1. 研究計畫內容（以10頁為限）：
2. 摘要

資通訊人才在資通訊產業快速發展下，成為國家經濟成長重要的一環，但是大學中，程式設計對於初學者十分具有挑戰，課程通過率低、容易失去興趣。同時，老師或助教因大班無法顧及學生需要。又許多學生在學習過程中，因缺乏回饋不知錯在哪裡，而影響學習成效與意願。因此，本研究希望依照學習者在程式設計不同的知識水平與困難點發展適性化學習系統，以能針對個別學習者推薦適合的學習路徑能夠快速找到困難點及獲得學習教材，進而提高學生的學習成就與學習意願。

關鍵字：適性化學習、程式設計教學、學習路徑

1. 研究動機與研究問題

近年來資訊產業的快速發展，教育部在2014年時通過新的十二年國民基本教育課程綱要總綱「108課綱」，中「科技資訊與媒體素養」跨領域核心素養，可以從此發現科技教育的重要性及需求。同時資通訊產業的快速發展，也使得資通訊人才的重要性大增。目前世界各國也在積極培養與爭取資訊人才，創新知識和技術人才在經濟發展中的作用越來越受到重視(Shuzhen, 2019)。在「2021-2023年重點產業人才供需調查及推估報告」指出，目前人才欠缺原因以「應屆畢業生供給數量不足」，占總原因比例26.3%最多，因此政府在各級學校及各產業，均在積極推動程式設計等課程，希望能訓練更多資訊人才。預計美國未來所需要的STEM專業人才數量將超過其培育的數量10年，對資通訊人才的需求預計成長快於所有職業的平均水平(Qian & Lehman, 2017)。2030年資通訊數位人才需求將接近10萬人，而台灣面臨各產業數位轉型的情況下，資通訊人才卻嚴重不足，可能對國家經濟發展有巨大影響，因此資訊教育更形重要。

程式設計為資通訊人才的基礎能力，在過去大為必修課程，但相較其他科目通過率卻較低，因此進而影響學生學習成效與畢業後從事資通訊產業意願。2021年時教育部與勞動部統計，2015年大專院校「資訊通訊科技學門」畢業生投保率85.65％，但在2019年時卻降到73.04％，5年內減少了12.61%，由此可知資通訊學生畢業後，有許多學生並未進入資通訊產業，可能影響國家在資通訊產業的發展。並且，程式設計的概念及原理非常抽象，且很難理解(王秀鶯，2013)，對多數學生在學習上有障礙。此外，傳統的程式設計在初期學習門檻較高，對許多數學生而言並不容易上手，且易引起恐懼造成學習興趣低落。同時，學生在學習期間也經常因為不清楚錯誤原因，失去方向，而導致學習成就較低(Adu-Manu et al., 2013)

在一般大學課堂教學中，老師無法一一解析錯誤答案，依照各個學生知識薄弱的區塊給予協助。在時間有限的課堂中，助教或輔導也不能滿足每一位學生的需求。因此在過去的研究中發現，適性化的教學能夠幫助學生有更好的學習成就與學習效率，因為適性化教學，可以依照學生個人不同的情形給予適合個人的學習方案( Tseng, 2008)，讓學生用更有效率的方式學習或加強其需要學習的內容，因此本研究希望可以探討程式設計課程適性化的可能性。

在程式學習的過程中，除錯應是一種基本能力，但事實上卻發現許多學生並沒有基本的除錯能力，進而使這些學生在學習程式設計過程中困難重重，不知道自己錯誤在哪裡，也不知如何更正 (Nagvajara & Taskin,2007)，因此有需要可以發展一個輔助系統，來幫助這些學生練習完成除錯以及完備正確的知識。如果可以發展基於錯誤類型的適性化學習系統，針對學生明顯需要改正的錯誤，推薦適合的學習路徑，應可以有效率地引導學生學習正確知識。

程式設計錯誤分類為語法錯誤、語義錯誤和邏輯錯誤（Hristova et al. 2003; McCall and Kölling,2014），其中語法錯誤，是可以在編譯器上直接提示錯誤，但對於新手來說，語義錯誤與邏輯錯誤是較不容易發現的(Eltegani and Butgereit,2015)。並且在這些基礎入門課程的負面學習成效，可能會對學習者對該領域的態度產生有害影響(Adu-Manu et al., 2013)。因此若發展一個幫助老師和助教教學，並針對每位學生需求的適性化學習系統，應輔導學生的學習成效與態度有很大的幫助。

