運用生成式 AI 工具輔助學習軟體系統架構之學習成效分析與探討

Analysis and discussion on learning effectiveness of using generative AI tools to assist learning software system architecture

孔崇旭，陳姵予 *，林峻劭
國立臺中教育大學 資訊工程學系
* bcs113117@gm.ntcu.edu.tw

【摘要】 本研究探討生成式人工智慧應用於物件導向軟體開發教學中的應用與學習成效，並開發一套結合生成式 AI 與統一建模語言(UML)的學習輔助平台。該平台允許學生以自然語言描述程式功能規格，並結合使用 UML 語言所繪製的軟體系統架構，自動生成所有模組功能並整合完整的軟體系統，以提升「軟體架構設計」的學習成效與系統規劃能力。研究採用實驗組與對照組設計，比較使用學習輔助平台與傳統學習方法的學生在軟體開發學習成效上的差異，並進行統計分析。結果顯示，學習輔助平台能顯著減少學生在開發過程中的時間成本，並提升中低成就學生對物件導向程式設計的理解與應用能力，相較於傳統方法更具學習成效。本研究證實生成式 AI 可作為程式設計教育的有效輔助工具，並為未來課程設計提供實證參考，進一步推動程式設計教學的創新與發展。

【關鍵字】 生成式人工智慧；輔助軟體系統開發；物件導向程式教學；統一建模語言；學習成效

Abstract: *Generative artificial intelligence has flourished in recent years, profoundly impacting various fields, including education. Despite this, research on programming education, particularly in object-oriented software system development, remains limited in effectively utilizing generative AI for instructional support. This study develops a teaching assistance platform integrating generative AI with Unified Modeling Language, enabling students to generate and integrate an entire system automatically. By combining natural language descriptions of program functions with code frameworks, this approach addresses the time-consuming nature of software development in educational settings and enhances students' ability to design complex software system architectures.*

Keywords: Generative Artificial Intelligence, Assisted Software System Development, Object-Oriented Programming Education, Unified Modeling Language, Learning Effectiveness

1. 前言

在軟體開發領域，傳統的開發流程通常包含需求分析、設計、開發、測試、上線等階段，這些過程往往需要大量人力與時間，導致開發週期冗長。此外，軟體開發的複雜性隨著系統規模的增長而提升，開發人員需要考慮多種架構設計模式、模組化開發方法以及效能優化策略，使得開發成本進一步提高。

近年來，生成式 AI 蓬勃發展，其能夠快速生成程式碼的特性，可以提高軟體開發的效率，使開發人員能夠更專注於架構設計與系統優化。儘管生成式 AI 十分便利，其在程式設計教學輔助方面仍存在一些限制，目前，生成式 AI 較難以建立較大且複雜的軟體系統，尤其在處理較複雜的操作流程（scenario）時，所生成的程式碼往往需要人工修改，且程式碼的品

質參差不齊。有時生成的程式碼語法簡單易懂，有時則過於複雜，使得初學者在沒有指導的情況下難以理解和應用。

物件導向程式設計的課程目標是培養軟體系統開發的高階人才，要能熟悉開發較大複雜的軟體系統，操作較複雜的操作流程，學生在學習物件導向程式設計時，往往面臨理解和應用複雜概念的挑戰。

如前所敘述可以得知，傳統的教學方式往往需要透過大量的理論講解與實作練習來培養學生的程式設計能力，但初學者在理解物件導向概念、設計模式以及大型系統架構時，常會因抽象程度過高而感到困難。此外，現有的學習資源雖然豐富，但缺乏能夠即時回饋並引導學生修正錯誤的工具，使得學習過程較為被動，進而影響學習成效，因此本論文開發一個運用生成式 AI 技術優點的系統，設計一個生成式 AI 物件導向軟體系統開發的輔助學習平台(Generative AI Object-Oriented System Development Platform，以下簡稱 GAIOOP)，用來協助軟體系統開發的教學，該系統能讓學生可以透過統一模型語言(Unified Modeling Language, UML) 以及自然語言以產生相對應之程式碼，輔助學生學習軟體系統開發和程式設計。

