教育研究與發展期刊　第二十二卷第一期　2026 年 3 月　頁 61-88

DOI　10.6925/SCJ.202603_22(1).0003

Evaluating the Applicability and Effectiveness of a CAL-Based Cognitive Apprenticeship System in Small and Large Class Instruction

Chien-Hung Lai

Assistant Professor, Department of Information and Computer Engineering, Chung Yuan Christian University

Liang-Chieh Ho

Research Assistant, Graduate Institute of Learning and Instruction, National Central University

Abstract

This study evaluates the cognitive apprenticeship model in introductory computer science courses across different class sizes to determine its impact on student outcomes and instructional practices. The research methodology involved a comparative analysis of midterm and final examination results, as well as programming assignment performance, in both small- and large-class settings. Key findings indicate that smaller classes foster more in-depth learning and more effective application of programming skills, thereby contributing significantly to long-term skill development and opportunities for deep learning. In contrast, no statistically significant differences were noted in the exam scores between small and large classes, suggesting limited impact on short-term assessment outcomes. Additionally, qualitative feedback revealed that students in smaller classes appreciated the personalized attention and systematic learning environment provided by the cognitive apprenticeship system. Conversely, students in larger classes experienced distractions and were critical of the flipped classroom and the associated grading systems. These insights emphasize the importance of class size in shaping educational strategies and student engagement in programming courses.

Keywords: cognitive apprenticeship; programming skills; class size instruction; learning assessment

Chien-Hung Lai's E-mail: soulwind@cycu.org.tw (Corresponding Author)

評估 CAL 認知學徒制對大小班規模教學的適用性與有效性研究

賴建宏 中原大學資訊工程學系助理教授
何亮頡 中央大學學習與教學研究所專任研究助理

摘　要

本研究評估了不同班級規模的計算機概論課程中的認知學徒模式，以確定其對學生成績和教學實踐的影響。我們的方法包括分析小班和大班的期中考和期末考以及程式設計作業。主要研究結果表明小班制可以促進更深入的學習和程式設計技能的有效應用，從而顯著增強長期技能發展和深度學習機會。相較之下，小班和大班之間的考試成績沒有顯著差異，這表明對短期評估結果的影響有限。此外，定性回饋指出，小班學生感受到個人化關注和系統化的學習環境，而大班學生則感到分心，並對翻轉教室和評分系統持批評態度。這些分析結果強調班級規模在制定教育策略和學生參與程式設計課程的重要性。

關鍵詞：認知學徒制、程式設計技能、班級教學、學習評估

【通訊作者】賴建宏 電子郵件：soulwind@cycu.org.tw

I. Introduction

In traditional apprenticeship models, instructional methods and content largely depend on the mentor's personal experience and preferences, potentially leading to a lack of systematic and structured knowledge transfer (Collins & Kapur, 2014). To address the shortcomings of traditional apprenticeships, the cognitive apprenticeship framework emphasizes active knowledge construction rather than passive knowledge reception, positing that cognitive processes play a more critical role in learning than the mere acquisition of practical skills.This learner-centered approach encourages and guides learners to think actively (Junus et al., 2019), enhancing their interest and motivation in learning, which positively impacts their performance (Raju et al., 2021). Within programming education, prior studies have demonstrated the effectiveness of applying cognitive apprenticeship principles by employing systems that function as cognitive mentors to guide and monitor learners' progress while teachers supplement knowledge and oversee progress (Lai et al., 2023).

Research in educational methodologies has increasingly recognized the limitations of traditional apprenticeship models, particularly in the context of technical disciplines that require not only theoretical knowledge but also the cultivation of practical skills. Traditional models often rely heavily on passive observation, which may not sufficiently engage students in the active learning processes necessary for complex skill acquisition. In contrast, cognitive apprenticeship is designed to strengthen students' active learning by integrating explicit teaching of thinking processes into the learning of skills. This approach makes the expert's tacit knowledge visible to students, thereby facilitating a deeper understanding and mastery of skills through guided practice and reflection(Butler et al., 2019; Cakmakci et al., 2025; Collins et al., 1991). Furthermore, a review of prior studies on cognitive apprenticeship reveals its positive impact on student performance (Tsukube & Matsuo, 2020) and shows that, when combined with other learning strategies, it can significantly enhance problem-solving abilities (Kuo et al., 2012). For instance, a study involving English as a Foreign Language (EFL) learners divided 60 participants into an experimental group and a control group. The experimental group received instruction based on the principles of cognitive apprenticeship, and the results

demonstrated that this group significantly outperformed the control group in oral proficiency. This finding suggests that cognitive apprenticeship is effective in enhancing language proficiency, even in relatively large participant groups (Ostovar-Namaghi et al., 2024). Similarly, another study focused on students with learning disabilities and English learners, albeit with a small sample size of only six participants. Despite the limited number of students, cognitive apprenticeship significantly improved their scientific explanation writing skills, indicating that the approach can yield positive educational outcomes even in small-scale implementations (Lee & Paz, 2021). In the domain of mathematics education, a study examining middle school students' problem-solving abilities divided 64 participants into an experimental group and a control group. The experimental group received cognitive apprenticeship instruction through multimedia teaching methods. The results revealed significant improvements in the problem-solving skills of the experimental group, highlighting the effectiveness of cognitive apprenticeship in medium-sized participant groups (Chethana & Menezes, 2017).

However, there is a notable lack of research exploring whether the number of participants influences the educational effects of cognitive apprenticeship (CA) and, consequently, student performance. Most previous studies have employed a fixed number of participants within a single experimental setup, without examining potential variations in outcomes across groups of differing sizes (Au et al., 2023; Doabler et al., 2018). Yet, class size is a critical factor that may significantly affect the effectiveness of CA. Core components of CA—such as coaching and scaffolding—require intensive teacher-student interaction to facilitate meaningful learning. Smaller class sizes enable more personalized attention and allow instructors to provide timely and tailored feedback that better addresses individual student needs, thereby enhancing both motivation and engagement. According to existing literature, small-class instruction not only improves learning outcomes but also enhances students' overall learning experiences (Richards et al., 2021). Thus, the moderating role of class size must not be overlooked when implementing CA—particularly in cognitively demanding disciplines such as programming, where interactive instructional support is essential.