所以本研究希望依照學生過去在練習與測驗中常發生的錯誤，以發展一個學習路徑的適性化學習系統，可以自動判斷學生是哪一類型的錯誤，並且根據錯誤給予適當的學習建議，讓學生能夠快速找到需要學習的重點，幫助老師教學，提高學習成效與學習效率。

故本研究的目的為：

1. 依照學生過去在練習與測驗中常發生的錯誤，以發展一個適性化學習路徑的系統，去判斷是哪一種類型的錯誤。
2. 根據錯誤給予適當的學習建議。

希望系統可以幫助老師適性化的教學，並同時可以讓學生快速找到需要學習的重點，引提高學習成效與學習態度。

1. 文獻回顧與探討

3.1程式設計教學

程式設計為一門學科，是以程式解決問題的過程，也是軟體開發的重要步驟(Adu-Manu et al., 2013)。這門課程通常開設在大學資通訊相關學系一年級時，教導程式編寫基礎。而近來台灣各大學也將程式設計這們課程納入畢業門檻，各系大一學生都需學習這門課程。程式設計教學含括編程工具和程式語言的知識、解決問題的技能。目前大多資通訊學系程式設計課程以Java,C,C++,C#,Python等程式語言作為入門語言。

程式設計的教學方式包含不同的方法：以教師為中心的講座、協作式學習、翻轉教室、遊戲式學習等方式(Hsu et al., 2018；Zhang et al., 2013)。雖有多元的教學方式，但學生學習成就較低的困境仍未太大改善。 Eltegani et al., ( 2015) 提出，學習策略、協作、自我效能、時間可用性、興趣、支持、腳手架( scaffolding)、設備和資源的可用性以及舒適度(comfort level)，均為影響參與程式設計的主要變數(Eltegani et al., 2015)。從文獻可以得知，學生在句法知識、概念知識和策略知識方面表現出各種誤解和其他困難(Qian & Lehman, 2017)。目前有些 IT 解決方案可以幫助證明高質量的學習體驗並提高參與度，例如教育管理系統、自適應系統和智能輔導系統 (Eltegani et al., 2015) 。

在過往研究中顯示，大約有1/3的學生在程式設計相關課程成績為不及格（Beise et al., 2003；Gill & Holton, 2006；Adu-Manu et al., 2013；Zhang et al.,  2013）。原因包含了：學生缺乏思維能力、缺乏對程式語言語法的了解、缺乏邏輯和推理能力等原因的干擾，因此通過率較低。

3.2 程式設計學習困難類型

關於程式設計課程失敗的研究，通常集中在學生理解概念的能力或弱點上。很多學生發現編程既困難又令人沮喪(Adu-Manu et al., 2013)，並進而可能會對學習者對該領域的態度產生有害影響(Ismail et al., 2010；Adu-Manu et al., 2013)。同時，大多數學生在理解語義、除錯技能和編程技能方面，存在很大困難(Eltegani et al., 2015)。研究也顯示，這並非台灣特別的現象，在其他國家的程式設計初學者也有相類似的問題(Ismail et al., 2010)。

計算機科學教育的研究人員也使用不同的術語，例如「困難」、「誤解」與「錯誤」等術語(Boulay, 1986; Sleeman et al., 1986; Pea, 1986; Qian & Lehman, 2017 )，來描述學生對計算機編程的不准確或不完整的理解。在過去的研究中學者，將程式設計學習困難作分類，包含：表面錯誤和深層錯誤（Sleeman et al., 1986），按原因對誤解進行分類（Qian & Lehman, 2017），語法錯誤、語義錯誤和邏輯錯誤（Hristova et al. 2003; McCall and Kölling 2014），語法知識困難、概念知識困難和策略知識困難(Qian & Lehman, 2017)。同時，研究也顯示，許多學生表現出脆弱的編程知識，很少人能夠清楚地理解編成策略(Nandigam and Bathula., 2013)。

表1 程式設計學習困難解釋與錯誤範例

|  |  |  |
| --- | --- | --- |
| 困難 | 解釋 | 錯誤範例 |
| 語法知識(syntactic knowledge) | 是關於語言特徵、基本事實和規則的知識。 | 1. 參數間缺少分號 2. 使用錯誤的關鍵字 3. 無效語法 |
| 概念知識(Conceptual knowledge) | 是指關於編程結構和原理如何工作以及計算機內部發生了什麼的知識。 | 對程式結構或執行原理的誤解，例如：   1. 條件式 2. 迴圈 3. 類別與物件 |
| 策略知識(Strategic knowledge) | 是指如何應用程序設計的句法和概念知識來解決新問題。 | 對應用程式設計解決問題的困難：   1. 未能選擇適當的循環結構 2. 未能對程式進行測試和除錯 |

資料來源：Qian, Y., & Lehman, J. (2017). Students’ misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE), 18*(1), 1-24.