2. 相關研究

2.1. 生成式人工智慧

生成式人工智慧(Generative Artificial Intelligence, 簡稱生成式 AI)是一種強大的深度學習技術，通常是基於 RNN (Yu et al., 2019)或是(Vaswani et al., 2017)等人提出的 Transformer 等架構，訓練過程中，模型學習大量文字、圖片、影音或是其他類型的資料，從而能夠模仿人類生成文字、圖片、影音或其他多種形式的內容。這項技術已經在多個領域取得了重大突破，包括自然語言處理、圖像生成、音樂創作和程式碼生成等。目前常見的生成式 AI 模型有 GPT-4、LLaMA、Gemini、Grok 等。GPT-3(Generative Pre-trained Transformer) (Floridi & Chiriatti, 2020) 是由 OpenAI 所開發的第三代自我迴歸語言模型(Autoregressive model)，它使用深度學習來產生類似人類的文本，會將稱為提示語(prompt)的來源輸入開始產生單字、程式碼或其他資料序列，例如，它在機器翻譯中用於統計預測單字序列。2018 年 GPT 的第一次迭代使用了 1.1 億個學習參數(即神經網路在訓練過程中嘗試優化的值)，一年後，GPT-2 使用了 15 億個學習參數，如今，GPT-3 使用 1750 億個參數，它使用微軟 Azure 的 AI 超級電腦(Scott2020)進行訓練，其訓練費用非常昂貴，估計花費了 1200 萬美元。通過大量文本訓練，能製造出類似人類的文字，這有助於多種自然語言處理工作，如翻譯、摘要、問答和聊天應用。ChatGPT 能夠以對話方式理解並回應人類輸入，非常適合互動應用。此外，ChatGPT 可以根據不同資料集和任務進行微調，從而實現定制化，處理特定任務並產生更準確、更實用的輸出。LLaMA(Large Language Model Meta AI)是由 Meta AI 公司於 2023 年 2 月發布的大型語言模型集合，使用參數從 70 億到 650 億不等。LLaMA-13B 的效能超過了 GPT-3，且其體積卻小了 10 倍以上，LLaMA-65B 與其他大型語言模型相比也相當有競爭力。綜上所述，生成式人工智慧是一項強大且多用途的深度學習技術，其帶來了多領域的應用，如翻譯、問答系統和互動應用。生成式人工智慧的不斷演進和新型模型的出現，為各領域帶來更多可能性，並提供了更多元化的選擇，有助於實現更精確且實用的應用。

2.2. 生成式人工智慧在教學上的應用

目前已經有許多人在研究生成式 AI 在各種領域上的影響，人們普遍認為，從問題的自然語言描述自動合成原始碼可能會大大提高專業開發人員的生產力(Peng et al., 2023)，尤其是學生能夠輕鬆地自動生成程式碼並評估解決方案，這引起了對已有深入研究的入門程式設計

領域的關注(Becker & Quille, 2019)，像是(Baidoo-Anu & Ansah, 2023)探討了 ChatGPT 在教育領域上的影響，ChatGPT 作為一種強大的語言生成模型，具有出色的互動性，它能夠在教育環境中發揮多種作用，包括教學輔助、研究輔助以及評估等。這種技術的應用大大縮短了完成這些工作所需的時間，使教學和評估變得更加高效。然而，與其優勢相對應的是一些挑戰，ChatGPT 雖然能提供快速回答問題，但有時也可能生成錯誤答案或編造不存在的內容，這可能會對教學產生負面影響。在教育中，確保生成式人工智慧的安全並具有建設性至關重要，制定相應的指南和原則，確保這種技術的正確應用，避免可能的負面影響，這是一個迫切需要討論的問題。

