教育學報，2021 年，第 49 卷第 2 期，頁 71–95

積木式程式設計之學習成效後設分析

邱仁一、崔夢萍*
國立臺北教育大學課程與教學傳播科技研究所

近年來積木式程式語言廣泛運用於各教育階段資訊課程，提供學習者導入程式語言學習的環境。本研究以後設分析法探討積木式程式設計教學對不同教育階段學生學習成效的影響，包含程式設計能力、問題解決、運算思維和學習動機，並探討影響學習成效的變項。本研究透過系統化文獻資料蒐集和篩選，針對 2003 至 2021 年 37 篇已發表的文獻共 5,430 名樣本進行分析。研究結果顯示積木式程式設計教學的整體效果量為中效果量（0.33），在程式設計能力、問題解決、運算思維和學習動機上皆有正向效果，支持積木式程式設計對學生學習有效。再者，程式設計能力受「教學時間」、「專案類型」和「研究地區」調節變項作用而有差異：問題解決能力受「專案類型」調節；運算思維受「研究地區」調節；學習動機則受「實驗人數」、「積木式程式設計環境」、「專案類型」、「研究地區」調節。本研究結果可作不同教育階段運用積木式程式語言教學的參考。

關鍵詞：程式設計；後設分析；積木式程式設計；學習成效

緒 論

隨着科技發展，如何令學生具備資訊素養，培養未來能力，是目前重要的教育目標之一。新媒體聯盟 2016 年的地平線報告指出，程式設計已成為重要的學習工具（New Media Consortium, 2017）。台灣教育部在 2019 年新課程綱要亦將程式設計列為重要學習內容，要求程式設計課程列為國高中的必修科目（國家教育研究院，2019），顯示程式設計已成為各階段教育的一項重要能力指標。

程式設計學習環境可分為文字式程式設計（text-based programming）和積木式程式設計（block-based programming），教學者可依據對象和程度選擇適用環境。文字式程式設計程式符號較為複雜，可能會降低初學者的學習興趣和成效。積木式程式

* 通訊作者：崔夢萍（mptsuei@mail.ntue.edu.tw）

設計環境則提供學生以拖曳拼圖的方式進行程式編寫，大幅降低錯誤編寫語法的可能，並能降低認知負荷（何昱穎等，2010），常見的環境如 Scratch、Alice 和 APP Inventor。

先前研究支持程式設計學習對學生不同層面的影響，如提升程式設計能力（林育慈、吳正己，2016）、問題解決能力（Korkmaz, 2016b）、運算思維能力（Pérez-Marín, et al., 2020）等。過去研究發現積木式程式設計環境比傳統教學（直接講述式教學、C 語言教學等）能顯著提升學生的程式設計能力（Durak, 2018; Wang et al., 2009），但部分研究則呈現不同結果（Korkmaz, 2016b; Mihci & Özdener, 2014）。問題解決能力方面，Lai & Yang（2011）指出積木式程式設計環境比傳統教學等更能提升學生的問題解決能力，然而有研究顯示兩者並無顯著差異（Nam et al., 2010; Oluk & Saltan, 2015）；學習動機方面，過去研究亦呈現不一致的結果（Korkmaz, 2016b; Shanmugam et al., 2019）。

後設分析（meta-analysis）屬統整性分析方法，將相同主題研究使用標準差為單位量尺進行概念比較（馬信行，2007），可進一步推論相同主題的研究結果。先前在程式設計教學後設分析方面，Y. K. C. Liao & Bright（1991）分析 1969 至 1989 年程式設計教學介入對學生認知能力的影響，整合 65 篇相關研究後發現整體效果量為 0.41。Umapathy & Ritzhaupt（2017）分析 18 篇研究，發現程式設計對學生學習成效效果量為 0.64，測驗成績效果量為 0.41，情意影響效果量為 0.08。Scherer et al.（2019）以程式設計對學生的學習遷移效應進行後設分析，整體效果量為 0.49。它們都顯示程式設計教學對學生認知能力的助益。

以積木式程式學習為題的後設分析研究方面，Costa & Miranda（2017）針對 2000 至 2014 年六篇 Alice 與傳統程式設計研究，發現總體效果量為 0.54。Xu et al.（2019）分析 13 篇使用積木式程式設計與文字式程式設計的文獻，發現效果量為 0.25，且在國小階段效果量最高（0.98）。

先前後設分析文獻中，以積木式程式設計環境作後設分析的研究較少，且未針對不同積木式程式設計環境進行比較分析，針對學習成效分析範圍較大，故本研究延伸先前的積木式程式後設分析研究：

1. 根據 Wu et al.（2020）指出華人地區和韓國的教師認為使用積木式程式教學是使學生學習程式語言的最好方式，本研究探討不同地區研究運用積木式程式設計教學對學生學習成效的影響；
2. 提升學生學習與創新、資訊科技素養和問題解決能力為目前教育的重點方向。本研究探討積木式程式設計對學生程式設計能力、問題解決能力、運算思維能力和學習動機的影響有其必要性；
3. 先前研究顯示教育階段（Xu et al., 2019）、國家地區（Oluk & Saltan, 2015; Shanmugam et al., 2019）、實驗人數（Nam et al., 2010）等因素對學習成效有不同

的影響，再者程式設計教學方式一直是教育者關注的問題（Köse, 2010）。Peng et al.（2019）以專題導向學習（project-based learning）對大學生教學，結果發現能顯著提升程式設計能力；亦有研究以任務導向學習（task-based learning）方式，例如逐題問題解決任務（Cárdenas-Cobo et al., 2020; Su et al., 2014），皆發現能提升程式設計能力。又根據 Batista et al.（2016）指出若以專題導向學習製作遊戲設計與機器人，能促進學生的運算概念。

然而，先前後設分析尚無針對學習導向進行調節效果的分析。因此，本研究擴大分析積木程式影響學生學習成效的調節變項。研究目的如下：

1. 探討積木式程式設計教學對學生整體學習成效的影響；
2. 探討積木式程式設計教學對學生程式設計能力、問題解決能力、運算思維能力、學習動機的影響；
3. 探討不同調節變項（實驗人數、教學時間、教育階段、程式設計環境、學習導向、學習成效、研究地區）對學習效果量的影響。

文獻探討

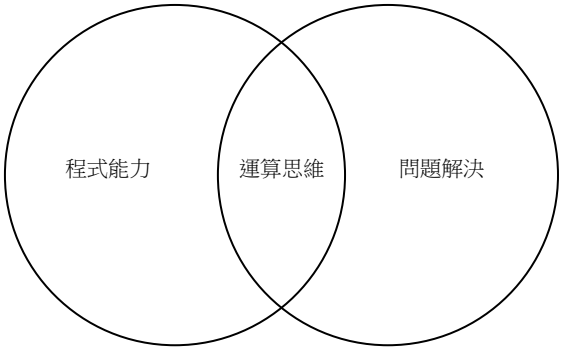
積木式程式設計

積木式程式設計意指將文字式程式設計打包為積木方塊，使學習者透過視覺化界面，拖曳積木組合程式語法，並於積木方塊設定參數功能，即可容易編輯和執行程式指令（Kelleher & Pausch, 2005）。

目前常見的積木式程式設計環境如下：Alice 由卡內基美隆大學於 1998 年推出，透過積木介面控制 3D 動畫，可將程式碼轉換為 C++ 或 JAVA（賴錦緣，2016）；Scratch 由麻省理工學院於 2006 年開發，使用積木編輯程式並能結合物聯網；APP Inventor 則為 Google 於 2009 年基於 Android 系統環境所開發，可連結陀螺儀、藍芽等硬體，培養手機程式概念（Papadakis et al., 2016）。積木式程式設計具有良好的除錯功能，若兩塊積木連結無法形成有效語法，系統會不予連結。因此，積木式程式設計環境可增加初學者的程式完成度，提高學習動機和成就感。

Zhang & Nouri（2019）認為積木式程式設計是資訊能力的表現，透過結合問題解決能力，可培養學生的運算思維能力。本研究參考 Zhang & Nouri（2019）的概念關係，提出運算思維能力、問題解決能力和程式設計能力的相關概念（見圖一）。此外，學生學習動機亦是積木式程式教學所關注的焦點，因此，本研究以上述四個依變項為學習成效範圍。

圖一：學習成效相關變項關係圖



積木程式設計學習成效

積木式程式設計學習對學生程式設計能力的影響

積木式程式設計學習環境提供友善的使用者介面，且不須精熟程式語法即可編寫，具容易學習、降低語法錯誤、提升學習興趣、降低認知負荷、提升高層次思考等優勢（林育慈、吳正己，2016）。先前有關積木式程式設計對不同教育階段學生程式能力的研究結果並不一致。Wang et al.（2009）發現 Alice 對 166 名台灣國中生程式能力的影響顯著高於接受 C++ 學習的學生。然而，王裕德等（2012）針對台灣 72 名高中生進行實驗，發現使用積木式程式設計與 C++ 程式設計能力並無顯著差異。

