每一節課中，老師會先進行該主題的教學，並適時讓學生編寫練習。課堂後半段利用學習單的練習與討論，且課後會有線上程式設計的作業。

表 3-1 每週教學主題暨課程內容

週次	課程主題	課程活動	課後作業
1	JavaScript 基本介紹	基本語法與 I/O 使用	線上作業(一)
2	基本運算與 if、else	基礎判斷式概念	線上作業(二)
3	判斷式種類	判斷式與應用	線上作業(三)
4	期中測驗	測驗先期內容	-
5	迴圈種類	迴圈概念與應用	線上作業(四)、(五)
6	函式	函式概念與應用	線上作業(七)
7	網頁專案	製作個人網站	完成個人專案
8	期末測驗	測驗全部內容	-

第四節　研究工具

根據前面小節的研究架構、研究流程規劃，本研究所需用到的工具包含 JSLA 教學平台、課程中相關的學習單、量表、評量測驗，其使用方法詳細說明如下：

一、　JSLA 教學平台

本研究會利用 JavaScript Learning Assistant(JSLA)來輔助授課。其中此平台會有課程講義、線上練習、討論區的功能，也包含了程式線上編寫、執行、回饋的機制。後端也會記錄學習者的留言與討論內容、針對題目系統回饋的內容與次數，實驗後的分析會依據平台中學生程式碼編寫的狀況與後端資料的蒐集為基礎。

二、　課堂學習單

課堂中控制組與實驗組都會配合該主題的學習單練習與討論，而在實驗組的部分，會有依照該題目的引導提示，並有相關的反思題型讓學生進行概念的消化與自我思考。

三、　程式設計期中、期末成就測驗

為瞭解學習者在整個程式設計課程中所學習的成效，課程中會有程式設計的期中、期末成就測驗。內容包含了相關課程概念的選擇題、問答題、改寫問題、解釋程式碼題型等題型。期中與期末滿分都為 100 分，期末考試內容會包含期中考的概念並延伸應用，且題目會較多與相較困難。在解釋與問答的題型中會依照學生填寫的正確、完整性給予分數，選擇類型的題目則是依照題目定值得給予分數。

四、　運算思維測驗

運算思維量表是選取了 Bebras 的測驗題，大部分是選取高中生的題目，有少數題目是國中生的程度。總共 14 題，每題一分，題目都是需要邏輯性思考的類型，以及圖像理解的能力，雖然都是選擇題但還是具備挑戰性。該運算思維量表只有在前測進行測驗，後測並沒有實施。

五、 態度問卷

態度問卷是以五點量表所設計的，其中分為實用性、程式設計之自我感受、動機意願、自信、問題解決能力五個面向，總共為 22 題。本問卷採用李克特五點量表的方式計分(非常同意 5 分、同意 4 分、既不同意也不反對 3 分、不同意 2、非常不同意 1 分)，其中還包含了反向題，最後總分越高表示該面向有越正向的態度。該問卷在前測與後測都有施實，進而探討控制組與實驗組兩個班級的變化與差異。詳細量表題目，如下表 3-3 所示。

表 3-3 以下為態度量表，分為實用性、程式設計之自我體會、動機意願、自信、問題解決能力五個面向

面向	題目
實用性	1.我願意花時間在學習程式設計上。
	2.我認為程式設計可以解決各種領域的問題。
	3.程式設計對我的生活或職業是有幫助的。
程式設計之自我感受	4.程式設計對我來說是困難的。
	5.學習程式設計時，常會遇到一些迷惑的地方。
	6.程式設計是具有挑戰性的。
動機意願	7.我覺得程式設計很有趣。
	8.我希望以後有機會學習更多的程式設計。
	9.我願意花時間完成程式設計任務。
自信	10.程式設計讓我有成就感。
	11.我能夠獨力想出程式問題的解法。
	12.我需要透過他人的協助才能順利完成程式設計。
	13.如果有足夠的時間，我通常可以完成程式。
	14.就算目前的程式很複雜，我也可以很專心地編寫。
	15.我無法清楚了解程式的運作方式。
	16.程式設計對我來說像是在組合許多不可理解的黑盒子。

問題解決能力	17.寫程式遇到困難時，我通常透過同儕幫助來完成。
	18.寫程式遇到困難時，我通常求助老師來完成。
	19.如果有提示一些程式碼的寫法，我可以完成程式。
	20.當寫程式發生錯誤時，我通常可以順利地解決問題。
	21.遇到程式設計的困難時，我通常不知道該如何解決。
	22.遇到程式設計的困難時,我可以自己蒐集資料尋求解答。

六、 半結構式訪談題目

半結構式訪談會在整個實驗結束後實施，訪談內容為質性的資料，且是抽選控制組與實驗組兩班學習成就前、中、後各兩人並無計算分數。內容包含對整個課程的感受，對程式設計該科目的想法，對教學策略與教學平台的建議與實際體驗的心得。控制組與實驗組訪談的題目會有所不同，實驗組還會針對教學回饋的部分進行訪問，供後續的探討分析。詳細題目請參考表 3-4。

表 3-4 為半結構式訪談兩班之題目

	問題	控制組	實驗組
課程面向	1	談談你現在對程式設計此科目的看法?	
	2	談談對於程式設計上課模式的看法?(上課->帶範例->學習單->課後作業>檢討)	談談對於程式設計上課模式的看法?(上課->帶範例->學習單、反思->課後作業、檢討回饋教材)
	3	談談你對於完成學習單、線上程式編寫的感覺?	

	4	你在學習程式設計哪個主題時感到比較困難(程式執行順序、邏輯運算、迴圈、判斷式、函式等等)? 為什麼?	
	5	你在做學習單都是如何完成的(自己獨力思考解決、利用線上編譯的輔助、問老師或同學等等)，為什麼?	
	6		你覺得學習單的討論、自我概念反思及引導有哪些幫助?為什麼?
系統面向	7	範例練習與課後線上習題，在編寫過程中最常發生的困難為何?(進度沒跟上寫不出來、看不懂題意、時間不夠等等)為什麼?	
	8	你覺得系統練習題配合圖形化的輸出有哪些幫助或感想?	
	9	在編寫程式時編譯器所提供的錯誤訊息是否有幫助(編譯器的 error message)?為什麼?	
	10		在編寫程式時系統依照該題目所提供的相關概念回饋是否有幫助(提示與批改的功能)?為什麼?
	11		檢討練習題與學習單時所提供的相關回饋教材是否有幫助(動畫與相關概念教材)?為什麼?
	12	在製作網頁專題(綜合所有課程概念)時有甚麼困難與感想?為什麼?	

七、 迷思概念與錯誤的語法知識之編碼表

以下列表結合了實驗前的文獻蒐集、實驗前迷思概念預測的資料、與實驗中所發現的課堂狀況，整理出了學生會發生的迷思概念與錯誤的語法知識之列表。根據此列表可以針對學生的迷思概念與錯誤的語法知識進行編碼，以供後續的量化分析與討論。詳細內容下表 3-5、3-6 所示。

表 3-5 錯誤的語法知識編碼表

	編碼	分類說明	解釋	範例
錯誤的語法知識	S0	空白	包含語法不熟或其他原因而空白沒作答。	沒寫。
	S1	邏輯運算規則	x--2、i++等等運算學生會以自己錯誤的方法計算，包含計算錯誤。	i+=2 會認為是 i+多少會等於 2。
	S2	I/O 方面的錯誤	因為 input 與 output 的失誤導致錯誤或格式問題。	input 錯誤資料、output 印出「""」、「 」、「 」等符號所引發錯誤。
	S3	程式語言語法不熟	語法不正確錯導致的錯誤、無法運行。	結尾少「；」、「：」或宣告方式錯誤等基本規則。

表 3-6 迷思概念編碼表

	編碼	分類說明	解釋	範例
迷思概念	M1	變數相關概念	認為宣告的變數都是字串型態、認為變數不需要特別宣告就可以使用、 認為程式的指派與邏輯判斷是一樣的、認為例外狀況是不用另外設定規則。	認為「=」與「==」是一樣的、認為變數宣告一定="變數名稱"(字串型態)。
	M2-1	if、else if、else 的執行順序	認為程式執行之順序有非從上到下的狀況、或認為該程式會有同時進行判斷兩行程式碼的狀況。	學生認為程式在判斷條件時會一起跑 if、else if、else，其實是有順序的。
	M2-2	多加上了條件的判斷式	認為 else 後面必須加上條件判斷才能把 if 判斷之外的狀況涵蓋進來。	學生會寫出 else(x>=0)的判斷式程式碼來額外作條件限制。
	M2-3	switch case 相關概念	認為程式執行之順序有非從上到下的狀況、認為 case 後還要多加條件來過濾狀況、認為 break 不影響執行。	同一區 break 後的 print 卻執行了、認為 case1、2、3 會同時進行判斷。

	M3-1	for 迴圈之執行順序	認為該程式運作之順序不是一行一行執行的，並認為可以跳行的執行。	忽略每個 for 在最後會做一次終止條件的檢查，才結束迴圈程式。
	M3-2	for 之初始設定	認為索引值一定要=0、終止值都是>的狀況、每次條件變化都是 i++或認為重複執行的功能無法寫成迴圈形式。	任務要求重覆三次，學生寫成(i=0；i<=3；i++)
	M4-1	巢狀 for 迴圈之執行順序	認為該程式運作之順序不是一行一行執行的狀況，並認為可以跳行的執行。	多重迴圈部分，for 在最後會做一次終止條件的檢查，才結束迴圈程式。
	M4-2	巢狀 for 之初始設定	認為索引值一定要=0、終止值都是>的狀況、每次條件變化都是 i++或認為重複執行的功能無法寫成迴圈形式。	多重迴圈部分，內或外部迴圈要求重覆三次，學生寫成(i=0；i<=3；i++)
	M5	不同迴圈間的轉換	認為 do while 與 for 迴圈是無法互相達成一樣的目的。	在 while 與 for 的轉換上無法正確達成，或執行上功能不一樣。
	M6	變數的生命週期	認為 function 之全域與區域變數是一樣的、認為 function 內外部資訊有互相共用。	學生會想在 function 外呼叫 function 內部的變數 x。

第五節　JSLA 輔助教學工具

本研究實驗使用的教學平台 JSLA(JavaScript Learning Assistant)主要分為三個部分，第一部分為讓學習者可以線上的觀看或下載教學講義的教學課程。第二部分是讓學習者可以針對課程練習題或相關內容做留言的實作討論區。第三部分是線上練習區，供學生線上練習、編寫、執行程式碼。本節會分別針對每一項功能與教學運用作詳細的說明與介紹。

一、　JSLA-教學課程

JSLA 是以網頁為基礎架構的教學平台，其中包含了教學課程、實作討論區、線上練習的功能。該平台會讓學生進行個人帳號的註冊，以供紀錄作業的上傳、下載，以及討論區留言與回饋的狀況，如下圖 3-2。

圖 3-2 為教學系統登入帳戶後的畫面

在教學課程裡學生可以在此線上觀看上課的講義，或是下載此課程講義。在此區可以看到整個教學課程的內容與程序，課程內容包含了 JavaScript 的基本介紹、判斷式的種類與概念、迴圈的概念、函式的概念、以及期末的網頁專案製作，如下圖 3-3 所示。

圖 3-3 為教學系統教學課程的畫面

二、 JSLA-實作討論區

在實作討論區可以讓學生進行線上的留言，包含針對該練習題的疑問、該課堂中相關概念的想法、與同儕間的想法互動。留言的內容會包含疑問、心得、討論互動、重點概念筆記等等，其中這些留言會經由平台依照個人帳號紀錄於後端的資料庫(包含時間、內容、留言者)，如下圖 3-4 所示。

圖 3-4 為教學系統實作討論區的畫面

三、 JSLA-線上練習

線上練習區為學生主要進行 JavaScript 練習的地方，此區有基本的程式編寫、除錯、執行、顯示結果、上傳下載作業的功能，以及實驗組額外的針對練習題回饋相關教材的功能，如下圖 3-5 所示，平台 ip 為 140.122.76.98/aoe0 與 140.122.76.98/aoe1。

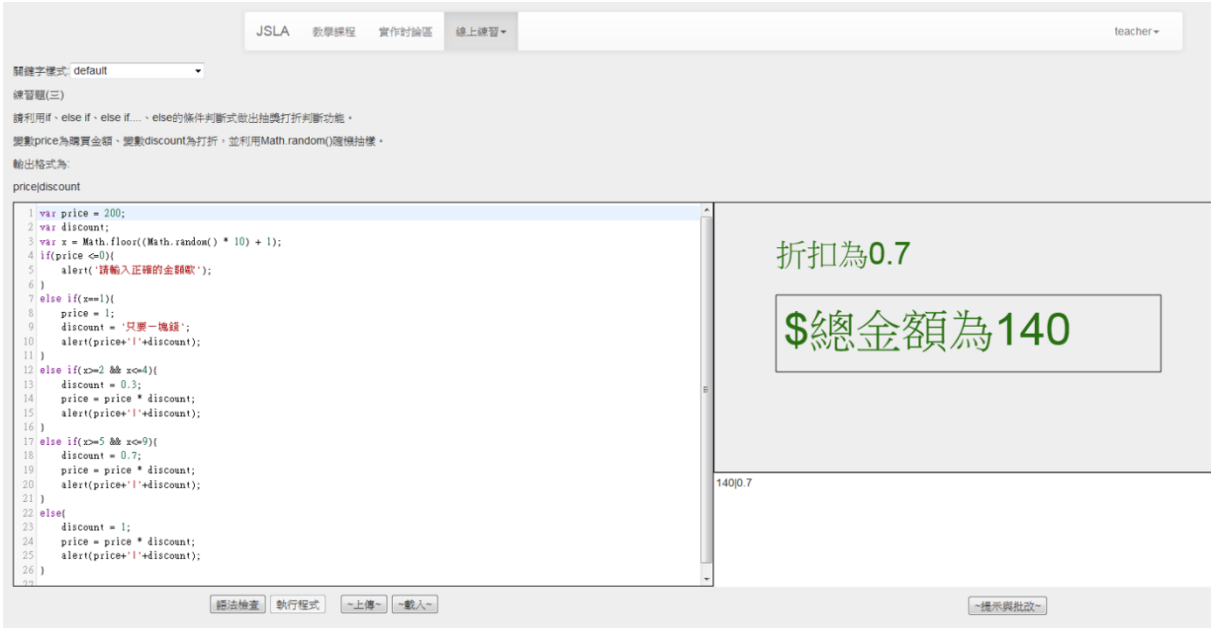


圖 3-5 為教學系統線上練習的畫面

在圖中 1 號區域是關鍵字變色與背景風格選擇，包含了不同種的色調與關鍵字強調變色處理，讓使用者可以依照個人的喜好與習慣去選擇，有了關鍵字變色的功能也可以提高程式編寫的效率與更快速、更清楚瀏覽程式碼的效果，如下圖 3-6、3-7 所示。

圖 3-6 為教學系統線上練習區的區域功能

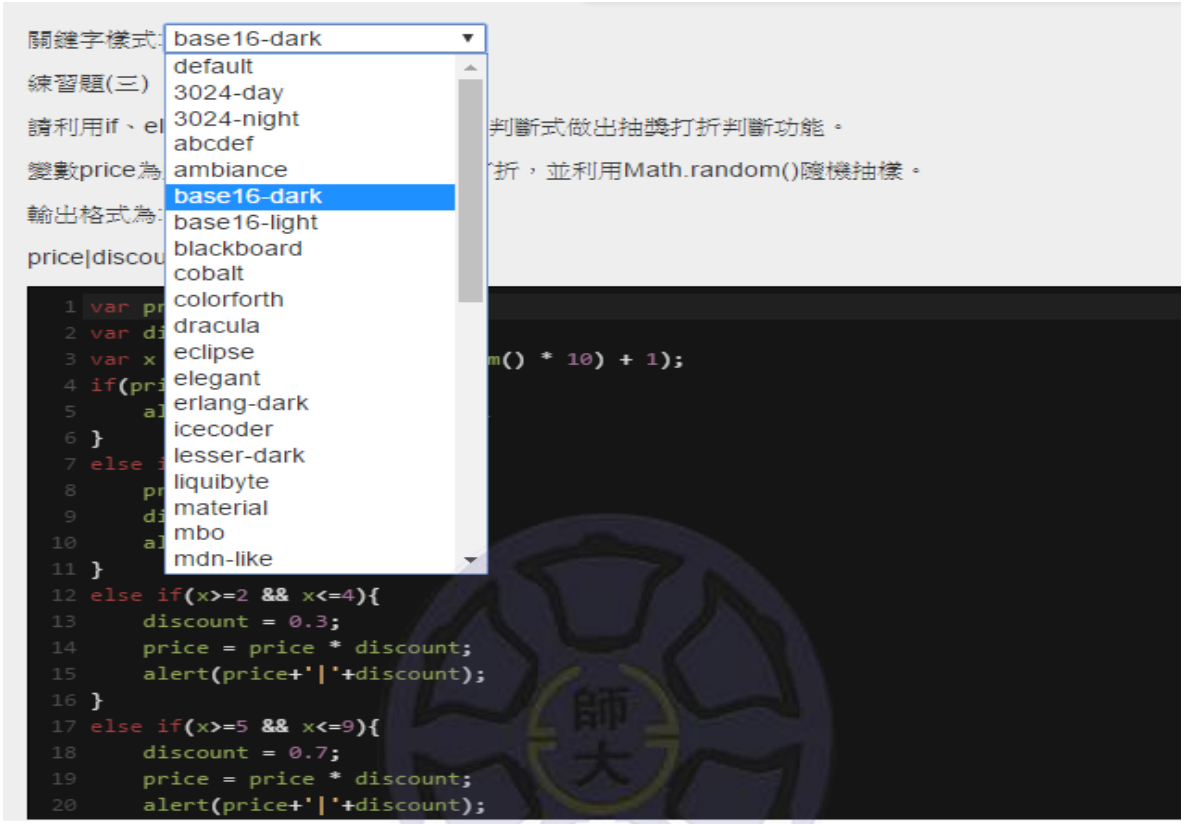


圖 3-7 為線上練習區編譯區風格設定功能

2 號區域為 JavaScript 程式碼編寫的地方，配合下方 3 號區域進行程式碼練習，包含語法檢查、程式執行與上傳、載入作業等功能。學生在此區練習的程序為編寫好程式碼，按下「語法檢查」的按鈕進行第一階段的後端語法除錯，如有錯誤會顯示錯誤訊息，如無語法上的錯誤，則可以按下「執行程式」的按鈕進行該程式的執行結果，如上圖 3-6 所示。

4 號與 5 號區域為程式執行的結果，5 號區域會顯示程式碼所執行後的輸出結果與語法錯誤時的錯誤訊息。4 號區域會依照 5 號所執行的結果進行後端的圖形化整合，針對該練習題配合更直觀的顯示方式或圖形化輸出，如下圖 3-8 所示。