因為錯誤的情況且狀況有很多種，本研究將針對概念知識困難與策略知識困難進行判別與輔助，並推薦學生需要花時間要複習哪一章節；因在程式撰寫過程中編譯器便會提示語法錯誤，因此系統主要輔導學生較不易判別的困難分類，以提高學生學習動機。

3.3適性化學習

適性化學習與因材施教的理念相同，依照學習者不同的認知方式、個人背景、學習成績等因素，給予不同的學習路徑、教材內容或呈現方式，以提高學生的學習成績與學習效率。適性化可參考的資料包括個人資料中的能力、興趣、學習風格等，以及在學習過程中所留下的活動紀錄，還有學習環境等資料 (Kobsa et al.,2001)

表2 適性化學習路徑模式

|  |  |
| --- | --- |
| 適性化學習模式 | 解釋 |
| 學習路徑 | 提供適當的網路學習路徑與導覽方式，是適性化學習重要的議題 |
| 教材內容 | 針對個人化網路教材的提供 |
| 呈現方式 | 在不同身分、職位或是有不同偏好時呈現不一樣的介面 |

資料來源：黃國禎, 蘇俊銘, & 陳年興. (2015). *數位學習導論與實務 (第二版)*. 博碩文化股份有限公司.

適性化學習目前已應用在多種不同領域的學科。Walkinton(2013)將模型追蹤方法結合到方程問題求解的過程中，發現經歷過適性化學習的學生能更有效地組合不同的代數表達式；張安緹與陳鴻仁(2018)使用即時適性英語字彙遊戲學習系統教學，低學習能力學生學習成就顯著提高，且學習動機有明顯的提升； Liu et al.(2017)利用適性化學習，輔導藥學專業學系學生學習專業科目，研究結果表明有助於解決化學科目的知識差距；陳又菁(2019)使用適性化引導發現式行動學習電子書(AGM-PLE)教學，學生反應式較可以有自己探索及反思的機會與使用適性化的程式設計電子書學習起來更有效率。本研究將以課堂已有現有的教材為基礎，依每位學生所需要學習不同進度，採用學習路徑作為適性化系統的模式，但以達到適性化教學目的。

* 1. 學習路徑

學者認為學習路徑可以解決很多不同的問題。不同的學習者有不一樣的瀏覽目的，可能是測驗、學習或精進；並且，學習資料以網站的方式呈現，不同人搜尋到網站的目的可能不同，網站隨著時間的成長與演變，之前的網頁結構可能會變得不適合，造成使用者無法網站上的學習資料；因此如果利用學習路徑，可以使網站上的學習資料有效的被使用 (Perkowitz & Etzioni, 1997；黃國禎、蘇俊銘、陳年興，2015) 。

適性化學習路徑可以畫分為五大類:直接導覽、鏈結適性化排序、適性連結隱藏、連結適性化註解以及導覽圖調適。(Brusilovsky et al., 1998；黃國禎、蘇俊銘、陳年興，2015)，以下分別說明之：

表3 學習路徑類型

|  |  |
| --- | --- |
| 學習路徑類型 | 類型解釋 |
| 直接導覽(Direct guidance) | 由系統推薦最適合學習者的學習路徑，利用”下一步”按鈕引導學習者進錄下一個學習頁面。 |
| 鏈結適性化排序(Sorting or adaptive ordering of visible links | 根據學習者模組對多個連結按照重要程度排序， 將最合適或關聯性大的連結排在最上面，最不合 適或關聯性小的連結排在底下。 |
| 適性連結隱藏(Adaptive hiding of links) | 依照學習者的學習目標，將多餘或過於複雜的資訊隱藏，避免學習者瀏覽到尚未能理解的教材，增加學習挫折。 |
| 連結適性化註解 | 在連結上加入個人化註解，以便告知學習者該連 結節點 (頁面 )之目前狀態(Brusilovsky,& Eklund, 1998; Eklund,& Sinclair, 2000) 。 |
| 導覽圖調適 | 提供學習者個人化的超媒體地圖。 |

資料來源：黃國禎, 蘇俊銘, & 陳年興. (2015). *數位學習導論與實務 (第二版)*. 博碩文化股份有限公司.