Korkmaz（2016b）的研究顯示 Scratch 與 C++ 對 49 名土耳其大學生的程式設計能力影響無顯著差異。Mihci & Donmez（2017）以土耳其 101 名大學生進行研究，發現使用 APP Inventor 與傳統物件導向式程式語言對程式設計能力影響無顯著差異。

積木式程式設計學習對學生問題解決能力的影響

問題解決能力指對於未知問題提出最佳解決方式或提出解決策略的能力，學生在面對未知情形時，若能有效運用自身經驗或演算法解決問題，可視為具問題解決能力（Woods et al., 1997）。

Schoenfeld（1987）提出解題六步驟模式，學生若採用此策略可幫助增進問題解決能力。程式設計亦呈現類似思考歷程。崔夢萍（1999）指出，學生使用 Logo 程式設計時，其學習歷程與問題解決的步驟十分相似。Lai & Yang（2011）亦認為積木式程式設計可使學生發展出具創意的問題解決方式，並以台灣 130 名六年級學生進行 Scratch 教學實驗，發現實驗組學生的問題解決能力顯著高於控制組學生。Oluk & Saltan（2015）以 Scratch 對土耳其 65 名六年級學生進行研究，發現 Scratch 與傳統教學方式

對學生問題解決能力的影響無顯著差異。Korkmaz（2016b）對土耳其 44 名大學生施以 Scratch 教學，發現使用 Scratch 介入的學生，其問題解決能力顯著高於使用 C++ 的學生。Dekhane et al.（2013）以 70 名大學生進行實驗，發現使用 APP Inventor 能顯著提升學生的問題解決能力。

積木式程式設計學習對學生運算思維能力的影響

運算思維能力是指透過電腦科學概念來理解並解決問題，其核心能力包含拆解問題、找出規律、歸納與抽象化以及設計演算法（Wing, 2006）。程式設計注重邏輯和演算的特性，能引導學生提出具系統性的問題解決方案（Ching et al., 2018）。Brennan & Resnick（2012）認為 Scratch 可培養學生多種運算概念，並進一步轉化為運算思維能力。Alice 和 APP Inventor 亦適合發展學生的運算思維能力（Touretzky et al., 2013）。Pérez-Marín et al.（2020）以 136 名國小學生進行實驗，發現以 Scratch 教導學生能顯著增加學生的運算思維能力。Tekerek & Altan（2014）以土耳其 60 名六年級學生進行研究，發現無論是接受傳統或 Scratch 教學的學生，其運算思維能力無顯著差異。Oluk & Saltan（2015）對土耳其 65 名六年級學生進行程式語法教學，結果亦顯示 Scratch 實驗組學生與傳統教學的控制組組學生在運算思維能力上無顯著差異。

積木式程式設計教學對學生學習動機之影響

Coto & Mora（2019）針對大學生學習程式設計調查，發現學生學習時最常感受的負面情緒為焦慮、挫折。積木式程式設計對學生學習動機的研究發現，馬來西亞 89 名大學生透過積木式程式設計學習後，其學習動機和自我效能皆優於以傳統方式學習的學生（Shanmugam et al., 2019）。何昱穎等（2010）針對 12 名修習 C++ 程式設計的大學生進行調查，發現經 Scratch 補救教學後，學生的學習愉悅感顯著提升。然而，Korkmaz（2016b）以土耳其 49 名大學生的研究指出，無論是透過 Scratch 或傳統 C++ 程式設計學習，其負向態度皆無顯著差異。

研究方法

研究篩選準則

本研究的文獻為近年發表的積木式程式設計相關文獻，包含以中、英語撰寫的期刊和會議論文，須採實驗研究法，提供效果量計算的量化數據。實驗組須使用積木式程式設計環境進行教學，控制組採用傳統教學（傳統、既有教學等詞彙）。本研究

的樣本文獻較少提及使用的教學策略，故將傳統教學界定為使用非積木式程式設計環境，如文字式程式設計（C++、JAVA）或 Flash 動畫等。上述準則中，最早且符合篩選準則的相關文獻出版自 2003 年，文獻蒐集截止時間為 2020 年 11 月。

研究樣本蒐集與分類

樣本文獻蒐集來源如下：“EBSCO-Academic Search Premier”、“Applied Science & Technology Source”、“ERIC”、“Library, Information Science & Technology Abstracts”、“PsycArticles”、“Teacher Reference Center”、“The Serials Directory”、“ACM Digital Library”、“IEEE Xplore Digital Library”；中文文獻則搜索「CEPS 中文電子期刊」、「CEPS 中文會議論文」、「Hyread 臺灣全文資料庫」和「中國知網」，並利用 Google 學術搜尋降低疏漏。文獻標題或關鍵詞、摘要須包含以下詞彙：“Scratch”、“Alice”、“Alice software”、“App Inventor”、“Block-based programming”、“Visual programming”、「積木式程式設計」或「圖形化編程」。

文獻在標題、關鍵詞和摘要瀏覽後，排除重複文獻並逐篇檢視，排除不符合篩選準則的文獻後有 52 篇。為避免整體效果量受到異常值（outlier）效果量的影響，計算效果量後將樣本信賴區間與整體信賴區間未重疊的文獻予以刪除，最後以 37 篇文獻進行編碼分析。

編碼準則

本研究依據 A. C. K. Cheung & Slavin（2016）提出教育研究後設分析編碼，包含樣本數、實驗方法、教育階段（國小、國高中、大學），並記錄各文獻實驗介入時間、積木式程式設計學習導向（依教學目標分為專案導向、任務導向）、專案類型（動畫設計、程式設計、遊戲設計、機器人）、作者、發表年代、標題、研究地區（亞洲、美洲、歐洲；華人地區、非華人地區）和使用工具。

依變項部分，由於各研究依變項的敘述略有不同，進行後設分析以使用依變項的涵蓋概念進行分類（馬信行，2007）。本研究依變項分為程式設計能力（程式設計成績、程式設計理論測驗分數等）、問題解決能力（問題解決能力、創造力、高階思考能力等）、運算思維能力（運算思維能力、邏輯思考能力等）和學習動機（自信、內在動機等）四項。

效果量計算

研究使用 Cohen’s d 與 Hedges’ g 為標準化效果量工具，使用 Comprehensive

Meta-analysis 第二版為計算工具。效果量採 Lipsey & Wilson（2001）提出的界定標準，當效果量小於 0.32 為小效果量，在 0.33 至 0.55 之間為中效果量；大於 0.56 則為大效果量。計算公式如下：

$$\text{Cohen's } d = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{(n_1+n_2-2)}}}$$

$\bar{X}_1$、$\bar{X}_2$ 和 n_1、n_2 分別代表實驗組和控制組的平均數和樣本數，S_1^2 和 S_2^2 代表兩組的標準差。Hedges' g 則如 Cohen's d 為效果量計算工具，但更適用於小樣本研究，相關公式如下：

$$J = 1 - \frac{3}{4(N-2)-1}$$

$$\text{Hedges' } g = J \times \text{Cohen's } d$$

同質性檢定部分，由於本研究蒐集的文獻涵蓋世界各地，亦包含各年齡層學生，可視為不同母群體，故採用隨機效果模式（random effect model）進行同質性檢定，以增進研究結果的推論（Borenstein et al., 2009）。

研究結果與結論

研究樣本描述統計

本研究共分析 37 篇量化文獻（表一），總樣本為 5,430 人，其中 2,463 位使用積木式程式設計環境，2,967 位使用傳統方式學習，期刊論文共 24 篇，會議論文共 13 篇。研究發現文獻數量自 2003 年起依年代增多，尤以 2015 至 2021 年最多（28 篇，75.68%）。45.95%（17 篇）的研究為 60 人以下的研究；54.05%（20 篇）的教學時間為 8 周或以上；以大學生為研究對象較多（17 篇，45.94%）；使用 Scratch 最多（20 篇，59.45%）。83.78%（31 篇）文獻使用專案導向進行教學，專案類型又以程式設計最多（16 篇，51.61%）。研究以探討程式設計能力的相關文獻最多（23 篇，62.16%）。亞洲文獻最多（27 篇，72.97%），研究地區以非華人地區較多（26 篇，70.27%）。

出版偏誤

本研究同時使用 Orwin's fail-safe N、Egger's test 和 trim and fill 方法進行出版偏誤檢定。Orwin's fail-safe N 部分，設定無效果量值為 0.1，可能影響結果的未尋獲或未顯著文獻篇數為 82 篇；Egger's test 雙尾檢定結果為 p = .45，顯示無明顯出版