Programming, being a technically oriented discipline, emphasizes the cultivation of practical skills. Students need not only to master theoretical knowledge but also to hone

skills through practical application, necessitating close guidance and feedback from teachers, especially when encountering errors and challenges. The personalized needs of programming students are pronounced, as students exhibit different learning styles, requiring teachers to adapt their guidance to each individual learner characteristics. In cognitive apprenticeship applications, a key aspect is that learners can observe the teacher's thought patterns through demonstrations, guidance, and scaffolding, allowing them to visualize, reflect on, and practice these processes (Brown et al., 1989). Moreover, programming often involves complex problem-solving that benefits immensely from the cognitive apprenticeship approach, where learners can observe the nuanced cognitive processes of more experienced programmers through guided demonstrations, scaffolding, and real-time feedback. This model facilitates a deeper understanding of not only "how" to perform specific tasks but also "why" certain solutions are effective, fostering a higher level of cognitive engagement and mastery of the material. Therefore, in larger student groups, teacher-student interactions may be challenged, potentially hindering personalized instruction and individual learning progress, especially in programming courses (Lai, 2023).

Although cognitive apprenticeship (CA) has been applied in programming education, research examining its effectiveness across different class sizes remains limited. The existing literature indicates that class size can significantly influence the efficacy of instructional strategies. For instance, Garcia (2023) notes that in larger classes, opportunities for personalized interaction between students and instructors are diminished, potentially impeding the individualized guidance crucial for effective programming instruction. Moreover, Wang et al. (2024) demonstrated that variations in class size significantly impact both cognitive and non-cognitive outcomes, further underscoring the importance of investigating CA across diverse classroom settings. In response to these gaps, the present study implements a CA-based instructional system in programming courses offered to both small and large class cohorts, aiming to evaluate its applicability and effectiveness under varying class-size conditions. This study addresses three research questions:

- RQ1 asks whether class size is associated with differences in implementation-oriented learning performance accrued across CAL assignments when instruction, content, assessments, and the CAL environment are held constant.

- RQ2 asks whether class size is associated with differences in performance on time-limited examinations, including the midterm and final, which are expected to be less sensitive to interactional density than iterative assignments.
- RQ3 asks how students describe the mechanisms that facilitate or hinder CA enactment under different class sizes, and whether observed process indicators such as help-seeking and monitoring intensity correspond with those accounts.

These questions link outcomes with process-level evidence to test a mechanism-based explanation of how class size conditions the activation of CA in a CAL-supported programming course.

II. Literature Review

A. Cognitive Apprenticeship in Skill Acquisition

CA is a pedagogical approach derived from constructivism that seeks to make the expert's thinking visible to learners. Unlike traditional apprenticeship, which focuses on physical skills, CA emphasizes the internalization of cognitive processes and the procedural "how-to" knowledge required for complex problem-solving (Lakshminarayanan & Rao, 2021). The framework consists of six core teaching strategies: modeling, coaching, scaffolding, articulation, reflection, and exploration.

Recent studies have validated the effectiveness of CA across various disciplines. For instance, in language learning, Ostovar-Namaghi et al. (2024) found that CA significantly enhanced oral proficiency in larger instructional groups, while Lee and Paz (2021) demonstrated its efficacy in improving scientific writing among students with learning disabilities. In STEM education, CA has been shown to improve problem-solving abilities by making tacit knowledge explicit, allowing students to actively construct knowledge rather than passively receive it.

B. Computer Assisted Learning (CAL) in Programming

This study developed a programmed learning system named "CAL" (Computer Assisted Learning), which incorporates the six core teaching strategies of cognitive

apprenticeship—modeling, coaching, scaffolding, articulation, reflection, and exploration—into the students' learning experiences, currently facilitated through self-study (Fennell et al., 2019). One of the distinctive features of cognitive apprenticeship is the explicit exposure of the cognitive processes, enabling learners to perceive otherwise abstract knowledge and fostering their self-monitoring and self-correction skills (Fennell et al., 2019). In the "CAL" system, instructors methodically explain each concept, and students immediately engage in coding practice after learning each new concept. This sequential learning approach ensures that students acquire a foundational understanding of relevant concepts before tackling programming challenges. Furthermore, as programming complexity increases, students may face numerous challenges in constructing new knowledge (i.e., writing programs) (Wang, 2019), particularly when their grasp of certain concepts is still emerging. Thus, instructors consolidate these foundational concepts in subsequent classes, deepening students' understanding of weekly topics and course concepts through cognitive activities such as application, analysis, evaluation, and creation, thereby reinforcing their acquired knowledge.

C. The Relationship between CA and CAL

Figure 1 illustrates the six instructional steps of cognitive apprenticeship, a pedagogical approach aimed at enhancing students' cognitive development and skill acquisition by emulating the traditional master-apprentice learning model. This strategy emphasizes learning as a socially mediated process, where students not only receive knowledge transmission but also deepen their understanding and application through practice, exploration, and reflection. Figure 2 presents the CAL system utilized in this study.