學習路徑的調整可分為兩種，系統自動調整與學習整自訂學習目標。前者目前常見的做法包含：學習者主動的利用問卷，詢問使用者欲學習內容，並推薦學習路徑，以及學習者被動的被系統分析資料，並利用資料探勘、網站探勘等方式，做分析來推薦學習路徑 (黃國禎、蘇俊銘、陳年興，2015)。

本研究系統預計採用直接導覽類型，透過學習路徑中的測驗結果，在不影響學習者的情況下，使系統自動調整學習路徑，讓學習者減少搜尋要重新學習哪一章節的時間，得到依其狀況需要學習的教材。

3.5 適性化教學在程式設計領域的應用

Gavrilović et al.(2018)在課程「Java編程語言」中，將適性化教學系統分為多個模組，每個模組之中都還有不同的模塊課程，不同課程之中有許多學習目標以及對應的多種教材。系統根據學生的先備經驗，決定學生的學習過程與組成元赤。他們認為在適性化學習系統中，學生沒有義務要學習已經具備或了解的知識，每隻章節學習完成之後，會獨立地進行學生學習評量才可以進入下一章節。若分數沒有達到標準，便返回包含問題答案的章節。同時，只有在評量結束後，學生分級才有可能發生變化，否則被歸類為低知識水平的學生，在學習過程中都會維持相同的分類。在學習過程中，學生可以自由的回到過去所學習過的章節進行複習。該研究在學習者滿意度調查問卷(每題0-10分)中，每一題平均得分皆高於7分。若將AI技術使用在適性化教學中，我們稱之為自適性學習。此外，除了調查之外，還分析了學生在使用自適應學習過程的學年課程中的表現。因此使用自適應學習過程和Java評分器被證明是學生通過教材、演示和實踐示範，以及作業學習Java編程的絕佳方式。

Troussas(2020)在程式設計(學習程式語言C#)課堂中，將學習理論與智能技術結合。他們將學生的經驗、能力和知識水平等個人因素使用布魯姆分類法(Bloom's taxonomy)進行分類，六種層級(L1-L6)分別對應不同的學習目標和學習活動。並且使用七種權重來表示學生學習程式設計語言C#的知識水平：新手(N)、初學者(B)、中級(I)、好(G)、非常好(VG)、高級(A)、專家(E)，同時再針對不同知識水平給予不同難度、類型和數量的適性化教學。實驗結果證實，適性化教學提供了以學生為中心的學習環境，進一步提高了知識獲取和學習成果；也因此增加了學生的學習動機。

1. 研究方法及步驟

本研究以迴圈及陣列為例，因為過去研究結果發現，學習表現較差的學生皆在「重複結構」遇到困難，連帶影響以重複結構為基礎的「列表」及「綜合應用」表現。(李恩萱,李忠謀 2018)

程式初學者在「陣列」及「迴圈」單元之前，會先進行「變數」、「運算子」及「條件分支」等基礎知識，因此學生在陣列及迴圈的學習狀況也會受到在先前其他單元的知識水平所影響；因此雖以迴圈及陣列為例，但輔助教學系統仍會包含其他基礎單元之資料。

* 1. 研究步驟

本研究目的為建置適性化學習系統，整體研究步驟如下：

* + - 1. 蒐集文獻中所提的錯誤類型及往年學生答題錯誤紀錄：

將含有錯誤的程式碼段落中，將錯誤原因拆解成單一運算子

* + - 1. 將錯誤紀錄進行分類：

依照過往論文研究主要區分為語法錯誤、語義錯誤、概念知識困難以及策略知識困難，再將錯誤紀錄細分到不同單元主題。

以下for迴圈題目錯誤為例：

題目：請輸出0-5

學生錯誤答案：

for( int i=0; i<5; i++){

System.out.print(“i”);