表一：研究文獻編碼

文獻作者和年分	出版類型	教育階段	工具	研究地區	時長	學習導向、專案類型	學習成效
Al-Linjawi & Al-Nuaim（2010）	期刊	大學	Alice	沙烏地阿拉伯	7 周	動畫	程式能力
Al-Tahat（2019）	期刊	大學	Alice	科威特	16 周	動畫	程式能力
Brown et al.（2008）	會議	國小	Scratch	美國	4 節課	動畫	問題解決
Cárdenas-Cobo et al.（2020）	會議	大學	Scratch	厄瓜多	17 周	任務導向	程式能力
De Kereki（2008）	會議	大學	Scratch	烏拉圭	15 周	動畫	程式能力
Deng et al.（2020）	期刊	高中	其他	中國	8 周	程式	運算思維；學習動機
Durak（2018）	期刊	國小	Scratch	土耳其	10 周	動畫	程式能力；學習動機
Dwarika & de Villiers（2015）	會議	大學	Alice	南非	3 周	動畫	程式能力
Erol & Kurt（2017）	期刊	大學	Scratch	土耳其	14 周	程式	學習動機；程式能力
Gul et.al（2017）	期刊	大學	APP Inventor	巴基斯坦	20 周	程式	問題解決
Kim & Lee（2017）	期刊	混合	APP Inventor	韓國	—	動畫	問題解決；學習動機
Korkmaz（2015）	會議	大學	Scratch	土耳其	6 周	任務導向	程式能力；問題解決；運算思維；學習動機
Korkmaz（2016a）	期刊	大學	Scratch	土耳其	6 周	機器人；遊戲	程式能力；運算思維；學習動機
Korkmaz（2016b）	期刊	大學	Scratch	土耳其	6 周	機器人；遊戲	程式能力；問題解決；學習動機
Lai & Yang（2011）	會議	國小	Scratch	台灣	18 周	程式	問題解決
A. Y. H. Liao & Huang（2021）	會議	大學	APP Inventor	台灣	12 周	程式	程式能力；學習動機

表一（續）

文獻作者和年分	出版類型	教育階段	工具	研究地區	時長	學習導向、專案類型	學習成效
Master et al.（2017）	期刊	國小	其他	美國	20 分鐘	機器人	學習動機
Mihci & Donmez（2017）	期刊	大學	APP Inventor	土耳其	6 周	程式	程式能力
Moreno-León et al.（2016）	期刊	國小	Scratch	西班牙	4 周	遊戲	程式能力
Moskal et al.（2004）	會議	大學	Alice	美國	9 周	動畫	程式能力
Nam et al.（2010）	會議	國小	Scratch	韓國	4 周	程式	問題解決
Oh et al.（2015）	期刊	國小	Scratch	韓國	10 周	遊戲	問題解決
Oluk & Saltan（2015）	期刊	國小	Scratch	土耳其	6 周	程式	運算思維；問題解決
Papadakis et al.（2016）	期刊	高中	Scratch；APP Inventor	希臘	18 周	程式	程式能力；學習動機
Rodríguez-Martínez et al.（2020）	期刊	國小	Scratch	西班牙	—	程式	運算思維
Su et al.（2014）	期刊	國小	Scratch	台灣	6 周	任務導向	程式能力
Sykes（2007）	期刊	大學	Alice	加拿大	3 周	動畫	學習動機
Topalli & Cagiltay（2017）	會議	大學	Scratch	土耳其	18 周	程式	程式能力
Topalli & Cagiltay（2018）	期刊	大學	Scratch	土耳其	7 周	程式	程式能力
Tsai（2019）	期刊	大學	APP Inventor	台灣	17 周	程式	程式能力
Wang et al.（2009）	會議	高中	Alice	台灣	8 周	動畫	程式能力；學習動機
Yünkül et al.（2017）	期刊	國小	Scratch	土耳其	18 周	—	程式能力
吳純慧等（2019）	會議	中學	Scratch	台灣	16 周	任務導向	程式能力
林督閔等（2018）	會議	國小	Scratch	台灣	—	程式	程式能力
崔志生（2018）	期刊	國小	Scratch	中國	10 周	—	問題解決；學習動機
傅騫等（2019）	期刊	中學	其他	中國	8 周	程式	運算思維
鄧文博、張文蘭（2015）	期刊	中學	APP Inventor	中國	8 周	程式	問題解決

偏誤；trim and fill 方法顯示在隨機效果模型中，填補 2 篇負向效果量可從 0.32 下降至 0.30，可視為無明顯出版偏誤（Borenstein et al., 2009）。

整體效果量

根據圖一，程式設計能力包含運算思維，且問題解決能力亦包含運算思維，因此本研究將運算思維分別納入程式設計和問題解決能力的效果量。

圖二顯示整體平均效果量為 $g = 0.33$（中效果量）。當效果量為正值時，顯示積木式程式設計的學習成效優於傳統教學，負值則反之。效果量計算方式採各文獻的平均效果量，標準化常態分配數值 $Z = 6.29$，$p < .00$，且 95% 信賴區間不包含 0，顯示積木式程式設計教學的學習成效顯著優於傳統教學。

不同調節變項在學習成效的效果量分析

表二為不同調節變項在學習成效的分析，效果量計算方式採各文獻提供的獨立效果量，見表三。

研究設計變項

一、實驗人數

依據 Y. K. C. Liao & Chen（2007），將樣本數分為 60 人以下、61–120 人和 121 人以上三組。表二顯示 61 至 120 人組呈現中效果量（$g = 0.36$），60 人以下呈現小效果量（$g = 0.23$），組內檢定具顯著差異（Z = 5.61, 2.85, $p < .001, .01$）；121 人以上為小效果量（$g = 0.17$），組內檢定不具顯著差異（$Z = 1.67$）。實驗人數對效果量不具調節效果（$Q = 2.93$）。表三顯示實驗人數在學習動機影響具顯著差異（$Q = 11.02$, $p < .01$）；事後檢定結果顯示 61–120 人的學習動機顯著高於 121 人以上（$Q = 10.44$, $p < .01$），其餘則無顯著差異。

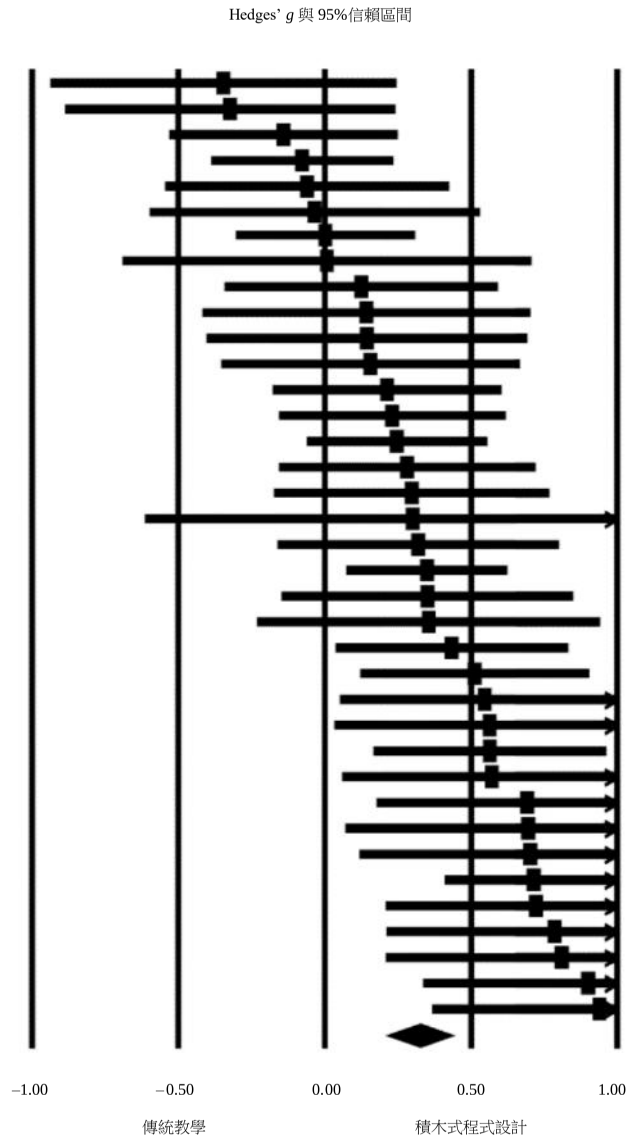
二、教學時間

Gall et al.（1996）指出實驗設計介入數個月或較長時間才可視為較強的介入效果，後設分析亦指出教學時間在 8 周以上有較好的學習動機（Richardson et al., 2017）。本研究分為 8 周以下和 8 周以上兩組。表二顯示 8 周以下組（$g = 0.09$）無組內顯著差異（$Z = 1.02$）；8 周以上組（$g = 0.36$）具組內顯著差異（$Z = 6.74$, $p < .001$）。教學時間在 8 周以上學習成效顯著高於 8 周以下（$Q = 7.30$, $p < .05$）。

表三顯示 8 周以上的程式設計能力與問題解決能力（$g = 0.47, 0.32$）顯著高於 8 周以下（$g = 0.10, –0.12$; $Q = 11.66, 5.30$, $p < .01, .05$）。其餘依變項則無顯著差異。