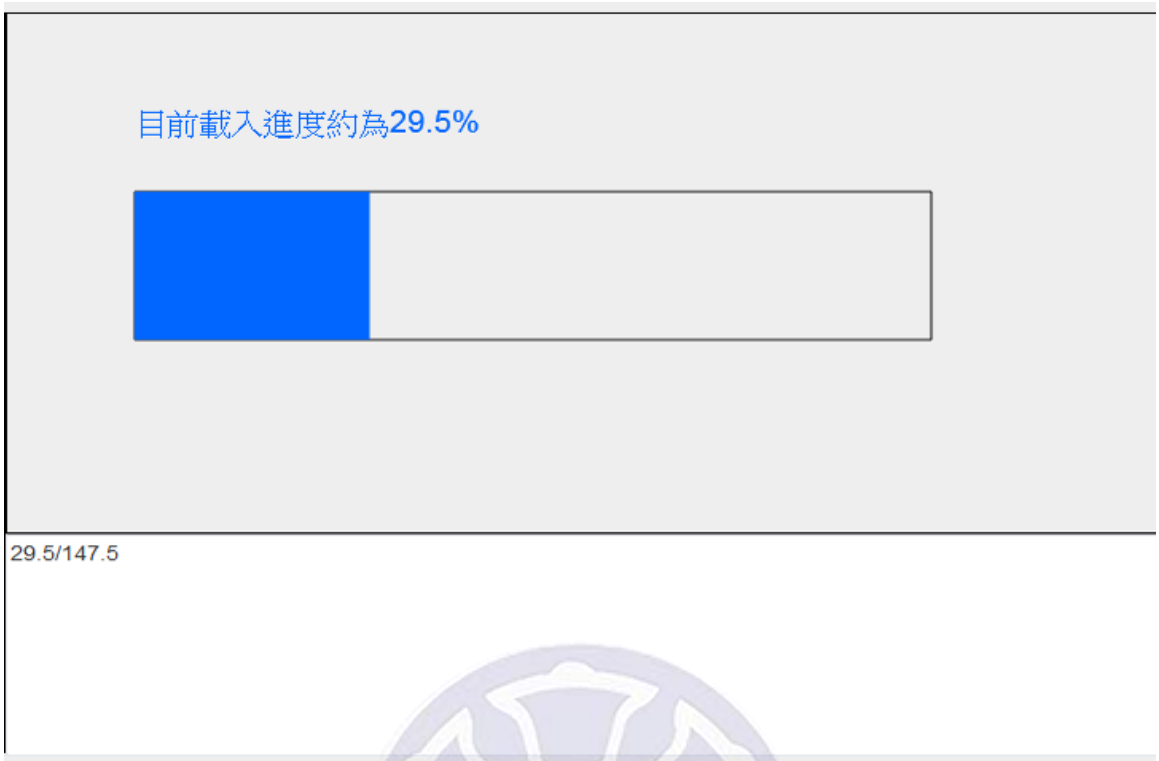


圖 3-8 更直觀的顯示方式或圖形化輸出

四、 迷思概念診斷與回饋機制

本研究將根據實驗前所蒐集、設計出來的迷思念列表來診斷學生的迷思概念。其中會有人為與系統兩類型的診斷，系統方面會根據學生在線上所編譯執行的程式，透過系統後端，根據輸出的結果，進行程式碼關鍵字或運算變數的錯誤比對，推斷學生可能造成錯誤的原因，若診斷出學生寫的程式可能因某項迷思概念所造成的錯誤，平台將給予相關的回饋教材，包含直接敘述錯誤的地方、配合圖文說明、以及供應相關概念語法資料。人為診斷部分包含了學習單、期中期末測驗學生所填答的狀況進行診斷，還有對線上程式練習所診斷的結果進行人為的二次診斷。

在 6 號區域(圖 3-6)的「提示與批改」的按鈕是實驗組才會有的功能，其功能會針對該練習題進行輸出結果的後端判斷。依照錯誤的方式回饋給學習者適當的補充教材，而教材方面會利用文字敘述、圖形與動畫等方式呈現，讓學習者更具體、有效的方式了解為何錯誤與相關的概念或語法知識，如下圖 3-9、3-10 所示。

圖 3-9 文字敘述的回饋

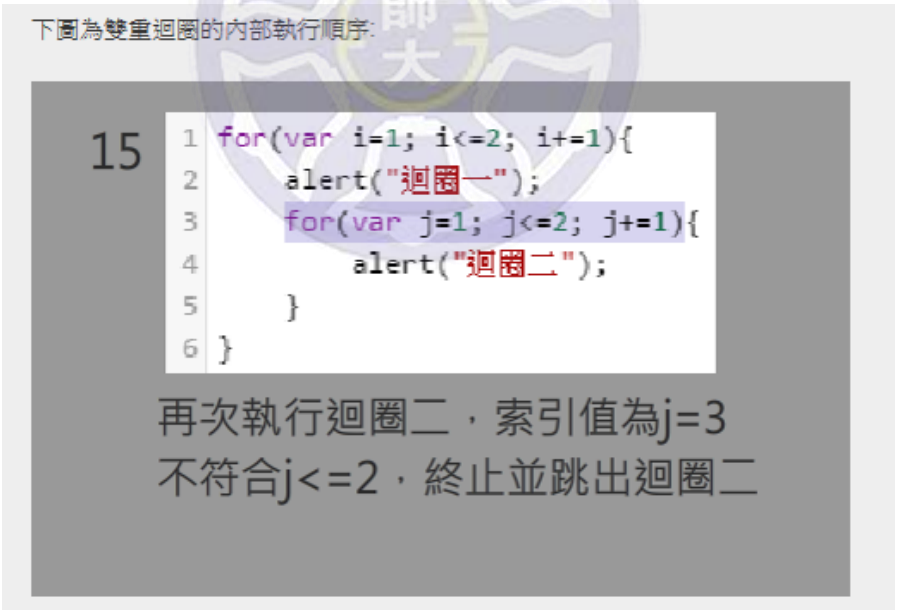


圖 3-10 圖形與動畫的回饋

第六節　資料蒐集與分析

一、　資料蒐集

1.量化資料蒐集

本研究根據所發展的迷思概念診斷與回饋教學策略，學生須在課堂中填寫學習單以進行練習與反思，課後亦須完成程式作業。為了解學生的學習狀況，亦進行期中與期末兩次成就測驗。研究所蒐集之量化資料除了學習單、課後作業、期中與期末成就測驗之成績之外，為了解學生撰寫程式時所發生迷思概念的狀況，亦將學習單填寫內容、課後程式作業、期中測驗、期末測驗之程式進行迷思概念編碼，以量化這些質性程式資料，便於後續的統計分析。系統方面會在後端記錄學生個人帳戶的回饋次數、時間、題型、模式，以及在留言區的討論、心得、問題等等之次數與時間。學生學習成就方面，會記錄學生課堂中的學習單、線上程式作業、期中期末測驗、專題分數、與整合的該課程總成績分數。量表方面，紀錄前後測的態度測驗結果，與前測的運算思維測驗結果。

2.質性資料蒐集

課程中的質性資料，會紀錄課程中每次學生在學習單的反思引導題所做的留言內容，以及教學系統的留言討論區之留言內容。課後的質性資料，課後會進行結構式的訪談，對控制組與實驗組該課程學習成就(根據學習成就之兩個標準差與平均值為基準)的前、中、後段各抽出兩位學生來做訪問。訪談內容包含對程式語言本身、程式設計課程的個人看法與該系統方面大致的使用情形，其中也會訪問到在整個程式設計課程中最困難的概念是哪個部分與再編寫程式的過程中最常面臨的問題等等相關資料。

二、　資料分析方法

1.對學習者程式設計學習成就之影響

根據準實驗研究法的理論，對實驗組與控制組做 ANCOVA 統計分析，可以探討學習者在兩種不同的程式設計教學策略中是否有顯著的差異。針對實驗組與控制組兩班的

期中與期末測驗的迷思概念與錯誤的語法知識進行量化分析，可以顯示出兩班在該方面是否有數量的不同。

2.對矯正學習者程式設計迷思概念之影響

將實驗組與控制組的迷思概念發生次數針對種類進行分類，並分析在那些迷思概念種類上，實驗組是否會少與控制組，且是否因為設計的相關教學回饋有所影響，並促進學習者進行反思與矯正該迷思概念，或是漸進的避免和減少發生的次數。

個別探討某學生在學習程式設計過程中的迷思概念矯正歷程，分析實驗組某學生在課程的學習單、線上作業、期中與期末測驗學習成就的變化，再根據整個過程的反思情況、系統回饋內容、迷思概念與錯誤語法知識類型和次數，進行質性分析與討論該過程的變化與影響。

3.對學習者程式設計學習態度之影響

對實驗組與控制組前後測的態度評分進行 ANCOVA 統計分析，探討兩班的差異與變化。根據態度評量的答題狀況，以及課程後的半結構式訪談內容來了解學生的想法，再討論為何會有這樣的差異與是甚麼原因造成這樣的結果。

第四章　分析結果與討論

本章共分為兩個小節，第一節說明本研究實驗後的資料統計分析之結果，並簡要的敘述其意義；第二節說明探討因為何種因素造成的結果差異與概念在學習歷程中的改變。

第一節 分析結果

一、　探究程式設計初學者常有的迷思概念

1. 程式設計迷思概念蒐集

在實驗前會先蒐集大部分程式設計之迷思概念相關文獻，並整理合併常發生的錯誤與誤解為列表以供後續參考的基礎與課程內容設計。其中包含了，常見的C++程式語言、Pascal、Python、Ruby、Java，以及使用不同的編寫平台介面，而初學者大部分會在程式結構、迴圈、判斷式、物件導向的概念上有困難，詳細整理之表格請參考附錄一。

2. 教學實驗前的程式設計迷思概念調查

在教學實驗前先針對程式設計初學者施測，以調查程式設計初學者之迷思概念，並對照由文獻所彙整之迷思概念，進而確認多數初學者會有的迷思概念。根據文獻所提的常見迷思概念，設計程式設計題目進行實際測驗。測驗內容包含了筆試部分與電腦實作部分，程式結構與概念包含邏輯運算、判斷式、迴圈、陣列、函式、遞迴(高中資訊科技課綱涵蓋之程式設計結構與概念)，程式語言為c語言。測驗參與者18人(14男、4女)，參與者包含高中與大學學生，且大部分是學過c語言，以下表4-1為測驗所整理出來的錯誤想法。

表4-1 教學實驗前程式設計測驗

	錯誤想法	相關概念
1	認為if、else if、else為同時執行的。	判斷式
2	認為條件不符合時，還是要執行內部程式。	判斷式
3	認為for迴圈在最後一次執行時，不需要檢查終止條件即可判斷是否停止。	迴圈

4	認為巢狀迴圈執行過程都不需要檢查終止條件，即可判斷是否到外層迴圈。	迴圈
5	認為迴圈終止值計算會被其他變數影響。	迴圈
6	認為 do while 迴圈只執行一次。	迴圈
7	認為陣列的第一個元素是 A[1]。	陣列

3. 教學實驗後程式設計初學者常有的迷思概念整理

該程式設計迷思概念列表示根據先前的文獻蒐集與實驗前的測驗改進的，並再根據教學實驗過程的狀況與學生的表現而加以修改的結果，其中可以看出學生在程式的執行順序上有明顯不同於專家的想法，進而導致後續各種的學習障礙與錯誤形式，如下表 4-2 所示。

表 4-2 實驗後所整理的迷思概念列表

編碼	分類項目	解釋	範例
M1	**變數相關概念**	認為宣告的變數都是字串型態、認為變數不需要特別宣告就可以使用、 認為程式的指派與邏輯判斷是一樣的、認為例外狀況是不用另外設定規則。	認為「=」與「==」是一樣的、認為變數宣告一定="變數名稱"(字串型態)。
M2-1	**if、else if、else 的執行順序**	認為程式指令執行之順序有非從上到下的狀況、或認為該程式會有同時進行判斷兩行程式碼的狀況。	學生認為程式在判斷條件時會一起執行 if、else if、else，其實是有順序的。
M2-2	**多加上了條件的判斷式**	認為 else 後面必須加上條件判斷才能把 if 判斷之外的狀況涵蓋進來。	學生會寫出 else(x>=0)的判斷式程式碼來額外作條件限制。
M2-3	**switch case 相關概念**	認為程式指令執行之順序有非從上到下的狀況、認為 case 後還要多加條件來過濾狀況、認為 break 不影響執行。	同一區 break 後的 print 卻執行了、認為 case1、2、3 會同時進行判斷。
M3-1	**for 迴圈之執行順序**	認為該程式指令運作之順序不是逐行執行的，並認為可以跳行地執行。	忽略每個 for 在最後會做一次終止條件的檢查，才結束迴圈程式。

M3-2	for 之初始設定	認為索引值一定要=0、終止值都是>的狀況、每次條件變化都是 i++或認為重複執行的功能無法寫成迴圈形式。	任務要求重覆三次，學生寫成(i=0；i<=3；i++)。
M4-1	巢狀 for 迴圈之執行順序	認為該程式指令運作之順序不是逐行執行的狀況，並認為可以跳行地執行。	多重迴圈部分，for 在最後會做一次終止條件的檢查，才結束迴圈程式。
M4-2	巢狀 for 之初始設定	認為索引值一定要=0、終止值都是>的狀況、每次條件變化都是 i++或認為重複執行的功能無法寫成迴圈形式。	多重迴圈部分，內或外部迴圈要求重覆三次，學生寫成(i=0；i<=3；i++)。
M5	不同迴圈間的轉換	認為 do while 與 for 迴圈是無法互相達成一樣的目的。	在 while 與 for 的轉換上無法正確達成，或執行上功能不一樣。
M6	變數的生命週期	認為 function 之全域與區域變數是一樣的、認為 function 內外部資訊有互相共用。	學生會想在 function 外呼叫 function 內部的變數 x。

二、 設計程式設計輔助教學平台與教學策略以矯正常見的程式設計迷思概念

在程式設計的課程中，有包含學習單、教學輔助系統的回饋，其中回饋有系統自動給的與老師批改後所給予的，內容有直接的提示錯誤原因、相關完整教材、圖文說明、動畫概念等等，下表 4-3 列出了各項迷思概念所對應給予的教學回饋。

表 4-3 迷思概念列表所對應的回饋資料

編碼	分類項目	回饋內容與資料
M1	變數相關概念	1.資料型態參考: http://taiwantc.com/js/js_tut_a3.htm#資料的型態 (Data Type)
M2-1	if、else if、else 的執行順序	1.邏輯判斷參考: https://msdn.microsoft.com/zh-tw/library/85yyde5c%28v=vs.94%29.aspx http://taiwantc.com/js/js_tut_a4.htm#邏輯運元 (Logical Operators) 2.判斷式參考:http://www.w3schools.com/js/js_if_else.asp 3.if-elseif-else.png（如下圖 4-1）

M2-2	多加上了條件的判斷式	1.判斷式參考:http://www.w3schools.com/js/js_if_else.asp 2.判斷式內之策略: https://www.youtube.com/watch?v=ImFNi1ALecg 3.h3_2.png
M2-3	switch case 相關概念	1.switch 參考資料: https://msdn.microsoft.com/zh-tw/library/hzc6t81t(v=vs.94).aspx http://www.w3schools.com/js/js_switch.asp
M3-1	for 迴圈之執行順序	1.迴圈執行狀況: http://163.32.98.15/teacher/benme/vb-player/forloopJs.html 2.h4.swf（如下圖 4-2）
M3-2	for 之初始設定	1.for 迴圈內部執行策略:http://www.w3schools.com/js/js_loop_for.asp https://msdn.microsoft.com/zh-tw/library/s1cyybdf(v=vs.94).aspx
M4-1	巢狀 for 迴圈之執行順序	1.for 迴圈參考資料: http://163.32.98.15/teacher/benme/vb-player/forloopJs.html 2.h5.swf（如下圖 4-3）
M4-2	巢狀 for 之初始設定	1.for 迴圈參考資料: http://www.w3schools.com/js/js_loop_for.asp https://msdn.microsoft.com/zh-tw/library/s1cyybdf(v=vs.94).aspx
M5	do while 之執行策略	1.While 參考資料: https://msdn.microsoft.com/zh-tw/library/6wsy66x9(v=vs.94).aspx http://www.w3schools.com/js/js_loop_while.asp
M6	function 之變數呼叫與回傳	1.函式參考資料: https://msdn.microsoft.com/zh-tw/library/4t2k5yhw(v=vs.94).aspx http://www.w3schools.com/js/js_functions.asp

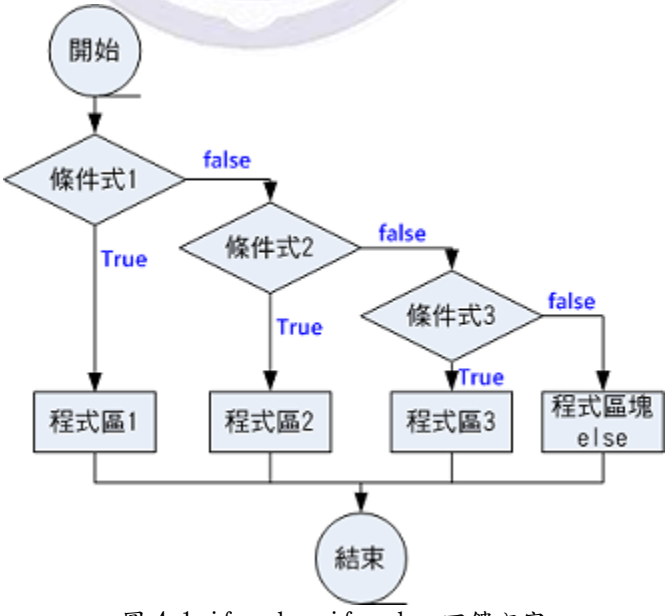


圖 4-1 if、else if、else 回饋內容

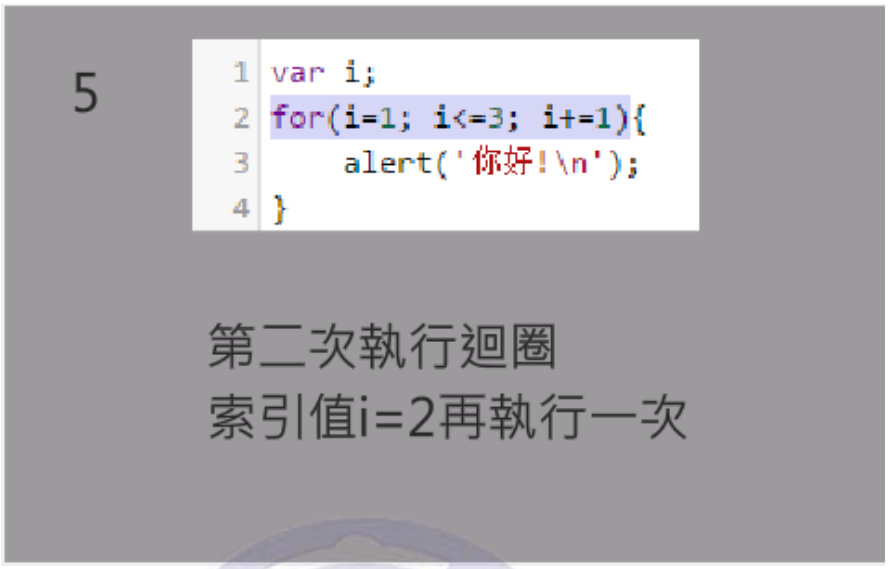


圖 4-2 for 執行順序的動畫截圖

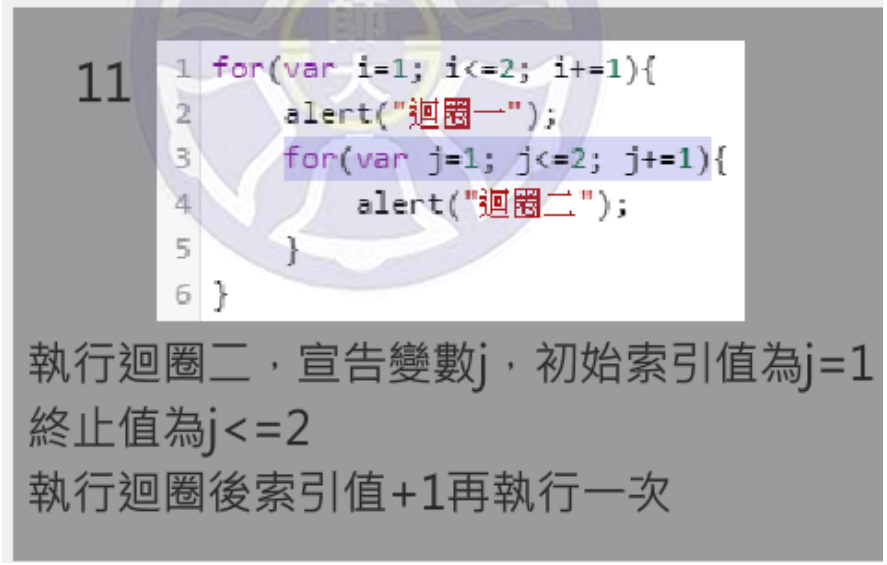


圖 4-3 巢狀 for 執行順序的動畫截圖

三、 評估所設計之程式設計輔助教學平台與教學策略之效能

1. 對學習者程式設計學習成就之影響

將實驗組與控制組該課程的學習成就與運算思維前測進行 ANCOVA 統計分析，下表 4-4 為兩組在學習成就的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將學習成就進行兩組的 ANCOVA 分析結果。