}

此範例有兩個錯誤：

1. for迴圈中的條件式，若初始值設為0則條件式應為i<6或是i<=5；此錯誤類型將劃分為for迴圈概念知識困難及if條件式概念知識困難，因為學生可能是不了解迴圈的執行結束條件，也可能是因為不了解if的概念，進而造成程式撰寫錯誤。
2. 迴圈動作敘述中的輸出，此題應要輸出變數i，而非字串型態i；此錯誤類型可劃分為變數語法知識困難，因學生錯誤原因為不了解變數的輸出方式，而造成程式碼編寫困難。
   * + 1. 進行機器學習訓練：

撰寫程式碼的題型最容易發現學生在程式設計中遇到的困難點，但也因為程式並非只有一種解法，且因變數的命名或是參數初始值不同就可能會出現許多不同的結果；因此程式碼題的問題判斷是傳統教學中若不使用人工的情況下最難處理的題型，因此本研究利用過去課程中所蒐集的數據，訓練機器學習的方式來預測學生可能有的困難點，提高分析困難點的效率。

* + - 1. 將教材進行分類與裁切：

參考Xie等人(2019)對程式設計能力的分級將教材依照不同程度進行重新整理，將每個單元的教材從S1-S4分為四個等級；並將教材與錯誤分類進行對應。

表4 程式設計技能分級

|  |  |
| --- | --- |
| 技能分級 | 能力 |
| S1 | 單行程式碼閱讀能力 |
| S2 | 單行程式碼撰寫能力 |
| S3 | 段落程式碼閱讀能力 |
| S4 | 段落程式碼撰寫能力 |

資料來源：Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., ... & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education, 29*(2-3), 205-253.

* + - 1. 設計評估題庫題：

使用課堂及課後練習未使用到的題目作為系統內評估的試題；學生程度評估題目以程式碼題作為主要測量內容，採用機器學習的方式判斷學生可能存在的困難點，並用單選題、多選題、填充題等題型，提高辨識學生程式設計學習困難點的準確度。

* + - 1. 設計學習路徑：

每章節先進行學生知識水平評估，給予測驗回饋，並由系統分析學生困難點所在，推薦學生最需要先進行學習內容，再依照學生程度給予相對應的學習教材。學習結束後，再進行知識水平評估，測量學生學習成效及分析學生困難點。評估—分析困難點—學習，循環發生直到學習結束。