圖二：整體效果量森林

文獻	學習成效	樣本數		統計數據						
		傳統	積木	Hedges' *g*	*SE*	變異數	下限	上限	*Z*	*p*
Korkmaz（2016a）	混合	24	20	–0.346	.301	.091	−0.936	0.244	–1.151	.250
Korkmaz（2015）	混合	24	24	–0.323	.287	.082	−0.885	0.240	–1.124	.261
Mihci & Donmez（2017）	程式設計	47	54	–0.141	.198	.039	−0.529	0.248	−0.710	.477
Topalli & Cagiltay（2017）	程式設計	352	45	–0.077	.158	.025	−0.387	0.233	−0.488	.625
Oluk & Saltan（2015）	混合	34	31	–0.060	.247	.061	−0.544	0.424	−0.244	.807
Rodríguez-Martínez et al.（2020）	運算思維	23	24	–0.034	.287	.082	−0.597	0.528	−0.120	.905
Wang et al.（2009）	混合	85	81	0.003	.155	.024	−0.302	0.307	0.017	.987
De Kereki（2008）	程式設計	15	15	0.007	.355	.126	−0.689	0.704	0.021	.983
崔志生（2018）	混合	35	35	0.125	.237	.056	−0.340	0.590	0.527	.598
Korkmaz（2016b）	混合	21	28	0.142	.285	.081	−0.417	0.701	0.499	.618
Oh et al.（2015）	問題解決	25	25	0.144	.279	.078	−0.403	0.690	0.516	.606
Cárdenas-Cobo et al.（2020）	程式設計	29	29	0.155	.259	.067	−0.353	0.664	0.599	.549
Lai & Yang（2011）	學習動機	34	96	0.213	.199	.040	−0.177	0.603	1.071	.284
Dwarika & de Villiers（2015）	程式設計	48	55	0.231	.197	.039	−0.154	0.617	1.175	.240
Topalli & Cagiltay（2018）	混合	274	48	0.247	.157	.025	−0.060	0.554	1.576	.115
Gul et.al（2017）	問題解決	40	40	0.282	.223	.050	−0.154	0.718	1.266	.205
Yünkül et al.（2017）	程式設計	33	36	0.297	.240	.057	−0.173	0.766	1.238	.216
Moskal et al.（2004）	混合	10	10	0.300	.466	.217	−0.613	1.213	0.644	.520
Su et al.（2014）	程式設計	34	34	0.320	.244	.060	−0.160	0.799	1.307	.191
鄧文博、張文蘭（2015）	問題解決	104	103	0.349	.140	.019	0.075	0.623	2.501	.012
Durak（2018）	混合	30	32	0.351	.253	.064	−0.146	0.847	1.385	.166
林督閔等（2018）	程式設計	22	22	0.356	.299	.089	−0.230	0.941	1.191	.234
Deng et al.（2020）	混合	49	50	0.434	.202	.041	0.039	0.830	2.152	.031
Brown et al.（2008）	問題解決	40	73	0.512	.198	.039	0.123	0.901	2.582	.010
Master et al.（2017）	混合	32	32	0.547	.252	.063	0.053	1.041	2.172	.030
Al-Tahat（2019）	程式設計	26	30	0.564	.270	.073	0.035	1.092	2.091	.037
吳純慧等（2019）	程式設計	50	50	0.565	.202	.041	0.168	0.961	2.789	.005
Nam et al.（2010）	問題解決	30	30	0.571	.260	.068	0.061	1.081	2.195	.028
A. Y. H. Liao & Huang（2021）	混合	30	30	0.692	.263	.069	0.177	1.207	2.635	.008
傅騫等（2019）	問題解決	23	18	0.695	.318	.101	0.072	1.319	2.187	.029
Papadakis et al.（2016）	混合	18	35	0.702	.297	.088	0.120	1.284	2.365	.018
Tsai（2019）	程式設計	84	96	0.714	.153	.024	0.413	1.014	4.651	.000
Kim & Lee（2017）	混合	45	23	0.722	.261	.068	0.210	1.233	2.766	.006
Moreno-León et al.（2016）	程式設計	25	24	0.786	.292	.085	0.214	1.359	2.691	.007
Al-Linjawi & Al-Nuaim（2010）	程式設計	21	24	0.810	.306	.093	0.211	1.409	2.650	.008
Erol & Kurt（2017）	混合	26	26	0.900	.287	.082	0.337	1.462	3.134	.002
Sykes（2007）	程式設計	17	72	0.939	.291	.085	0.368	1.509	3.225	.001
整體效果量				0.331	.053	.003	0.228	0.435	6.288	.000



不同教育階段

本研究將教育階段分為國小、國高中和大學組，其中 1 篇文獻因實驗對象同時包含大學生和國小生而予以排除（Kim & Lee, 2017）。

表二顯示國高中組學習成效呈現中等效果量，國小和大學為小效果量，組內檢定皆呈現顯著差異（g = 0.36, 0.29, 0.19; Z = 3.54, 4.40, 2.44, p < .001, .001, .05）。教育階段的不同對於效果量不具調節效果（Q = 1.96），顯示積木式程式設計環境在各教育階段皆有正向成效。表三顯示教育階段的不同對於問題解決能力具有調節效果（Q = 8.16, p < .05）；事後檢定結果顯示國高中的問題解決能力效果（g = 0.42）顯著高於大學（g = –0.11）。但教育階段的不同在程式設計、運算思維和學習動機上則不具調節效果。

積木式程式設計環境

表二顯示三種積木式程式設計環境對學生學習成效皆具正效果量，APP Inventor 為中效果量（g = 0.58），Scratch 與 Alice 皆為小效果量（g = 0.20, 0.33）。組內檢定皆呈現顯著差異（Z = 4.94, 3.58, 2.58, p < .001 ,001, .05）。積木式程式設計環境的不同對於效果量具調節效果（Q = 8.97, p < .05），事後檢定結果顯示 APP Inventor 的學習成效顯著高於 Scratch（Q = 8.74, p < .01）。

表三顯示在學習動機部分，三種積木程式之間呈現顯著差異（Q = 9.23，p < .05），事後檢定結果顯示 APP Inventor 顯著高於 Scratch（Q = 9.13, p < .01）。本研究並無 Alice 與 APP Inventor 在運算思維的相關文獻，故不進行分組比較。其餘變項無顯著差異。

學習導向

一、專案導向與任務導向

表二顯示兩者皆具小效果量（g = 0.28, 0.25），但專案導向的組內檢定具顯著差異（Z = 5.12, p < .001），任務導向組內檢定不具顯著差異（Z = 1.99）。兩者之間無顯著差異（Q = 0.05）。表三顯示學習導向的不同對四項學習成效皆無調節效果。

二、專案類型

表二結果顯示動畫設計和程式設計為中效果量（g = 0.37, 0.41），組內檢定具顯著差異（Z = 3.94, 5.61, p < .001, .001）。遊戲設計和機器人呈負效果量，與傳統教學無顯著差異（g = –0.15, –0.06, Z = –1.14, –0.31）。專案類型的不同對於效果量具調節效果（Q = 18.58, p < .001），事後檢定結果顯示動畫設計的效果量顯著高於遊戲設計

表二：不同調節變項在學習成效效果量分析表

	效果量與 95%信賴區間							異質性			組間比較
	K	*g*	*SE*	下限	上限	*Z*	*p*	*Q*	*df*	*p*	
整體效果（*n* = 37）	—	0.33	.05	.23	.44	6.29	.00***				
研究設計變項											
樣本數											
60 人以下	37	0.23	.08	.07	.39	2.85	.00**				
61–120 人	26	0.36	.06	.00	.48	5.61	.00***				
121 人以上	10	0.17	.10	–.03	.37	1.67	.10				
同質性檢定								2.93	2	.23	
教學時間											
8 周以下	28	0.09	.09	–.08	.26	1.02	.31				2 > 1
8 周以上	41	0.36	.05	.25	.46	6.74	.00***				
同質性檢定								7.30	1	.01*	
教育階段											
國小	21	0.29	.07	.16	.41	4.40	.00***				
國高中	14	0.36	.10	.16	.56	3.54	.00***				
大學	36	0.19	.08	.04	.34	2.44	.02*				
同質性檢定								1.96	2	.38	
積木程式設計環境											
Alice	13	0.33	.13	.08	.59	2.58	.01*				2 > 3
App Inventor	10	0.58	.12	.35	.81	4.94	.00***				
Scratch	39	0.20	.06	.09	.31.	3.58	.00***				
同質性檢定								8.97	2	.01*	
學習成效											
程式設計	35	0.32	.06	.21	.43	5.50	.00***				
問題解決	20	0.18	.09	.01	.35	2.03	.04*				
運算思維	7	0.28	.10	.10	.47	2.95	.00***				
學習動機	25	0.27	.09	.10	.44	3.05	.00***				
同質性檢定								1.89	3	.60	
學習導向											
專案導向	59	0.28	.06	.17	.39	5.12	.00***				
任務導向	7	0.25	.13	.00	.49	1.99	.05				
同質性檢定								0.05	1	.82	
專案類型											
動畫設計	20	0.37	.09	.19	.55	3.94	.00***				1 > 3, 4
程式設計	24	0.41	.07	.27	.56	5.61	.00***				2 > 3, 4
遊戲設計	8	−0.15	.13	.13	–.39	−1.14	.25				
機器人	7	−0.06	.19	–.43	.31	−0.31	.75				
同質性檢定								18.58	3	.00***	
研究地區											
世界地區											
亞洲	59	0.21	.05	.11	.31	4.23	.00***				2 > 1
美洲	7	0.54	.12	.28	.80	4.02	.00***				
歐洲	6	0.59	.19	.21	.96	3.08	.00**				
同質性檢定								8.23	2	.02*	
華人地區與其他											
華人地區	26	0.30	.07	.17	.43	4.62	.00***				
非華人地區	47	0.25	.06	.12	.37	3.71	.00***				
同質性檢定								0.38	1	.54	