(Cooper, 2023)針對 ChatGPT 在科學教育的應用上也做了一些探索性研究，在學生使用上，ChatGPT 在回答問題時通常會提供相關答案，但其可信度不足且有倫理問題，例如版權問題。在教學中，可利用 ChatGPT 生成 5Es表單以進行科學教學，但其輸出結果仍有待改善，且教師應對 AI 資源進行批判性評估，並根據情況調整，所以 AI 尚不能取代教師的專業知識。在研究中，ChatGPT 能協助作者修改句子、提升描述清晰度，但使用 AI 平台時需要更透明，也需要明確指導方針以確保科學知識的推進。

程式教育是現代教育體系中的重要組成部分，因為程式設計技能在當今數位化社會中的重要性越來越高。生成式 AI 技術的崛起為改進程式教學方式提供了新的機會和挑戰。知名的幾個基於大型語言模型的程式生成工具，包含 OpenAI 的 Codex、DeepMind 的 AlphaCode、Amazon 的 CodeWhisperer 等，都能自 6 動生成程式碼，從簡單的範例到複雜的演算法，這使得初學者能夠更容易地理解程式設計概念，並減少錯誤，同時節省了時間。(Denny et al., 2024; Sarsa et al., 2022)針對了 Codex 進行評估，並提出了利用生成式 AI 輔助學習程式設計的優點，其優點是能提供更多的練習資源，利用 Codex 生成的程式設計練習題目，約有 80% 的練習題是嶄新的，且與人工生成的練習題目相比，其數量也遠超過人工生成的，若是再加以修改，甚至能針對學生進行個性化生成，且 Codex 所生成的程式碼可以通過 CodeCombat 所制定的電腦科學一級(CS1)測驗，其成績表現在兩次考試中，得分皆是 15.7/20(78.5%)，若是使用 Codex 來輔助學生解決類似 CS1 問題的程式考試，普遍有使用 Codex 學生的成績，相較於未使用 Codex 的學生還好（(Finnie-Ansley et al., 2022)。

其他研究表明，這類工具在處理 Insertion Sort 或是 Tree Traversal 等常見的演算法問題時，其所生成的程式碼較正確有效(Dakhel et al., 2023)，也可以為資料結構和演算法等課程中的典型問題生成正確程式碼(Finnie-Ansley et al., 2023)。

2.3. 物件導向技術學習之困難

多年以來，物件導向程式設計課程的輟學率和不及格率居高不下，因此吸引許多研究者探討其成因與解決方法，根據(Tan et al., 2009)的研究，他們調查了 182 位大學生對於學習程式語言時所面臨的困難，他們在設計解決特定任務的程式、分割功能成各個程式、學習語法、找出程式錯誤等方面感到挑戰。此外，在程式語言的各種主題中，封裝、繼承和多型是他們最難以理解的部分，但這三部分恰好是 Java 中最為重要的概念。該研究也發現實作練習、小組練習和個人練習，都能有效增進學生的學習。提供範例程式和練習題目也被認為是最有幫助的學習材料。而且，導致學生成績不好的主要原因中，包含了練習範例過少和教學大綱過於重視理論知識等問題。

在教學中，評估學生的程式設計作業往往僅著眼於最終結果，而忽略了學生在解決問題過程中的發展和學習， (Hosseini et al., 2014)的研究強調了評估學生 Java 程式設計作業時應考慮到過程和結果的平衡。不僅僅關注最終成果，而是對學生在解決問題過程中的學習和進

步進行更全面的評估，這將有助於更好地指導學生的學習策略並提高其程式設計能力。最後的結論是，約有 77.63%的學生通常從簡單的程式開始，隨著概念的增加和測試案例的擴展，逐步提升程式的正確性，且能夠有效改善學習成效。