控制組 31 人學習成就平均為 50.44，標準差為 15.12。實驗組 32 人學習成就平均為 63.39，標準差為 15.24，且兩班顯著性為 0.807 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.000 小於 0.05 表示兩班在學習成就方面具有顯著差異的，且實驗組高於控制組。

表 4-4 實驗組與控制組學習成就之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	50.44	15.12	31	23.742	0.000***
Experimental	63.39	15.24	32		

將實驗組與控制組該課程的學習單成績與運算思維前測進行 ANCOVA 統計分析，下表 4-5 為兩組在學習單成績的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將學習單成績進行兩組的 ANCOVA 分析結果。

控制組 31 人學習單平均為 65.24，標準差為 17.24。實驗組 32 人學習單平均為 72.50，標準差為 16.08，且兩班顯著性為 0.570 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.020 小於 0.05 表示兩班在學習單方面具有顯著差異的，且實驗組高於控制組。兩組在學習單上的比較可以了解引導反思模式的效果與建立正確概念的影響。

表 4-5 實驗組與控制組學習單之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	65.24	17.24	31	5.722	0.020*
Experimental	72.50	16.08	32		

將實驗組與控制組該課程的線上程式練習成績與運算思維前測進行 ANCOVA 統計分析，下表 4-6 為兩組在線上程式練習的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將線上程式練習分數進行兩組的 ANCOVA 分析結果。

控制組 31 人線上程式練習平均為 43.11，標準差為 14.47。實驗組 32 人線上程式練習平均為 53.02，標準差為 15.86，且兩班顯著性為 0.737 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.002 小於 0.05 表示兩班在線上程式練習方面具有顯著差異的，且實驗組高於控制組。

表 4-6 實驗組與控制組線上程式練習之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	43.11	14.47	31	10.934	0.002**
Experimental	53.02	15.86	32		

將實驗組與控制組該課程的專題成績與運算思維前測進行 ANCOVA 統計分析，下表 4-7 為兩組在專題的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將專題分數進行兩組的 ANCOVA 分析結果。

控制組 31 人專題平均為 64.03，標準差為 28.79。實驗組 32 人專題平均為 79.83，標準差為 32.85，且兩班顯著性為 0.965 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.007 小於 0.05 表示兩班在專題方面具有顯著差異的，且實驗組高於控制組。而專題部分是整合了每個主題的網頁製作專案，間接代表了程式設計實作與整合應用能力。

表 4-7 實驗組與控制組專題之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	64.03	28.79	31	7.672	0.007**
Experimental	79.83	32.85	32		

將實驗組與控制組該課程的期中測驗成績與運算思維前測進行 ANCOVA 統計分析，下表 4-8 為兩組在期中測驗的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將期中測驗分數進行兩組的 ANCOVA 分析結果。

控制組 31 人期中測驗平均為 52.54，標準差為 20.84。實驗組 32 人期中測驗平均為 63.81，標準差為 23.22，且兩班顯著性為 0.880 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.009 小於 0.05 表示兩班在期中測驗方面具有顯著差異的，且實驗組高於控制組。

表 4-8 實驗組與控制組期中程式設計概念測驗之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	52.54	20.84	31	7.247	0.009**
Experimental	63.81	23.22	32		

將實驗組與控制組該課程的期末測驗成績與運算思維前測進行 ANCOVA 統計分析，下表 4-9 為兩組在期末測驗的描述性統計與同質性檢定，並以運算思維前測成績為共變數，將期末測驗分數進行兩組的 ANCOVA 分析結果。

控制組 31 人期末測驗平均為 34.64，標準差為 20.40。實驗組 32 人期末測驗平均為 55.71，標準差為 19.53，且兩班顯著性為 0.625 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.000 小於 0.05 表示兩班在期末測驗方面具有顯著差異的，且實驗組高於控制組。

表 4-9 實驗組與控制組期末程式設計概念測驗之描述性統計資料與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	34.64	20.40	31	27.749	0.000***
Experimental	55.71	19.53	32		

2. 對矯正學習者程式設計迷思概念之影響

將兩班期中測驗的錯誤的語法知識與迷思概念次數做比較，控制組錯誤的語法知識 72 次、迷思概念 55 次，實驗組錯誤的語法知識 57 次、迷思概念 42 次。期中測驗結果控制組的班級在錯誤的語法知識與迷思概念上都比實驗組來的多，如圖 4-4 所示。

將兩班期末測驗的錯誤的語法知識與迷思概念次數做比較，控制組錯誤的語法知識 304 次、迷思概念 71 次，實驗組錯誤的語法知識 248 次、迷思概念 63 次。期末測驗結果控制組的班級在錯誤的語法知識與迷思概念上都比實驗組來的多，如圖 4-5 所示。

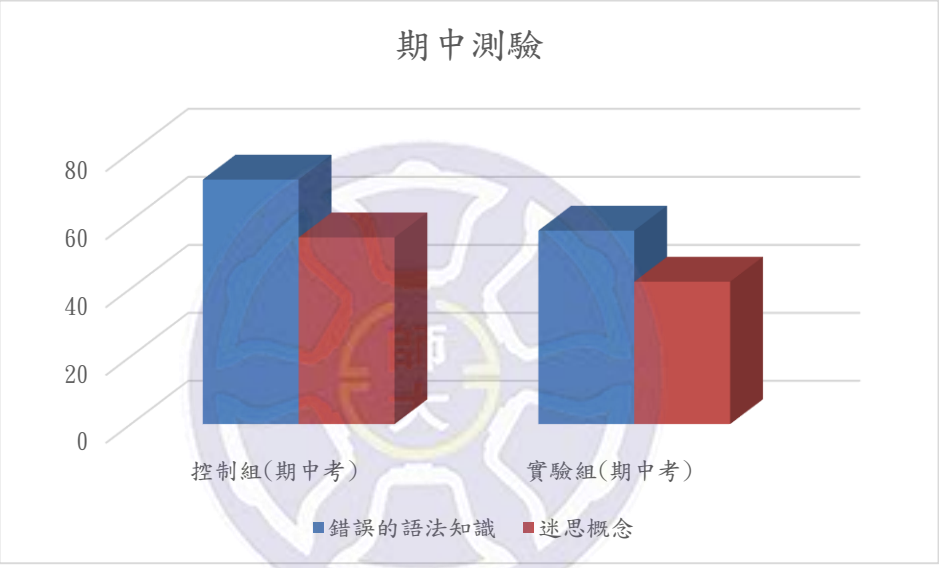


圖 4-4 兩班期中測驗錯誤的語法知識與迷思概念的次數

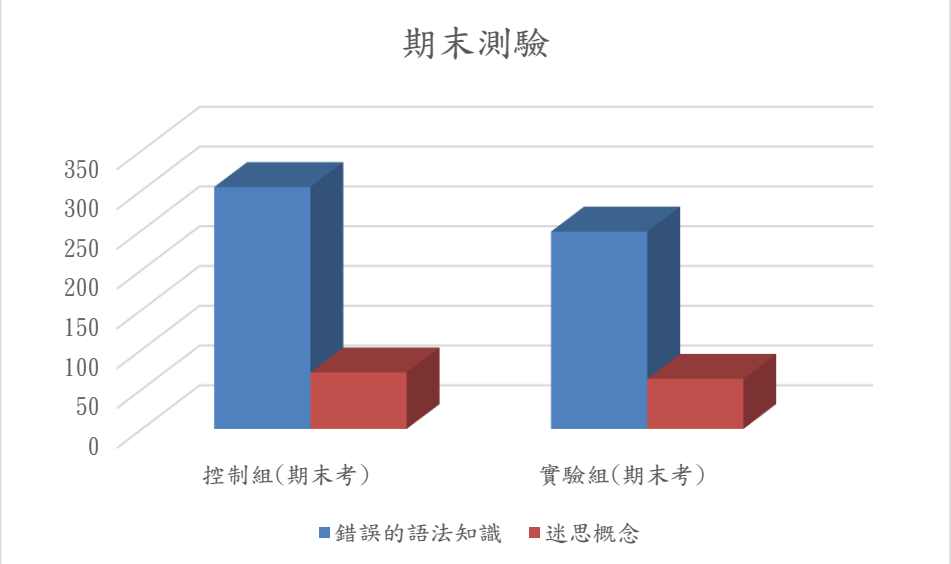


圖 4-5 兩班期末測驗錯誤的語法知識與迷思概念的次數

將控制組與實驗組兩班的錯誤的語法知識與迷思概念編碼進行排序，依照課堂的順序來列出下表資料。結果顯示控制組在錯誤的語法知識上比實驗組來的多，且大多都是 S0(空白)的狀況，如下表 4-10 所示。

表 4-10 為控制組與實驗組兩班的錯誤的語法知識依時間順序列表

控制組	S0	S1	S2	S3	實驗組	S0	S1	S2	S3
線上程式練習 1	2	0	10	2	線上程式練習 1	0	0	2	0
學習單 2	16	3	26	10	學習單 2	11	0	20	7
線上程式練習 2	17	4	4	5	線上程式練習 2	0	1	7	2
學習單 3	3	0	12	0	學習單 3	12	0	2	0
線上程式練習 3	1	0	4	24	線上程式練習 3	6	0	4	15
期中測驗	44	14	2	12	期中測驗	29	6	3	19
學習單 4	86	6	11	11	學習單 4	83	0	16	4
線上程式練習 4	25	0	4	1	線上程式練習 4	17	0	0	0
學習單 5	26	0	0	7	學習單 5	20	0	0	0
線上程式練習 5	32	0	0	0	線上程式練習 5	26	0	0	0
線上程式練習 7	2	0	6	0	線上程式練習 7	4	0	0	0
期末測驗	188	16	14	86	期末測驗	139	15	10	84

表 4-11 與表 4-12 結果顯示，控制組在不同迷思概念類型的錯誤狀況是相較實驗組少的，因為控制組大部分狀況都是被歸類到錯誤的語法知識，實驗組在不同類型上都是多於等於控制組。

表 4-11 為控制組的迷思概念依時間順序列表

控制組	M1	M2-1	M2-2	M2-3	M3-1	M3-2	M4-1	M4-2	M5	M6
線上程式練習 1	11	0	0	0	0	0	0	0	0	0
學習單 2	0	1	0	0	0	0	0	0	0	0
線上程式練習 2	9	2	0	0	0	0	0	0	0	0
學習單 3	23	17	22	18	0	0	0	0	0	0
線上程式練習 3	24	26	1	0	0	0	0	0	0	0
期中測驗	5	25	15	10	0	0	0	0	0	0
學習單 4	0	0	0	0	0	0	0	10	1	0
線上程式練習 4	0	0	0	0	0	2	0	0	0	0
學習單 5	0	5	0	0	0	5	0	0	0	25
線上程式練習 5	0	0	0	0	0	0	0	2	0	0
線上程式練習 7	0	0	0	0	0	0	0	0	0	0
期末測驗	0	23	0	5	22	8	5	1	0	7

表 4-12 為實驗組的迷思概念依時間順序列表

實驗組	M1	M2-1	M2-2	M2-3	M3-1	M3-2	M4-1	M4-2	M5	M6
線上程式練習 1	11	0	0	0	0	0	0	0	0	0
學習單 2	0	7	0	0	0	0	0	0	0	0
線上程式練習 2	31	2	0	0	0	0	0	0	0	0
學習單 3	3	19	21	40	0	0	0	0	0	0
線上程式練習 3	20	14	1	0	0	0	0	0	0	0
期中測驗	12	24	4	2	0	0	0	0	0	0
學習單 4	0	0	0	0	0	2	14	14	12	0
線上程式練習 4	0	0	0	0	0	4	0	0	0	0
學習單 5	0	0	0	0	0	0	0	0	0	11
線上程式練習 5	1	0	0	0	0	0	0	6	0	0
線上程式練習 7	0	0	0	0	0	0	0	0	0	0
期末測驗	0	11	0	5	24	12	0	9	0	2

3. 對學習者程式設計學習態度之影響

態度的評測方面會將學生態度量表(分為實用性、程式設計之自我體會、動機意願、自信、問題解決能力五個面向)進行加總，其中含有反向的題目。實驗組與控制組以態度前測分數為共變數，將兩班態度後測分數進行 ANCOVA 分析，並配合半結構式訪談的內容做進一步的討論。

控制組 31 人態度評測平均為 32.90，標準差為 10.65。實驗組 32 人態度評測平均為 36.28，標準差為 14.09，且兩班顯著性為 0.875 大於 0.05 表示兩班為同質性的。在 ANCOVA 檢定部分，控制組與實驗組顯著性為 0.757 大於 0.05 表示兩班在態度評測方面不具有顯著差異，如下表 4-13 所示。態度前後總分控制組與實驗組有顯著差異，經過成對樣本 t 檢定顯示後測總分有下降的現象，控制組 41.38 降為 32.90，實驗組 45.71 降為 36.28，如下表 4-14 所示。

表 4-13 實驗組與控制組態度後測之描述性統計與 ANCOVA 分析

Group	*Mean*	*SD*	*n*	*F*	*p*
Control	32.90	10.65	31	0.097	0.757
Experimental	36.28	14.09	32		

表 4-14 為兩班態度前後測成對樣本 t 檢定

Group	test	*n*	*Mean*	*SD*	*df*	*p*
Control	Pre test	31	41.38	9.97	30	0.001**
	Pro test		32.90	10.65		
Experimental	Pre test	32	45.71	12.83	31	0.000***
	Pro test		36.28	14.09		

態度量表題目信度分析，分為五個面向總共 22 題，實用性三題 Cronbach's alpha 為 0.834，程式設計之自我體會三題 Cronbach's alpha 為 0.846，動機意願三題 Cronbach's alpha 為 0.913，自信六題 Cronbach's alpha 為 0.753，問題解決能力六題 Cronbach's alpha 為 0.719，其中 4、5、6、12、15、17、18、21 題為反向題，如下表 4-15 所示。而第 16 題(16.程式設計對我來說像是在組合許多不可理解的黑盒子)因為 Cronbach's alpha 過低所以將其刪除。

表 4-15 為態度量表個面向的信度分析結果，斜體字為反向題(上方為控制組、下方為實驗組的數據)

面向	題目	Cronbach's alpha	前測 平均	後測 平均
實用性	1. 我願意花時間在學習程式設計上。	0.834	控制組 3.05 實驗組 3.62	控制組 2.62 實驗組 3.08
	2. 我認為程式設計可以解決各種領域的問題。		控制組 3.11 實驗組 3.85	控制組 2.14 實驗組 3.74
	3. 程式設計對我的生活或職業是有幫助的。		控制組 3.30 實驗組 3.68	控制組 2.97 實驗組 2.77
程式設計之自我體會	*4. 程式設計對我來說是困難的。*	0.846	控制組 3.86 實驗組 3.68	控制組 4.34 實驗組 4.32
	5. 學習程式設計時，常會遇到一些迷惑的地方。		控制組 3.86 實驗組 3.77	控制組 4.51 實驗組 4.32
	6. 程式設計是具有挑戰性的。		控制組 4.08 實驗組 4.22	控制組 4.42 實驗組 4.23
動機意願	7. 我覺得程式設計很有趣。	0.913	控制組 3.13 實驗組 3.31	控制組 2.60 實驗組 2.70
	8. 我希望以後有機會學習更多的程式設計。		控制組 2.94 實驗組 3.31	控制組 2.45 實驗組 2.74
	9. 我願意花時間完成程式設計任務。		控制組 3.02 實驗組 3.57	控制組 2.68 實驗組 2.94
自信	10. 程式設計讓我有成就感。	0.753	控制組 3.37 實驗組 3.68	控制組 2.61 實驗組 2.88
	11. 我能夠獨力想出程式問題的解法。		控制組 2.68 實驗組 2.97	控制組 2.23 實驗組 2.52

	12. 我需要透過他人的協助才能順利完成程式設計。		控制組 3.67 實驗組 3.57	控制組 4.11 實驗組 4.05
	13. 如果有足夠的時間，我通常可以完成程式。		控制組 3.27 實驗組 3.60	控制組 2.85 實驗組 3.08
	14. 就算目前的程式很複雜，我也可以很專心地編寫。		控制組 2.83 實驗組 3.22	控制組 2.38 實驗組 2.58
	15. 我無法清楚了解程式的運作方式。		控制組 3.21 實驗組 3.02	控制組 3.55 實驗組 3.50
問題解決能力	*17. 寫程式遇到困難時，我通常透過同儕幫助來完成。*	0.719	控制組 3.81 實驗組 4.05	控制組 4.05 實驗組 4.26
	18. 寫程式遇到困難時，我通常求助老師來完成。		控制組 3.16 實驗組 3.34	控制組 3.08 實驗組 3.17
	19. 如果有提示一些程式碼的寫法，我可以完成程式。		控制組 3.37 實驗組 3.72	控制組 3.11 實驗組 3.47
	20. 當寫程式發生錯誤時，我通常可以順利地解決問題。		控制組 2.62 實驗組 2.88	控制組 2.29 實驗組 2.53
	21. 遇到程式設計的困難時，我通常不知道該如何解決。		控制組 3.16 實驗組 3.22	控制組 3.61 實驗組 3.58
	22. 遇到程式設計的困難時，我可以自己蒐集資料尋求解答。		控制組 2.75 實驗組 3.20	控制組 2.47 實驗組 2.88

第二節　討論

一、　初學者常見之迷思概念

初學者在此 Javascript 程式語言的課程中，可以觀察到大部分的學生會對於程式的執行順序有所疑惑與誤解、以及對於變數、迴圈、判斷式方面有迷思概念的狀況。而學生對於執行順序上會有非逐行或認為可以同時執行的狀況，該迷思概念明顯影響到後續的課程，此概念會牽涉到判斷式與迴圈的概念建立與編寫應用。

二、 討論評估所設計之程式設計輔助教學平台與教學策略之效能

1. 對學習者程式設計學習成就之影響

控制組與實驗組兩班在學習成就方面有統計上的顯著差異，且實驗組較控制組高。學生學習成就是由學習單成績、線上練習成績、專題製作成績、的期中測驗成績、期末測驗成績所總合。

兩班學生在課堂中學習單的填答狀況相較課後線上程式練習的完整，所以在學習單的分數方面都比線上程式練習來的高。且學生表示線上程式練習時有些人看不懂題目想表達的問題、有些是知道該題目的任務，卻無法用程式語言來呈現，表示在讀完題目後進行程式語言抽象化的過程有所困難、有些是程式語言上有所不熟悉或是迷思概念所造成的錯誤。而在網頁專題上的製作實驗組的分數也是高於控制組，且大多數學生在此專題上表示相較前面的課程，對該專題製作的課程非常感興趣，上課與實作方面也較認真設計和有創意，所以整體專題製作分數都較高。

