Journal of Educational Computing Research
https://doi.org/10.1177/07356331231170382

MANUEL B. GARCIA
www.manuelgarcia.info

Research Article

Facilitating Group Learning Using an Apprenticeship Model: Which Master is More Effective in Programming Instruction?

Manuel B. Garcia [a,b] *

[a] Educational Innovation and Technology Hub, FEU Institute of Technology, Manila, Philippines
[b] College of Education, University of the Philippines Diliman, Quezon City, Philippines

*** Correspondence:**
Manuel B. Garcia, University of the Philippines Diliman and FEU Institute of Technology.
mbgarcia@feutech.edu.ph

How to cite this article:
Garcia, M. B. (2023). Facilitating Group Learning Using an Apprenticeship Model: Which Master is More Effective in Programming Instruction?. *Journal of Educational Computing Research*. https://doi.org/10.1177/07356331231170382.

Article History:
Received: 4 Jan 2023
Revised: 23 Feb 2023
Accepted: 31 Mar 2023
Published: 20 Apr 2023

Abstract:

Computer programming is a difficult course for many students. Prior works advocated for group learning pedagogies in pursuit of higher-level reasoning and conceptual understanding. However, the methodological gaps in existing implementations warrant further research. This study conducted a three-armed cluster-randomized controlled trial to comparatively evaluate the social and cognitive effects of group learning pedagogies in computer programming. Following an apprenticeship model, each group has a designated master: drivers in pair programming (PP), peer leaders in peer-led team learning (PLTL), and practitioners in practitioner-assisted group learning (PAGL). In all course deliverables, the PP group received the lowest mean scores. Meanwhile, no significant difference was found between the PLTL and PAGL groups. Except for psychological safety, social factors such as task cohesion, interdependence, and group potency were significantly different between the groups. Both PLTL and PAGL groups reported a significant increase in social factors after 14 weeks of intervention. These findings provide a rationale for educational leaders and teachers to formulate curricular plans that integrate PLTL and PAGL in computer programming education. Overall, this study contributes to the literature on group learning, expands the pedagogies in computer programming, and serves as additional empirical evidence on cognitive apprenticeship and sociocultural perspectives of learning.

Keywords:

Computer Programming, Group Learning, Active Learning, Peer-Led Team Learning, Pair Programming, Practitioner-Assisted Group Learning

INTRODUCTION

Computer programming holds a reputation for being an academically challenging course, especially for undergraduate students with little or no computing background (Garcia, 2021) and outside a mainstream computer science degree program (Jacobs et al., 2016). Unfortunately, this negative connotation is often exacerbated by high dropout and failure rates, which are regularly as high as 50% (Margulieux et al., 2020). Although the source of learning difficulties is multifaceted, it is considered that at least part of the problems originates from teaching practices. Numerous studies have indicated that traditional methods (e.g., instructor-centered lectures) for teaching computer programming are insufficient (Gamage, 2021; Khan et al., 2020; Malik & Coldwell-Neilson, 2017). Part of the concern about the efficiency of traditional teaching emanates from the pedagogical format that places students in a passive role with minimal opportunity to develop critical and metacognitive thinking skills. This inadequacy enabled substantial revisions of programming teaching methodology in pursuit of higher-level reasoning and conceptual understanding (Eteng et al., 2022). According to a systematic review (Vihavainen et al., 2014), various programming teaching interventions (e.g., collaboration and peer support) improve passing rates and increase retention, unlike traditional lecture and lab-based approaches.

Accordingly, many researchers have emphasized the necessity for group learning pedagogies in computer programming education. From a macro perspective, the zone of proximal development (ZPD) of Vygotsky (1978) is often cited as the fundamental theoretical basis to justify why novice programmers should collaborate with more competent peers (e.g., Demir & Seferoglu, 2020). At a classroom level, Garcia (2021) underscored the propensity of novice programmers to initiate group discussions during laboratory activities and after lecture sessions. The information interchange in these interactions counterpoises the fear of coding stemming from the complexity and negative perception of programming courses. When students exhibit a positive attitude towards a course, it is more probable that they will develop a sense of self-efficacy and improve their academic performance (Garcia, Enriquez, et al., 2022). Conversely, when working in isolation, this fear factor often leads to a sense of confusion, lack of comfort, and self-uncertainty. The presence of collaborators consequently proclaims the significance of establishing a learning environment where support and guidance are available, especially to novice programmers (Garcia, 2021; Lou et al., 2001) and repeat students (Sheard & Hagan, 1998). In the sphere of computer programming education, many researchers have investigated various group learning pedagogies, such as collaborative learning (Hayashi et al., 2015), peer-assisted learning (Altintas et al., 2016), and cooperative learning (Garcia, 2021), to name a few.

Despite the innumerable evidence in support of these group learning pedagogies, there are still gaps worth exploring. The present study justifies the necessity for further research by pinpointing pedagogical shortcomings and difficulties from the vantage points of students (i.e., group composition) and teachers (i.e., instructional guidance). First, although there are several group formation techniques (e.g., student-selected and randomly-assigned) and pedagogical strategies (e.g., heterogeneous group configuration and mixed-gender partnership), all groups are still composed of novice programmers who are at the same stage of their learning journey. The supervision of collaborative tasks imposes high levels of cognitive load on students, who often lack the appropriate regulatory skills to synchronize their collective cognition, emotions, and behaviors (Järvelä et al., 2016). When members are occupied by self-regulation of learning and consequently cannot work productively in their teams, the absence of collaborative interaction decreases group potency. More importantly, group activities often fail without teamwork (Kreijns et al., 2003). These flaws explain why computing students cite incompetent and unreliable group members as the primary driver of their preference to work alone (Schulz et al., 2022; Waite et al., 2004). Meanwhile, the tendency for this negative collaborative experience emphasizes the significant role of teachers in fostering positive student interaction, diagnosing the progress of the group, and intervening when necessary. Instructional guidance is therefore a requisite for collaborative activities (Gamage, 2021), yet almost nonexistent in computer science courses (Schulz et al., 2022). The systematic review by Berssanette and de Francisco (2021) cited the laborious work required from the teachers for executing active learning pedagogies as the primary difficulty. One scenario is that teachers need to monitor several groups simultaneously while providing support and guidance. With these obstacles, the low adoption of this methodology not only in computer programming (Schulz et al., 2022) but also across science, technology, engineering, and math (STEM) courses remains a problem (Nguyen et al., 2021). These pedagogical shortcomings thus warrant the recruitment of an additional workforce who are competent in programming and can provide instructional guidance during group learning.