註：*K* 為效果量數，*g* 為效果量，*SE* 為標準誤。

* *p* < .05, ** *p* < .01, *** *p* < .001

表三：不同調節變項在學習成效相關變項效果量分析表

	程式設計		問題解決		運算思維		學習動機	
	g (SE, K)	*Q (K)*	*g (SE, K)*	*Q (K)*	*g (SE, K)*	*Q (K)*	*g (SE, K)*	*Q (K)*
整體學習成效（*K*）	0.32 (.06, 35)	—	0.18 (.09, 20)	—	0.29 (.10, 7)	—	0.27 (.09, 25)	—
研究設計變項								
樣本數								
60 人以下	0.33 (.09, 20)	0.02 (35)	0.04 (.15, 10)	2.28 (20)	0.14 (.17, 4)	1.38 (7)	0.19 (.16, 11)	11.02** (25)
60–120 人	0.32 (.09, 10)		0.26 (.11, 9)		0.39 (.12, 3)		0.49 (.09, 10)	
121 人以上	0.30 (.14, 5)		0.35 (.14, 1)		—		−0.05 (.14, 4)	
教學時間								
8 周以上	0.47 (.07, 20)	11.66** (33)	0.32 (.07, 11)	5.30* (18)	0.48 (.13, 3)	3.42 (6)	0.28 (.11, 13)	1.84 (24)
8 周以下	0.10 (.09, 13)		−0.12 (.18, 7)		0.10 (.16, 3)		0.20 (.17, 11)	
教育階段								
國小	0.35 (.13, 8)	5.09 (35)	0.14 (.12, 8)	8.16* (19)	0.14 (.19, 2)	4.86 (7)	0.33 (.10, 6)	0.52 (24)
國高中	0.49 (.15, 7)		0.42 (.10, 4)		0.48 (13, 3)		0.19 (.18, 7)	
大學	0.23 (.08, 20)		−0.11 (.17, 7)		−0.01 (.21, 2)		0.24 (.18, 11)	
積木程式設計環境								
Alice	0.40 (.10, 6)	2.67 (30)	0.33 (.35, 2)	4.32 (15)	—	(3)	0.26 (.25, 5)	9.23* (20)
App Inventor	0.58 (.26, 4)		0.43 (.13, 3)		—		0.76 (.16, 3)	
Scratch	0.25 (.07, 20)		0.05 (.12, 10)		0.11 (.16, 3)		0.20 (.09, 12)	
學習導向								
專案導向	0.34 (.15, 27)	0.32 (33)	0.20 (.09, 18)	(18)	0.28 (.10, 7)	(7)	0.27 (.10, 21)	0.01 (22)
任務導向	0.25 (.07, 6)		—		—		0.24 (.29, 1)	
專案類型								
動畫設計	0.40 (.09, 8)	11.22* (27)	0.57 (.14, 4)	11.36* (18)	—	2.69 (7)	0.29 (.17, 8)	8.62* (21)
程式設計	0.41 (.10, 15)		0.30 (.09, 9)		0.37 (.11, 5)		0.60 (.16, 5)	
遊戲設計	−0.10 (.21, 2)		−0.18 (.27, 3)		0.03 (.30, 1)		−0.11 (.19, 4)	
機器人	−0.16 (.20, 2)		−0.34 (.29, 2)		−0.05 (.28, 1)		0.12 (.27, 4)	
研究地區								
世界地區								
亞洲	0.31 (.06, 28)	1.80 (34)	0.16 (.09, 18)	3.30 (20)	0.32 (.10, 6)	1.39 (7)	0.14 (.09, 19)	12.44** (25)
美洲	0.10 (.25, 2)		0.51 (.20, 1)		—		0.74 (.16, 4)	
歐洲	0.54 (.21, 4)		−0.03 (.29, 1)		−0.03 (.29, 1)		0.69 (.40, 2)	
華人與非華人地區								
華人地區	0.48 (.06, 13)	5.96* (35)	0.31 (.09, 8)	2.04 (20)	0.48 (.13, 3)	4.64* (7)	0.15 (.12, 8)	1.17 (25)
非華人地區	0.23 (.08, 22)		0.08(.13, 12)		0.07 (.14, 4)		0.33 (.12, 17)	

註： *K* 為效果量數，*g* 為效果量，*SE* 為標準誤。

* $p < .05$, ** $p < .01$, *** $p < .001$

與機器人（Q = 10.58, 4.12, p < .01, .05），程式設計的效果量亦顯著高於遊戲設計與機器人（Q = 14.45, 5.44, p < .001, .05）。

表三顯示專案類型的不同對於程式設計能力（Q = 11.22, p < .05）、問題解決能力（Q = 11.36, p < .05）和學習動機（Q = 8.62, p < .05）具有顯著調節效果。事後檢定結果顯示，在程式設計能力部分，動畫設計顯著高於遊戲設計和機器人（Q = 4.32, 5.91, p < .05, .05），程式設計顯著高於遊戲設計和機器人（Q = 4.71, 6.38, p < .05, .05）；動畫設計和程式設計對問題解決能力的影響顯著高於機器人（Q = 7.87, 4.32, p < .01, .05），動畫設計對問題解決能力的影響顯著高於遊戲設計（Q = 5.80, p < .05）；學習動機部分，程式設計顯著高於遊戲設計（Q = 8.22, p < .05）。

學習成效

表二顯示，程式設計能力、問題解決、運算思維和學習動機皆為小效果量（g = 0.32, 0.18, 0.28, 0.27）。組內檢定皆具顯著差異（Z = 5.50, 2.03, 2.95, 3.05, p < .001, .05, .001, .001），學習成效對於效果量不具調節效果（Q = 1.89）。

研究地區

本研究分為亞洲、美洲和歐洲三地，排除 1 篇研究為非洲（Dwarika & de Villiers, 2015）。表二顯示歐洲具大效果量（g = 0.59），美洲為中效果量（g = 0.54），亞洲為小效果量（g = 0.21）。事後檢定結果顯示，美洲效果量顯著高於亞洲地區（Q = 5.30, p < .05），其餘地區則無顯著差異。各地區在學習動機呈現顯著差異（Q = 12.44, p < .01）（表三）。事後檢定結果顯示，美洲地區在學習動機上顯著高於亞洲地區（Q = 11.43, p < .01），其餘依變項則無顯著差異。

至於華人與非華人地區，研究結果顯示（表二）兩組皆呈現小效果量（g = 0.30, 0.25），且組內檢定呈現顯著差異（Z = 4.62, 3.71, p < .001, .001）。兩組學習成效無顯著差異（Q = 0.38）。華人地區的程式設計能力和運算思維顯著高於非華人地區（Q = 5.96, 4.64, p < .05, .05），其餘依變項則無顯著差異（表三）。

研究結果討論

整體效果量方面

本研究延伸先前研究，聚焦於積木式程式設計環境的文獻。結果發現積木式程式設計對整體學習成效為中效果量（g = 0.33），顯示積木式程式設計對學生學習成效具

顯著正向影響，與先前研究結果一致（Costa & Miranda, 2017; Xu et al., 2019），支持積木式程式設計對學生整體學習成效有效。

本研究結果顯示積木式程式設計教學對學生程式設計能力為小效果量（g = 0.32），這結果呼應 Scherer et al.（2019）的學習近遷移程式技能與程式知識的效果值為 0.75。本研究結果顯示在問題解決能力為小效果量（g = 0.18），與先前研究結果一致（Lai & Yang, 2011），Y. K. C. Liao（2000）後設分析程式語言對問題解決能力具顯著正向效果（g = 0.58）。

本研究顯示積木式程式設計對於培養學生運算思維為小效果量（g = 0.28），這結果與先前研究結果一致，支持運用積木式程式設計能提升學生的運算思維能力（Pérez-Marín et al., 2020）。但礙於運算思維相關文獻較少，僅蒐錄 6 篇文獻（7 效果量），未來研究可進一步分析。此外，本研究顯示積木式程式設計比傳統教學能顯著提升學習動機（g = 0.27），與 Xu et al.（2019）的結果一致（g = 0.19）。