兩班在期中測驗與期末測驗方面也是實驗組高於控制組，在期中測驗時實驗組平均為 63.81，略高於控制組平均 52.54，而在期末測驗方面實驗組平均為 55.71，遠高於控制組平均 34.64。可能造成原因為期中測驗的迷思概念延續的影響後續的新知識建立(D. N. Perkins & Simmons, 1988; Putnam et al., 1986; Treagust, 1988)、期末測驗的題目偏向概念整合型問題、以及期末測驗題目數量與考試範圍都比期中測驗來的多。

2. 對矯正學習者程式設計迷思概念之影響

(1)變數

在變數相關概念的題目中，經常會發現學生有兩類狀況。第一類:學生會對變數的型態產生錯誤的應用，且會誤解該型態的意思。以下圖 4-6 為例，如題目規定 num 變數宣告為數值，x 變數宣告為字串，但學生會將兩個變數都利用「”” 」宣告為字串型態，雖然在輸出方面結果會一樣，但是其意義不同。在這種狀況下會回饋給學生圖 4-7 的資料型態種類列表，以及範例與相關應用，讓學生了解其資料型態種類的差異。且舉相關例子讓學生思考，把數值宣告成字串之後，雖然在單純輸出顯示方面會一樣，但如要做相關數值的運算，將會無法計算。

```
1 var num ="310265",X="#0088A8"
2 alert('(你好!)'+'('+num+')'+'(再見~)'+'('+X+')')
3
4
```

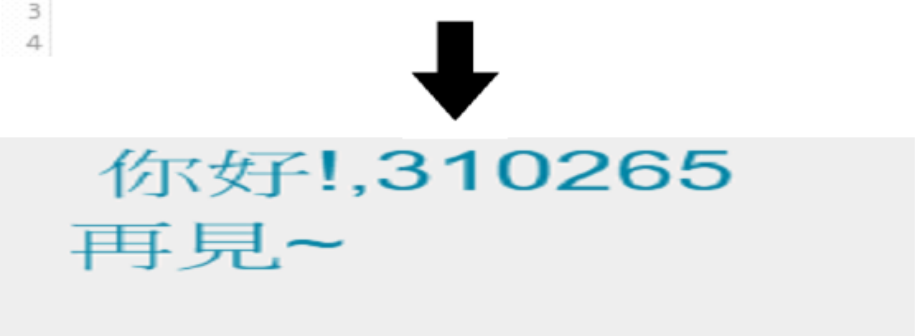


圖 4-6 為 JavaScript 程式碼與輸出解果

資料型態	例子	說明
字串 (String)	var name = "Tom"; var model = "1001";	字串通常是用雙引號(") 或單引號(')括著， 例子中表示了一個 Tom 字串和一個 1001 字串。

數目
(Number)
	var age = 16;
	var pi = 3.141592653589793;
	var expo = 54e7;
	var octal_num = 010;
	var hex_num = 0x10;
		數目可以是實數和整數， 整數包括10 進制、8 進制和 16 進制。 數目前面是 0 代表 8 進制， 數目前面是 0x 代表 16 進制

布林 (Boolean)	var is_enabled = true; var ie3 = false;	布林資料只有兩個值， 就是 **true** 或者 **false** 任何一個， 適合用作一些只有兩個可能的運算。
空白 (Null)	var no_content;	資料沒有內容， 例如你只定義它變數的名稱， 但沒有指定內容， 那麼它就等於 null 了。
物件 (Object)	var dw = document.write; var myname = document.myform.myname;	通常是為了方便而定義物件變數， 例如可以用 dw 代替 documen.write， 那就不用打很多字了
陣列 (Array)	var even_num = new Array(2,4,6,8,10); var fruit = new Array("apple","orange");	陣列是用來儲存一組相同型態的資料， 在本篇稍後詳述。

圖 4-7 為 JavaScript 資料型態例子與說明

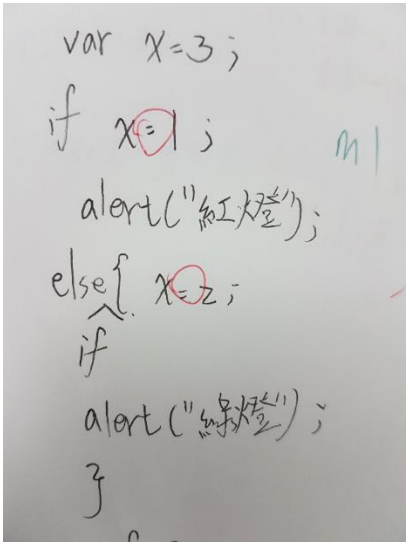


圖 4-8 為「=」誤用之範例

第二類:學生會將在作相關變數的運算時，會有將數學領域與程式設計的「=」誤用。數學中的意義是將「=」左邊定義相等於「=」右邊，而在程式設計中「=」是將右邊的數值指派為「=」左邊的變數。如上圖 4-8 為範例，學生將判斷式中的邏輯判斷單元誤用為「=」，所以程式在運行到此行時，就把原本變數 x 之數值改變了。此狀況回饋會以口述的方式，釐清其數學與程式設計在「=」上的概念，並讓學生實際動手做相關的範例練習。

參考下表 4-16，為控制組與實驗組兩班在 M1(變數相關概念)迷思概念發生的次數，且此表示依照時間所排序的。兩班在線上程式練習 1 與學習單 2 的結果顯示，M1 迷思概念發生次數無差異。但在學習單 3、線上程式練習 3 的結果顯示，實驗組發生 M1 的次數比控制組還少，前幾次兩班結果差不多，後續漸漸實驗組有所進步，可以證實該教學回饋是有效果的。而在期中測驗有關 M1 的相關題目，由於結合不同語法與概念，控制組較多空白的狀況所以 M1 次數低於實驗組。

表 4-16 為課程中 M1 迷思概念依照時間排序

控制組	S0	S1	S2	S3	**M1**	實驗組	S0	S1	S2	S3	**M1**
線上程式練習 1	2	0	10	2	**11**	線上程式練習 1	0	0	2	0	**11**
學習單 2	16	3	26	10	**0**	學習單 2	11	0	20	7	**0**
線上程式練習 2	17	4	4	5	**9**	線上程式練習 2	0	1	7	2	**31**
學習單 3	3	0	12	0	**23**	學習單 3	12	0	2	0	**3**
線上程式練習 3	1	0	4	24	**24**	線上程式練習 3	6	0	4	15	**20**
期中測驗	44	14	2	12	**5**	期中測驗	29	6	3	19	**12**

(2)條件判斷式

條件判斷式之相關概念的題目中，迷思概念分為 3 類。第一類:M2-1(if、else if、else 的執行順序)對於判斷式的程式運作流程有所迷思，有些學生會認為只需要檢查某幾個條件而已，大部分學生的狀況是會認為每個條件判斷是同時執行的，同時進行條件判斷是否符合。以下圖 4-9 為範例，根據程式執行的順序而言，程式輸出結果為”違規超速!罰 2000 元”，但有大多數學生認為判斷的條件是同時進行檢查的，所以這題會以為輸出結果為”罰 4000 元”。該迷思概念的回饋，會以圖 4-10 來解釋其程式執行順序，以及如何檢

查條件。再舉例更精簡版的相似題目來讓學生練習，排除學生因為判斷的內容與題目情境所影響。

```
1)    var speed = 120;
2)    if(speed >= 45){
3)          alert("違規超速!罰2000元");
4)    else if(speed >= 120){
5)          alert("罰4000元");      }
6)    else if(speed <= 30){
7)          alert("違規過慢!罰600元");
8)    else {
9)          alert("無違規");      }
```

圖 4-9 為 M2-1 迷思概念範例

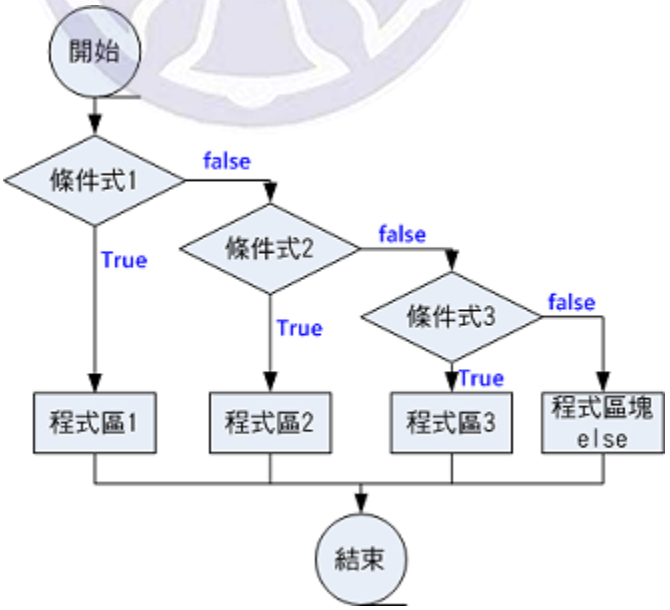


圖 4-10 if、else if、else 執行順序之回饋內容

第二類:M2-2(多加條件進行判斷)迷思概念，發生在學生寫 if、else 判斷式時，會認為在 else 的部分需要再配合額外的條件，程式才會順利執行。下圖 4-11 為學生在該題型發生 M2-2 的狀況，根據題目的規定把分數做等第分類，但學生會認為最後的 else 部分加上剩餘的狀況之條件，雖然本題在執行上不會影響輸出結果，但在某些情形程式就會有問題。M2-2 類的回饋教材，會以釐清判斷式整體架構與定義為主，給予相關的資料補充判斷式條件設定的功能與 else 的功能應用。讓學生了解 else 有把剩下來所有狀況都包刮了的概念，且在正常編寫程式時，不會如題目一樣知道進來的變數之值是甚麼，所以如額外設定剩餘的條件，會導致有些數值被錯誤的過濾掉。

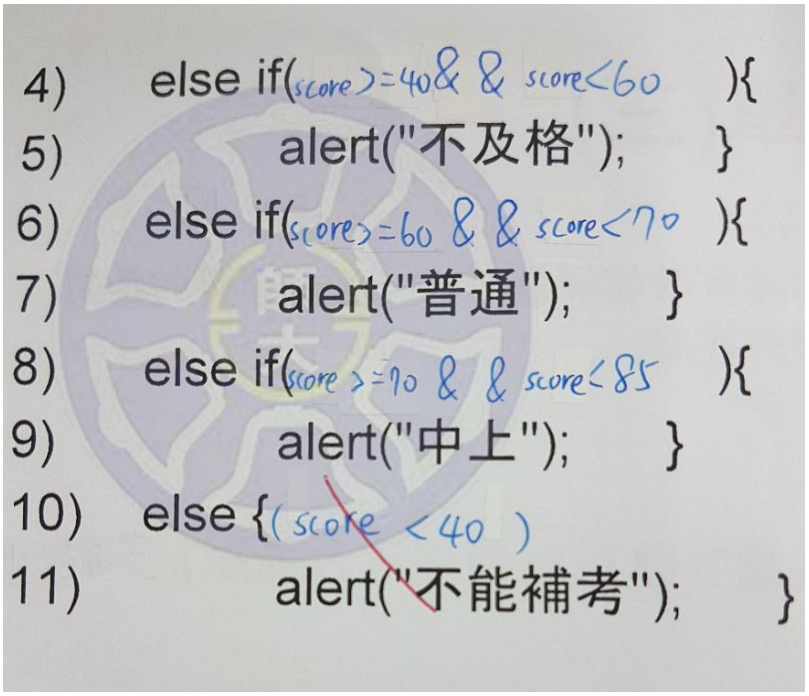


圖 4-11 M2-2 迷思概念狀況

第三類:M2-3(switch case 相關概念)迷思概念包含了，學生會認為 switch case 的各項狀況需要給條件做相關判斷，以及學生認為 switch case 在程式執行上會同時判斷所有條件，或者認為若以人腦判斷不符合 case 的所有條件，則 switch 中的指令便不必執行，所以在某些情形下會漏跑幾行或輸出結果錯誤。以下圖 4-12 為範例，如果變數 x=4 的狀況，學生會認為所有 switch 中的指令並不會執行，而其實是要全部指令執行並檢查過每個

case，才結束此判斷式的。此狀況會回饋給學生圖 4-13，讓學生更具體理解 switch case 的架構與執行順序，並連結 if、else 的相關概念與轉換技巧。

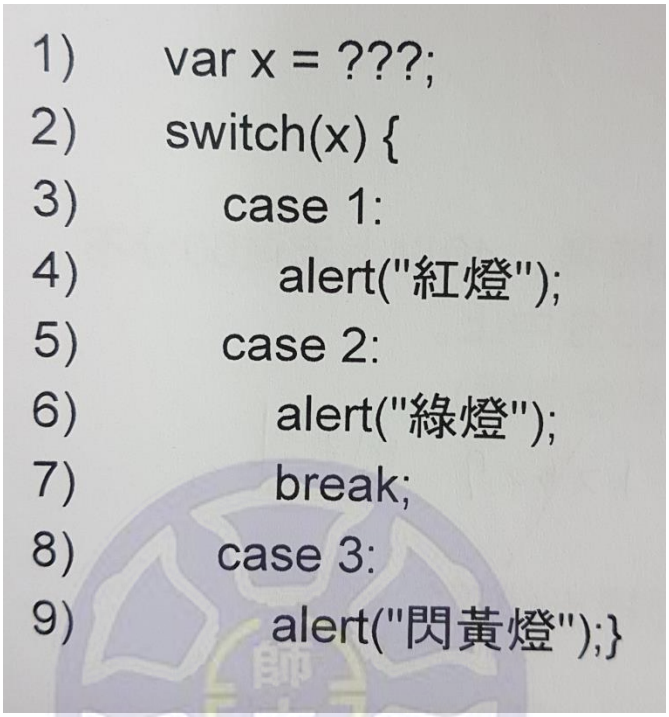


圖 4-12 M2-3 迷思概念狀況

參考下表 4-17，為控制組與實驗組兩班在 M2-1、M2-2、M2-3 的迷思概念發生的次數，且此表示依照時間所排序。整體 M2-1、M2-2 迷思概念發生次數控制組有比實驗組多的趨勢，配合回饋的教材實驗組有漸漸矯正迷思概念的現象，並證實相關回饋是有正向影響的。而在 M2-3 控制組與實驗組無明顯的差異，可能是因為相關 switch case 的練習次數不足，所以在回饋上次數也不多，導致學生在該方面較不熟悉，進步趨勢不顯著。

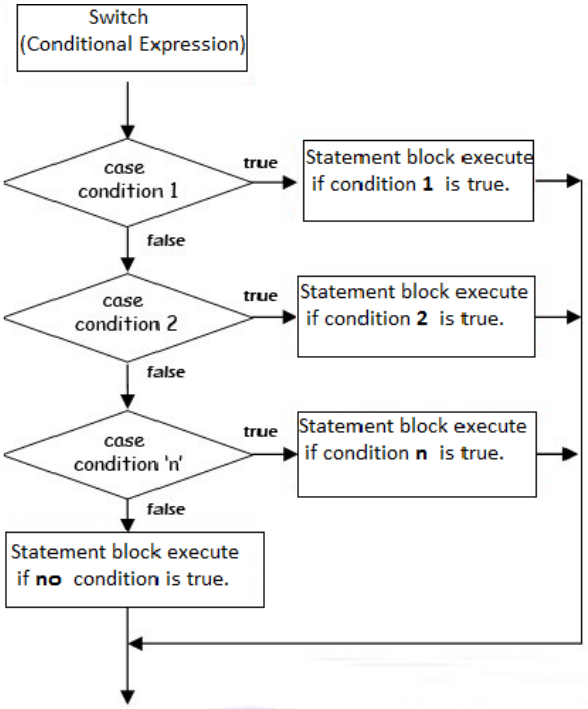


圖 4-13 M2-3 迷思概念圖文回饋

表 4-17 為課程中 M2-1、M2-2、M2-3 迷思概念依照時間排序

控制組	S0	S1	S2	S3	M2-1	M2-2	M2-3	實驗組	S0	S1	S2	S3	M2-1	M2-2	M2-3
學習單 2	16	3	26	10	1	0	0	學習單 2	11	0	20	7	7	0	0
線上程式練習 2	17	4	4	5	2	0	0	線上程式練習 2	0	1	7	2	2	0	0
學習單 3	3	0	12	0	17	22	18	學習單 3	12	0	2	0	19	21	40
線上程式練習 3	1	0	4	24	26	1	0	線上程式練習 3	6	0	4	15	14	1	0
期中測驗	44	14	2	12	25	15	10	期中測驗	29	6	3	19	24	4	2
期末測驗	188	16	14	86	23	0	5	期末測驗	139	15	10	84	11	0	5

(3)迴圈

在迴圈的迷思概念分為兩類，第一類 M3-1(迴圈執行順序)與 M4-1(巢狀迴圈執行順序)的迷思概念，學生會認為程式在執行迴圈時，可以同時執行某幾行，且非依序執行；或者認為迴圈指令的執行是做完迴圈判斷一定須進入迴圈，以下圖 4-14 為範例，學生會認為程式的執行順序上為 1>2>3>2>3>2>3，然後程式就跳出此迴圈部分。實際程式在運作是 1>2>3>2>3>2>3>”2”，最後還是要到前面條件進行檢查，才可以確認是否符合終止條件。該類型的迷思概念回饋會利用動畫的方式，讓學生更具體的體會整體執行的流程，且會解釋每一步驟的功能，詳細回饋內容請參考圖 4-2 與圖 4-3。

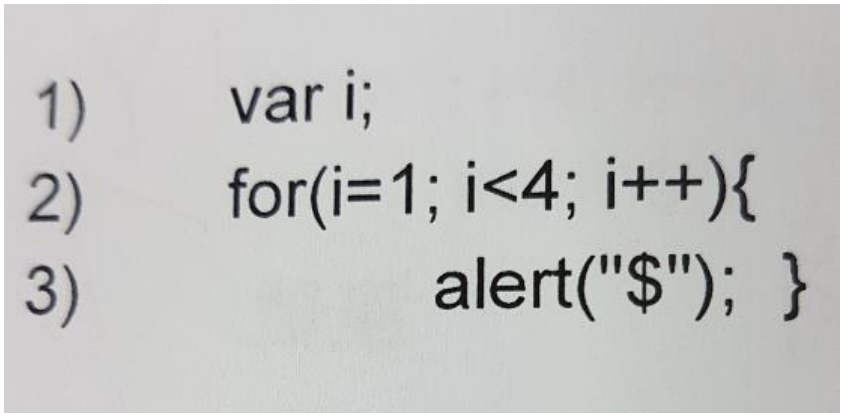
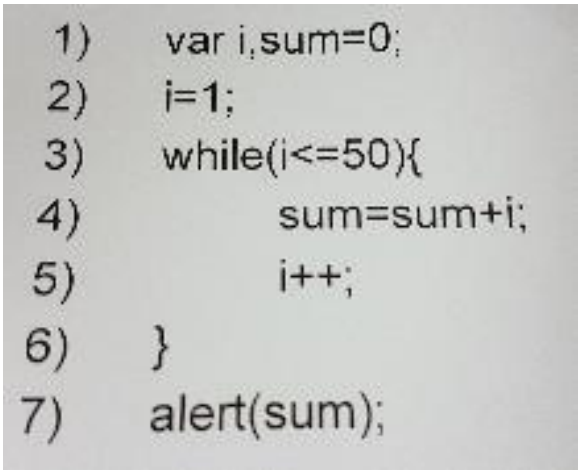