The present study begins with an acknowledgment that teachers are a vital resource in a computer science classroom, which is why the intervention only affects the laboratory part of the introductory programming course. By having the same teacher in the lecture sessions, this experiment setup likewise ensures a constant source of theoretical programming knowledge. Nevertheless, this study recruited the assistance of peer leaders and software practitioners as the additional workforce to achieve a two-fold objective. First, the extensively studied pair programming (PP) was compared as a baseline with two group learning strategies: peer-led team learning (PLTL) and practitioner-assisted group learning (PAGL). Unlike PP, these groups are under the guidance of external facilitators who have additional expertise. The second objective is to introduce PAGL as a new group learning strategy, which is a modified standard version of PLTL where peer leaders are substituted by industry practitioners. While PP and PLTL have been continually investigated in programming and other STEM courses, respectively, the literature is scarce when it comes to PAGL. Most group learning strategies are often facilitated by students, teachers, and other internal stakeholders but not by external partners. From a theoretical perspective, the common denominator between these strategies is the application of a cognitive apprenticeship model where a master (i.e., drivers in PP, peer leaders in PLTL, and practitioners in PAGL) teaches the apprentice (i.e., novice programmers). This model provides a framework for understanding how effective teaching and learning can occur in computer programming education. In essence, the present study likewise investigates the effectiveness of group learning strategies through their designated masters (i.e., more experienced people serving as secondary teachers) in teaching coding to novice programmers.

LITERATURE REVIEW

Active Learning and Collaboration

The rise to prominence of active learning has prompted the education sector to challenge the theoretical underpinnings of traditional, teacher-centered learning. A recent systematic review in computer programming education reports that active learning pedagogies increase self-confidence, stimulate classroom engagement, enable instruction flexibility, and improve the learning experience (Berssanette & de Francisco, 2021). By definition, active learning is a student-centered approach that encompasses several different pedagogies (e.g., problem-solving, discussions, case studies, and team-based learning) where students are actively or experientially involved in their learning journey. Commonly characterized as antithetical to lecturing, active learning pedagogies require students to be dynamic participants in the teaching and learning process, particularly in facilitating their construction and use of knowledge. According to Lombardi and Shipley (2021), the concept of active learning has been tagged into differing ontological categories, such as instructional pedagogies and strategies, psychological and social constructs, and design principles. Regardless of the wide variety of connotations, at its core, active learning follows the basic fundamental tenet of constructivism: *students learn by doing rather than observing*. Major (2020) noted that educators are turning to group learning pedagogies to accomplish active learning because of the solid evidentiary basis for its benefits. In computer programming, the benefits of teamwork and collaboration are profound because real-life software projects require the coordinated efforts of a team, especially with the increasing complexity of modern software systems (Demir & Seferoglu, 2020; Garcia, 2021; Schulz et al., 2022).

Pair Programming

PP is a collaborative programming methodology in which two programmers share a single workstation as they develop an information system together. Under this practice, the two programmers assume either the role of the *driver* (i.e., writes the code) or the *navigator* (i.e., reviews the code) and regularly switch roles resulting in a highly interactive, adaptive development process. This software development technique was originally employed in the industry during the 1970s but has become popular in academia in recent years (Hanks et al., 2011). In educational settings, PP is frequently used in teaching a wide range of programming courses and languages from basic to advanced levels making it a subject of considerable investigation (e.g., Bodaker & Rosenberg-Kima, 2022; Demir & Seferoglu, 2020). Hawlitschek et al. (2022) analyzed empirical studies (n = 61) on the implementation of PP in higher education published between 2010 and 2020 and reported positive effects in comparison with solo programming. In addition, students under PP arrangements exhibited more confidence in their skills, better performance in assessments, and a higher probability of course completion. This approach can lead to better code quality and fewer errors since both programmers are reviewing and discussing the code as it is being written (Padberg & Muller, 2003). Thus, it can help catch mistakes early in the development process, which can save time and resources in the long run.

Although the benefits of PP are compelling, there are nonetheless implementation challenges. Demir and Seferoglu (2020) noted that practitioners often disregard group compatibility by pairing students sitting close to each other thereby limiting the academic power of PP. Partner incompatibility interferes with student learning, which means an additional workload for teachers who need to address any arising issues. For instance, there is the possibility of the academically advanced member of the pair doing the bulk of the work. In that case, the higher-skilled partner would tend to overlook the suggestions by the lower-skilled partner and the lower-skilled partner would simply concur with any approach proposed by the higher-skilled partner (Hanks et al., 2011). Thus, Hawlitschek et al. (2022) recommend that teachers should monitor collaborative work to ensure equal participation and meaningful interaction, especially if students are novices. Following the fundamental reason for active learning hesitancy (i.e., laborious work for monitoring several groups simultaneously), this recommendation warrants further investigation of other group learning strategies in which each group receives equal attention and guided instruction from someone knowledgeable on the topic.

Peer-Led Team Learning

PLTL is a collaborative learning variation that has been widely adopted in STEM disciplines, such as Chemistry (Chan & Bauer, 2015), Biology (Winterton et al., 2020), Engineering (Muller et al., 2018), and Mathematics (Mills et al., 2020). Akin to other active learning strategies, PLTL exemplifies a departure from instructor-centered to learner-centered instructions, providing students the opportunity to manage their learning. Under the PLTL model (Gafney & Varma-Nelson, 2008), small groups of students and trained peer leaders meet regularly to collaborate in solving problems related to the topics covered in lectures by their teachers. In the realm of conceptual understanding, this collaboration between students and their more capable peers constitutes the ZPD (Vygotsky, 1978). The review of studies (*n* = 67) that examined efforts to integrate PLTL in STEM classrooms by Wilson and Varma-Nelson (2016) found positive results on course grades and retention, especially the critical components published in *Peer-Led Team Learning: A Guidebook* (Gosser et al., 2001) are properly implemented.