不同調節變項對學習效果量的影響

研究結果發現實驗人數為 61–121 人時為中效果量（g = 0.36），60 人以下和 121 人以上亦具有小效果量（g = 0.23, 0.17），顯示實驗樣本人數太少或太大可能對學習成效產生影響，尤其對學習動機更為明顯。

教學時間在 8 周以上為中效果量，8 周以下為小效果量（g = 0.36, 0.09），8 周以上對程式設計影響顯著高於 8 周以下。因此，程式設計能力的培養以長期教學活動能產生較佳效果；8 周以下在程式設計、問題解決和運算思維的效果值較小，顯示教學時間太短，學生停留在語法學習階段，未能對問題解決或運算思維能力有深入影響。

對象為國高中學生呈現中效果量（g = 0.36），國小和大學生的學習成效皆呈現小效果量（g = 0.29, 0.19），各教育階段間無顯著差異，這結果與 Xu et al.（2019）的研究一致。本研究顯示教育階段對問題解決能力具顯著調節效果，其餘學習成效皆無顯著影響，大學生在問題解決和運算思維呈現負效果量，結果與 Hembree（1992）的研究部分相同。J. C. Y. Cheung et al.（2009）指出積木式程式環境對初學者或國中小學生容易學習，但對大學生程式設計缺乏彈性，影響學習成效。

比較三種積木式程式環境發現，APP Inventor 對學生整體學習成效呈現大效果量（g = 0.58），Alice 和 Scratch 則為小效果量（g = 0.33, 0.20），對學生學習成效皆顯著優於傳統教學。教師可依各環境特性進行教學，如 Alice 可用於 3D 動畫製作、Scratch 用於機器人程式設計、APP Inventor 適合於手機應用程式。再者，程式設計環境在學生學習動機上具顯著的調節效果，APP Inventor 顯著高於 Scratch，與過去研究相符。Papadakis et al.（2016）認為 APP Inventor 擁有開發手機應用程式的特性，比 Scratch 較易提升學生的學習動機。

不同學習導向的積木式程式教學發現，積木式程式設計以專案和任務導向學習的整體學習成效皆呈現小效果量（g = 0.28, 0.25），專案導向學習成效顯著高於傳統教學，且專案類型在動畫設計和程式設計具顯著較佳的學習成效。同時，動畫設計和程式設計皆呈現中效果量（g = 0.37, 0.41），支持使用動畫設計和程式設計為專案類型可顯著提升學生的整體學習成效。而遊戲設計和機器人則呈現負效果量（g = –0.15, –0.06），二者或因蒐錄效果量較少，較難呈現準確效果量。

專案類型在程式設計、問題解決和學習動機皆具顯著調節效果，以動畫和程式設計進行不同面向的學習成效具有較正向的影響。正如 Cooper et al.（2003）指出 Alice 提供動畫相關的物件讓學生設定參數，能降低認知負荷而專注於互動設計。而遊戲程式設計尚須學習邏輯運算，需要較高階的程式技巧才能設計遊戲關卡，較不易於短期達成效果。

按不同地區比較發現，歐洲在積木式程式設計對整體學習成效呈現大效果量（g = 0.59），美洲和亞洲則分別呈現中和小效果量（g = 0.54, 0.21），支持積木式程式設計在各地區教學的學習有效性。且美洲的整體效果量和學習動機顯著高於亞洲，原因可能是亞洲學生有較大的課業考試壓力，學習內容又較多，影響整體學習成效。亦有可能是本研究蒐集的亞洲文獻最多，導致差異性較大而影響結果。

以華人和非華人地區進行比較發現，兩組在整體學習成效皆呈現顯著的小效果量（g = 0.30, 0.25），組間皆具顯著差異。華人地區的程式設計能力和運算思維（g = 0.48, 0.48）顯著高於非華人地區（g = 0.23, 0.07）。學習動機部分，非華人地區為中效果量（g = 0.33），華人地區則為小效果量（g = 0.15）。這結果顯示積木式程式設計在華人地區程式設計和運算思維能力的教學上具有相當顯著的效果。

結論與建議

本研究旨在透過後設分析探討積木式程式設計教學對學生學習成效的影響，發現積木式程式設計具有中效果量，支持使用積木式程式設計對學生學習成效的助益，在程式設計、問題解決、運算思維和學習動機方面皆顯著優於傳統教學。

教學時間、程式設計環境、不同專案類型、研究地區在積木式程式設計教學對學生學習成效皆產生影響。教學時間以 8 周以上效果顯著高於 8 周以下，APP Inventor 效果顯著高於 Scratch，專案導向又以動畫設計、程式設計的效果較佳，研究地區則以美洲地區高於亞洲地區。

本研究延伸先前研究分析，可作積木式程式相關研究的參考依據。研究結果發現積木式程式設計在各教育階段皆具有正向學習成效，惟對大學生在問題解決和運算

思維能力上無明顯助益，建議高等教育教師或可採用高階程式進行教學，或兼用積木式和文字式程式。此外，長期教學對學生學習成效影響較大，對程式設計能力影響最為顯著。建議教師與研究者須規劃較長的課程以獲得較佳成效。本研究顯示在 61–120 人組顯著高於傳統教學的學習成效，可作未來研究人數的參考。

在積木式程式設計環境的選擇上，結果顯示 APP Inventor 的整體學習成效顯著高於 Scratch，其中學習動機項目呈現顯著差異。APP Inventor 由於其目的是設計手機應用程式，具有較明確且真實的目標，教師在設計 Alice 和 Scratch 課程時，應結合教學目標與現實問題解決，依據不同教學目標採用不同特性的積木式程式環境，以促進學生不同面向的學習成效。

本研究發現不同專案導向影響積木式程式設計的學習成效，其中動畫設計和程式設計的學習成效優於遊戲設計和機器人，教師可優先以動畫和程式設計兩專案類型進行教學。未來研究可據此延伸探究更多元的專案設計對學習成效的影響。本研究礙於任務導向相關文獻較少，建議未來可針對積木式程式設計的任務導向學習進一步分析。

美洲地區的整體學習成效顯著高於亞洲，在學習動機部分亞洲地區則顯著較低。在程式設計和運算思維部分，華人地區顯著高於非華人地區，支持積木式程式設計學習在華人地區的有效性。在設計積木式程式設計課程時，建議華人地區教師可普遍運用積木式程式教學培養學生的程式設計和運算思維能力，並以更多實際問題解決作教學範例，以提升學生的問題解決能力和內在學習動機。

本研究受限於文獻蒐集的篩選和蒐集準則，研究限制如下：

1. 本研究僅蒐集 2003 至 2021 年已出版的期刊和會議論文，而工作紀錄、學位論文、成果報告不在蒐集範圍，未來研究可加入更多類型的研究文獻進行分析。
2. 本研究選用樣本文獻中較常見的依變項作效果量分析，但仍無法涵蓋所有積木式程式設計學習成效的依變項，未來研究可再納入更多依變項進行分析。
3. 本研究在運算思維、遊戲設計、機器人和非亞洲的相關積木式程式設計文獻蒐錄較少，建議未來後設分析與實務研究可再延伸上述變項的相關研究。
4. 本研究以分析 Alice、APP Inventor 和 Scratch 積木式程式為主，建議未來可針對更多相關程式設計環境或不插電教學方式進行研究。
5. 本研究結果僅分析專案和任務學習導向的文獻，並未分析各教育階段教師所實施的教學策略對學習成效的影響，主要因為大多數文獻並未說明教學策略的實施方式，建議未來研究可針對不同教學策略運用於積木式程式教學進行深入探討。

參考文獻

王裕德、陳元泰、曾鈴惠（2012）。〈機器人問題導向程式設計課程對女高中學生學習程式設計影響之研究〉。《科學教育月刊》，第 354 期，頁 11–29。

何昱穎、張智凱、劉寶鈞（2010）。〈程式設計課程之學習焦慮降低與學習動機維持 —— 以 Scratch 為補救教學工具〉。《數位學習科技期刊》，第 2 卷第 1 期，頁 11–32。

吳純慧、許坤明、顏義和、方俞珮（2019，5 月）。〈數位遊戲式學習融入國中程式設計課程之學習成效〉。文章發表於第八屆工程、技術與科技教育學術研討會，彰化，台灣。

林育慈、吳正己（2016）。〈運算思維與中小學資訊科技課程〉。《教育脈動》，第 6 期，頁 5–20。

林督閔、林琨越、楊肅健、林秋斌（2018，10 月）。〈回授法對國小六年級生學習 Scratch 程式設計成效與態度之影響〉。文章發表於 2018 TANET 臺灣網際網路研討會，桃園，台灣。

馬信行（2007）。〈後設分析之方法論問題之探討〉。《αβγ 量化研究學刊》，第 1 卷，頁 170–183。

國家教育研究院（2019）。〈新課綱「程式設計」，學邏輯解問題〉。https://epaper.naer.edu.tw/edm.php?grp_no=2&edm_no=134&content_no=2672