圖 4-14 M3-1 迷思概念例題

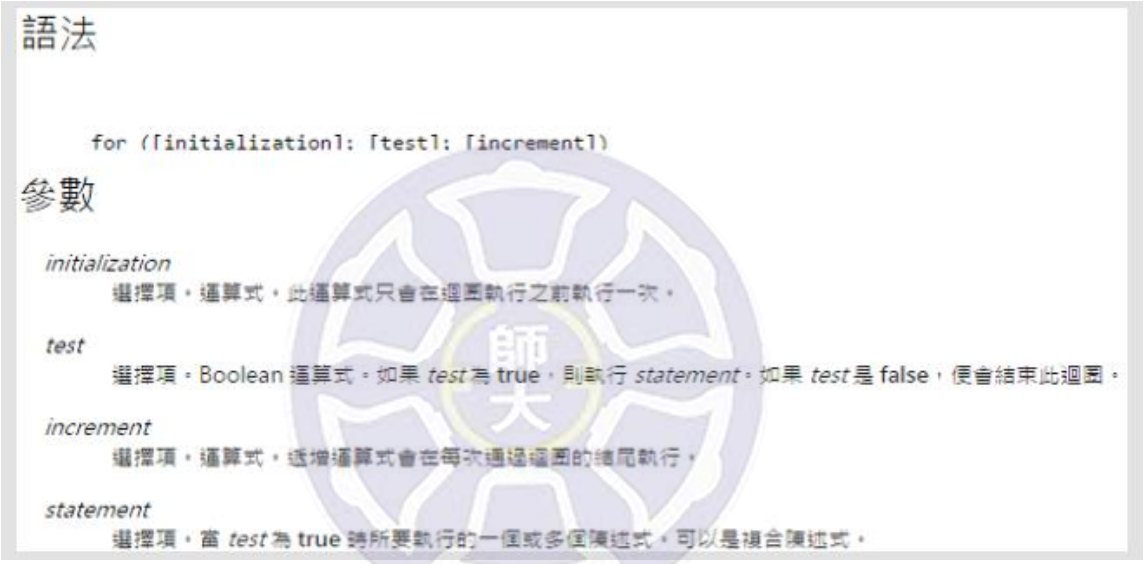
第二類 M3-2(初始設定)與 M4-2(巢狀初始設定)的迷思概念，學生在寫迴圈的題型時，會認為迴圈的初始條件設定某些變數是固定的，包含認為索引值固定為 0、終止值都大於 0、每次變化量都為+1，這樣的迷思會讓學生在編寫迴圈時，無法靈活控制迴圈的運作，或無法掌控迴圈的執行結果，導致學生迴圈內部執行策略有誤、寫不出來。如下圖 4-15 為範例，學生在轉換 while 迴圈為 for 迴圈時，會認為終止值要大於 0，導致該迴圈重複次數有誤。該種迷思概念會以圖 4-16 給予回饋，讓學生清楚區隔初始設定的每項功能與差異，且說明如何利用這些變數做相關的迴圈變化，且配合練習讓學生思考如何用不同的設定，產生一樣的輸出結果。

參考下表 4-18，為控制組與實驗組兩班在 M3-1、M3-2、M4-1、M4-2 的迷思概念發生的次數，且此表示依照時間所排序。雖然依照時間來看，實驗組本身無顯著的進步，但相較控制組而言，實驗組在相關的迴圈問題出現迷思概念(期末測驗裡發生之次數)，而控制組卻大部分還處於寫不出來的狀況，或是不想嘗試的情形。如延長迴圈部分之教學課程，實驗組可能就會有明顯進步趨勢，所以在相關回饋方面，還是有正向效果的。



```
1)    var i,sum=0;
2)    i=1;
3)    while(i<=50){
4)          sum=sum+i;
5)          i++;
6)    }
7)    alert(sum);
```

圖 4-15 M3-2 迷思概念例題



語法

```
for ([initialization]; [test]; [increment])
```

參數

initialization
選擇項，通算式，此通算式只會在迴圈執行之前執行一次。

test
選擇項，Boolean 通算式，如果 *test* 為 true，則執行 *statement*，如果 *test* 是 false，便會結束此迴圈。

increment
選擇項，通算式，遞增通算式會在每次通過迴圈的結尾執行。

statement
選擇項，當 *test* 為 true 時所要執行的一個或多個陳述式，可以是複合陳述式。

圖 4-16 M3-2 迷思概念回饋資料

表 4-18 為課程中 M3-1、M3-2、M4-1、M4-2 迷思概念依照時間排序

控制組	S0	S1	S2	S3	M3-1	M3-2	M4-1	M4-2	實驗組	S0	S1	S2	S3	M3-1	M3-2	M4-1	M4-2
學習單 4	86	6	11	11	0	0	0	10	學習單 4	83	0	16	4	0	2	14	14
線上程式練習 4	25	0	4	1	0	2	0	0	線上程式練習 4	17	0	0	0	0	4	0	0
學習單 5	26	0	0	7	0	5	0	0	學習單 5	20	0	0	0	0	0	0	0
線上程式練習 5	32	0	0	0	0	0	0	2	線上程式練習 5	26	0	0	0	0	0	0	6
期末測驗	188	16	14	86	22	8	5	1	期末測驗	139	15	10	84	24	12	0	9

(4)函式

在函式相關課程中，所發現的迷思概念 M6(函式內變數之呼叫與回傳)，學生會認為函式的變數宣告是相當於全域的效果，但其實是限定在函式內部而已。下圖 4-17 範例中，學生會認為該程式可以正常執行，且結果會輸出”36”，但在函式外是無法對內部的變數做直接的呼叫。這類的迷思概念回饋，會先給予函式基本的定義釐清之補充資料，再連結相關概念的補充。如以 c 語言為例，會涉及變數的生命週期問題，分類為全域變數、區域變數、區塊變數，並解釋與各別舉相關範例，讓學生了解其概念與原因，建立正確的函式應用。

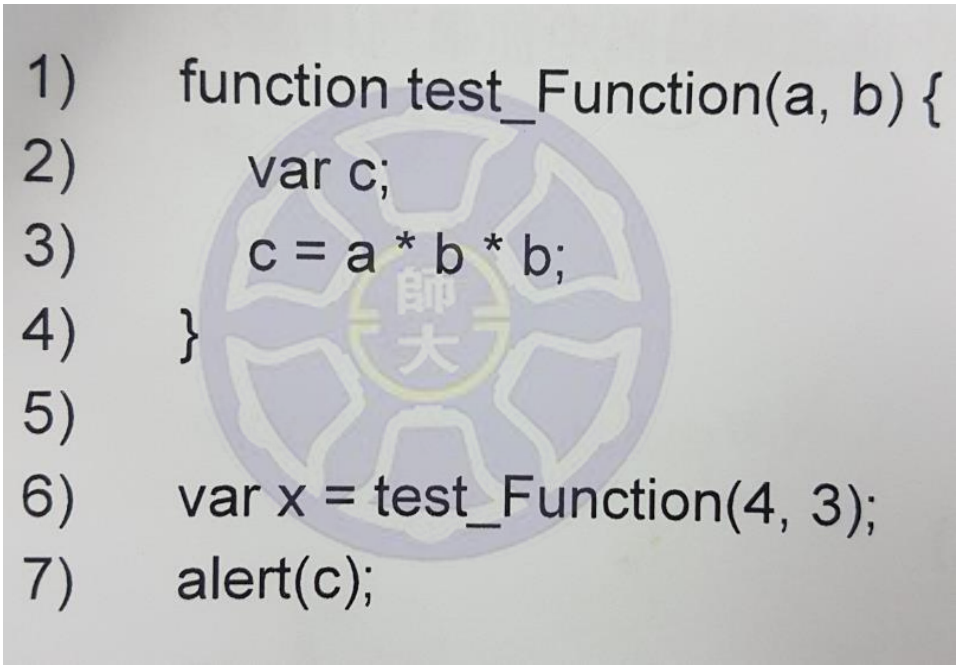


圖 4-17 M6 迷思概念範例

參考下表 4-19，為控制組與實驗組兩班在 M6 的迷思概念發生的次數，且此表示依照時間所排序。在函式相關的題目中，兩班在期末測驗結果顯示，都有進步的趨勢，且實驗組相較控制組明顯的進步，所以可以證實該回饋有正向的影響。

表 4-19 為課程中 M6 迷思概念依照時間排序

控制組	S0	S1	S2	S3	M6	實驗組	S0	S1	S2	S3	M6
學習單 5	26	0	0	7	25	學習單 5	20	0	0	0	11
期末測驗	188	16	14	86	7	期末測驗	139	15	10	84	2

如將實驗組的學生之學習成就分類為高、中、低，並以系統回饋次數與學生高、中、低學習成就進行 ANOVA 分析，結果顯著性為 0.921 大於 0.05，表示實驗組內學生高、中、低學習成就與系統回饋次數上無顯著的差異，如下表 4-20 所示。

表 4-20 為系統回饋次數與學生高、中、低學習成就進行 ANOVA 分析

Group	*Mean*	*n*	*F*	*p*
高成就	77.53	12		
中成就	64.30	8	0.083	0.921
低成就	48.65	12		

3. 對學習者程式設計學習態度之影響

在學生學習態度方面控制組與實驗組兩班無統計上的顯著差異，而實驗後兩班所測的態度分數低於實驗前，成對樣本 t 檢定結果顯示兩班均顯著降低。但可能是因為學生在實驗前對程式語言的認知程度較低，所以在該方面量表填答時，會錯估自己的程式方面能力與低估程式設計的複雜度與延伸應用問題，導致在課程後對程式設計有一定的了解，進而態度分數略為下降。

態度量表五個面向實用性、程式設計之自我體會、動機意願、自信、問題解決能力，控制組與實驗組兩班相較前測，認為程式設計對自己生活中的實用性是下降的，可能是因為現在學的程式語言功能較基礎無法想像其延伸應用面。而在程式設計之自我體會方面相較前測，學生認為程式設計該科目是比原本想像中的還要有挑戰性、更困難、更花時間的。動機意願方面相較前測，學生結果顯示是降低的，學生可能因為作業無法達成或不想花時間在該科目，所以量表填答的狀況都是偏向動機下降的情形。控制組與實驗組兩班相較前測，在自信方面是降低的，代表學生對

自己的程式設計能力感到無信心、認為程式設計是非常困難的、面對程式問題時無法順利完成且成就感不高。在問題解決能力方面相較前測，學生在該能力方面顯示都會偏向尋求他人的幫助，而對於獨立解決程式問題較無把握，所以整體數據有降低。

在半結構式訪談、平台討論與學習單意見留言的結果顯示，控制組與實驗組兩班雖然在後測時對於程式設計的實用性是有降低的情形，但是大部分的學生都還是認為學習程式設計是一定會有所幫助的，不管是邏輯能力上的增強或是對於未來工作上的應用。而在授課過程中控制組學生多數認為時間有些緊湊，對於整體授課、範例、學習單、課後作業、檢討問題的流程上沒甚麼問題。實驗組在授課時間分配上也反映有些不夠，而對於整體授課、範例、學習單、課後作業、回饋教材、反思、檢討問題方面覺得尚可，認為引導式教學與反思是有效果的。

控制組與實驗組在整個學習過程方面，學習單、線上程式編寫上如成功的完成了，學生表示不管多困難還是很有成就感的。而在程式設計概念部分迴圈是大部分學生所認為最困難的單元，且對於迴圈的執行順序上有所障礙，不清楚迴圈目前執行的位置與狀況，在巢狀迴圈的題型也有一樣的情形，在對該方面概念有所困難或疑惑的學生，期中期末測驗會無法作答(S0 空白)，且該情形控制組多於實驗組。

問題解決方面，控制組與實驗組在遇到問題時都是會自己先試著解決，當無法順利完成時就會尋求幫助，問同學、老師、或查資料，且較少學生可以自行獨立完成整個問題。實驗組在引導式教學與概念反思方面，多數人認為如互相討論課程相關概念是有正向幫助的，而學習單引導式的教學模式可以讓學生自我思考核心概念，進而內化其概念並延伸應用與連結其他相關概念，為新知識建立穩固的基礎觀念。

控制組與實驗組在範例練習、課後線上作業方面，認為線上教學系統在練習題的圖形化輸出上效果普通，而程式編譯器的錯誤訊息是很有幫助的，可以快速知道程式是哪一行出錯，但要如何修改或是哪個概念造成又是一個困難點。對於教學系統所提供的回饋提示與老師給予的回饋教材，實驗組多數認為有正向的幫助，且學生表示視覺化的回饋比起單純的文字提示更有效果、更為具體，圖片或動畫的呈現會讓學生更有意願觀看。

在網頁專題製作方面，控制組與實驗組一致認為以實際的生活例子來做練習，可以更有效地引起學生學習動機與興趣，進而提高學習成效與教學效率。但其實要設計這樣的網頁專題課程，是需要前面所先學習的基本程式概念與語法，才有辦法實施與製作完整的專案，且這樣的教學模相較需要更長的時間。

三、 綜合討論

1. 程式設計學習成就與迷思概念影響

在該 JavaScript 迷思概念診斷與矯正的課程中，將實驗組與控制組兩班學習成就(包含學習單、線上實作練習、期中期末測驗、專題)進行 ANCOVA 統計分析，結果顯示控制組與實驗組有顯著性的差異，在學習成就方面實驗組高於控制組，詳細資料請參考上表 4-4。證實了該迷思概念診斷與矯正的教學策略，對於學生學習 JavaScript 語言與正確概念的建立有正向的影響的。在教授程式設計的課程中，如適當的配合教學的回饋方式與內容，不但可以增進學生程式設計的學習成就，也可以提升教學的效率。

程式設計迷思概念方面，學生在變數、判斷式、函式之概念相較迴圈、巢狀迴圈概念有明顯的減少的趨勢，也可以觀察到在不同的程式設計主題中，學生對於程式的執行順序上有迷思概念，且經過這樣的診斷與矯正，最有顯著的減少。而在迴圈、巢狀迴圈概念中，雖然兩班無顯著的迷思概念減少現象，但比照控制組在迴圈方面的學習單、期中期末測驗、線上程式練習的作答狀況，可以觀察出是比實驗組的迷思概念狀況還嚴重(例如:題目空白)，或是迴圈方面的知識建立不完善所導致。

2. 平台對程式設計學習的影響

該實驗教學系統在實驗組的部分有初步診斷回饋的功能，整體迷思概念在變數、條件判斷式、迴圈、函式的回饋中，學生誤解概念或語法功能誤用方面，經過重點概念釐清、以及強調程式功能的回饋，可以減少學生在該主題之迷思概念的發生。而在程式的執行順序與架構方面，給予正確與錯誤的實際範例進行相關解說，可以讓學生一步一步的了解每一行程式碼的功能與用意，進而矯正學生在程式執行順序之迷思概念。

3. 學生學習態度的影響

在以上的結果討論中可以得知，學生在學習程式設計時，所持有的態度會間接影響到學習成就。如在該課程中，有學生本身對於程式設計興致低落，會導致上課意願降低，或進而影響其他同學，也有可能影響到整班的上課風氣，最終導致相關測驗無法正確作答，或是空白(空白並非迷思概念)。從前後測的態度量表的動機意願面相結果顯示，也有可能是學生對於上課內容，隨著進度漸漸的感到概念逐漸複雜且無成就感，所以動機面相降低導致學習意願降低。而在這方面的學生主要建議為，於相對複雜的核心概念與課程內容上，可以給予充足的時間，讓學生多次的練習與充足的時間自行反思，再配合相關的矯正教材會更有效果。

第五章　結論與建議

本研究利用迷思概念診斷與矯正來輔助學生建立正確的程式設計概念、或減少迷思概念的產生，而本章會統整本研究的結果與討論，針對該種教學模式給予結論與建議。本章共分為兩個小節，第一節依據研究結果彙整出結論；第二節則針對未來進行程式設計迷思概念方面之研究提出些許建議。

第一節　結論

本研究利用迷思概念診斷與矯正，並配合回饋之教學模式，來輔助學生建立正確的程式設計概念、或減少迷思概念的產生。透過課堂學習單、期中期末測驗、專題成果、線上程式練習、平台與學習單反思討論、態度量表、半結構式訪談的量化與質性資料結果進行分析討論。根據上述研究討論結果，本研究發現以下結論：

一、　初學者在程式之執行順序上常有迷思概念

從本研究結果可以觀察到，初學者不管在哪一個課程主題中，都會對程式的執行順序有所迷思概念。例如在 if else、switch case 判斷式、迴圈、巢狀迴圈，初學者常有程式執行順序的迷思概念為：

(1) 程式指令可以同時執行。

(2) 程式指令可以由下而上執。

(3) 程式指令可以跳行的執行。

而以上的執行順序與想法並不符合正確的程式執行狀況。程式執行順序之矯正部分，可以利用一步步拆解的方式，釐清每一步驟的功能與原因，或利用步驟圖文解說的方式，讓學生體會到該順序的重要性，並有效地矯正此迷思概念(Bayman & Mayer, 1983; Du Boulay, 1986)。

二、　本研究迷思概念診斷與矯正教學模式對於學生學習程式設計是有幫助的

本研究發現，如學生在某主題上有迷思概念，可以配合教學策略的引導反思與回饋教材，讓學生在有迷思概念之處輔助正確概念的建立，以及釐清類似與易誤用的概念，進而矯正迷思概念。且從文獻探討中可以得知，程式設計的迷思概念對於學習該科目是有負面影響的，而針

對迷思概念進行矯正，對於程式設計學習成就是有正向效果的(Lahtinen et al., 2005; D. Perkins & Martin, 1986; D. N. Perkins & Simmons, 1988; Revival, 2014)。對於學生在學習單、期中期末測驗、專題成果、線上程式練習時，所發生的狀況進行迷思概念診斷，在配合教學的過程給予適當的回饋。讓學生可以具體了解程式的架構與執行狀況，並釐清類似與易誤用的概念，以及連結相關學習的範例與概念，建立穩健的基礎邏輯(Brinko, 2008; Hattie & Timperley, 2007)。

第二節　建議

根據本次研究之迷思概念診斷與矯正的教學模式進行過程，所面臨的相關問題，提供以下教學之建議，期望未來若有相關程式設計迷思概念之教學研究時，在資料蒐集與實際教學運用時更加順利。

一、　迷思概念項目建置更完善

可以將迷思概念的主題與內容建置的更廣闊，加入非初學程式語言的迷思概念部分，或是可以連結、應用到其它的程式語言。而迷思概念的項目可以有更明確的判斷原則與相關範例，利於後續的診斷作業。

二、　系統之迷思概念診斷

系統的自動診斷功能，可以配合更多元化的題型，並能正確的診斷迷思概念。可以偵測使用者輸入的全部程式碼，再篩選出關鍵之程式碼，進行一系列的迷思概念診斷流程。不過由於錯誤的語法或迷思概念，導致程式無法執行或輸出結果不正確的排列組合狀況非常多，所以在何時判定語法與何種診斷迷思概念的模式影響甚大。

三、　矯正回饋資料

矯正資料的完善建置，根據每個迷思概念都可以有多元的矯正資料，包含圖文、動畫、概念圖、流程圖，如可以配合相關的概念，且設計出針對此概念的動手實作輔助教材，也可以具體讓學生體會到該概念的重點。且矯正資料的給予時機，都盡量在發生此迷思概念當下，即時的回饋給學習者是有較好的成效。

參考文獻

中文部份

陳建偉. (2008). 高三學生液體界面現象迷思概念之研究 A Study of Misconceptions of Liquid Interface Phenomenon of the 12th Graders (pp. 1–120).

彭泰源, & 張惠博. (1999). 國小五年級學童力與運動概念學習之研究.

廖焜熙. (2001). 理化科學概念及過程技能之研究回顧與分析.