* + - 1. 系統建置：

學生學習與作答介面

課堂管理者監測介面，以方便老師及助教掌握學生學習狀況對教學內容進行調整或是輔導學生。

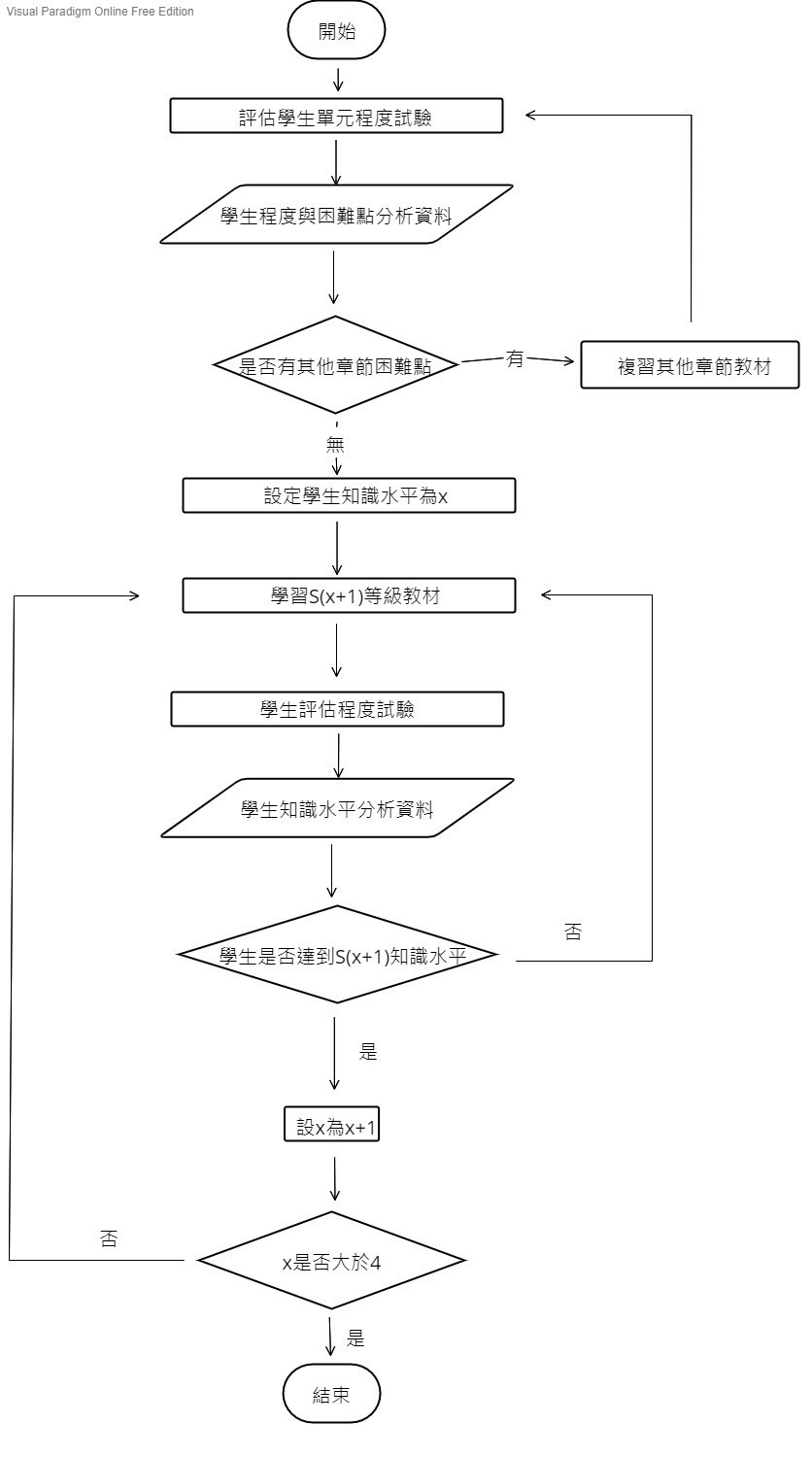
適性化學習路徑

學習過程紀錄功能(學習路徑、學習時間、作答時間與作答狀況等)

* + - 1. 系統成效評估：

在資管系大一程式設計課堂中進行測試，包含學生端與管理者端，從實際的情況中從中研究適性化學習系統對於程式設計學生是否有幫助，以及可以改善哪些不足使適性化推薦內容更為準確。

* 1. 適性化學習路徑設計
     + 1. 將每一學習目標模組化，並建置相對應的評估試題與教材，依照課程規劃按照單元順序學習。
       2. 每一章節開始學習前，先以測驗評估學生在該章節的知識水平，以及檢視是否有前面單元未解決的學習困難。
       3. 若有前面章節的學習困難項目則先學習該困難點所需知識；當排除前面章節的學習困難後。
       4. 便依照系統所推薦程度之教材學習。教材根據知識水平分為四個等級(S1至S4)，當知識水平愈高時，可能會發生的困難點也會更多。
       5. 學習完一個等級後，往下一個等級學習，直到S4學習評估通過。



以下題目為題庫範例

|  |  |  |
| --- | --- | --- |
| 題目分類 | 題目 | 題目解釋 |
| for loop-S1-單選題 | 請選擇輸出結果為何?  int sum = 0;  for(int i=0; i<6; i++){  sum+=i;  }  System.out.print(sum);  A. 21  B. 15(正解)  C. 0  D. 6 | 此題目的為測驗學生對於for迴圈中不只一項控制項時的熟悉程度。  若學生選擇錯誤答案A依照以往資料判斷極有可能對於迴圈中的條件式有學習困難，將安排學生測驗迴圈判斷式S1-S4的題目確認學生困難點類型，並依照學生學習狀況油系統推薦適合的測驗難度；作答C或D的學生則是對於+=或是for迴圈整體架構不熟悉，會根據學生過去學習過程資料安排評估測驗。 |
| for loop-S4-簡答題 | 請於空格處填入正確的程式碼，讓此程式可計算出1到100間可被3整除的數值之總和：  int sum = 0, i; //宣告累加值 sum 與迴圈變數 i  for (i=1;i<=100;i++) //跑100次的for迴圈  if ( {#1} ) //當i的值可以被3整除時  {#2} ; //將i的值累加到sum  System.out.println("目前累加值為" + sum); | 本題需利用if-else條件是對控制項進行判別，學生需要了解迴圈，及在迴圈中使用條件式以及安排適當的輸出順序。較偏向實際在撰寫時所需應對的狀況。  簡答題使用機器學習辨認學生困難原因，並依照判斷給予評估測試。 |