綜合以上觀點，我們了解到學生在學習物件導向程式語言時所遇到的困難，尤其是在 Java 的封裝、繼承和多型等重要概念的理解上。因此，未來的教學方法和資源提供應更加注重實作練習，提供更多的範例程式和練習題目，以幫助學生更好地理解並應用這些核心概念。通過改善教學方式，我們希望降低學生學習曲線的陡峭度，提高學生對於程式語言的理解和應用能力，從而有效提升學生的學習成效。透過對學生程式設計作業的評估的研究，我們意識到過程中的發展和學習對於程式設計能力的提升至關重要。而在我們開發的輔助學習平台中，我們特別強調了這一點。除了強調最終成果外，我們平台所提供的程式碼版本控制功能，使得學生能夠追踪其程式作業的演進過程。這使得學生能夠更好地理解和分析自己作業的每一個階段，從而更有效地提升其程式設計能力，而且通過觀察和評估學生程式碼在不同階段的變化，我們也能夠更全面地了解他們的學習路徑和程式設計思維模式，從而更有針對性地提供指導。

3. 研究目標與方法

3.1. 研究目標

本研究旨在探討生成式人工智慧（Generative AI）在輔助學習軟體系統開發中的應用，並評估其對學生學習成效的影響。具體目標如下：

3.1.1 開發生成式 AI 輔助學習平台

設計並實作一套結合生成式 AI 與統一建模語言的學習輔助平台（GAIOOP），幫助學生透過自然語言描述程式功能，並自動生成對應的軟體架構與程式碼，以提升學習效率與理解能力。

3.1.2. 比較 AI 輔助與傳統學習方法的成效

採用實驗組與對照組設計，分析使用 GAIOOP 學習的學生與採用傳統教學方法的學生，在學習成效、程式設計能力與系統開發能力上的表現差異，並透過統計分析驗證其有效性。

3.2. 研究對象

本研究以臺灣中部某大學資訊工程學系修習「物件導向程式設計」課程之學生，共 34 位，主要修習課程之學生為資訊工程學系二年級的學生。

3.3. 研究架構

本研究之研究架構圖如圖 1 所示，各項說明如下：

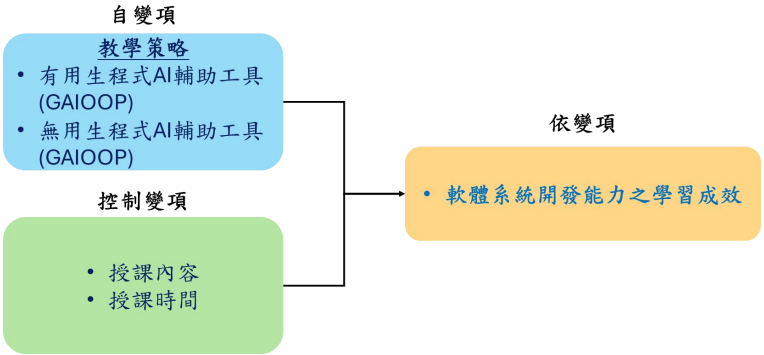

圖 1 實驗架構圖

3.3.1. 自變項 以「教學策略」為自變項，其中包含兩種策略的使用，分別為「有使用生成式 AI 輔助工具(GAIOOP)」及「無使用生成式 AI 輔助工具(GAIOOP)」。

(1)「有使用生成式 AI 輔助工具(GAIOOP)」：將學生學習時，將使用 GAIOOP 系統來輔助產生程式碼，在 GAIOOP 系統中會有結合 UML 的架構觀念及軟體規格，來產生透過 GAIOOP 系統與 ChatGPT 來溝通，並經過 GAIOOP 系統來做 Source Code 的產生來做系統的整合來產生完整的應用程式。

(2)「無使用生成式 AI 輔助工具(GAIOOP)」：將學生學習時，將自行使用繪製 UML 的架構觀念，學生依據軟體規格，自行撰寫程式碼，自行產生 Source Code，並自行來做系統整合產生完整的應用程式。