英文部份

0zdener, N. (2008). A comparison of the misconceptions about the time-efficiency of algorithms by various profiles of computer-programming students. *Computers and Education*, *51*(3), 1094–1102. http://doi.org/10.1016/j.compedu.2007.10.008

Akbaş, Y., & Gençtürk, E. (2011). The effect of conceptual change approach to eliminate 9th grade high school students' misconceptions about air pressure. *Kuram ve Uygulamada Egitim Bilimleri*, *11*(4), 2217–2222.

Anseel, F., Lievens, F., & Schollaert, E. (2009). Reflection as a strategy to enhance task performance after feedback. *Organizational Behavior and Human Decision Processes*, *110*(1), 23–35. http://doi.org/10.1016/j.obhdp.2009.05.003

Baker, B. S. (1992). A Program for Identifying Duplicated Code. *Computing Science and Statistics*, *24*, 49–57.

Bayman, P., & Mayer, R. E. (1983). A diagnosis of beginning programmers' misconceptions of BASIC programming statements. *Communications of the ACM*, *26*(9), 677–679. http://doi.org/http://doi.acm.org/10.1145/358172.358408

Bonar, J., & Soloway, E. (1985). Preprogramming Knowledge: A Major Source of Misconceptions in Novice Programmers. *Human-Computer Interaction*, *1*(2), 133–161. http://doi.org/10.1207/s15327051hci0102_3

Booth, J. L. (2011). Why Can't Students Get the Concept of Math ?, (October), 31–35.

Branch, W. T., & Paranjape, A. (2002). Feedback and Reflection: Teaching Methods for Clinical Settings. *Academic Medicine : Journal of the Association of American Medical Colleges*, *77*(12 Pt 1), 1185–1188. http://doi.org/10.1097/00001888-200212000-00005

Brandt, C. (2008). Integrating feedback and reflection in teacher preparation. *ELT Journal*, *62*(1), 37–46. http://doi.org/10.1093/elt/ccm076

Brinko, K. T. (2008). The Practice of Giving Feedback to Improve Teaching. *The Journal of Higher Education*, *64*(5), 574–593.

Britos, P., Rey, E. J., Rodríguez, D., & García-martínez, R. (2008). Work in Progress - Programming Misunderstandings Discovering Process Based On Intelligent Data Mining Tools, 1–2.

Chen, C.-L., Cheng, S.-Y., & Lin, J. M.-C. (2012). A study of misconceptions and missing conceptions of novice Java programmers, 1–7. Retrieved from http://worldcomp-proceedings.com/proc/p2012/FEC2866.pdf

Chen, M.-P. (2007). The Effects of Instructional Approach and Programming Tools on Novices' Learning Computer Programming. *Journal of Taiwan Normal University: Science Education*, *52*(1&2), 1–21. http://doi.org/10.6300/JNTNU.2007.52.01

Chick, H. L., & Baker, M. (2005). Investigating Teachers'Responses To Student Misconceptions. *Psychology of Mathematics Education*, *2*, 249–256. Retrieved from http://www.emis.ams.org/proceedings/PME29/PME29CompleteProc/PME29Vol2Adl_Fre.pdf#page=255

Danielsiek, H., Paul, W., & Vahrenhold, J. (2012). Detecting and Understanding Students' Misconceptions Related to Algorithms and Data Structures. *ACM Technical Symposium on Computer Science Education (2012)*, 21–26. http://doi.org/10.1145/2157136.2157148

Du Boulay, B. (1986). Some Difficulties of Learning to Program. *Journal Of Educational Computing Research*, *2*(1), 57–73. http://doi.org/10.2190/3LFX-9RRF-67T8-UVK9

Eckerdal, A., & Thuné, M. (2005). Novice Java programmers' conceptions of "object" and "class", and variation theory. *ACM SIGCSE Bulletin*, *37*(3), 89. http://doi.org/10.1145/1151954.1067473

Goldman, K., Gross, P., Heeren, C., Herman, G., Kaczmarczyk, L., Loui, M. C., & Zilles, C. (2008). Identifying important and difficult concepts in introductory computing courses using a delphi process. *ACM SIGCSE Bulletin*, *40*(1), 256. http://doi.org/10.1145/1352322.1352226

Hancock, C. H. (1940). An evaluation of certain popular science misconceptions. *Science Education*, *24*(4), 208–213. http://doi.org/10.1002/sce.3730240409

Hattie, J., & Timperley, H. (2007). The Power of Feedback. *Review of Educational Research*, *77*(1), 81–112. http://doi.org/10.3102/003465430298487

Holland, S., Griffiths, R., Woodman, M., Hall, W., Keynes, M., & Kingdom, U. (1997). Avoiding Object Misconceptions. *SIGCSE Technical Symposium on Computer Science Education*, *29 Issue 1*, 131–134.

Hwang, W. Y., Shadiev, R., Wang, C. Y., & Huang, Z. H. (2012). A pilot study of cooperative programming learning behavior and its relationship with students' learning performance. *Computers and Education*. http://doi.org/10.1016/j.compedu.2011.12.009

Kaczmarczyk, L. C., Petrick, E. R., East, J. P., & Herman, G. L. (2010). Identifying student misconceptions of programming. *Proceedings of the 41st ACM Technical Symposium on Computer Science Education - SIGCSE '10*, 107–111. http://doi.org/10.1145/1734263.1734299

Lahtinen, E., Ala-Mutka, K., & Järvinen, H.-M. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin*, *37*(3), 14. http://doi.org/10.1145/1151954.1067453

Lwo, L. S., Chang, C. C., Tung, Y. P., & Yang, W. C. (2013). Marine science literacy and misconceptions among senior high school students. *Journal of Research in Education Sciences*, *58*(3), 51–83. http://doi.org/10.6209/JORIES.2013.58(3).03

Moons, J., & De Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education*, *60*(1), 368–384. http://doi.org/10.1016/j.compedu.2012.08.009

Morales, Z. A. (2014). Analysis of Students ' Misconceptions and Error Patterns in Mathematics : The Case of Fractions, 1–10.

Perkins, D., & Martin, F. (1986). Fragile knowledge and neglected strategies in novice programmers. In *Empirical studies of programmers* (pp. 213–229). Retrieved from http://books.google.com/books?hl=en&lr=&id=sswoYivNQVUC&oi=fnd&pg=PA213&dq=Fragile+Knowledge+and+Neglected+Strategies+in+Novice+Programmers&ots=afQcU6lc_F&sig=z1XL1J_F7uXpWMQ6dATyayoRwao

Perkins, D. N., & Simmons, R. (1988). Patterns of Misunderstanding: An Integrative Model for Science, Math, and Programming. *Review of Educational Research*, *58*(3), 303–326. http://doi.org/10.3102/00346543058003303

Putnam, R. T., Sleeman, D., Baxter, J. A., & Kuspa, L. K. (1986). A Summary of Misconceptions of High School Basic Programmers. *Journal of Educational Computing Research*, *2*(4), 459–472. http://doi.org/10.2190/FGN9-DJ2F-86V8-3FAU

Ragonis, N., & Ben-Ari, M. (2005). A long-term investigation of the comprehension of OOP concepts by novices. *Computer Science Education*, *15*(February 2015), 203–221. http://doi.org/10.1080/08993400500224310

Revival, G. (2014). What is a misconception? Common Misconceptions in Basic Mathematics, 1–46. http://doi.org/10.1038/sj.embor.7400842

Rittle-Johnson, B., & Schneider, M. (2015). Developing conceptual and procedural knowledge of mathematics. *Oxford Handbook of Numerical Cognition*, 1118–1134. http://doi.org/10.1093/oxfordhb/9780199642342.013.014

Running, D. M., Ligon, J. B., & Miskioglu, I. (2016). Individual Differences in Perception of Applied Music Teaching Feedback. *Journal of Composite Materials*, *33*(10), 928–940. http://doi.org/0803973233

Rutar, N. (University of M., Almazan, C. (University of M., & Foster, J. (University of M. (2004). A comparison of bug finding tools for Java. *International Symposium on Software Reliability Engineering (ISSRE)*, 245–256. http://doi.org/10.1109/ISSRE.2004.1

Sanders, K., & Thomas, L. (2007). Checklists for grading object-oriented CS1 programs. *ACM SIGCSE Bulletin*, *39*(3), 166. http://doi.org/10.1145/1269900.1268834

Sekiya, T., & Yamaguchi, K. (2013). Tracing quiz set to identify novices' programming misconceptions. *Proceedings of the 13th Koli Calling International Conference on Computing Education Research*, 87–95. http://doi.org/10.1145/2526968.2526978

Sirkiä, T., & Sorva, J. (2012). Exploring programming misconceptions: An analysis of student mistakes in visual program simulation exercises. *Proceedings - 12th Koli Calling International Conference on Computing Education Research, Koli Calling 2012*, 19–28. http://doi.org/10.1145/2401796.2401799

Spohrer, J. C., & Soloway, E. (1986). Novice Mistakes : Are the Folk Wisdoms Correct ? *Communications of the ACM*, *29*(7), 10.

Taylor, A. K., & Kowalski, P. (2012). Students ' misconceptions in Psychology : How you ask matters … sometimes. *Journal of the Scholarship of Teaching and Learning*, *12*(3), 62–77. Retrieved from http://search.proquest.com/docview/1314328305?accountid=13042

Treagust, D. F. (1988). Development and use of diagnostic tests to evaluate students' misconceptions in science. *International Journal of Science Education*, *10*(2), 159–169. http://doi.org/10.1080/0950069880100204

Van Den Boom, G., Paas, F., Van Merriënboer, J. J. G., & Van Gog, T. (2004). Reflection prompts and tutor feedback in a web-based learning environment: Effects on students' self-regulated learning competence. *Computers in Human Behavior*, *20*(4), 551–567. http://doi.org/10.1016/j.chb.2003.10.001

附錄一 程式設計迷思概念蒐集表格

以下為本研究所蒐集的相關程式設計迷思概念的列表，後續整合編修成初學者常發生的迷思概念，並配合設計課程內容。

語言 或 環境	類別	迷思概念種類	迷思概念描述
Ruby 互動式測驗題	combining conditional and loop	Change Variable In Condition	Interpret variables in the conditional part of conditionals as control variables.
		Change Variable In Condition 2	Interpret variables in the conditional part of conditionals as the end value of the enclosing loop.
		Change Variable In Loop	Interpret variables in conditionals as control variables.
		Change Variable In Loop 2	Interpret variables in conditionals as the end value of the enclosing loop.
		Change Variable In Simple Loop	Interpret variables in the body of conditionals as control variables.
		Neglect(ignore) For Loop	Ignore loop when the body has no control variables.
		Regard As Array	Interpret substitution to keep the previous values.
Python UUhistle (VPS)	assignment, control structures, and/or booleans	Inverted assignment	The student assigns the value of the left-hand variable to the right-hand side variable, rather than the other way around.
		Wrong branch	Even though the conditional evaluates to False, the student jumps to the ‘then’ clause.
		Wrong False	As soon as the conditional evaluates to False , the student proceeds to return False from the function.
		Conditional into accessed variable	After evaluating the expression, the student assigns its value to divisor.
		Conditional into loop control variable	After evaluating the expression, the student assigns its value to i.
	functions object-oriented topics	Executing function instead of defining it	The def command defines a new function: the student is supposed to store the function (object) in memory.Instead, the student starts executing the function body.
		Unevaluated parameters	The student tries to start the function call before they have evaluated the sum.
		Parameter in the wrong frame	The student creates parameter variables in the caller’s frame, not in the callee’s.
		Misplacing return value	The student creates the variable result in calculate’s frame instead of returning the value.
		Return value into variable	The student assigns the result of the multiplication back into the variable intermediate.

		Failing to store return value	After returning a value, the student does not store assign the return value to the caller's local variable.Instead, they fetch the function call to the evaluation area again.
		Assignment copies object	The student creates a new object rather than copying a reference.
		Failing to create second object of a class	A new object should be created, but the student instead makes a copy of a a reference to an existing object.
		Local variable becomes instance variable	Instead of creating a new local variable to store the reference, the student creates an instance variable for the new object.
		Instance variable becomes local variable	Instead of creating a new instance variable for the object, the student creates a new local variable.
		Method call without recipient	The student tries to call the method fuel from the Car class before accessing the variable for a reference to the target object.
		Assigning reference to uninitialized object	The student assigns the reference to the variable before calling the __init__ method.
		Returning self-reference instead of dereferencing it	The student returns self rather than the value of the instance variable __profession.
CS1 programing Java	memory models, and data assignment	T1: Students misunderstand the relationship between language	Semantics to semantics (Student applies real-world semantic understanding to variable declarations.)
		elements and underlying memory usage.	All Objects same size (Student thinks all Objects are allocated the same amount of memory regardless of definition and instantiation.)
			Instantiated no memory allocation (Student thinks no memory is allocated for an instantiated Object.)
			Uninstantiated memory allocation (Student thinks memory is allocated for an uninstantiated Object.)
			Off by 1 array construction (Student thinks an array's construction goes from 0 to length, inclusive.)
			Primitive no default (Student thinks primitive types have no default value.)
			Primitives don't have memory (Student thinks primitives without a value have no memory allocated.)
		T2: Students misunderstand the process of while loop operation.	
		T3: Students lack a basic understanding of the Object concept	
		T4: Students cannot trace code linearly.	

Java(Greenfoot)	object- oriented concepts	Static data members vs. constant data members	Confused the properties of static data members with constant data members
	certain programming constructs	Classes vs. Objects	Failed to recognize that classes were actually user-defined data types that could be used to define variables just like built-in data types such as integer and Boolean
		Constructors	(1) Erroneously thought that constructors of all classes defined in a project were invoked automatically when the project was opened
			(2) Defined the constructor(s) of a class as its private member(s) when they should be public members instead
			(3) Did not realize that constructors, as a special type of method, did not require a return type
		Arguments passing for method calls	Confused formal parameters with actual parameters
		Polymorphism	(1) Mistakenly considered that declaring one superclass variable instead of two subclass variables means declaring two variables at the same line
			(2) Mistakenly considered that referencing two subclass objects successively by the same superclass variable means substituting two superclass objects for two subclass objects
		Access modifiers of methods	Mistakenly considered that a private method could not invoke methods defined in other classes
OOP Java(BlueJ)	1.Object vs. Class	1.1: The nature of a class as a template	1.1.1 Difficulties in understanding the static aspect of the class definition.
			1.1.2 Difficulties in understanding that a method can be invoked on any object of the class.
			1.1.3 Misconception: You can define a method that doesn't access any attribute.
			1.1.4 Misconception: You can define a method that adds an attribute to the class.
			1.1.5 Difficulties in understanding the classification of methods that we used (constructors, mutators, accessors, ''others'').
			1.1.6 Misconception: You can invoke a method on an object only once.
		1.2: Connections between objects and classes.	1.2.1 Misconception: A class is a collection of objects, rather than a template for creating objects.
			1.2.2 Misconception: You can define a non-constructor method to create a new object.
			1.2.3 Misconception: You can define a method that replaces the object itself.
			1.2.4 Misconception: You can define a method that destroys the object itself.
			1.2.5 Misconception: You can define a method that divides the object into two different objects.
		1.3: Object creation	1.3.1 Difficulties in understanding the process of creating an object.

		1.4: Identification of objects.	1.4.1 Misconception: Two objects of the same class cannot have equal values for their attributes.
			1.4.2 Misconception: Two objects can have the same identifier if there is any difference in the values of their attributes.
			1.4.3 Misconception: An attribute value can be used as the object identifier.
			1.4.4 Misconception: The object identifier is one of the object's attributes.
			1.4.5 Difficulties recognizing an object due to multiple representations (the set of values of attributes, the BlueJ icon and the graphical rendering).
	2.Instantiation and Constructors	2.1: General understanding of instantiation.	2.1.1 Misconception: There is no need to invoke the constructor method, because its definition is sufficient for object creation.
			2.1.2 Misconception: Constructors can include only assignment statements to initialize attributes.
			2.1.3 Misconception: Instantiation involves only the execution of the constructor method body, not the allocation of memory.
			2.1.4 Misconception: Invocation of the constructor method can replace its definition.
		2.2: Understanding instantiation in a composed class.	2.2.1 Misconception: If objects of the simple class already exist, there is no need to create the object of the composed class.
			2.2.2 Misconception: Creation of an object of a composed class automatically creates objects of the simple class that appear as attributes of the composed class.
			2.2.3 Difficulties in understanding where objects of the simple class are created,before the creation of the object of the composed class.
		2.3: Understanding instantiation is affected by its implementation in the programming language.	2.3.1 Difficulties understanding the empty constructor.
			2.3.2 Difficulties understanding objects if their attributes are not explicitly initialized.
			2.3.3 Initializing an attribute with a constant as part of its declaration causes confusion in distinguishing between a class and an object.
			2.3.4 Initializing an attribute with a constant within the constructor declaration causes confusion in distinguishing between a class and an object.
			2.3.5 Misconception: If the attributes are initialized in the class declaration there is no need to create objects.
	3.Simple vs. Composed Classes	3.1: Understanding encapsulation.	3.1.1 Misconception: An object cannot be the value of an attribute.
			3.1.2 Misconception: The attributes of the composed class include all the attributes of the objects of a simple class, instead of the objects themselves.

			3.1.3 Misconception: The attributes of the composed class include all the attributes of the objects of a simple class, in addition to the objects themselves.
			3.1.4 Misconception: Methods that are declared in the simple class have to be declared again in the composed class for each of the simple objects.
			3.1.5 Misconception: There is no need for mutators and accessors for attributes that are of the simple class within the composed class.
			3.1.6 Misconception: To change the value of an attribute of an object of a simple class that is the value of an attribute in an object of a composed class, you need to construct a new object.
			3.1.7 Misconception: Methods can only be invoked on objects of the composed class, not on objects of the simple class defined as values in its attributes.
		3.2: Understanding modularity.	3.2.1 Misconception: After a composed class is defined, new methods cannot be defined in the simple class.
			3.2.2 Methods from the simple class are not used; instead, new equivalent methods are defined and duplicated in the composed class.
			3.2.3 Methods in different classes are not distinguished if they have the same signature.
		3.3: The class as a collection of objects.	3.3.1 Misconception: Objects of a simple class, used as values of the attributes of a composed class, have to be identical.
			3.3.2 Misconception: In a composed class you can develop a method that adds an attribute of a simple class to the composed class.
			3.3.3 Misconception: In a composed class you can develop a method that removes an attribute of a simple class from the composed class.
		3.4: Understanding information hiding.	3.4.1 Misconception: Attributes of the simple class must be directly accessed from the composed class instead of through an interface.
		3.5: Personification.	3.5.1 Misconception: Attributes in a simple class are automatically replicated in the composed class by transferring its meaning.
		3.6: Understanding invocation on the implicit object.	3.6.1 Misconception: A method must always be invoked on an explicit object. (We did not teach the use of an explicit ‘‘this,’’ so the students invented an imaginary object upon which to invoke the method.)
	4.Program Flow	4.1: Understanding executions of methods.	4.1.1 Misconception: Methods are executed according to their order in the class definition.
			4.1.2 Misconception: Every method can be invoked only once.
			4.1.3 Difficulties distinguishing when there is a need to explicitly write the identifier of the object.
			4.1.4 Difficulties understanding the influence of method execution on the object state.