* 1. 預計進行實驗對象

過去研究結果發現，學習表現較差的學生皆在「重複結構」遇到困難，連帶影響以重複結構為基礎的「列表」及「綜合應用」表現 (李恩萱 與 李忠謀，2018) 。參照過去銘傳大學大一必修課程程式設計(一)，學生在109學年度的表現，在迴圈與陣列之章節成績相較於前幾單元，會出現成績分水嶺，且答題錯誤原因故不相同，雖可大致分為語法錯誤、邏輯錯誤、編譯錯誤等問題。

本研究以銘傳大學資管系110學年度與111學年度修習「程式設計(一)」「程式設計(二)」之學生為研究對象，探討其使用輔助學習系統對於學習迴圈與陣列章節是否有顯著的幫助。

* 1. 前導研究

2021年11月26日於程式設計課程中進行10題for迴圈單元題目測試，並以人工的方式閱卷，分析學生作答所發生的錯誤與困難點，並且依照作答狀況給予學生不同的學習建議與輔導，參加人數共計53位；12月17日公布測驗結果，給予適性化解析與評語，並蒐集測驗回饋問卷。此次問卷以Sáez-López (2016)等人所使用的問卷作為模板，以李克特量表調查(非常不同意1-非常同意5)；有效問卷人數共計51位。參與題目測驗並繳交有效問卷人數共47人。

表5(6) 問卷統計

|  |  |
| --- | --- |
| 問卷構面 | 平均分數 |
| 測驗的主動情況 | 4.27 |
| 測驗對未來的影響 | 4.43 |
| 感知有用性 | 4.40 |
| 學習情緒 | 4.06 |

前導研究問卷平均分數均在4分以上，問卷學生也表示「透過測驗能夠更明確知道自己哪裡不會，可以及時找到問題解決」、「對學習是有幫助的」等回饋，由此可知利用練習題發現學生困難點並進行輔導的方法是可行的，因此將利用此概念開發適性化學習系統。

* 1. 評估方法
     + 1. 以Troussas et al.(2020)與Gavrilović et al.(2018)的問卷為範本進行修改後，對學生調查自身感受以及適性化系統(Troussas et al., 2020; Gavrilović et al., 2018 )
       2. 抽樣訪談學生對於適性化系統的感受與認知
       3. 與前一學年度學生成績成長幅度進行比較

1. 預期結果

本研究的目的希望可以，針對學生在程式設計中的困難點發展一個適性化學習路徑系統，來幫助學生可以快速找到學習其所需要的教材，提高在程式上的學習成效與態度。老師也可以由此系統，了解學生在程式設計中所面對的困難，並依其調整或補充教材，進而提高課程通過率與學生的學習效果。

1. 參考文獻

中文文獻

王秀鶯. (2013). 導入 Scratch 程式教學對國中生自我效能與學習成就之探究—以程式設計課程為例. *國立臺灣科技大學人文社會學報, 9*(1), 1-15.

李恩萱, & 李忠謀. (2018). 教育學院大學生運算思維與程式設計學習成就. *TANET2018 臺灣網際網路研討會*, 2592-2596.

黃國禎,蘇俊銘,陳年興(2015)。數位學習導論與實務。台灣：博碩

張安緹, & 陳鴻仁. (2018). 即時適性英語字彙遊戲學習系統教學之學習成效與學習動機分析. *數位學習科技期刊, 10(*4), 31-58.

陳又菁(2019)。適性化引導發現式程式設計行動學習。教育部教學實踐研究計畫成果報告(PEE107130)。

英文文獻

Anderman, E. M., Urdan, T., & Roeser, R. (2003). The patterns of adaptive learning survey: History, development, and psychometric properties. In Indicators of Positive Development Conference, Washington, DC.

Adu-Manu, K., Kingsley, J., & Owusu, P. Y. (2013). Causes of failure of students in computer programming courses: The teacher-learner Perspective. *International Journal of Computer Applications, 77*(12), 27-32.

Beise, C., VanBrackle, L., Myers, M., & Chevli-Saroq, N. (2003). An examination of age, race, and sex as predictors of success in the first programming course.