The success of the PLTL model has been attributed to its type of interaction (student-to-student) that promotes a non-judgmental environment. Winterton et al. (2020) reasoned that these interplays often occur without the supervision of someone (e.g., teachers) with the authority to assign grades in the course. A glaring difference between a typical teacher and a peer leader is that the latter can relate to students based on recent personal experience and success as former students in the same course. Typically, the role of peer leaders is performed by students who are recent completers and academically exceptional in the course, which means that they have additional expertise in the subject matter. Wilson and Varma-Nelson (2016) noted that the fundamental philosophy behind the employment of peer leaders as facilitators of group work is "*to establish the dynamic of slightly more advanced learners scaffolding the education of novices*". Although peer leaders are not yet experts in their field, their classification as students positions them well to act as facilitators of group work in a non-threatening environment.

Practitioner-Assisted Group Learning

The present paper proposed and defined PAGL as an active learning strategy that provides a collaborative environment for students to engage in intellectual discussions with experts in the field. From a methodological perspective, it is a modified standard version of PLTL where industry practitioners (e.g., computer programmers, software developers, and other information technology professionals) substitute for peer leaders. The exchange of resource persons is partly attributable to role model perception. This concept refers to the process by which individuals identify and emulate the characteristics, behaviors, and attitudes of others whom they perceive to be successful or competent in a particular domain. In PLTL, students considered peer leaders their role models for being exemplars of success, as someone who excelled in academics (Winterton et al., 2020). Additionally, peer leaders inspire others to be better and achieve more by showing what they accomplished as fellow students. It is reasonable to expect that practitioners are viewed in the same light because being successful is a key reason why someone would be considered a role model (Gladstone & Cimpian, 2021). More importantly, practitioners as role models operate as representations of what is possible which then results in the personification of students' existing goals (e.g., to be a computer programmer as well) (Morgenroth et al., 2015). By interacting with experts in the field, students are exposed to successful professionals who possess the skills and knowledge required to succeed in their respective domains. This exposure can serve as a source of inspiration and motivation for students, who may be more inclined to emulate the behaviors and attitudes of successful industry practitioners than their peers (Lockwood & Kunda, 1997; Marx & Ko, 2012).

From this perspective, PAGL follows the central tenet of Bandura's (1977) social cognitive theory that emphasizes the vital role of social environments in shaping self-efficacy – one's belief in their capacity to learn and do well in a domain. Through observational learning, a key source of self-efficacy is watching a role model succeed on comparable tasks. Demonstrating task completion to required standards requires competence (i.e., general attributes such as intelligence or skill), which predicts desirability – one of the three role model qualities proposed by Morgenroth et al. (2015). As seasoned individuals, practitioners have already developed expertise and acquired a great deal of content knowledge. By working alongside experts in the field, students can observe firsthand the skills and knowledge required to complete tasks to the required standards. In his proposal of a social construction approach in computer science education, Machanick (2007) exemplified the workflow of expert practitioners as a legitimate approach to convincing students to value fundamental concepts, reasoning from first principles, and fusing knowledge from multiple sources. Although teachers possess similar technological knowledge, one notable difference is that practitioners can share industry experience and intuitive specialist knowledge that may not be available in academia. In the realm of social media, TikTok videos about one's programming career are the most-watched and digitally-engaged programming topics (Garcia, Juanatas, et al., 2022). Thus, the mental process of how practitioners address and solve real-world problems through coding could provide valuable insights and improve the technological skills of novice programmers.

METHODOLOGY

Theoretical Framework

In computer science education, Machanick (2007) contended that not all phenomena observed in teaching and learning can be explained by purely cognitive models. His proposal of a social construction approach aims to make students more active learners as early as in their introductory courses. Following the intent of promoting active and social learning experiences in a computing classroom, the present study applied a sociocultural perspective with a particular emphasis on the ideas of cognitive apprenticeship and situated cognition. The sociocultural theory advocates a collaborative practice that transcends social constructivism to build learning communities and knowledge societies. Polly et al. (2017) underscored the significant practical applications of sociocultural theory concerning the establishment of student-centered learning environments through collaboration with peers and experts. As collaborative environments are a central component of social learning theories, active learning strategies leveraging interactions such as peer-to-peer (PP), peer-to-leader (PLTL), and peer-to-expert (PAGL) fall under the sociocultural umbrella. Moreover, the seminal work of Brown et al. (1989) on situated cognition posited the inseparable theoretical positions of enculturation and social participation in learning instead of solely occurring inside the mind of an individual. By interacting with others in the social sphere, learning stimulates a variety of internal developmental processes. The situated nature of cognition is likewise acknowledged by allowing learners to discover, observe, and engage in expert practices. Hennessy (1993) accentuated cognitive apprenticeship in constructing domain expertise and indoctrinating novices into the profession. Therefore, the designation of drivers, peer leaders, and software practitioners as group masters thus follows the apprenticeship model.

Research Design

The present study employed an experimental research design to comparatively evaluate the effectiveness of group learning strategies for teaching computer programming. Formally surfaced in educational psychology, the experimental method is suitable for inferring causal relationships between the intervention (e.g., learning resources, pedagogies, or curriculums) and its outcome (e.g., academic performance or learning attitude). Similar to the methodology employed by Garcia and Yousef (2022) in a website development course, this study adopted a prospective, three-armed, cluster-randomized controlled trial with PP as the control group and PLTL and PAGL as the intervention groups. This design complies with the highest level of the test (i.e., innovation vs. enhanced treatment) for experimental treatment because it applied a treatment of unknown efficacy (PAGL) rather than the customary *treatment vs. no treatment* or *innovation vs. standard treatment* arrangements (Taber, 2019). Rather than individual learners, class sections (i.e., clusters) were appointed as the unit of analysis due to ethical considerations and university enrollment policies. Three class sections were randomly selected and automatically enrolled into a masterclass (i.e., a single online classroom for asynchronous instruction) by the program director but each section has its corresponding synchronous meeting channel to prevent contamination. These channels are connected to the main channel (see figure 1) where the professor taught the course using the same syllabus and schedule (time and day) to avoid a potential confounding effect. Treatment allocation was determined using a web-based randomization program. The key ethical principles mandated by the university and in the Declaration of Helsinki were followed to protect participants' rights.