			4.1.5 Difficulties understanding the invocation of a method from another method.
		4.2: Understanding data flow.	4.2.1 Where do the values of the parameters come from?
			4.2.2 To where does the return value of a method go?
		4.3: Things happen with no cause.	4.3.1 Misconception: Objects are created by themselves.
			4.3.2 Misconception: Attributes values are updated automatically according to a logical context.
			4.3.3 Misconception: The system does not allow unreasonable operations.
		4.4: ‘‘How does the computer know?’’	4.4.1 How does the computer know what the class attributes and methods are?
			4.4.2 How does one class recognize another?
		4.5: Difficulties in understanding the overall flow of execution: What happens and when?	
問卷調查多種程式語言 C++ Java Pascal other languages		What kind of issues you feel difficult in learning programming?	Using program development environment
			Gaining access to computers/networks
			Understanding programming structures
			Learning the programming language syntax
			Designing a program to solve a certain task
			Dividing functionality into procedures
			Finding bugs from my own program
		Which programming concepts have been difficult for you to learn?	Variables (lifetime, scope)
			Selection structures
			Loop structures
			Recursion
			Arrays
			Pointers, references
			Parameters
			Structured data types
			Abstract data types
			Input/output handling
			Error handling
			Using language libraries
Java2D and Swing	OO misconceptions	Instance / class conflation	Classes identical except for class names
			Classes identical except for property values that could be set in constructors

			Classes identical except for very minor changes
			Classes rarely/never instantiated more than once
			Superclass/subclass used instead of class/instance
		Problems with linking and interaction	All code in a single class
			Code in single class instead of composite class and parts
			Classes defined but not linked in
			Work in methods exclusively done by assignment
			Objects passed as parameters but not used
		Class / collection conflation	Class whose instances all model elements of some collection
		Problems with abstraction	Classes identical except for property values that could be set Problems with abstraction in constructors
			Duplicate code not factored into superclass
			Duplicate method signatures in different classes not defined as interface
		Problems with modelling	Classes joined that should be separate or vice versa
			Classes do not correspond to objects in domain
			Failure to use inheritance to model hierarchical domain
		Classes just text	Failure to delete irrelevant code when adapting
		Identity/attribute conflation	Variables with names that are really values of attributes
		Objects are only data records	No classes with methods other than constructors or accessor/mutators
		Problems with encapsulation	Methods/variables in different classes always have different names

附錄二 運算思維測驗題目

在本研究中的運算思維測驗是在實驗前所實施的，控制組與實驗組兩班題目相同，以供後續 ANCOVA 統計分析。

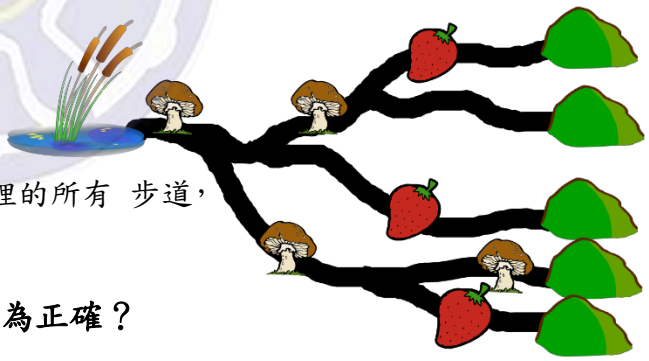
國際運算思維能力

1.樹林漫步

樹林裡有五條步道，每條步道的 ft 發點皆為海貍池塘（下圖左側）終點分別為樹林裡五座 ft 的 ft 頂（下圖右側）。每條步道都包含一些路段，所謂「路段」是兩個 分叉點中間的路，或者是最後一個分叉點和終點之間的路。如下圖所示，有些路 段可能會隸屬於不只一條步道。步道上可能發現蘑菇和（或）草莓。

我們可以用式子描述這座樹林；例如下面的四個式子 A、E、G 和 F 分別代表不同 的意思：

A：在樹林裡的所有步道 E：在樹林裡至少有一條步道 G：其每個路段都有……F：其某個路段至少會有一顆……



舉例說明「AG（蘑菇和草莓）表示「在樹林裡的所有 步道，每個路段都有蘑菇和草莓。」

請問用以下哪個式子描述這個樹林較為正確？

A) AF（草莓）

B) AG（草莓或蘑）

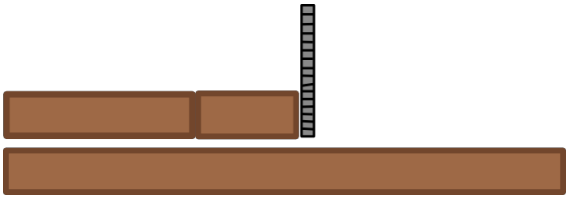
C) EG（草莓或蘑）

D) EG（草莓）

答案:C

2.切割水管

X 教授手上有長度 4 公尺、7 公尺及 100 公尺的水管各一條。現在有個工程需要一條 13 米的水管，但是 X 教授找不到捲尺和標記筆，所以他切割水管時，只能 利用現有的水管長度來當作測量的依據。如下圖：



X 教授想要盡可能的保留 100 公尺水管的長度，留下愈長的水管愈好。

請問 X 教授切割出一條 13 公尺的水管之後，原本 100 公尺的水管最多剩下多少 公尺呢？

A) 87

B) 85

C) 82

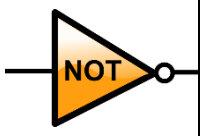
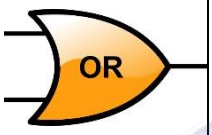
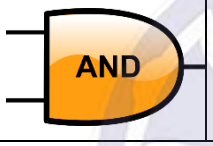
D) 81

答案:B

3.邏輯電路

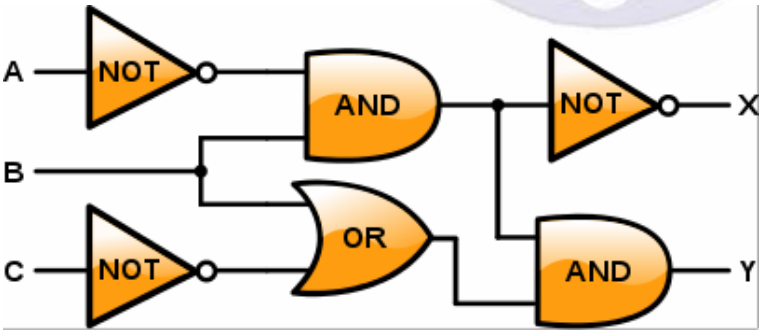
邏輯閘（Logic gates）左方為輸入，右方為輸 ft；邏輯閘可能有 1 或 2 個輸入，

但不管哪種邏輯閘都只會有 1 個輸 ft，而且不是 ON 就是 OFF。每個邏輯閘的輸 ft 都取決於由 ON 或 OFF 所組成的輸入。三種邏輯閘規則如下圖說明：

NOT 閘		如果輸入為ON ,則輸出為OFF。 如果輸入為OFF，則輸出為ON。
OR 閘		除非兩個輸入均為OFF，否則輸出為ON。
AND 閘		只有在兩個輸入均為ON 時，輸出為ON。

下圖中，如果輸入 A 為 OFF，而輸入 B 及 C 均為 ON，請問輸出 X 及輸出 Y 各

是什麼？



A) X 為 OFF，Y 為 OFF

B) X 為 OFF，Y 為 ON

C) X 為 ON，Y 為 OFF

D) X 為 ON，Y 為 ON

答案:B

4.矩形

小機器人的專長是畫矩形，以下是操作它的指令：

Orange: 畫一單位長的橘色直線

Black: 畫一單位長的黑色直線

Turn: 向右轉 90 度

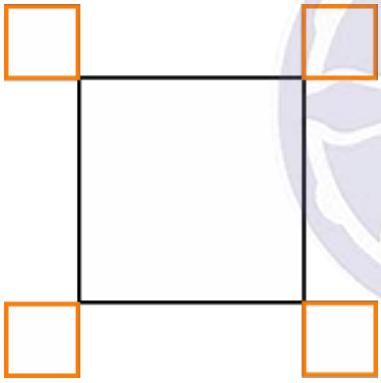
以下是小機器人能接收的指令規則：

指令 A, 指令 B：先執行指令 A，再執行指令 B

n × B：重覆執行指令 B 共 n 次

n × ( )：重覆依序執行括號內的指令共 n 次

機器人想畫出下列橘黑相間的圖案。而下列四個指令中，只有一個指令無法畫出 這個圖案。請問是哪一個？



A)4 x (2 x (Orange , Turn) , Orange , 3 x Black , Orange , Turn)

B)4 x (3 x Black , 3 x (Orange , Turn) , Orange)

C)4 x (2 x (Orange , Turn) , 3 x Black , 2 x (Orange , Turn))

D)4 x (Black , 3 x (Orange , Turn) , Orange , 2 x Black)

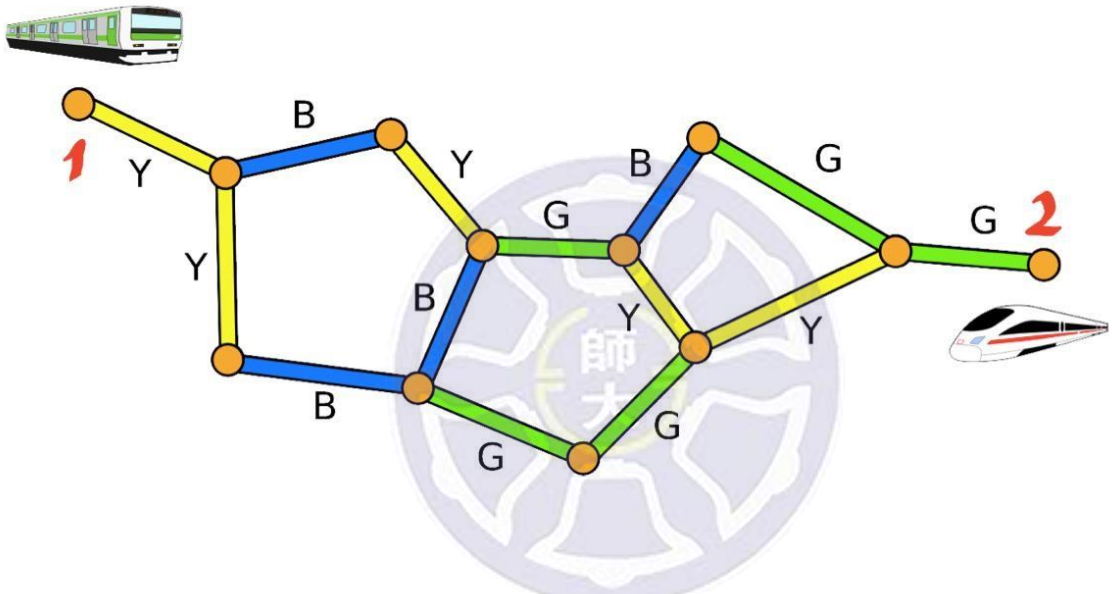
答案:C

5.快速直達車

兩列火車分別從車站 1 和 2 往彼此方向 ft 發。下面的地圖標示 ft 它們之間的所有車站以及彩色的鐵路。

只要其中一列火車在移動，另外一列火車就必須在車站等待。當火車移動時，其 行經鐵路的色彩會被記錄下來；遺憾的是，從記錄中我們不知道是哪部火車在移 動。

例如，若記錄了 BG，可能是有一列火車先經過藍色鐵路再經過綠色鐵路，但也可 能是有一列火車經過藍色鐵路，緊接著另外一列火車經過綠色鐵路。



如果兩列火車最後相遇了，下面哪個可能是兩列火車相遇前記錄下的顏色？

A) GYGBGYBB

B) YYBYGGBG

C) GBYBYGY

D) YBBYBYY

答案:A

6.飯店房間號碼

有一間飯店，裡面所有的房間號碼都是由左到右的兩個數字組成；第一個數字代表房間所在樓層，第二個數字代表從電梯到房間的步行距離。有個顧客上門想要 住宿，但他不太喜歡走路，所以步行距離愈短的房間愈好。若有多間房間的步行 距離相同，這位顧客比較喜歡樓層低一點的房間。

請將下列飯店空房間的號碼，依照這位顧客的喜好排列：

12, 25, 11, 43, 22, 15, 18, 31, 44, 52

越左邊的號碼，對應這位顧客越喜歡的房間。請問下列何者的順序正確？

A) 18, 15, 12, 11, 25, 22, 31, 44, 43, 52

B) 52, 43, 44, 31, 22, 25, 11, 12, 15, 18

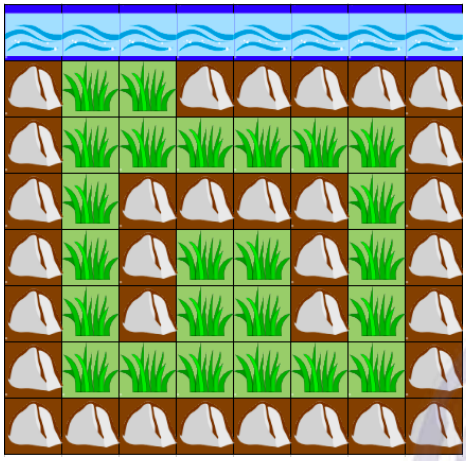
C) 11, 31, 12, 22, 52, 43, 44, 15, 25, 18

D) 11, 12, 15, 18, 22, 25, 31, 43, 44, 52

答案:C

7.水壩

有隻海貍在河中蓋了一個水壩當作自己的家， 結果使得水流阻塞並造成河水氾濫到岸邊的土地上。如下圖所示，河水( ) 每小時會往四方（上、下、左、右）擴散，並各淹沒一格的草地（ ）而地圖上標示為岩石地形的土地( ) 則能 阻隔水流擴散而不被淹沒。



請問幾個小時之後，地圖上的草地全部會被河水淹沒？

A) 9 小時

B) 10 小時

C) 11 小時

D) 12 小時

答案:C

8.海貍約翰的秘密訊息

海貍們採用特殊的加密法在海貍網路裡傳送祕密訊息。為了不讓其他不相關的海貍看懂訊息內容，每隻海貍會選一個神祕數字；而原文中的每個英文字母，皆會 依照神祕數字向後推算，替換成另一個字母。

舉例來說，海貍安娜的神祕數字是 3；當她想送個祕密訊息，所有訊息裡的字母會換成該字母往後推移 3 個的字母，所以 A 會被 D 取代，B 會被 E 取代，以此類推。若是取代的字母順序已經到字母表結尾，那就回到字母表的開始字元 A 繼續 算；因此， Y 就用 B 代替，因為 Y→Z→A→B。

安娜傳送祕密訊息時的字母對照表，如下圖所示：

你的好朋友海貍約翰想要傳祕密訊息給你；雖然你非常急切地想知道海貍約翰送 來的訊息內容，但你卻忘了他所採用的神祕數字。好在你從他之前傳來的訊息得 知，若原來的訊息是 USE，則加密後的訊息會變成 BZL。

請問約翰送給你的祕密訊息“WHYAFHAZPE”，是想要約你做什麼事情? 請依 你解出的訊息中是否出現給定之英文字判斷。

A) 熬夜通宵玩電玩（GAME）

B) 跟他一起去玩足（SOCCER）

C) 晚上去參加宴會(PARTY）

D) 星期五到外面去跑步（RUN）

答案:C

9.奇怪的單字

海貍們會運用只由 a,b,c 三個字母所組成的奇怪單字，並這些單字進行下列三種指令操作：

指令 1：將每一個字母 a 用 aa 取代。
指令 2：將某些字母 b 用字母 c 取代。
指令 3：在單字的任意位置插入字母 c。

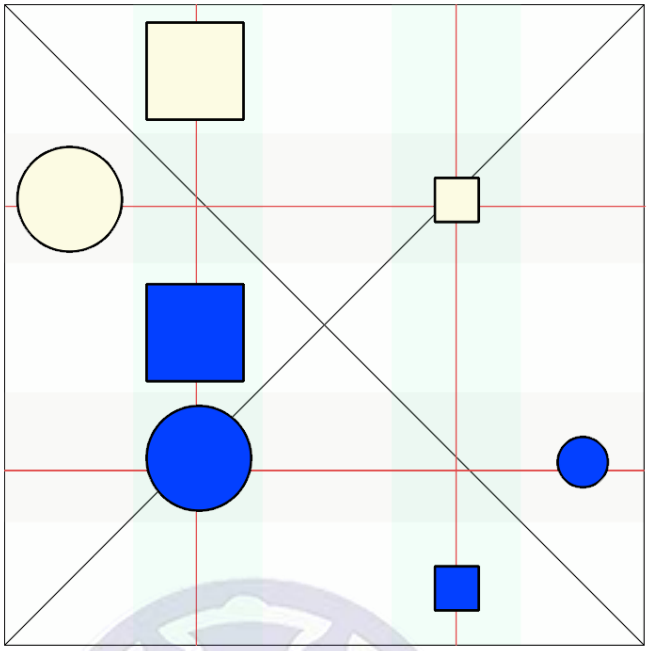
舉例來說，若原始單字為 abbbcaab，先對其執行指令 1，海貍們會得到新單字：**aa**bbbc**aaaa**b；接著對這個新單字再執行指令 2，便得到 aab**c**bcaaaab 第二個 新單字；最後，海貍們再對第二個新單字執行指令 3，就會得到 aabcbcaa**c**aab。 而海貍們可任意選擇是否執行這三個指令，或是指令的先後順序。

如果有一個起始單字為 aabbbbaabbccbbabbc，則下列那一個新單字無法經由 上述的三種指令產 Th？

A) abbbaabbccbaaaabbc
B) aaaaccbbaacaaccccbbaabbc
C) caaccccaaccccccacccc
D) acacbcbcbcbcacacbcbcccccbcbcacbcbcc

答案:A

10.圓形與方形



小明與小華在玩一個叫做「真或假」的遊戲，遊戲一開始時小明放了七張牌在桌上，接下來小明藉由描述這七張牌的形狀（圓形或方形）顏色（藍色或白色） 尺寸（大型牌或小型牌）與相對位置寫下了四個句子，每個句子只會是「真」或「假」最後請小華找 ft 那個句子是「真」的。

請問以下的四個句子中，那一個的描述是真的？

A) 桌面上可以找到兩張牌 X 和 Y，X 是藍色且 Y 是白色，而 X 在 Y 的上 面

B) 桌面上任意兩張牌 X 和 Y，若 X 是方形 Y 是圓形，則 X 在 Y 之上

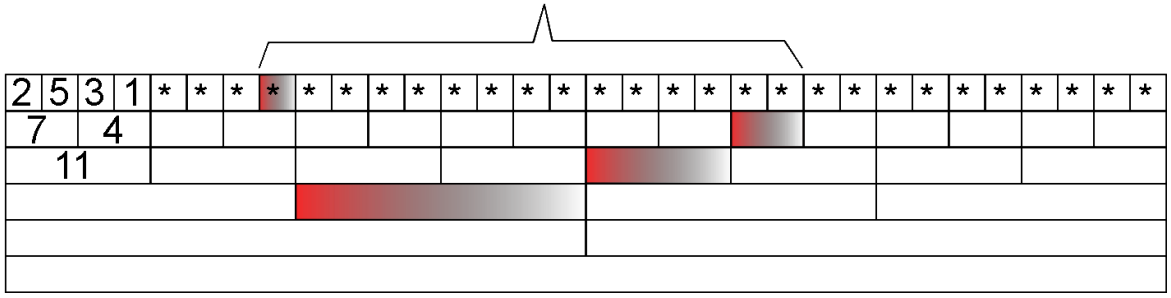
C) 桌面上任意兩張牌 X 和 Y，若 X 是小型牌 Y 是大型牌，則 X 在 Y 之右

D) 桌面上任意兩張牌 X 和 Y，右 X 是白色而 Y 是藍色，則 X 在 Y 的下面

答案:C

11. 紅蘿蔔倉庫

小兔子魯比為了幫助村民度過即將來臨的寒冬，建造了個有 32 間儲藏室的倉庫來存放紅蘿蔔。為了掌握紅蘿蔔的量，魯比用表格登記每間儲藏室的紅蘿蔔重量。 以下表第一列為例，魯比在前四間儲藏室分別存放了 2 公斤、5 公斤、3 公斤和 1 公斤的紅蘿蔔。其他儲藏室的紅蘿蔔重量先用星號（*）隱藏起來。