Figure 1

The Six Instructional Steps of Cognitive Apprenticeship

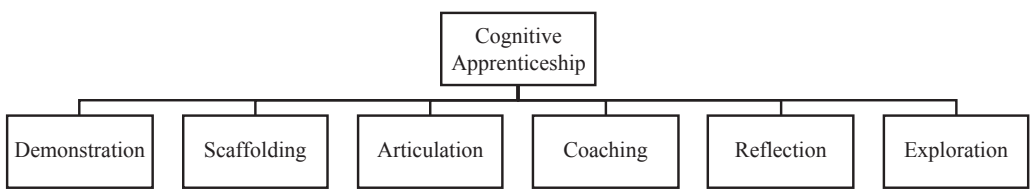
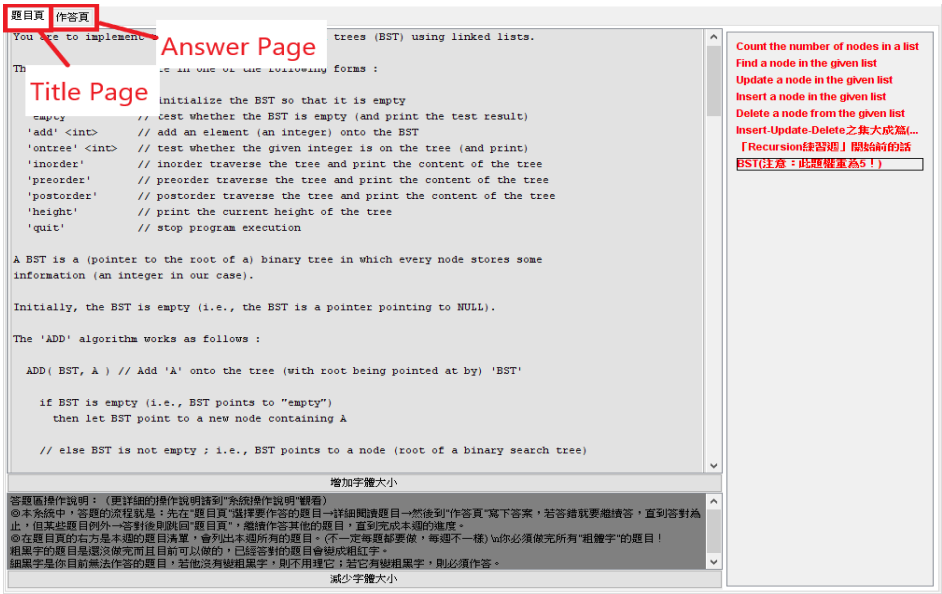


Figure 2

System Main Screen



1. Demonstration Phase

During this phase, the mentor's demonstration reveals not only the practical steps but also the reasoning behind these actions, helping apprentices understand the purpose and significance of each step. According to this instructional principle, the system divides the main screen into two parts: a problem and demonstration area and a list of problems (Figure 2), simulating the mentor's demonstration process conducted in the presence of the apprentice. The left side displays fundamental concepts and syntax of programming, paralleling a mentor's verbal demonstration, allowing students to understand program structures and operational principles.

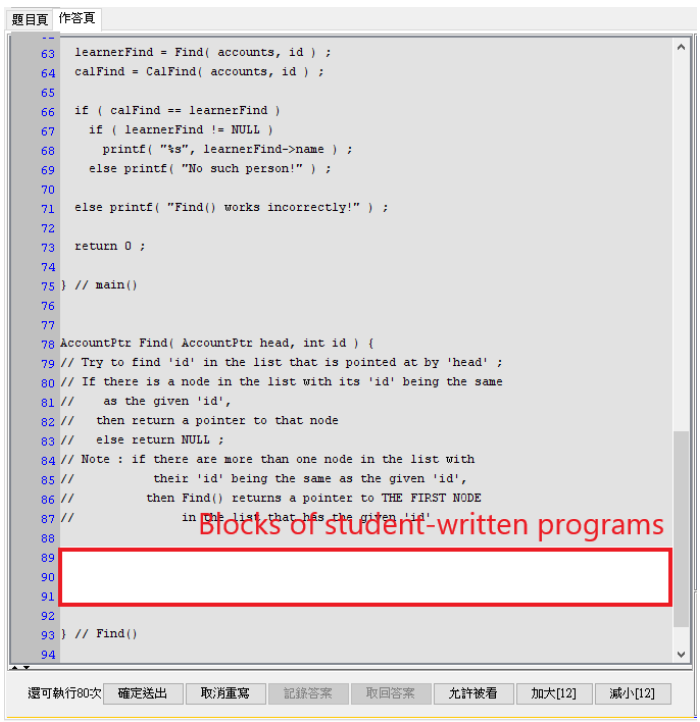
2. Scaffolding and Articulation Phase

In cognitive apprenticeship, the scaffolding phase involves the mentor providing temporary support to help students overcome difficulties encountered during task completion. As students' capabilities improve, this support is gradually removed, enabling them to complete tasks independently. The articulation phase requires students to explain and reflect on their learning process, including how they approached problems

and their insights, which helps deepen their understanding of the content and fosters critical thinking. In the present system, these two phases are integrated to support students in gradually building their programming knowledge and skills. When students are ready to start coding, they can select the Answer Page next to the Title Page, transitioning to the coding environment. Here, the CAL system provides the necessary learning frameworks, beginning with small tasks like completing a function. As shown in Figure 3, students are required to complete the Find function. As students advance, the system progressively guides them toward independently solving more complex programming problems. Furthermore, our system encourages students to write comments during their coding, which serve both as an articulation practice and as a means of explicating their thought processes and logic. This approach allows students to deepen their understanding of the content and develop the ability to clearly express their reasoning through code and comments under the guidance of a mentor.