3.3.2. 依變項 研究中的依變項為「軟體系統開發能力之學習成效」，其中「軟體系統開發能力之學習成效」以開發軟體系統為主，分別使用前後測採上機實作評量，物件導向使用技術採上機實作評量。

3.3.3. 控制變項 為了減低自變項以外的其他變項對實驗效果產生影響，本研究在實驗過程中盡量掌控除了自變項之外的其餘教學條件，並使其盡可能相同。實驗進行時兩組之教學者皆由研究者親自擔任，兩組教學時間同為 16 節課，教學內容均以物件導向程式設計課程內容為主。

3.4. 研究流程

研究實施流程如圖 2 所示，各項說明如下：

1. 一般上課：包含基本物件導向能力技術的培養 7 週，期中上機考試 1 週。
2. 分組：依據教學方式分成 2 組，實驗組使用「GAIOOP 輔助教學系統」，對照組使用「一般 PPT 教學教材」。
3. 教學實驗：共 6 週，實驗組使用「GAIOOP 輔助教學系統」，對照組使用「一般 PPT 教學教材」，並實施 2 週的後測，並收集成績資料。

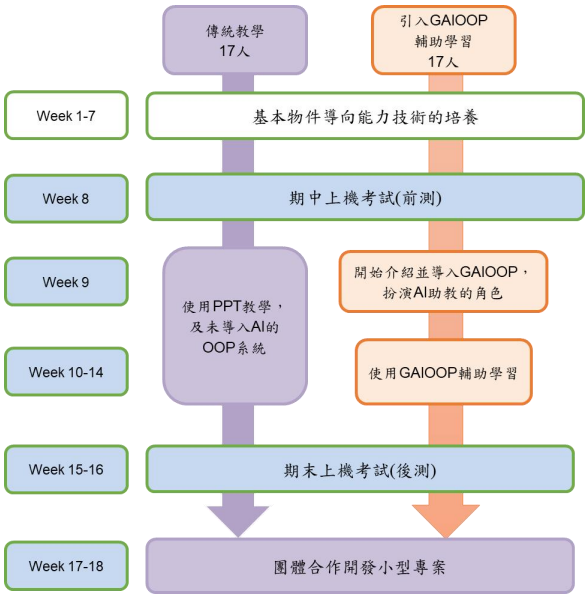

圖 2 研究實施流程圖

3.5. 研究工具與使用流程

3.5.1. GAIOOP 輔助教學平台系統功能 GAIOOP 輔助教學教學平台的系統功能有規格表單，用以引導學生填寫函式的細部需求功能規格；程式碼解析器用於學生上傳程式架構時，將多

個檔案與整串的程式架構解析承擔個函式；線上程式碼編輯器可以讓學生在線上即時修改生成的程式碼，或者式修改生成式 AI 生成的程式錯誤；版本控制可以讓學生倒回任一歷史版本的需求規格表單與程式碼；儲存容器負責分類與儲存學生上傳的程式框架、需求規格表單、完成生成後的完整軟體系統程式；Log 檔紀錄使得教師可以詳細分析每個學生與生成式 AI 的問答紀錄、使用的 token 數量與時間；自動程式碼合併功能則是負責將所有生成完成後的程式碼自動合併至該函式對應的物件。