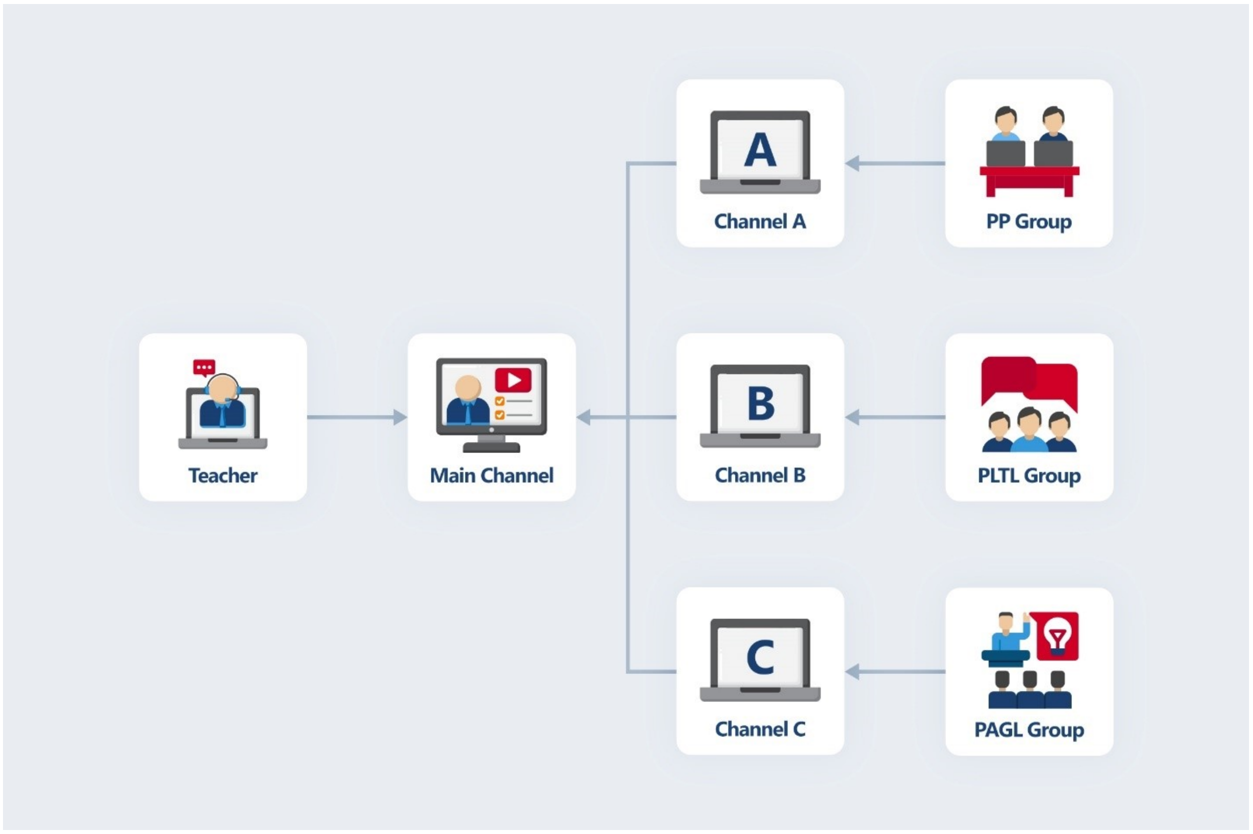

Figure 1: Virtual Classroom Setup for Synchronous Sessions

Setting and Sample

The experiment was carried out at one of the leading institutes of technology in the Philippines from April to August 2022. This non-sectarian, private university offers four-year information technology and computer science programs. Despite having multiple specializations, these degree programs position computer programming courses as a potent foundation for a career in the computing industry. One of these courses is *Computer Programming 2* with both lecture (CCS0007) and laboratory (CCS0007L) classes designed to expand the knowledge of programming students by focusing on more advanced features of the *C*++ language. These features include pointers, file handling, and an introduction to object-oriented programming, which are a continuation of the basic concepts discussed in CCS0003 and CCS0003L (see Garcia, 2021). It is noteworthy that this introductory programming course is the second among the many coding-related courses in the programs and it was deliberately selected to secure a programming knowledge baseline. Final course grades from the previous programming course were used to measure prior programming knowledge. Ensuring that there is no significant difference between the groups in terms of programming knowledge can help control for potential confounding variables in the experiment. A total of three sections with 50 students each were recruited and randomly assigned to either PP, PLTL, or PAGL groups. Students were informed that their participation was not required and have no impact on their grades. All students (*n* = 150) agreed to participate and submitted a signed informed consent form (and parent consent for minors).

Table 1. The Coverage of the Assessments for Lecture and Laboratory Classes

Module	Topics	Lecture	Laboratory
1	User-Defined Functions and Parameters	Summative Assessment 1	Technical Assessment 1
2	Two-Dimensional and Multidimensional Arrays		Technical Assessment 2
3	Character and String Manipulation	Summative Assessment 2	Technical Assessment 3
4	Structures (struct)		Technical Assessment 4
5	Pointers and Dynamic Memory Allocation	Summative Assessment 3	Technical Assessment 5
6	Linked List Data Structure		Technical Assessment 6
7	File Handling	Summative Assessment 4	Technical Assessment 7
8	Introduction to Object-Oriented Programming		Technical Assessment 8

Note: The coverage of the midterm examination was modules 1 to 4 while the final examination covered all modules.