Figure 3

Scaffolding and Articulation Phase

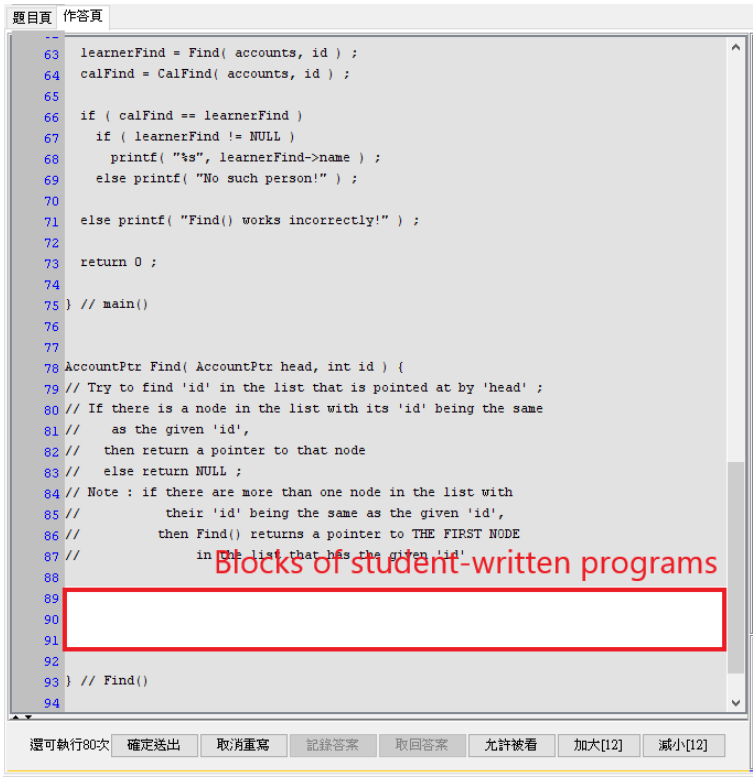


3. Coaching and Reflection Phase

Our system offers instant feedback during the coaching phase; when students start writing code, the system automatically checks the code style and compilation, providing corresponding suggestions and corrections. As shown in Figure 4, the system displays the results of executing the program, students can see the difference between their own program and the standard answer. This is comparable to the immediate guidance and feedback a mentor would provide during practice, helping students promptly identify and correct errors and understand how to optimize their code. The reflection phase allows students to compare their code execution results with the teacher's standard answers after submission, thereby promoting self-reflection on their problem-solving methods and the identification of areas for improvement.

Figure 4

Coaching and Reflection Phase



4. Exploration Phase

The exploration phase of cognitive apprenticeship refers to apprentices independently seeking new problems and higher-level tasks with mentor guidance and attempting to solve them. This phase focuses on allowing learners to apply their acquired knowledge and skills in practical scenarios, thereby deepening their understanding and capacity for application. Our system supports this exploration phase by providing a structured list of problems, sequenced from simpler to more complex. After completing basic problems, students can choose to tackle advanced problems based on their progress and interests, encouraging them to actively explore and solve more complex challenges using their programming knowledge and skills.

D. Conceptual Framework

Based on the literature, this study proposes a conceptual framework where Class Size acts as a critical moderator between the CA-based CAL Intervention and Student Outcomes.

- Input: The CAL system implements CA strategies (Modeling, Coaching, Scaffolding, Articulation, Reflection, Exploration).
- Moderator: Class Size (Small vs. Large) influences the interaction density, specifically affecting the human-dependent aspects of CA (Coaching intensity, personalized Scaffolding, and Articulation opportunities).
- Process: Students engage in iterative programming practice, help-seeking, and self-reflection.
- Outcomes: These are measured via Implementation Performance (CAL assignments) and Examination Performance (Midterm/Final).

We hypothesize that while CAL standardizes the "Modeling" phase, the "Coaching" and "Scaffolding" phases are sensitive to class size, as larger classes may dilute the teacher's ability to supplement system feedback. Students in small class settings will exhibit significantly higher learning outcomes than those in large class settings, due to increased opportunities for individualized guidance and interaction. Addtionally, class size will moderate the effectiveness of cognitive apprenticeship, such that smaller class sizes will reduce students' cognitive load and enhance their ability to benefit from the CA instructional framework.

III. Methodology

A. Research Design

This study employs a quasi-experimental design using mixed methods to evaluate the applicability and effectiveness of a CAL-based cognitive apprenticeship system. A non-equivalent control group design was utilized because the groups were formed based on pre-existing academic classes rather than random assignment. The study integrates quantitative analyses of learning outcomes with qualitative content analysis of student feedback to provide a mechanism-based explanation of the results.

B. Participants

The participants consisted of two intact cohorts of first-year college students enrolled in the College of Electrical and Information Engineering.

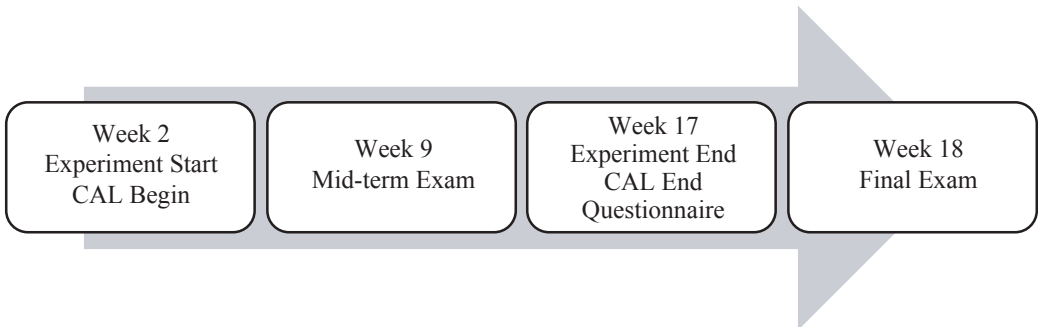
- Large Class Group: 71 students from the Undergraduate Program in Intelligent Computing and Big Data.
- Small Class Group: 17 students from the Undergraduate Program in Electrical Engineering & Computer Science.

Both cohorts were taught by the same instructor, followed an identical syllabus for the *Introduction to Computing course* (*C language*), and used the same CAL system. A pre-test conducted in the first week confirmed no statistically significant differences in prior programming knowledge between the two groups ($p > 0.05$).

C. Data Collection and Procedure

The experimental procedure spanned 16 weeks (from Week 2 to Week 17), as illustrated in the experimental process (see Figure 5). Students used the CAL system for pre-class preparation and weekly programming assignments. The system guided them through CA phases, while in-class sessions focused on the application of concepts to real-life problems and live coding activities.