3.5.2. 使用流程 本研究所開發的 GAIOOP 輔助教學平台使用流程如圖 3 所示，各項說明如下：

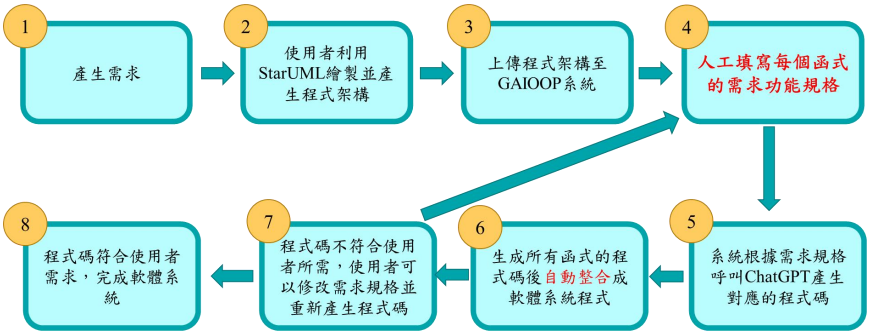

圖 3 使用流程圖

學生在產生需求後，先於 StarUML 上繪製類別圖(Class Diagram)，並通過 StarUML 中的 Java generate code 插件，將類別圖轉換成空白的程式架構。接著將程式架構上傳至 GAIOOP 輔助教學平台，並在 GAIOOP 輔助教學平台完成程式碼的分割後，以單個函示為單位依序填寫每個函式的需求功能規格，並在填寫完後送出需求規格，並等待生成式 AI 生成程式。

當所有函式的需求功能規格填寫完後，可以將各個函式自動整合成軟體系統程式。

4. 實驗結果分析

本研究旨在探討生成式 AI 輔助學習軟體系統（GAIOOP）是否能夠提升學生在軟體系統開發與程式設計方面的學習成效。為了達成此目標，我們依據學生的前測成績進行分組，並於學期末進行後測，以評估學習成效。後測採用期末上機評量的方式，包含五道難度相同的試題，學生將隨機抽取一道題目，需從無到有撰寫出一個完整的軟體系統，以進行上機實作考試。本研究採用共變異數分析（ANCOVA）進行分析來評估 GAIOOP 對學習成效的影響。以下為詳細的結果分析。

4.1. 共變異數分析（ANCOVA）

為確認 ANCOVA 分析使用前提是否成立，進行了迴歸係數同質性檢定。交互作用項（group × pretest）未達顯著水準（$p = 0.575 > 0.05$），表示實驗組與對照組之間的前測與後測之回歸斜率相近，符合迴歸係數同質性假設。因此可進一步使用 ANCOVA 比較兩組在後測成績的差異。表 1 為 ANCOVA 分析的結果。

分析顯示，在控制前測成績後的後測成績比較結果，實驗組後測平均數為 40.28，高於對照組的 24.69，調整後平均數分別為 41.37 與 25.17，顯示 GAIOOP 對學習成效有正向影響，但差異未達統計顯著性（$F = 2.606$，$p = 0.117 > 0.05$）。

表 1 ANCOVA 分析結果

變異數	組別	個數	平均數	調整後平均數	標準差	*F*	*p*
教學策略	實驗組	17	40.28	41.37	29.8	2.606	.117
	對照組	17	24.69	25.17	27.3		

因此，本研究將高分群分離，實驗組與對照組分別減去 4 個最高分的數據(17*0.75=12.75)，以此取得實驗數據中的後 75%數據，用來分析中低分組。首先進行迴歸係數同質性檢定。交互作用項（group × pretest）未達顯著水準（$p = 0.127 > 0.05$），符合迴歸係數同質性假設。因此可進一步使用 ANCOVA 比較兩組在後測成績的差異。表 2 為中低分群 ANCOVA 分析結果。

數據顯示，實驗組後測平均數為 27.4，高於對照組的 11.6，調整後平均數分別為 28.35 與 10.73，顯示 GAIOOP 對中低成就學生的學習成效具有正向影響，兩組間差異已達統計顯著（$F = 5.923$，$p = 0.0231 < 0.05$），相較於全體樣本的分析結果（$p = 0.117$），中低分組的 p 值顯著降低，顯示 GAIOOP 對中低分學生的影響更為明顯。

表 2 中低分組 ANCOVA 分析結果

變異數	組別	個數	平均數	調整後平均數	標準差	*F*	*p*
教學策略	實驗組	13	27.4	28.35	20.65	5.923	.0231
	對照組	13	11.6	10.73	14.34		