Research Instruments

Following a similar experiment on group effectiveness (Olivera & Straus, 2004), the present study leans on the primary perspectives on group learning: cognitive and social. While the cognitive dimension examines the effects of group work on individual and team cognitive processes, the social dimension scrutinizes the social factors conceiving successful performance. According to Nokes-Malach et al. (2015), both cognitive and social factors drive collaborative failure and success. The cognitive component was measured by replicating the study of Garcia and Revano (2021) in evaluating gamification as a teaching strategy in Python programming. Accordingly, the study assessed students' academic performance using summative assessments in lecture (individual) and performance-based assessments in the laboratory (group). Module topics and the coverage of the assessments were presented in Table 1. The deliberate inclusion of individual assessments in the analysis intends to measure the group-to-individual transfer of learning, which is important for several reasons. Although group projects are prevalent in the computer science industry, it is conventional for members of the team to work independently and integrate their outputs afterward to produce a collective product. Programmers writing different modules to produce a computer application is one example. It is likewise possible for individuals to work solo, especially on small and easy projects while working in a collaborative group, either sequentially or simultaneously. Finally, the social component was measured using the Team Learning Beliefs & Behaviors Questionnaire (TLBBQ) developed by Van den Bossche et al. (2006) for investigating the cognitive and social factors driving teamwork in collaborative learning environments. Social constructs such as psychological safety (e.g., "*My knowledge and skills are* *valued by other members of the team*"), task cohesion (e.g., "*The team is united in reaching its goal*"), interdependence (e.g., "*The team members agree on what we want to accomplish*"), and group potency (e.g., "*This team believes it can be very effective*") were used in this study. Social cohesion was not included because it does not predict team learning behavior.

Procedure and Data Analysis

For the duration of one trimester (14 weeks), the experiment was conducted in a virtual computer science classroom. Although the literature is inadequate, some studies underscore the positive effects of PP (Bodaker & Rosenberg-Kima, 2022) and PLTL (Young & Lewis, 2022) when implemented virtually. Online synchronous meetings were twice a week and two hours per meeting. The first week was allotted for the course and experiment orientations and review of the prerequisite course with individual programming activities while the last week was allocated for the group project presentations and final examination. Following the guidelines from previous studies (e.g., Hawlitschek et al., 2022; Wilson & Varma-Nelson, 2016), three custom instructional guides were prepared to outline the delivery of each group learning strategy. In terms of group size, the ratio of master to student is 1:2 for PP (*n* = 25 pairs), 1:5 for PLTL (*n* = 10 groups), and 1:10 for PAGL (*n* = 5 groups). PP was implemented as a classroom strategy while PLTL and PAGL were executed as workshop events outside official class hours. The TLBBQ was distributed via an online learning management system (LMS). All groups answered the survey questionnaire on April 20 and July 22, 2022, as pretest and posttest, respectively. With consent from all parties, assessment scores were extracted from students' LMS accounts after the final examination. The collected data were analyzed using IBM SPSS Statistics (version 28.0.1). The comparison of the social aspect was measured using the Kruskal-Wallis *H* test (between-group) and Wilcoxon Signed-Rank Test (within-group) while the comparison of the cognitive aspect was measured using one-way Analysis of Variance (ANOVA) and Multivariate Analysis of Variance (MANOVA).

RESULTS

The objectives of this study were to comparatively examine the cognitive and social effectiveness of group learning strategies and introduce PAGL as a pedagogy for teaching computer programming. Following a cognitive apprenticeship model and an experimental research design, three cohorts of students were randomly assigned to either PP, PLTL, or PAGL groups for a 14-week educational intervention (equivalent to one trimester). All groups used the same syllabus, and each lecture session was delivered by the same professor on the same schedule. Conversely, the laboratory sessions followed the designated group learning pedagogy either as a classroom strategy (PP) or whatever techniques students learned during their workshop sessions (PLTL and PAGL). A demographic survey revealed that the participants were dominated by male students (*n* = 134; 89.33%) with a mean age of 19.56 years. The mean course grade from the prerequisite course among the three groups was 86.38 (*SD* = 8.56), and the one-way ANOVA analysis revealed that all students possessed the same prior knowledge of introductory computer programming (*F* = .513, *p* = 0.694) before the intervention.

Individual Academic Performance

Although the focal point of investigation of this study was the pedagogical effectiveness of group learning strategies (PP, PAGL, and PLTL), the assessment of individual academic performance reflects the group-to-individual transfer of learning. As previously argued, it is conventional for computer programmers to work independently and integrate their outputs afterward to produce a collective product. Measuring individual performance in a group learning strategy allows for a more nuanced understanding of how students are benefiting from the group learning experience. Table 2 shows the results of individual cognitive assessments in lecture classes. Accordingly, the PP group received the lowest mean scores for both summative assessments (75.44 ± 6.67) and major examinations (60.24 ± 12.34). On the other hand, the PLTL group received the highest mean scores for the summative assessments (81.15 ± 6.52) while the PAGL group received the highest mean scores for the major examinations (67.68 ± 11.47).

Table 2. Individual Cognitive Assessments in Lecture Classes

Assessment	**PP Group** (***n*** **= 50**)	**PLTL Group** (***n*** **= 50**)	**PAGL Group** (***n*** **= 50**)	***F***	**Sig.**	**Partial Eta Squared**
SA1 - Summative Assessment	84.04 (6.55)	85.42 (7.37)	87.58 (7.35)	.439	.645	.006
SA2 - Summative Assessment 2	77.98 (4.59)	79.08 (6.49)	81.02 (6.18)	3.505	.033	.046
SA3 - Summative Assessment 3	68.40 (9.28)	79.70 (6.18)	75.66 (7.21)	.862	.425	.012
SA4 - Summative Assessment 4	71.34 (7.11)	80.38 (5.97)	79.64 (5.52)	4.083	.019	.053
ME - Midterm Examination	60.10 (15.29)	69.42 (10.24)	67.74 (12.05)	.575	.564	.008
FE - Final Examination	60.38 (10.34)	65.06 (10.34)	67.62 (10.27)	6.333	.002	.079