Figure 5
The Experimental Process



Assessments:

- Programming Assignments: Collected weekly from Week 2 to Week 17 via the CAL system to measure practical implementation skills.
- Midterm Exam: Administered in Week 9, consisting of written conceptual questions (28%) and practical programming tasks (72%).
- Final Exam: Administered in Week 18, consisting of a comprehensive computer-based test (100%).
- Questionnaire: A standardized teaching evaluation survey along with open-ended feedback was collected in Week 17 to assess student perceptions.

D. Data Analysis

Both quantitative and qualitative techniques were employed:

- Quantitative Analysis: A one-way analysis of variance (ANOVA) was conducted to compare the effects of class size on programming assignments, midterm exams, and final exams. Effect sizes were calculated to determine the magnitude of observed differences.
- Qualitative Analysis: Open-ended student responses were analyzed using conventional content analysis (Hsieh & Shannon, 2005). Two researchers independently coded the data to identify themes related to learning barriers, support mechanisms, and system usability, followed by consensus discussions to

refine categories. For detailed theme definitions, coding categories, and representative excerpts, please refer to Table 1, which supplements the summary in Table 6.

Table 1

Summary of Themes, Definitions, and Representative Quotes

Theme	Definition	Sample Codes	Representative Quote
Theme 1: Barriers to CAL Engagement in Large-Class Contexts	Students in large classes encountered difficulties in maintaining focus and actively engaging with CAL due to noise, limited access to teacher feedback, and low motivation.	"too noisy", "hard to ask questions", "homework is overwhelming", "confused after CAL"	"The class is really noisy, and I can't concentrate at all." / "After doing the CAL tasks I still didn't understand—I felt like collapsing."
Theme 2: Personalized Support Facilitates CAL Engagement in Small Classes	Students in small classes reported stronger teacher support, clearer concept integration, and more structured learning experiences, which enhanced their CAL engagement.	"systematic learning", "teacher explains well", "motivated to finish homework"	"I hope I can study in a more systematic way." / "The teacher is conscientious and responsible."
Theme 3: Student Perceptions of CAL Usability and Learning Impact	Students' perceptions of CAL system usability and feedback design influenced their ability to engage in independent problem-solving and error reflection—both essential elements of the CA model.	"CAL is difficult", "need model answers", "improved skills", "interface confusing"	"Why is CAL so difficult?" / "I hope there will be standard answers after doing the homework."

IV. Results

To examine the impact of class size on student learning outcomes, this study conducted a one-way analysis of variance (ANOVA) on three performance indicators: midterm exam scores, final exam scores, and CAL programming assignment scores. The results are presented in Table 2 through Table 4, and effect sizes (η^2) were calculated to evaluate the magnitude of the effects.

A. Insights from Learning Assessments and Instructional Strategies

1. Interpretation of Assessment Differences

Class size had the greatest impact on programming implementation performance, a marginal effect on midterm exam outcomes (which focus on conceptual understanding), and no observable effect on final exam results (which focus on integration skills). These findings suggest that different types of assessment tools, when shaped by class interaction dynamics and instructional strategies, yield differentiated patterns of learning outcomes.

Table 2

ANOVA Analysis Results of CAL Scores

	SS	*df*	*MS*	*F*	*p*
Between groups	1484.354	1	1484.354	4.186	0.044
Within Group	30494.96	86	354.593		
Sum	31979.31	87			

Table 3

ANOVA Analysis Results of Mid-Term Exam

	SS	*df*	*MS*	*F*	*p*
Between groups	1016.736	1	1016.736	3.323	0.072
Within Group	26314.355	86	305.981		
Sum	27331.091	87			

Table 4

ANOVA Analysis Results of the Final Exam

	SS	*df*	*MS*	*F*	*p*
Between groups	38.207	1	38.207	0.044	0.835
Within Group	75041.972	86	872.581		
Sum	75080.179	87			

2. Characteristics and Functions of CAL vs. Exams

This may be because the nature of programming assignments differs from that of midterm and final exams. Programming assignments require students to continuously apply the knowledge and skills they have learned throughout the semester to solve practical problems, emphasizing the internalization and long-term accumulation of knowledge and skills. This ongoing application not only tests students' understanding of the course content but also assesses their ability to translate theoretical knowledge into practical coding tasks. In contrast, midterm and final exams are one-time assessments that are more likely to evaluate students' short-term memory and ability to solve problems under time constraints (de Bruin, 2019). Overall, programming assignment results more accurately reflect the effects of students' continuous learning and progressive skill accumulation throughout the semester (Table 5), while exams primarily assess the ability to recall and apply knowledge in the short term.

Table 5

Impact of Class Size on Programming Assignments

CAL	Small class	Large Classes
Average score	89.18	78.77
Pass rates	94.12%	87.32%

3. Feedback and Instructional Response Differences

In smaller class settings, the limited number of students allows for more personalized guidance from teachers. This targeted instruction substaintially enhances students' comprehension of the course content and their ability to effectively apply this knowledge in programming tasks. Consequently, students in smaller classes often exhibit deeper conceptual understanding and more robust skill application, as evidenced by their performance on programming assignments (de Bruin, 2019). Conversely, in larger classes, the higher student-to-teacher ratio may restrict the amount of individual attention

each student receives, potentially leading to less personalized guidance and weaker performance on programming assignments. Despite this, students in larger classes may still achieve exam scores comparable to those in smaller classes. This phenomenon can be attributed to students' reliance on short-term learning strategies focused on exam preparation, which do not necessarily contribute to long-term knowledge retention or deep understanding (Lai et al., 2021). Therefore, while students in both settings may perform similarly on exams, the depth of learning and skill application is likely more substantial in smaller classes.

4. Implications for Instructional Strategy in Large vs. Small Classes

Classroom interactions also play a pivotal role in explaining the differences in outcomes. In the larger class, the random selection process for questioning students during class resulted in fewer opportunities for individual students to actively participate. Many students were not called upon at all during the semester, which led to reduced engagement and a lower sense of accountability. In contrast, in the smaller class, each student was asked questions three to four times on average, significantly increasing engagement and the opportunity for personalized feedback. Moreover, students in the smaller class felt more comfortable initiating questions, while students in the larger class were often hesitant due to the more intimidating classroom environment and experienced difficulty articulating their questions effectively. Although peer discussions were encouraged in both classes, lower confidence and reduced willingness to participate among students in the larger class limited the effectiveness of this instructional approach.