Dziuban, C., Moskal, P., Johnson, C., & Evans, D. (2017). Adaptive learning: A tale of two contexts. *Current Issues in Emerging eLearning, 4*(1), 3.

Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research, 2*(1), 57-73.

Eltegani, N., & Butgereit, L. (2015). Attributes of students engagement in fundamental programming learning. In 2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE) (pp. 101-106). IEEE.

Gavrilović, N., Arsić, A., Domazet, D., & Mishra, A. (2018). Algorithm for adaptive learning process and improving learners’ skills in Java programming language.*Computer Applications in Engineering Education, 26*(5), 1362-1382.

Gill, G., & Holton, C. F. (2006). A self-paced introductory programming course. *Journal of Information Technology Education: Research, 5*(1), 95-105.

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers & Education, 126, 296-310.

Hristova, M., Misra, A., Rutter, M., & Mercuri, R. (2003). Identifying and correcting Java programming errors for introductory computer science students. *ACM SIGCSE Bulletin, 35*(1), 153-156.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). Instructional strategy in the teaching of computer programming: a need assessment analyses. *TOJET: The Turkish Online Journal of Educational Technology, 9*(2).

Kerr, P. (2016). Adaptive learning. Elt Journal, 70(1), 88-93.

Kobsa, A., Koenemann, J., & Pohl, W. (2001). Personalised hypermedia presentation techniques for improving online customer relationships. *The knowledge engineering review, 16*(2), 111-155.

Liu, M., McKelroy, E., Corliss, S. B., & Carrigan, J. (2017). Investigating the effect of an adaptive learning intervention on students’ learning. *Educational technology research and development, 6*5(6), 1605-1625.

McCall, D., & Kölling, M. (2014). Meaningful categorisation of novice programmer errors. In 2014 IEEE Frontiers in Education Conference (FIE) Proceedings (pp. 1-8). IEEE.

Nandigam, D., & Bathula, H. (2013). Competing dichotomies in teaching computer programming to beginner-students.*American Journal of Educational Research, 1*(8), 307-312.

Nagvajara, P., & Taskin, B. (2007). Design-for-debug: A vital aspect in education. In 2007 IEEE International Conference on Microelectronic Systems Education (MSE'07) (pp. 65-66). IEEE.

Pea, R. D. (1986). Language-independent conceptual “bugs” in novice programming.*Journal of educational computing research, 2*(1), 25-36.

Perkowitz, M., & Etzioni, O. (1997). Adaptive sites: Automatically learning from user access patterns. In Proc. 6th Int. World Wide Web Conf., Santa Clara, California (Vol. 149, p. 158).

Perkowitz, M., & Etzioni, O. (1997). Adaptive web sites: an AI challenge. In IJCAI (1) (pp. 16-23).

Qian, Y., & Lehman, J. (2017). Students’ misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE), 18*(1), 1-24.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion.*Computer science education, 13*(2), 137-172.

Shuzhen, H. (2019). Innovative talents training mode of science and engineering universities based on the human resources demand of modern enterprise in IoT technology. *Journal of Intelligent & Fuzzy Systems, 37*(3), 3303-3310.

Sleeman, D., Putnam, R. T., Baxter, J., & Kuspa, L. (1986). Pascal and high school students: A study of errors. *Journal of Educational Computing Research, 2*(1), 5-23.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using “Scratch” in five schools. C*omputers & Education, 97*, 129-141.

Tseng, J. C., Chu, H. C., Hwang, G. J., & Tsai, C. C. (2008). Development of an adaptive learning system with two sources of personalization information. *Computers & Education, 51*(2), 776-786.

Troussas, C., Krouska, A., & Sgouropoulou, C. (2020). A novel teaching strategy through adaptive learning activities for computer programming. *IEEE Transactions on Education, 64*(2), 103-109.

Walkington, C. A. (2013). Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. *Journal of Educational Psychology, 105*(4), 932.

Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., ... & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education, 29*(2-3), 205-253.

Zhang, X., Zhang, C., Stafford, T. F., & Zhang, P. (2013). Teaching introductory programming to IS students: The impact of teaching approaches on learning performance. *Journal of Information Systems Education, 24*(2), 147-155.

1. 需要指導教授指導內容
   * + 1. 研究計畫實施方式
       2. 研究是否有效的評估方法
       3. 錯誤類型的分類
       4. 成果報告撰寫