崔志生（2018）。〈基於 Scratch 的小學 STEAM 教學設計與應用〉。《中國教育技術裝備》，第 19 期，頁 31–35。

崔夢萍（1999）。〈資訊教育中的創造思考學習歷程 —— 理論探討與研究之分析〉。《課程與教學季刊》，第 2 卷第 4 期，頁 9–26, 145。

傅騫、解博超、鄭婭峰（2019）。〈基於圖形化工具的編程教學促進初中生計算思維發展的實證研究〉。《電化教育研究》，第 4 期，頁 122–128。

鄧文博、張文蘭（2015）。〈基於 APP Inventor 培養中學生創造性思維的設計研究〉。《電化教育研究》，第 8 期，頁 95–99。

賴錦緣（2016）。〈Alice 程式設計環境中配對與個別之學習成效比較〉。《中科大學報》，第 3 卷第 1 期，頁 177–190。

Al-Linjawi, A. A., & Al-Nuaim, H. A. (2010). Using Alice to teach novice programmers OOP concepts. *Journal of King Abdulaziz University: Science*, *22*(1), 1–20. https://doi.org/10.4197/Sci.22-1.4

Al-Tahat, K. (2019). The impact of a 3D visual programming tool on students' performance and attitude in computer programming: A case study in Jordan. *Journal of Cases on Information Technology*, *21*(1), 52–64. https://doi.org/10.4018/JCIT.2019010104

Batista, A. L. F., Connolly, T., & Angotti, J. A. P. (2016, October). *A framework for games-based construction learning: A text-based programming languages approach*. Paper presented at the 10th European Conference on Games Based Learning, Paisley, Scotland.

Borenstein, M., Hedges, L. V., Higgins, J. P. T., & Rothstein, H. R. (2009). *Introduction to meta-analysis*. John Wiley & Sons.

Brennan, K., & Resnick, M. (2012, April). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the annual meeting of the American Educational Research Association, Vancouver, Canada.

Brown, Q., Mongan, W., Kusic, D., Garbarine, E., Fromm, E., & Fontecchio, A. (2008, June). *Computer aided instruction as a vehicle for problem solving: Scratch boards in the middle years classroom*. Paper presented at the 2008 Annual Conference and Exposition, Pittsburgh, Pennsylvania.

Cárdenas-Cobo, J., Puris, A., Novoa-Hernández, P., Galindo, J. A., & Benavides, D. (2020). Recommender systems and Scratch: An integrated approach for enhancing computer programming learning. *IEEE Transactions on Learning Technologies*, *13*(2), 387–403. https://doi.org/10.1109/TLT.2019.2901457

Cheung, A. C. K., & Slavin, R. E. (2016). How methodological features affect effect sizes in education. *Educational Researcher*, *45*(5), 283–292. https://doi.org/10.3102/0013189X16656615

Cheung, J. C. Y., Ngai, G., Chan, S. C. F., & Lau, W. W. Y. (2009). Filling the gap in programming instruction: A text-enhanced graphical programming environment for junior high students. *ACM SIGCSE Bulletin*, *41*(1), 276–280. https://doi.org/10.1145/1539024.1508968

Ching, Y. H., Hsu, Y. C., & Baldwin, S. (2018). Developing computational thinking with educational technologies for young learners. *TechTrends*, *62*(6), 563–573. https://doi.org/10.1007/s11528-018-0292-7

Cooper, S., Dann, W., & Pausch, R. (2003). Using animated 3D graphics to prepare novices for CS1. *Computer Science Education*, *13*(1), 3–30. https://doi.org/10.1076/csed.13.1.3.13540

Costa, J. M., & Miranda, G. L. (2017). Relation between Alice software and programming learning: A systematic review of the literature and meta-analysis. *British Journal of Educational Technology*, *48*(6), 1464–1474. https://doi.org/10.1111/bjet.12496

Coto, M., & Mora, S. (2019, June). *Are there any gender differences in students' emotional reactions to programming learning activities?* Paper presented at the XX International Conference on Human Computer Interaction, Gipuzkoa, Spain.

De Kereki, I. F. (2008, October). *Scratch: Applications in computer science 1*. Paper presented at the 2008 38th Annual Frontiers in Education Conference, Saratoga Springs, NY, U.S.

Dekhane, S., Xu, X., & Tsoi, M. Y. (2013). Mobile app development to increase student engagement and problem solving skills. *Journal of Information Systems Education*, *24*(4), 299–308.

Deng, W., Pi, Z., Lei, W., Zhou, Q., & Zhang, W. (2020). Pencil Code improves learners' computational thinking and computer learning attitude. *Computer Applications in Engineering Education*, *28*(1), 90–104. https://doi.org/10.1002/cae.22177

Durak, H. Y. (2018). Digital story design activities used for teaching programming effect on learning of programming concepts, programming self-efficacy, and participation and analysis

of student experiences. *Journal of Computer Assisted Learning*, *34*(6), 740–752. https://doi.org/10.1111/jcal.12281

Dwarika, J., & de Villiers, M. R. (2015). Use of the Alice visual environment in teaching and learning object-oriented programming. In R. J. Barnett, L. Cleophas, D. G. Kourie, D. B. Le Roux, & B. W. Watson (Eds.), *Proceedings of the 2015 Annual Research Conference on South African Institute of Computer Scientists and Information Technologists* (Article 14). Association for Computing Machinery. https://doi.org/10.1145/2815782.2815815

Erol, O., & Kurt, A. A. (2017). The effects of teaching programming with Scratch on pre-service information technology teachers' motivation and achievement. *Computers in Human Behavior*, *77*, 11–18. https://doi.org/10.1016/j.chb.2017.08.017

Gall, M. D., Borg, W. R., & Gall, J. P. (1996). *Educational research: An introduction* (6th ed.). Longman.

Gul, S., Asif, M., Ahmad, S., Yasir, M., Majid, M., & Malik, M. S. A. (2017). A survey on role of Internet of things in education. *International Journal of Computer Science and Network Security*, *17*(5), 159–165.

Hembree, R. (1992). Experiments and relational studies in problem solving: A meta-analysis. *Journal for Research in Mathematics Education*, *23*(3), 242–273. https://doi.org/10.5951/jresematheduc.23.3.0242

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys*, *37*(2), 83–137. https://doi.org/10.1145/1089733.1089734

Kim, S. W., & Lee, Y. (2017). The effects of programming education using App Inventor on problem-solving ability and self-efficacy, perception. *Journal of the Korea Society of Computer and Information*, *22*(1), 123–134. https://doi.org/10.9708/jksci.2017.22.01.123

Korkmaz, Ö. (2015, June). *The effect of Scratch game based learning students' activities on students' computer programming academic achievement and attitude*. Paper presented at the ERPA International Congresses on Education, Athens, Greece.

Korkmaz, Ö. (2016a). The effect of Scratch- and Lego Mindstorms Ev3-based programming activities on academic achievement, problem-solving skills and logical-mathematical thinking skills of students. *Malaysian Online Journal of Educational Sciences*, *4*(3), 73–88.

Korkmaz, Ö. (2016b). The effect of Scratch-based game activities on students' attitudes, self-efficacy and academic achievement. *International Journal of Modern Education and Computer Science*, *8*(1), 16–23. https://doi.org/10.5815/ijmecs.2016.01.03

Köse, U. (2010). A web based system for project-based learning activities in "web design and programming" course. *Procedia — Social and Behavioral Sciences*, *2*(2), 1174–1184. https://doi.org/10.1016/j.sbspro.2010.03.168

Lai, A. F., & Yang, S. M. (2011, September). *The learning effect of visualized programming learning on 6th graders' problem solving and logical reasoning abilities*. Paper presented at the 2011 International Conference on Electrical and Control Engineering, Yichang, China.

Liao, A. Y. H., & Huang, S. P. (2021). The development and evaluation of a smart E-learning platform for programming instruction. In L. Barolli, A. Poniszewska-Maranda, & H. Park (Eds.), *Innovative mobile and Internet services in ubiquitous computing. IMIS 2020. Advances in intelligent systems and computing* (Vol. 1195, pp. 415–425). Springer. https://doi.org/10.1007/978-3-030-50399-4_40

Liao, Y. K. C. (2000). A meta-analysis of computer programming on cognitive outcomes: An updated synthesis. In J. Bourdeau & R. Heller (Eds.), *Proceedings of ED-MEDIA 2000 — World conference on educational multimedia, hypermedia & telecommunications* (pp. 598–604). Association for the Advancement of Computing in Education.

Liao, Y. K. C., & Bright, G. W. (1991). Effects of computer programming on cognitive outcomes: A meta-analysis. *Journal of Educational Computing Research*, *7*(3), 251–268. https://doi.org/10.2190/E53G-HH8K-AJRR-K69M

Liao, Y. K. C., & Chen, Y. W. (2007). The effect of computer simulation instruction on student learning: A meta-analysis of studies in Taiwan. *Journal of Information Technology and Applications*, *2*(2), 69–79. https://doi.org/10.6302/JITA.200709_2(2).0003

Lipsey, M. W., & Wilson, D. B. (2001). *Practical meta-analysis*. Thousand Oaks, CA: Sage.