B. Qualitative Analysis

The qualitative findings were generated based on the inductive analysis procedure described in Section 3.4. Table 6 summarizes the thematic structure derived from student responses, including theme labels, definitions, and representative quotes.

Table 6

Results of Students' Qualitative Feedback

Theme 1: Barriers to CAL Engagement in Large-Class Contexts	
n (No. of Students)	37 (from large class)
Representative Quotes	"The class is often very noisy, making it hard to focus on learning. When I try to preview the CAL exercises before class, I often get confused with the loop or condition structure, and there's no quiet space to ask for clarification, so I quickly fall behind." (Student A in a large class) "Sometimes CAL only tells me 'wrong answer' without showing which part of my code logic failed. Since there are too many students, I couldn't get timely help from the teacher to understand how to fix my mistakes." (Student B in a large class) "I stopped doing the CAL exercises after a few weeks because no one followed up on our progress. Without regular checks or reminders, I felt no pressure or motivation to continue." (Student C in a large class)
Theme 2: Personalized Support Facilitates CAL Engagement in Small Classes	
n (No. of Students)	15 (from small class)
Representative Quotes	"When I submitted wrong answers in CAL, the teacher went over them with me in class. It helped me understand where I went wrong." (Student D in the small class) "Because there are fewer students, I feel more accountable for completing the CAL tasks before class. The teacher checks our progress weekly and asks about specific exercises, so I stay on track." (Student E in a small class) "As the course progressed, I understood how to use CAL to trace code logic, and the error messages helped me correct mistakes more easily." (Student F in a small class)
Theme 3: Student Perceptions of CAL Usability and Learning Impact	
n (No. of Students)	52
Representative Quotes	"Doing exercises repeatedly in the CAL system helped me become more confident in writing logic structures. I now make fewer syntax errors and can debug my code more effectively." (Student G from the small class) "The CAL interface is sometimes hard to navigate. I wasn't sure where to write my code or how to read the error messages. Without in-class guidance, I often didn't know how to continue." (Student H in large class)

Note: Some students' feedback contributed to more than one theme. The frequency count (n) reflects the number of students whose comments were categorized under each theme, based on primary concern or emphasis.

1. Theme 1: Barriers to CAL Engagement in Large-Class Contexts

This theme addresses structural and contextual challenges that emerged in large-class teaching environments. These include issues external to the CAL system itself, such as difficulties in receiving timely instructor feedback, distractions caused by classroom

noise, and reduced accountability resulting from limited monitoring, rather than problems inherent to the CAL platform.

Students described multiple challenges that impeded their engagement with the CAL-based learning process, including classroom noise, difficulty maintaining concentration, and limited access to direct instructor feedback. These barriers are closely tied to the coaching and scaffolding components of the CA model, which rely heavily on real-time guidance, tailored support, and frequent learner-instructor interactions. In an overcrowded environment, the instructor's ability to identify individual learning gaps, offer timely hints, or appropriately adjust the level of difficulty is substantially constrained.

Moreover, limited interaction opportunities diminish students' engagement in articulation and reflection, two key mechanisms through which learners are encouraged to externalize their thinking and critically examine their problem-solving approaches. Without personalized prompts or follow-up questions from instructors or peers, students in large classes may complete exercises in a mechanical manner, missing the opportunity to internalize deeper conceptual understanding through guided reflection.

Therefore, this theme reveals how environmental and structural factors specific to large-class settings can disrupt the effective implementation of multiple cognitive apprenticeship components, thereby diminishing the pedagogical impact of the CAL system.

> "The class is often very noisy, making it hard to focus on learning. When I try to preview the CAL exercises before class, I often get confused about the loop or condition structure, and there's no quiet space to ask for clarification, so I quickly fall behind." (Student A, large class)
>
> "Sometimes CAL only tells me 'wrong answer' without showing which part of my code logic failed. Since there are too many students, I could not receive timely help from the teacher to understand how to fix my mistakes." (Student B, large class)
>
> "I stopped doing the CAL exercises after a few weeks because no one followed up on our progress. Without regular checks or reminders, I felt no pressure or motivation to continue." (Student C, large class)

Among the 71 students in the large-class group, 37 (approximately 52%) expressed concerns about distractions, insufficient teacher guidance, or lack of accountability in completing CAL tasks. This points to the need for supplemental strategies, such as peer facilitation or automated diagnostics to support CAL engagement in large-class settings.

2. Theme 2: Personalized Support Facilitates CAL Engagement in Small Classes

In small-class settings, students reported higher levels of engagement with the CAL system, supported by greater instructor attentiveness and more frequent opportunities for individualized guidance. The availability of systematic feedback, emotional encouragement, and conceptual clarification aligns closely with cognitive apprenticeship strategies of modeling, coaching, and scaffolding. Thereby fostering deeper learning and more sustained engagement with CAL.

> "When I submitted wrong answers in CAL, the teacher went over them with me in class. It helped me understand where my reasoning went wrong." (Student D, small class)
> "Because there are fewer students, I feel more accountable for completing the CAL tasks before class. The teacher checks our progress weekly and asks about specific exercises, so I stay on track." (Student E, small class)
> "As the course progressed, I learned how to use CAL to trace code logic, and the error messages helped me identify and correct mistakes more easily." (Student F, small class)

These reflections emphasize that the effectiveness of CAL is not just dependent on the system itself but also on the quality of human mediation. Of the 17 students in the small-class group, 13 (approximately 76%) commented positively on the instructor's personalized feedback, regular monitoring of CAL progress, and the increased sense of responsibility associated with smaller class sizes. This strong majority highlights the supportive instructional environment enabled by small-class contexts.