為了快速得知連續幾間儲藏室中紅蘿蔔的總重量，魯比擴充原本的表格；將儲藏 室兩兩為一組，並且將同一組儲藏室的紅蘿蔔重量加起來，填到表格的下一列。如上表，前兩間儲藏室為一組，加總得到 7 公斤後填到下一列；第三和四間儲藏室為一組，加總得到 4 公斤。魯比持續將數字兩兩為一組加總並填到下一列，直 到只剩下一個數字為止（這個數字相當於倉庫中所有紅蘿蔔的重量）。

使用這個表格可以快速計算連續儲藏室的紅蘿蔔重量總和。舉例來說，如果要知 道第八間到第二十二間儲藏室的紅蘿蔔重量總和，魯比不用一間一間累加，他只 要將上表中四個紅色格子的數字加總即可。因此，原先要將 15 個數字加總，透過這個表格只要將 4 個數字加總即可。

請問透過這樣的方式，用最少數字相加來計算任意連續儲存室的紅蘿蔔重量總和， 最多需要將幾個數字加總？

A) 6

B) 7

C) 8

D) 15

答案:C

12.六角形漫遊(進階)

海克薩剛國的海貍特別喜歡六角形，因此他們將居住的土地劃分成數個六

角形城市每一個城市都按照圖 A 的方式編定座標座標 $(k,0)$ 會在 $(k-1,0)$ 的右下方）並且設定兩個相鄰的城市距離為 1 單位長（如圖 B）

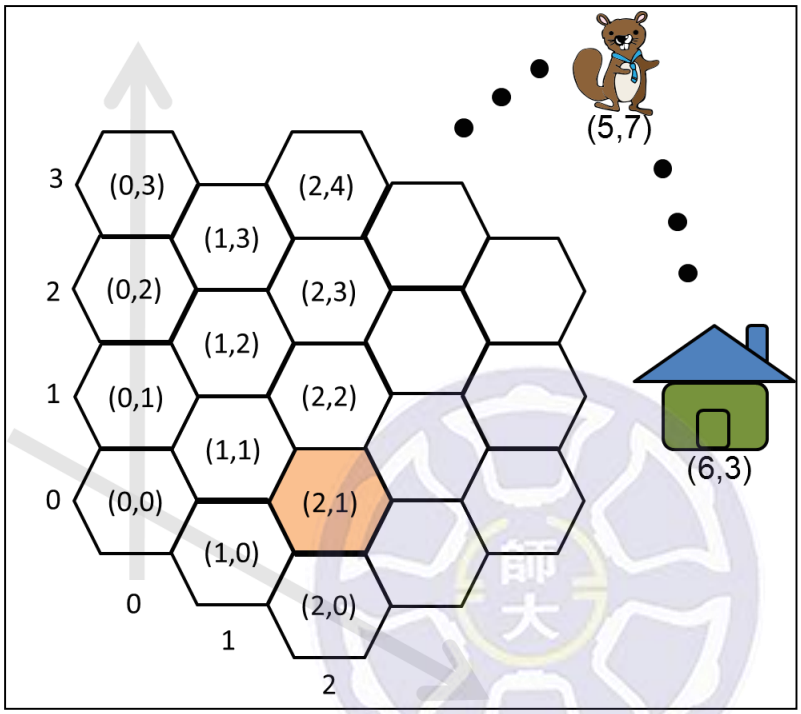


圖 A

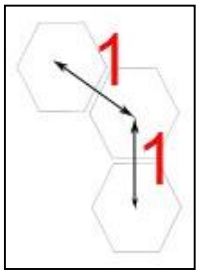


圖 B

有一個小海貍目前位在城市 (2, 1)的位置，而他要前往城市（5, 7）找他哥哥，再一同返回位在城市(6, 3)的家。請問小海貍路程的最短距離是幾單位 長？

A) 8 單位長

B) 9 單位長

C) 10 單位長

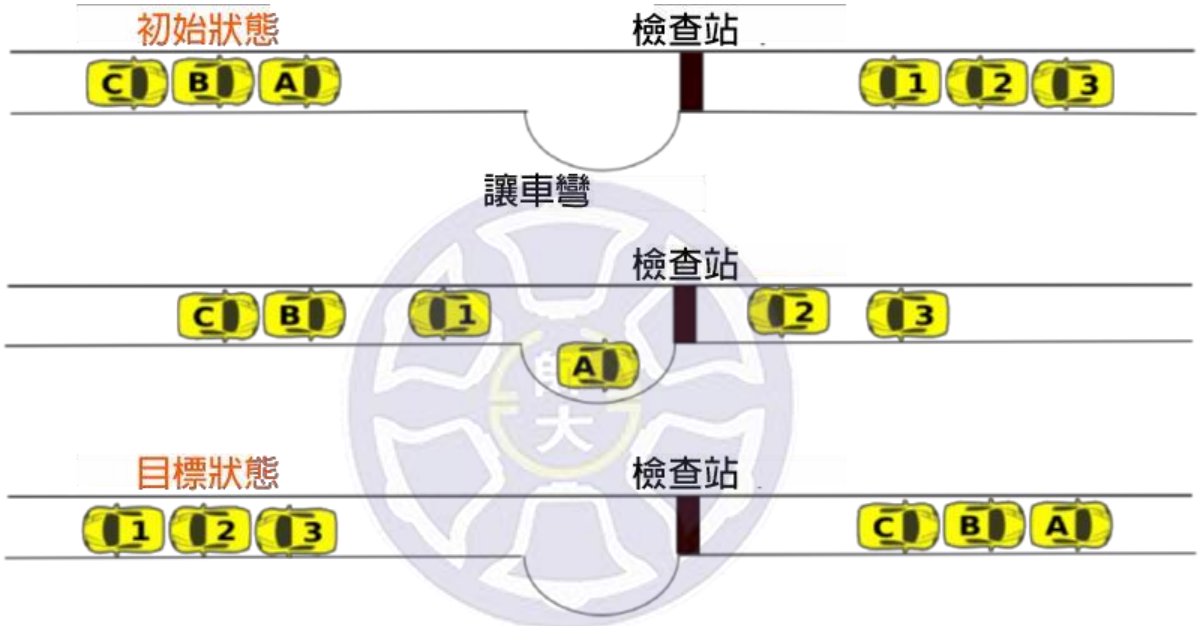
D) 11 單位長

答案:D

13.會車

如下圖所示，有六台車被困在一條狹小的道路上，其中三台車（A、B和C）要由左向右走，而另外的三台車（1、2及3）要由右向左走。圖上有一個檢查站，檢查站設有感應器可以計算有幾台車經過(同一台車前進或後 退都會各感應一次)。

雖然道路狹窄無法讓兩台車併行，但是檢查站的旁邊有一個讓車彎，可供 一台車暫時停靠，讓對向來車通過。



請問這六台車從最開始給定的初始狀態到會車完成的目標狀態，感應器會 記錄有多少車次經過？

A) 6 車次

B) 9 車次

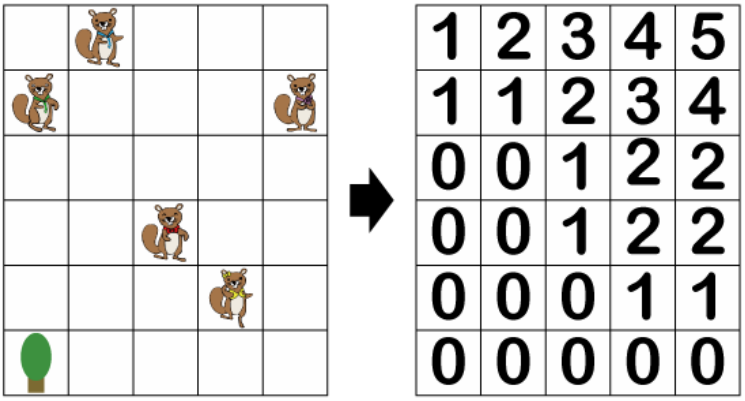
C) 15 車次

D) 18 車次

答案:D

14.加總區域表

一群小海貍居住在一大片矩形的濕地上，並可分割成 6x5 的格狀區域如下：



1	2	3	4	5
1	1	2	3	4
0	0	1	2	2
0	0	1	2	2
0	0	0	1	1
0	0	0	0	0

以一棵座落在溼地左下角的大樹當作參考點，為了不讓其他動物知道海貍 家的實際位置，小海貍製作了一份加密的地圖。加密的方法是把地圖中每 一格裡面填入一個整數。上面左圖是去年的實際位置而右圖則是加密後的 結果。

今年加密後的地圖如下，請問實際上有多少海貍住在圖中粗框線的區域裡呢？

1	3	4	7	9
1	3	4	6	8
1	2	3	5	6
1	2	3	4	5
0	1	1	2	3
0	0	0	0	1

A) 5

B) 4

C) 3

D) 2

答案:C

附錄三 每週課堂學習單

每週會設計課堂學習單，控制組與實驗組學習單不相同，實驗組會有引導與反思的部分(斜體)。而本研究學習單成績計算包含學習單(二)、學習單(三)、學習單(四)、學習單(五)。

JSLA 學習單(二)

1. 請問執行完下列程式碼後 x=?

```
var x = 10;
if(x>=5){
      x-=6;
      x+=2;
}
else{x=0;}
```

2. 請問執行完下列程式碼後 x=?

```
var x = 5;
if(x<=10){
      x+=5;
      if(x<=10){x=1;}
      else{x+=2;}
}
else{x+=2;}
```

3. 請回答下列問題

(1)請逐行說明程式碼功能?

(2)請問執行完下列程式碼後會顯示甚麼?

```
var speed = 70;
if(speed >= 45){
    alert("違規超速!");
}
else{
    alert("無違規");
}
```

//////////////////////背面有題目//////////////////////

4. 請依照此題狀況，依序寫出程式內部執行順序?

ex: 寫代號 12345

1) var speed = 70;

2) if(speed >= 45){

3) alert("違規超速!");

}

4) else{

5) alert("無違規");

}

JSLA 學習單(三)

填寫學習單可引導你個人的學習步驟，為了讓你能有效地學習二十一世紀必備的程式設計技能，請務必自己思考並作答，勿抄襲同學的答案。

一、

(1)請問執行完以下程式碼後 alert 會輸出甚麼?*(你可以使用平台測試)*

(2)請寫出此程式的執行順序。

例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

(3)(a) 請與核對執行順序正確答案。

(b) 你的執行順序是對的嗎?若否，知道哪裡出錯嗎?

```
1) var speed = 120;
2) if(speed >= 45){
3)       alert("違規超速!罰 2000 元");     }
4) else if(speed >= 120){
5)       alert("罰 4000 元");              }
6) else if(speed <= 30){
7)       alert("違規過慢!罰 600 元");              }
8) else {
9)       alert("無違規");                  }
```

二、

(1)請完成下列判斷式的條件設定:40 分以下不能補考、40 以上未達 60 分不及格、60 分以上未達 70 為普通、70 分以上未達 85 分中上。

(2)承上題寫出此程式的執行順序?*(你可以使用平台測試)*

例如: 順序為 1>2>3>4,5,6 **(4、5、6 同時執行)**

(3) (a) 請與核對執行順序正確答案。

(b) 你的執行順序是對的嗎?若否，知道哪裡出錯嗎?

```
1) var score = 69;
2) if(score >= 85 && score <= 100){
3)     alert("優良");    }
4) else if(                    ){
5)     alert("不及格");    }
6) else if(                    ){
7)     alert("普通");      }
8) else if(                    ){
9)     alert("中上");      }
10)else {
11)     alert("不能補考");    }
```

<討論>你在 if elseif else 這個語法的實作上，覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。

三、

(1)請問 x = 2 時，下列程式碼會 alert 輸出甚麼?*(你可以使用平台測試)*

(2)請問 x = 4 時，此程式的執行順序?

例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

(3) (a) 請與核對執行順序正確答案。

(b) 你的執行順序是對的嗎?若否，知道哪裡出錯嗎?

```
1) var x = ???;
2) switch(x) {
3)       case 1:
4)            alert("紅燈");
5)       case 2:
6)            alert("綠燈");
7)            break;
8)       case 3:
9)            alert("閃黃燈");}
```

四、

(1)下列程式碼會 alert 輸出甚麼?*(你可以使用平台測試)*

(2)此程式的執行順序?

例如：順序為 1>2>>3>4,5,6 (4、5、6 同時執行)

(3)　(a) 請與核對執行順序正確答案。

(b) 你的執行順序是對的嗎?若否，知道哪裡出錯嗎?

```
1) var meal = 3;
2) switch(meal) {
3)       case 1:
4)            alert("吃早餐");
5)            break;
6)       case 2:
7)            alert("吃中餐");
8)            break;
9)       case 3:
10)           alert("吃晚餐");
11)      default:
12)           alert("吃消夜");}
```

<討論>你在 switch case 這個語法的實作上，覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。

JSLA 學習單(四)

填寫學習單可引導你個人的學習步驟，為了讓你能有效地學習二十一世紀必備的程式設計技能，請務必自己思考並作答，勿抄襲同學的答案。

一、

(1)請問執行完以下程式碼後 alert 會輸出甚麼?

(2)請寫出此程式的執行順序。

例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

(3)你的答案是否正確?若否，你認為是哪裡出問題?從這個錯誤中你學到什麼?

```
1) var x = 5;
2) do {
3)       alert(″骰子的數字有″ + x);
4)       alert("\n");
5)       x--;
6) }
7) while (x > 0)
```

二、

(1)請逐行解釋以下程式碼的功能?

(2)請問執行完以下程式碼後 alert 會輸出甚麼?

(3)請利用 for(　;　;　)迴圈改寫?

(4)你的答案是否正確?若否，你認為是哪裡出問題?從這個錯誤中你學到什麼?

(5)　a. 若將(i<=50)改成(i<50)，結果 alert 會輸出甚麼?

b. 若將(i<=50)改成(i<51)，結果 alert 會輸出甚麼?

```
1) var i,sum=0;
2) i=1;
3) while(i<=50){
4)       sum=sum+i;
5)       i++;
6) }
7) alert(sum);
```

<討論>你在 while 與 for 迴圈語法的實作上，覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。

三、

(1)請問逐行解釋以下程式碼的功能?

(2)請問執行完以下程式碼後 alert 會輸出甚麼?

(3)請寫出此程式的執行順序。

例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

(4)你的答案是否正確?若否，請寫出原因、思考的方式?你認為是哪裡出問題?從這個錯誤中你學到什麼?

(5)若將程式中的 j<=size-1 改成 j<size，會有什麼效果?

```
1) var i, j;
2) var size = 2;
3) for (i = 1; i <= size; i++) {
4)       for (j = 1; j <= size-i; j++) {
5)            alert(" "); }
6)       for (j = 1; j <= 2 * i - 1; j++) {
7)            alert("*"); }
8) alert("\n"); }
```

<討論>你在 for 迴圈的巢狀結構語法的練習上，覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。

JSLA 學習單(五)

填寫學習單可引導你個人的學習步驟，為了讓你能有效地學習二十一世紀必備的程式設計技能，請務必自己思考並作答，勿抄襲同學的答案。

一、

(1)請問逐行解釋以下程式碼的功能?

(2)請問執行完以下程式碼後 alert(c)會輸出甚麼?

(3)如答案不正確，是哪裡出問題?從這個錯誤中你學到什麼?

```
1) function test_Function(a, b) {
2)         var c;
3)         c = a * b * b;
4) }
5)
6) var x = test_Function(4, 3);
7) alert(c);
```

<討論>你在全域變數與區域變數中，覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。

二、

(1)請問執行完以下程式碼後 alert(answer)會輸出甚麼?

(2)如答案不正確，是哪裡出問題?從這個錯誤中你學到什麼?

```
1)  function first(a, b, c) {
2)          var x = a * b *c;
3)          return second(x);
4)  }
5)
6)  function second(a){
7)          return a * a;
8)  }
9)
10) var answer = first(5, 3, 6);
11) alert(answer);
```

三、

(1)請問執行完以下程式碼後 alert(answer);會輸出甚麼?

(2)如將第 10 行改 var answer = first(5)變為 var answer = first(2), 那 alert(answer)會輸出甚麼?

(3)如(1)或(2)答案不正確，是哪裡出問題?從這個錯誤中你學到什麼?

```
1)  function first(a) {
2)          var i,ans = 0;
3)          for(i=0; i<=a; i++){
```

```
4)          ans = ans + i;}
5)          if(ans > 6){return second(ans);}
6)          else{return ans;}
7)  }
8)  function second(a){
9)          return a + a; }
10) var answer = first(5);
11) alert(answer);
```

*<討論>你在 function 概念與功能上的實作,**覺得最困難的地方在哪裡?請在平台上發表想法並與同學討論(為了讓你有更充分的反思時間，請務必於平台上討論，勿直接面對面討論)。***

附錄四 期中期末測驗題目

本研究課堂中，會有期中測驗和期末測驗，最後會結合課程中其它的成績，加權總合為該科目學習成就，而期中與期末測驗控制組與實驗組兩班相同。

期中測驗

一、

(1)請逐行說明程式碼的功能?

(2)請問執行完下列程式碼後會輸出甚麼?

(3)此程式的執行順序?

例如: 順序為 1>2>>3>4,5,6 (4、5、6 同時執行)

```
1)  var x = 5;
2)  var y = 10;
3)  if(x<=10){
4)        x+=5;
5)        if(x<=10) {
6)              x=1;
7)              y-=2; }
8)        else{y+=2; }
9)  }
10) else{x+=2; }
11) alert(y);
```

二、

(1)請逐行解釋以下程式碼的功能?

(2)請問執行完以下程式碼後 alert 會輸出甚麼?

(3)請利用 if 、 else if、else 改寫?

```
1)  var x = 3;
2)  switch(x) {
3)       case 1:
4)         alert("紅燈");
5)         break;
6)        case 2:
7)         alert("綠燈");
8)         break;
9)        case 3:
10)              alert("閃黃燈");}
11)           break;
```

期末測驗

一、

(1)請逐行解釋以下程式碼的功能?

(2)請問此程式的執行順序?例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

(3)請利用 switch …..case……. 改寫?

```
1)  var x = 4;
2)  if(x>5){
3)        x += 3;
4)        alert(x); }
5)  else if(x==8){
6)        x -= 6;
7)        alert(x); }
8)  else{
9)        alert(x); }
```


二、

(1)請逐行解釋以下程式碼的功能?

(2)請利用 for 迴圈改寫出一樣效果的程式碼?

```
1)  var x = 5;
2)  do{
3)        x += 2;
4)              alert(x);
5)              alert("\n");
6)  }
```

```
7)    while(x<15);
```

三、

(1)請逐行解釋以下程式碼的功能?

(2)請問此程式的執行順序?例如: 順序為 1>2>3>4,5,6 (4、5、6 同時執行)

```
1)    var i;
2)    for(i=1; i<4; i++){
3)          alert("$");    }
```

四、

(1)請逐行解釋以下程式碼的功能?

(2)請問執行完下列程式碼後會輸出甚麼?(畫整齊才好改　感謝)

```
1)    var i;
2)    var j;
3)    for(i=1; i<4; i++) {
4)              for(j=0; j<7; j+=2) {
5)              alert("$") ;
6)          }
7)          alert("\n") ;
8)    }
```

五、

(1)請問逐行解釋以下程式碼的功能?

(2)請問為何程式無法執行?

(3)如無法執行，要如何更改?

```
1)  function test_Function(a, b) {
2)          var c ;
3)          c = a + b + b ;
4)  }
5)
6)  var x = test_Function(4, 3) ;
7)  alert(c) ;
```