Master, A., Cheryan, S., Moscatelli, A., & Meltzoff, A. N. (2017). Programming experience promotes higher STEM motivation among first-grade girls. *Journal of Experimental Child Psychology*, *160*, 92–106. https://doi.org/10.1016/j.jecp.2017.03.013

Mihci, C., & Donmez, N. O. (2017). Teaching GUI-programming concepts to prospective K12 ICT teachers: MIT App Inventor as an alternative to text-based languages. *International Journal of Research in Education and Science*, *3*(2), 543–559. https://doi.org/10.21890/ijres.327912

Mihci, C., & Özdener, N. (2014, March). *Programming education with a blocks-based visual language for mobile application development*. Paper presented at the 10th International Conference on Mobile Learning, Madrid, Spain.

Moreno-León, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it belong in the K-12 curriculum? *Journal of Information Technology Education: Research*, *15*, 283–303. https://doi.org/10.28945/3521

Moskal, B., Lurie, D., & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. *ACM SIGCSE Bulletin*, *36*(1), 75–79. https://doi.org/10.1145/971300.971328

Nam, D., Kim, Y., & Lee, T. (2010, November). *The effects of scaffolding-based courseware for the Scratch programming learning on student problem solving skill*. Paper presented at the 18th International Conference on Computers in Education, Putrajaya, Malaysia.

New Media Consortium. (2017). *Horizon report: K–12 edition, 2009–2017*. https://library.educause.edu/resources/2017/12/horizon-report-k-12-edition-2009-2017/

Oh, J. C., Lee, J. H., Kim, J. A., & Kim, J. H. (2012). Development and application of STEAM based education program using Scratch: Focus on 6th graders' science in elementary school. *The Journal of Korean Association of Computer Education*, *15*(3), 11–23.

Oluk, A., & Saltan, F. (2015). Effects of using the Scratch program in 6th grade information technologies courses on algorithm development and problem solving skills. *Participatory Educational Research*, *2*(Special Issue II), 10–20. https://doi.org/10.17275/per.15.spi.2.2

Papadakis, S., Kalogiannakis, M., Zaranis, N., & Orfanakis, V. (2016). Using Scratch and App Inventor for teaching introductory programming in secondary education: A case study. *International Journal of Technology Enhanced Learning*, *8*(3–4), 217–233. https://doi.org/10.1504/IJTEL.2016.10001505

Peng, J., Wang, M., Sampson, D., & van Merriënboer, J. J. G. (2019). Using a visualization-based and progressive learning environment as a cognitive tool for learning computer programming. *Australasian Journal of Educational Technology*, *35*(2), 52–68. https://doi.org/10.14742/ajet.4676

Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and Scratch to teach computer programming to children? *Computers in Human Behavior*, *105*, Article 105849. https://doi.org/10.1016/j.chb.2018.12.027

Richardson, J. C., Maeda, Y., Lv, J., & Caskurlu, S. (2017). Social presence in relation to students' satisfaction and learning in the online environment: A meta-analysis. *Computers in Human Behavior*, *71*, 402–417. https://doi.org/10.1016/j.chb.2017.02.001

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: An experiment with sixth-grade students. *Interactive Learning Environments*, *28*(3), 316–327. https://doi.org/10.1080/10494820.2019.1612448

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology*, *111*(5), 764–792. https://doi.org/10.1037/edu0000314

Schoenfeld, A. H. (1987). What's all the fuss about metacognition? In A. H. Schoenfeld (Ed.), *Cognitive science and mathematics education* (pp. 189–215). Lawrence Erlbaum.

Shanmugam, L., Yassin, S. F., & Khalid, F. (2019). Enhancing students' motivation to learn computational thinking through mobile application development module (M-CT). *International Journal of Engineering and Advanced Technology*, *8*(5), 1293–1303.

Su, A. Y. S., Yang, S. J. H., Hwang, W. Y., Huang, C. S. J., & Tern, M. Y. (2014). Investigating the role of computer-supported annotation in problem-solving-based teaching: An empirical study of a Scratch programming pedagogy. *British Journal of Educational Technology*, *45*(4), 647–665. https://doi.org/0.1111/bjet.12058

Sykes, E. R. (2007). Determining the effectiveness of the 3D Alice programming environment at the Computer Science I level. *Journal of Educational Computing Research*, *36*(2), 223–244. https://doi.org/10.2190/J175-Q735-1345-270M

Tekerek, M., & Altan, T. (2014). The effect of Scratch environment on student's achievement in teaching algorithm. *World Journal on Educational Technology*, *6*(2), 132–138.

Topalli, D., & Cagiltay, N. E. (2017, November–December). *An enriched introduction to*

programming course with Scratch. Paper presented at the International Conference on Information Theoretic Security, Hong Kong, China

Topalli, D., & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers and Education*, *120*, 64–74. https://doi.org/10.1016/j.compedu.2018.01.011

Touretzky, D. S., Marghitu, D., Ludi, S., Bernstein, D., & Ni, L. (2013, March). *Accelerating K-12 computational thinking using scaffolding, staging, and abstraction*. Paper presented at the 44th ACM Technical Symposium on Computer Science Education, New York, NY, U.S. https://doi.org/10.1145/2445196.2445374

Tsai, C. Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior*, *95*, 224–232. https://doi.org/10.1016/j.chb.2018.11.038

Umapathy, K., & Ritzhaupt, A. D. (2017). A meta-analysis of pair-programming in computer programming courses: Implications for educational practice. *ACM Transactions on Computing Education*, *17*(4), Article 16. https://doi.org/10.1145/2996201

Wang, T. C., Mei, W. H., Lin, S. L., Chiu, S. K., & Lin, J. M. C. (2009, October). *Teaching programming concepts to high school students with Alice*. Paper presented at the 2009 39th IEEE Frontiers in Education Conference, San Antonio, TX, U.S. https://doi.org/10.1109/FIE.2009.5350486

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Woods, D. R., Hrymak, A. N., Marshall, R. R., Wood, P. E., Crowe, C. M., Hoffman, T. W., Wright, J. D., Taylor, P. A.,Woodhouse, K. A., & Bouchard, C. G. K. (1997). Developing problem solving skills: The McMaster problem solving program. *Journal of Engineering Education*, *86*(2), 75–91. https://doi.org/10.1002/j.2168-9830.1997.tb00270.x

Wu, L., Looi, C. K., Multisilta, J., How, M. L., Choi, H., Hsu, T. C., & Tuomi, P. (2020). Teacher's perceptions and readiness to teach coding skills: A comparative study between Finland, Mainland China, Singapore, Taiwan, and South Korea. *The Asia-Pacific Education Researcher*, *29*(1), 21–34. https://doi.org/10.1007/s40299-019-00485-x

Xu, Z., Ritzhaupt, A. D., Tian, F., & Umapathy, K. (2019). Block-based versus text-based programming environments on novice student learning outcomes: A meta-analysis study. *Computer Science Education*, *29*(2–3), 177–204. https://doi.org/10.1080/08993408.2019.1565233

Yünkül, E., Durak, G., Çankaya, S., & Misirli, A. Z. (2017). The effects of Scratch software on students' computational thinking skills. *Electronic Journal of Science and Mathematics Education*, *11*(2), 502–517.

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K–9. *Computers and Education*, *141*, Article 103607. https://doi.org/10.1016/j.compedu.2019.103607

The Meta-analysis of Block-based Programming on Students' Learning Outcomes

Jen-I CHIU & Mengping TSUEI

Abstract

Block-based programming is receiving attention in different educational stages. It provides learners with the access to introductory computer science courses. This meta-analysis study aimed to examine the effects of block-based programming, comparing to traditional instruction, on students' learning outcomes, including programming skills, problem-solving skills, computational thinking, and learning motivation. Database search yielded 37 publications with 5,430 samples. Results indicated that block-based programming had a significantly larger effect size than did traditional instruction for overall learning outcomes (0.33). More specifically, we found positive effect sizes for programming skills, problem-solving skills, computational thinking, and learning motivation. These study findings indicated the benefits of block-based programming education for students' learning outcomes. "Study duration," "project types," and "research areas" as moderator variables had statistically significant impacts on the mean effect size in students' programming skill. "Project types" had statistically significant impacts on the mean effect size in problem-solving skills. "Research areas" had statistically significant impacts on the mean effect size in computational thinking skills. Moreover, "sample size," "block-based programming languages," "research areas," and "project types" had also statistically significant impacts on the mean effect size in students' learning motivation. The findings helped teachers implement block-based programming instruction for facilitating students' learning outcomes for students in different education stages.

Keywords: programming; meta-analysis; block-based programming; leaning outcomes

CHIU, Jen-I（邱仁一）is a doctoral student in the Graduate School of Curriculum and Instructional Communications Technology, National Taipei University of Education.

TSUEI, Mengping（崔夢萍）is Professor in the Graduate School of Curriculum and Instructional Communications Technology, National Taipei University of Education.