3. Theme 3: Student Perceptions of CAL Usability and Learning Impact

In contrast to Theme 1, which focuses on structural classroom barriers, this theme highlights system-level limitations within the CAL platform itself. These include user interface challenges, unclear system-generated feedback, and the difficulties students encountered when navigating the platform's features—regardless of class size.

Many students appreciated the opportunity to engage in repeated practice, receive immediate system-generated feedback, and work through progressively more challenging programming tasks. These features align with the modeling, exploration, and scaffolding elements of cognitive apprenticeship. For example, by analyzing example code and engaging in exploratory coding tasks, students are exposed to expert-like thinking processes, which fosters independent problem-solving.

On the other hand, several students noted difficulties in navigating the CAL interface or interpreting system feedback, which compromised their ability to benefit from coaching and reflection. When learners encounter unclear system prompts or lack real-time clarification, they are less likely to pause, articulate their misunderstandings, or revise their logic, key steps in the cognitive apprenticeship process. Additionally, without consistent human oversight, the coaching that typically helps students link procedural steps to conceptual reasoning becomes fragmented.

Thus, while the CAL system embodies several CA principles, its effectiveness is mediated by interface design, feedback quality, and supplementary instructional support. This theme emphasizes that technological tools alone cannot fully replicate the nuanced interpersonal dynamics that make cognitive apprenticeship successful.

> "Doing exercises repeatedly in the CAL system helped me become more confident in writing logic structures. I now make fewer syntax errors and can debug my code more effectively." (Student G, small class)
>
> "The CAL interface is sometimes hard to navigate. I wasn't sure where to write my code or how to read the error messages. Without in-class guidance, I often didn't know how to continue." (Student H, large class)

Multiple students from both large and small classes expressed confusion when navigating the CAL system, particularly regarding its interface and usability. For example, several noted that "the system interface was hard to operate," or that "feedback from CAL was difficult to understand without guidance."

Across both class sizes, 21 students (17 from the large class, 4 from the small class) mentioned issues related to the CAL system's interface design, feedback clarity, or navigation. This cross-group pattern suggests that these concerns are rooted in the CAL

system's design rather than instructional context alone. Conversely, 18 students (12 from the small class, 6 from the large class) described how the CAL system improved their programming understanding through guided tasks and timely feedback. This divergence was further evidenced by process indicators regarding help-seeking behaviors. The use of the online Q&A/LINE channel was widely adopted in the small class but remained limited in the large class. This disparity highlights that smaller cohorts facilitate more robust coaching and accountability mechanisms, whereas structural constraints in larger classes may inhibit active engagement

V. Discussion and Conclusion

The findings of this study indicate that the integration of a Cognitive Apprenticeship model within a CAL system is generally effective for introductory programming instruction, but its impact is significantly moderated by class size. The discussion is structured around three key findings:

A. Short-Term Performance is Stable Across Class Sizes

The non-significant difference in midterm and final exam scores between the large and small classes suggests that the automated components of the CAL system successfully delivered the core CA strategies (Modeling and Scaffolding) necessary for short-term knowledge acquisition and assessment preparation (Garcia, 2023; Konstantopoulos & Shen, 2023). The system effectively standardized the foundational learning experience, thereby mitigating the expected disadvantage for the larger class in traditional lecture settings.

B. Deep Learning and Long-Term Skill Acquisition Thrive in Small Classes

A significant finding was the superior performance of the small class group in the weekly programming assignments. This indicates that the small-class environment better supported the more interaction-heavy CA strategies: Coaching, Articulation, and Reflection. The higher interaction density allowed the instructor to provide timely and

personalized feedback (Coaching) and created a safer learning space for students to engage in public articulation and reflection, which are critical for developing deep problem-solving skills and long-term retention (Barab & Duffy, 2012; Loughran, 2002).

C. Class Size Moderates the Effectiveness of Pedagogical Interventions

The qualitative data indicated that students in the small class valued the personalized attention and instructional accountability, while those in the large class expressed feelings of distraction and disconnection from the instructor. This disparity confirms the conceptual framework: while the CAL system standardizes the content delivery, class size determines the efficacy of the crucial human-mediated cognitive support (Fennell et al., 2019; Wang, 2019). The structural constraints of large classes dilute the essential human element of the apprenticeship model.

D. Conclusion

In conclusion, a CAL-based CA system is an effective instructional foundation for programming education across different class sizes, ensuring baseline learning outcomes. However, optimizing for deep learning and the robust application of programming skills requires instructional environments—such as small classes—that naturally support the human-intensive aspects of cognitive apprenticeship. Educational institutions must consider class size not just as an administrative variable but as a critical factor moderating pedagogical effectiveness.

E. Limitations and Future Work

This study utilized a quasi-experimental design with non-randomized groups, which limits the generalizability of the findings. Future research shouldadopt randomized controlled trial designs. Furthermore, developing computational supports within the CAL system that can better simulate personalized coaching and facilitate articulation for large cohorts remains a critical direction for future investigation..

References

Au, M. L., Tong, L. K., Li, Y. Y., Ng, W. I., & Wang, S. C. (2023). Impact of scenario validity and group size on learning outcomes in high-fidelity simulation: A systematics review and meta-analysis. *Nurse Education Today*, *121*, 105705. https://doi.org/10.1016/j.nedt.2022.105705

Barab, S. A., & Duffy, T. (2012). From practice fields to communities of practice. In D. H. Jonassen & S. M. Land (Eds.), *Theoretical foundations of learning environments* (pp. 29-65). Routledge.

Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning. *Educational Researcher*, *18*(1), 32-42. https://doi.org/10.3102/0013189X018001032

Butler, B. A., Butler, C. M., & Peabody, T. D. (2019). Cognitive apprenticeship in orthopaedic surgery: Updating a classic educational model. *Journal of Surgical Education*, *76*(4), 931-935. https://doi.org/10.1016/j.jsurg.2019.01.009

Cakmakci, G., Aydeniz, M., Brown, A., & Makokha, J. M. (2025). Situated cognition and cognitive apprenticeship learning. In B. Akpan & T. J. Kennedy (Eds.), *Science education in theory and practice: An introductory guide to learning theory* (pp. 293-311). Springer. https://doi.org/10.1007/978-3-030-43620-9_20

Chethana, D., & Menezes, L. A. (2017). Effectiveness of cognitive apprenticeship model on problem solving skills in mathematics through multimedia instructional approach. *Anveshana*, *7*(2), 91-103. https://doi.org/10.23872/aj/2017/v7/i2/168840

Collins, A., Brown, J. S., & Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. *American Educator*, *15*(3), 6-11.

Collins, A., & Kapur, M. (2014). *Cognitive apprenticeship*. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 109–127).Cambridge University Press. https://doi.org/10.1017/CBO9781139519526.008

de Bruin, L. R. (2019). The use of cognitive apprenticeship in the learning and teaching of improvisation: Teacher and student perspectives. *Research Studies in Music Education*, *41*(3), 261-279. https://doi.org/10.1177/1321103X18773110

Doabler, C. T., Clarke, B., Kosty, D., Kurtz-Nelson, E., Fien, H., Smolkowski, K., & Baker, S. K. (2018). Examining the impact of group size on the treatment intensity of a tier 2

mathematics intervention within a systematic framework of replication. *Journal of Learning Disabilities*, *52*(2), 168-180. https://doi.org/10.1177/0022219418789376

Fennell, H., Lyon, J. A., Madamanchi, A., & Magana, A. J. (2019). *Computational apprenticeship: Cognitive apprenticeship for the digital era*. https://doi.org/10.31235/osf.io/jy328

Garcia, M. B. (2023). Facilitating group learning using an apprenticeship model: Which master is more effective in programming instruction? *Journal of Educational Computing Research*, *61*(6), 1207 - 1231. https://doi.org/10.1177/07356331231170382

Hsieh, H. F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research*, *15*(9), 1277-1288. https://doi.org/10.1177/1049732305276687

Junus, K., Suhartanto, H., R-Suradijono, S. H., Santoso, H. B., & Sadita, L. (2019). The Community of Inquiry Model Training Using the Cognitive Apprenticeship Approach to Improve Students' Learning Strategy in the Asynchronous Discussion Forum. *Journal of Educators Online*, *16*(1). https://doi.org/10.9743/jeo.2019.16.1.7

Konstantopoulos, S., & Shen, T. (2023). Class size and teacher effects on non-cognitive outcomes in grades K-3: A fixed effects analysis of ECLS-K:2011 data. *Large-scale Assessments in Education*, *11*, Article 33. https://doi.org/10.1186/s40536-023-00182-8

Kuo, F.-R., Hwang, G.-J., Chen, S.-C., & Chen, S. Y. (2012). A cognitive apprenticeship approach to facilitating web-based collaborative problem solving. *Journal of Educational Technology & Society*, *15*(4), 319-331.

Lai, C.-H. (2023). Design interactive response system BASED on hot option for improving the Participation of students: A case study. *Education Journal*, *51*(2), 77-99.

Lai, C.-H., Jong, B.-S., Hsia, Y.-T., & Lin, T.-W. (2021). Association questions on knowledge retention. *Educational Assessment, Evaluation and Accountability*, *33*, 375-390. https://doi.org/10.1007/s11092-020-09337-5

Lai, C.-H., Tseng, C.-Y., & Haia, Y.-T. (2023). Apply flipped teaching integrating cognitive apprenticeship into programming teaching. *Journal of Liberal Arts and Social Sciences*, *20*(3), 275-301.

Lakshminarayanan, S., & Rao, N. J. (2021). Types and time of interaction for teaching introductory programming using instruction method of extreme apprenticeship. *Cogent Education*, *8*(1). https://doi.org/10.1080/2331186X.2021.1969880

Lee, Y., & Paz, S. D. L. (2021). Writing scientific explanations: Effects of a cognitive apprenticeship for students with LD and English learners. *Exceptional Children*, *87*(4), 458-475. https://doi.org/10.1177/0014402921999310

Loughran, J. J. (2002). Effective reflective practice: In search of meaning in learning about teaching. *Journal of Teacher Education*, *53*(1), 33-43. https://doi.org/10.1177/0022487102053001004

Ostovar-Namaghi, S. A., Morady Moghaddam, M., & Veysmorady, K. (2024). Empowering EFL learners through cognitive apprenticeship: A pathway to success in IELTS speaking proficiency. *Language Teaching Research*. https://doi.org/10.1177/13621688241227896

Raju, B. P., Jayanthi, N., & Cherian, V. K. (2021). Effect of cognitive apprenticeship model of teaching on enhansing life skills among secondary school students. *Webology*, *18*(6), 5955-5961.

Richards, K. A. R., Hemphill, M. A., & Flory, S. B. (2021). A Collaborative Approach to Manuscript Revisions and Responses to Reviewer Comments. *Journal of Teaching in Physical Education*, *41*(3), 341-346. https://doi.org/10.1123/jtpe.2021-0118

Tsukube, T., & Matsuo, M. (2020). The impact of cognitive apprenticeship on the perceived growth of junior doctors. *Journal of Workplace Learning*, *32*(7), 489-499. https://doi.org/10.1108/JWL-04-2020-0055

Wang, B.-T. (2019). Learning how to design apps through the cognitive apprenticeship approach and collaborative learning in a Taiwanese classroom. *International Journal of Information and Education Technology*, *9*(3), 222-226. https://doi.org/10.18178/ijiet.2019.9.3.1203

Wang, R., Zulkifli, N. N., & Mohd Ayub, A. F. (2024). Investigating the impact of the stratified cognitive apprenticeship model on high school students' math performance. *Education Sciences*, *14*(8), 898. https://doi.org/10.3390/educsci14080898

Manuscript received: Sep. 20, 2024
Modified: Jan. 23, May. 3,Jun 3,Jul.9,Sep.9&Nov.24 2025
Accepted: Dec. 25, 2025