The MANOVA shows that there was a statistically significant difference in the individual academic performance of students in computer programming, $F = 7.95$, $p = .001$; Wilk's $\Lambda = .791$, partial $\eta^2 = .111$. The tests of between-subjects effects using univariate ANOVAs revealed that group learning pedagogies have a statistically significant effect on SA2 ($F = 3.505$, $p = .033$; partial $\eta^2 = .046$), SA4 ($F = 4.083$, $p = .019$; partial $\eta^2 = .053$), and FE ($F = 6.333$, $p = .002$; partial $\eta^2 = .079$). To assess the significance of differences between pairs of group means, Tukey's Honest Significant Difference (HSD) post hoc tests were used. In SA2, there was a significant difference in the performance between PP and PAGL ($p = .027$), but not between PAGL and PLTL ($p = .221$) and PP and PLTL ($p = .612$). In SA4, there was a significant difference in the performance between PP and PLTL ($p = .019$), but not between PP and PAGL ($p = .099$) and PAGL and PLTL ($p = .783$). Finally, there was a significant difference in the performance between PP and PAGL (p

= .002), but not between PAGL and PLTL (p = .431) and PP and PLTL (p = .064) in FE. An emerging pattern from these significant results is that the PP group performed significantly worse than PAGL and PLTL groups while these two groups had a similar academic performance. Thus, it can be concluded that peer leaders (PLTL group) and practitioners (PAGL) had a more significant impact than drivers (PP group) in the group-to-individual transfer of learning. This finding provides insight into how group learning can be leveraged to improve individual performance. More specifically, it can help to ensure that each student is benefiting from the collaborative learning experience and address any individual learning needs.

Performance-Based Assessments

Unlike traditional lecture and laboratory-based approaches, collaboration and peer support have a better significant impact on improving passing rates and increasing retention in computer programming courses (Vihavainen et al., 2014). Thus, many researchers have implemented and explored the installation of group learning pedagogies to teach coding and computational thinking. The goal of this study is similar and the results of performance-based assessments in laboratory activities are presented in Table 3. Comparable to the individual achievements in lecture classes, the PP group received the lowest mean scores for both technical assessments (83.87 ± 4.57) and the final project (76.32 ± 14.90). On the other hand, the PLTL (85.82 ± 4.93) and PAGL (85.66 ± 5.31) groups have almost equal mean grades in the technical assessments. For the final project, students from the PAGL group received the highest mean score (84.34 ± 9.38).

Table 3. Group Cognitive Assessments in Laboratory Activities

Assessment	PP Group (n = 25)	PLTL Group (n = 10)	PAGL Group (n = 5)	F	Sig.	Partial Eta Squared
TA1 - Technical Assessment 1	92.64 (3.36)	93.96 (3.27)	94.84 (3.03)	.126	.882	.002
TA2 - Technical Assessment 2	95.06 (3.07)	94.84 (3.30)	96.02 (3.25)	1.908	.152	.025
TA3 - Technical Assessment 3	76.70 (4.81)	78.78 (5.70)	80.26 (5.94)	5.280	.006	.067
TA4 - Technical Assessment 4	76.88 (4.61)	80.30 (5.86)	80.84 (5.46)	8.100	.000	.099
TA5 - Technical Assessment 5	77.60 (4.49)	80.90 (5.66)	78.68 (5.47)	5.169	.007	.066
TA6 - Technical Assessment 6	78.30 (4.91)	82.18 (6.13)	80.50 (5.58)	6.616	.002	.083
TA7 - Technical Assessment 7	77.68 (4.40)	81.28 (5.34)	78.62 (6.42)	5.871	.004	.074
TA8 - Technical Assessment 8	96.12 (3.29)	94.34 (2.96)	95.50 (3.16)	1.147	.118	.013
FP - Final Project	76.32 (14.90)	82.66 (8.73)	84.34 (9.38)	6.951	.001	.086

The MANOVA shows that there was a statistically significant difference in the academic performance of groups of students, $F = 9.35$, $p = .000$; Wilk's $\Lambda = .604$, partial $\eta^2 = .223$. The tests of between-subjects effects using univariate ANOVAs revealed that group learning pedagogies have a statistically significant effect on TA3 ($F = 5.280$, $p = .006$; partial $\eta^2 = .067$), TA4 ($F = 8.100$, $p = .000$; partial $\eta^2 = .099$), TA5 ($F = 5.169$, $p = .007$; partial $\eta^2 = .066$), TA6 ($F = 6.616$, $p = .002$; partial $\eta^2 = .083$), TA7 ($F = 5.871$, $p = .004$; partial $\eta^2 = .074$), and FP ($F = 6.951$, $p = .001$; partial $\eta^2 = .086$), but not on TA1, TA2, and TA8. One possible explanation was that the first two technical assessments were simply a review of the prerequisite programming course. Therefore, all students have experience in writing programs with arrays and functions. Meanwhile, the last technical assessment (object-oriented programming) is simply an overview of the next programming course. The machine problem was not as complicated as the previous technical assessments. Only the core principles of this programming paradigm and how it differs from the procedural style of coding were discussed in this module. Other advanced concepts such as encapsulation, inheritance, polymorphism, and abstraction were not thoroughly explained. To assess the significance of differences between pairs of group means, Tukey's HSD post hoc tests were used. Like the emerging pattern in the individual cognitive assessments, the PP group performed significantly lower than the PAGL group in TA3 ($p = .004$) and TA5 ($p = .002$), and the PLTL group in TA4 ($p = .006$), TA6 ($p = .001$), TA7 ($p = .003$), TA8 ($p = .014$), and FP ($p = .016$). Meanwhile, the PAGL and PLTL groups had similar academic performances.

Team Learning Beliefs

The theoretical underpinning of most prior works on group learning in computer programming education was Vygotsky's (1978) ZPD, making social aspects a salient consideration. As argued in the previous experiment (Garcia, 2021), novice programmers tend to group themselves during laboratory activities and after lecture sessions to discuss the lessons and how to solve machine problems. In addition, although the transfer of learning transpired primarily due to cognitive load, group effectiveness typically leans on both cognitive and social perspectives (Olivera & Straus, 2004). To measure the social facets, programming students self-assessed the social factors driving their teamwork using the TLBBQ (Van den Bossche et al., 2006). The within-group and between-group comparisons are presented in Table 4. Using Kruskal-Wallis *H* tests, the results of the pretest showed no significant difference between the groups, in terms of psychological safety ($\chi^2 = 5.188$, $p = .751$), task cohesion ($\chi^2 = 6.828$, $p = .661$), interdependence ($\chi^2 = 3.763$, $p = .152$), and group potency ($\chi^2 = 5.051$, $p = .080$). This finding implies that students' team learning beliefs are homogenous before the intervention. After 14 weeks of online group collaboration, social factors such as task cohesion ($\chi^2 = 11.661$, $p = .033$), interdependence ($\chi^2 = 22.547$, $p = .000$), and group potency ($\chi^2 = 7.318$, $p = .026$) were significantly different between the groups. The PAGL group had the highest rating on task cohesion (3.92 ± 0.85) while it was the PLTL group for interdependence (4.14 ± 0.83) and group potency (4.12 ± 0.82). Only the psychological safety construct was found to be insignificantly different between the groups.