5. 結論與未來展望

本研究探討生成式人工智慧（Generative AI）在輔助學習軟體系統開發中的應用，並透過開發 GAIOOP 學習輔助平台，分析其對學生學習成效的影響。研究結果顯示，GAIOOP 在提升學生學習效率、改善物件導向程式設計開發能力方面具有正向效果。分析進一步顯示，移除高分群後，GAIOOP 對中低分程度學生的影響較為顯著，效果達顯著門檻，顯示其對中低分學生的助益較好，生成式 AI 輔助學習顯著提升了其軟體架構開發能力，相較於傳統教學方法更能降低學習門檻，幫助學生掌握軟體架構設計的核心概念。然而，分數較高的學生在使用 GAIOOP 時成效未顯著提升，可能是因為高分學生已具備較強的程式設計能力，生成的程式碼對其幫助有限，因此生成式 AI 在不同學習程度學生中的應用效果仍需進一步探討。

未來研究將聚焦於檢索增強生成（Retrieval-Augmented Generation, RAG）技術的應用，以提升 GAIOOP 生成程式碼的品質與準確性。RAG 技術可透過動態檢索相關學習資源與最佳實踐，輔助生成式 AI 在程式碼生成過程中進行自動校正，有效降低語法錯誤與邏輯缺陷，確保輸出內容的可讀性與可執行性。此外，GAIOOP 系統將新增學生使用行為紀錄功能，以支援後續的學習行為分析，也會加入系統之易用性量表、學習動機量表及學習態度量表進一步評估生成式 AI 輔助學習的效果。

6.誌謝

本研究承蒙教育部「教學實踐研究計畫」(計畫編號：PEE1137600)與國立臺中教育大學「高教深耕計畫」之經費支持與資源挹注，特此致謝。亦感謝本校師長與研究團隊在研究過程中的協助與建議，使本研究得以順利進行。

引用文獻

Baidoo-Anu, D., & Ansah, L. O. (2023). Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning. *Journal of AI*, *7*(1), 52-62.

Becker, B. A., & Quille, K. (2019). 50 years of cs1 at sigcse: A review of the evolution of introductory programming education research. Proceedings of the 50th acm technical symposium on computer science education,

Cooper, G. (2023). Examining science education in ChatGPT: An exploratory study of generative artificial intelligence. *Journal of science education and technology*, *32*(3), 444-452.

Dakhel, A. M., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M. C., & Jiang, Z. M. J. (2023). Github copilot ai pair programmer: Asset or liability? *Journal of Systems and Software*, *203*, 111734.

Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., Luxton-Reilly, A., Reeves, B. N., Santos, E. A., & Sarsa, S. (2024). Computing education in the era of generative AI. *Communications of the ACM*, *67*(2), 56-67.

Finnie-Ansley, J., Denny, P., Becker, B. A., Luxton-Reilly, A., & Prather, J. (2022). The robots are coming: Exploring the implications of openai codex on introductory programming. Proceedings of the 24th Australasian computing education conference,

Finnie-Ansley, J., Denny, P., Luxton-Reilly, A., Santos, E. A., Prather, J., & Becker, B. A. (2023). My ai wants to know if this will be on the exam: Testing openai's codex on cs2 programming exercises. Proceedings of the 25th Australasian Computing Education Conference,

Floridi, L., & Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, *30*, 681-694.

Hosseini, R., Vihavainen, A., & Brusilovsky, P. (2014). Exploring problem solving paths in a Java programming course.

Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of ai on developer productivity: Evidence from github copilot. *arXiv preprint arXiv:2302.06590*.

Sarsa, S., Denny, P., Hellas, A., & Leinonen, J. (2022). Automatic generation of programming exercises and code explanations using large language models. Proceedings of the 2022 ACM Conference on International Computing Education Research-Volume 1,

Tan, P.-H., Ting, C.-Y., & Ling, S.-W. (2009). Learning difficulties in programming courses: undergraduates' perspective and perception. 2009 International Conference on Computer Technology and Development,

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation*, *31*(7), 1235-1270.