Table 4. Within-Group and Between-Group Comparisons of Team Learning Beliefs

Social Factors	Within-Group Comparison						Between-Group Comparison	
	PP Group		PLTL Group		PAGL Group			
	Mean ± SD	*p*-value	Mean ± SD	*p*-value	Mean ± SD	*p*-value	χ^2	*p*-value
Psychological Safety		.219		.019		.464		
Pretest	3.46 ± 1.03		3.32 ± 1.06		3.78 ± 1.04		5.188	.751
Posttest	3.72 ± 1.09		3.82 ± 0.85		3.94 ± 0.84		5.875	.646
Task Cohesion		.348		.082		.035		
Pretest	3.66 ± 1.12		3.52 ± 1.23		3.44 ± 1.18		6.828	.661
Posttest	3.42 ± 1.09		3.88 ± 0.75		3.92 ± 0.85		11.661	.033
Interdependence		.059		.000		.069		
Pretest	3.46 ± 1.05		3.26 ± 1.10		3.68 ± 1.06		3.763	.152
Posttest	3.08 ± 0.94		4.14 ± 0.83		4.10 ± 0.89		22.547	.000
Group Potency		.112		.012		.236		
Pretest	3.16 ± 1.10		3.62 ± 1.16		3.60 ± 1.14		5.051	.080
Posttest	3.52 ± 1.20		4.12 ± 0.82		3.86 ± 0.81		7.318	.026

In terms of the within-group comparison, the Wilcoxon Signed-Rank Test showed mixed findings across the groups. The mean pretest scores ranged from 3.16 ± 1.10 to 3.78 ± 1.04 while the mean posttest scores ranged from 3.08 ± 0.94 to 4.14 ± 0.83. For the PP group, no significant difference was found before and after the intervention in all social factors. However, it is worth noting that it is the only group with decreased scores after the intervention, specifically in task cohesion (3.66 ± 1.12 → 3.42 ± 1.09) and interdependence (3.46 ± 1.05 → 3.08 ± 0.94). Meanwhile, the PLTL group reported a significant increase in psychological safety (3.32 ± 1.06 → 3.82 ± 0.85, *p* = .019), interdependence (3.26 ± 1.10 → 4.14 ± 0.83, *p* = .000), and group potency (3.62 ± 1.16 → 4.12 ± 0.82, *p* = .012). Conversely, only in task cohesion (3.44 ± 1.18 → 3.92 ± 0.85, *p* = .035) did the PAGL group report a significantly increased score. Finally, the highest mean posttest scores were reported by the PLTL group in interdependence (4.14 ± 0.83) and group potency (4.12 ± 0.82) and the PAGL group in psychological safety (3.94 ± 0.84) and task cohesion (3.92 ± 0.85). Although with mixed results, both the PLTL and PAGL groups elicit significant findings in the team learning beliefs among computer programming students.

DISCUSSION

As alternatives to conventional lecture and laboratory-based approaches, numerous studies recommended various programming teaching interventions such as collaboration and peer support to improve passing rates and increase retention (Vihavainen et al., 2014). More importantly, the benefits of teamwork and cooperation in computer programming education are profound because real-life software projects necessitate the coordinated efforts of a team to meet the increasing complexity of modern software systems (Demir & Seferoglu, 2020; Garcia, 2021;

Schulz et al., 2022). Nevertheless, as argued in this paper, existing group learning strategies have shortcomings from the vantage points of students (group composition) and teachers (instructional guidance). These pedagogical gaps led to the proposals of applying PLTL in computer programming and of PAGL as a new group learning strategy. Both pedagogies follow the active learning paradigm and the cognitive apprenticeship model. To evaluate these recommendations, a prospective, three-armed, cluster-randomized controlled trial was adopted to evaluate the group effectiveness with PP as the control group and PLTL and PAGL as the intervention groups.

In terms of cognitive assessments, it was apparent that students from the PP group received significantly lower mean scores on all summative assessments, machine problem activities (technical assessments), major examinations, and a final project. From a cognitive apprenticeship perspective, these assessment results indicate that peers or classmates as masters are less effective. This finding is unsurprising since their programming knowledge and expertise are insufficient compared to peer leaders and software practitioners. It also confirms the assertion of Morgenroth et al. (2015) that the ideal role model for students is slightly older and more advanced in the field. Therefore, although PP is better than solo programming (Hawlitschek et al., 2022), it is not the best option when it comes to group learning pedagogies. According to Garcia, Rull, et al. (2022), it is possible that the intensity of interaction decreases as the group size becomes smaller (e.g., students feel shy in a one-on-one discussion). Despite several meta-analyses of PP (e.g., Hannay et al., 2009; Umapathy & Ritzhaupt, 2017), research is scarce on the effectiveness of PP in student learning. This study extends the literature by assessing the cognitive performance of novice programmers using experimental research. While the focal point of the investigation was group learning strategies, it is worth noting that another significant contribution of the study was the evaluation of group-to-individual transfer of learning (Olivera & Straus, 2004). This analysis provides insight into how group learning can be leveraged to improve individual performance. For example, if students who participated in group learning do not perform as well on individual coding activities as those who learned independently, this may indicate that the group process did not effectively support the transfer of learning to individual tasks. Meanwhile, Tennyson et al. (2018) noted that the objective of PP is to improve code quality, unlike peer-assisted learning whose goal is to enhance learning. Therefore, this study calls for further research to examine which characteristics (e.g., leadership, peer influence, cohesion, conflict, and quality) of peer-assisted learning (Ala et al., 2021) are lacking in PP.

Students from PLTL and PAGL groups did not perform significantly differently from each other on all their deliverables. A common denominator between these groups is that their masters are more advanced programmers capable of scaffolding the education of novices (Wilson & Varma-Nelson, 2016). While software practitioners are seasoned individuals who have already developed their expertise, peer leaders are recent completers and academically exceptional in the course. Unlike drivers in the PP group, the masters in PLTL and PAGL groups have already acquired a great deal of content knowledge that they can effortlessly share with novice programmers. Following the motivational theory of role modeling, they also possess the qualities (i.e., desirability, attainability, and goal embodiment) that influence several role modeling processes (e.g., identification and vicarious learning). Overall, this result also echoes the conclusion of prior works that PLTL has a positive impact on student academic performance (Chan & Bauer, 2015; Muller et al., 2018), especially in STEM education (Wilson & Varma-Nelson, 2016). In addition, it provides preliminary empirical evidence on the effectiveness of PAGL as a teaching strategy. As a managerial implication, these findings provide a rationale and evidence for school leaders and teachers to formulate curricular plans that integrate PLTL and PAGL as pedagogies in computer programming education. As posited by Garcia (2022), some traditional class sessions may be replaced to allow collaboration between students.

The successful implementations of PLTL and PAGL may have something to do with relatability and role model status. According to Winterton et al. (2020), these factors assist in mitigating feelings of intimidation in introductory STEM environments, increasing student achievement, and reducing course attrition. Unlike traditional teachers who may intimidate some students, peer leaders are more relatable because they were former students in the same course (Wilson & Varma-Nelson, 2016). On the other hand, students considered software practitioners as role models because they are exemplars of success and the personification of their goals to be computer programmers (Morgenroth et al., 2015). It is also possible that students may have improved their self-efficacy beliefs after witnessing the programming competence of peer leaders and software practitioners. Peeking through the social cognitive theory perspective (Bandura, 1977), a key source of self-efficacy is watching a role model succeed in comparable tasks. According to the systematic review of the educational literature, there is a positive correlation between self-efficacy and academic performance (Honicke & Broadbent, 2016). In essence, highly efficacious students improve their learning because they exert more effort and are more determined to overcome obstacles when hardships arise. The systematic review also found that time on task (the amount of time spent on studying) moderates the relationship between self-efficacy and academic performance. As stated in the procedures, PP was employed as a classroom strategy while PLTL and PAGL were executed as workshop events outside official class hours.

In terms of the social perspective, all factors were found to be significantly different after the experiment although it depends on which intervention group. While students from the PAGL group reported a significant increase in task cohesion, it was team potency, psychological safety, and interdependence for students from the PLTL group. On the other hand, the control group did not report a significant increase in all factors, and it was the only group with decreased scores after the intervention. This finding supports the earlier assertion that PP may not be the best small-group learning pedagogy in computer programming. For students from the PAGL group, the significant increase in their shared commitment toward the task at hand indicates that industry practitioners were more capable of forming highly cohesive groups than peer leaders. Following the relational cohesion theory that equates a higher trust level with increased group cohesion (Lawler et al., 2000), it is possible that software practitioners were able to form groups that trust each other (e.g., through their leadership styles) or that students trust them more (e.g., due to their proven track record). Meanwhile, the beliefs of students from the PLTL group that they need each other, their contributions will be valued by other members, and they can be effective together indicates that the student-to-student type of interaction was more effective in these social factors. According to Winterton et al. (2020), peer leaders can naturally position themselves as facilitators of group works in a nonthreatening setting because they are technically students as well. An all-student group headed by a competent leader may have promoted a harmonious learning environment and an atmosphere of connection and belongingness.

In addition to the comparative evaluation of group learning strategies, another objective of this study was to introduce PAGL as a pedagogy in computer programming education. Recruiting industry practitioners as secondary teachers can bring many benefits to the education system. For example, they can provide students with valuable industry knowledge and experience that can help bridge the gap between academic theory and practical application. As shown in this study, students from the PAGL group exhibited superior cognitive abilities and better performance in certain social dimensions than PP and PLTL, respectively. Despite these positive findings, prospective adopters should carefully consider the challenges associated with this approach. First, it may be difficult to find practitioners who are willing to commit to an additional teaching task as they often work in demanding positions with stringent timeframes. Of greater significance is the possibility that these practitioners may lack teaching experience, thereby posing a potential limitation. They may require additional training and preparation to become effective secondary teachers. In addition, it should be noted that conducting onsite sessions requires access to classroom facilities and school resources, which may pose organizational challenges. Finally, it is also important to consider that adopting this approach may incur additional costs related to compensating practitioners for their services. Overall, it is recommended that prospective adopters should conscientiously weigh the challenges that PAGL entails before implementing it.

Although the present study employed a rigorous experiment, some limitations should be noted when interpreting the research findings. First, the intervention was delivered remotely since online education was the only option at that time. As posited recently (Goñi et al., 2020), online group work may have involved fewer team deliberation sessions. Future research should therefore validate the findings of the experiment during in-person or blended classes. Second, despite the effort in preventing contamination using a customized virtual classroom setup, it is still possible that students from different groups talked to each other. During the COVID-19 pandemic, students were found to communicate with one another outside of official class hours to exchange information (Fung et al., 2022). If so, it could have inadvertently exposed them to the intervention. An additional strategy employed in this study was the use of a consistent programming syllabus, instructor, and schedule to minimize potential confounding effects. However, this study did not control whether students consult other professors or find other learning resources online. One example is the growing popularity of TikTok as a knowledge source for programming learners (Garcia, Juanatas, et al., 2022). Finally, albeit the successes and failures were affixed to the groups, their assigned treatments, and corresponding masters, it is important to note that the cognitive and social outcomes were ultimately dependent on the performance of all students and the effectiveness of their online interactions.

CONCLUSION

Computer programming is a difficult course for many students. The traditional teaching format is deemed unfavorable to novice programmers since it places them in a passive role with minimal opportunity to develop critical and metacognitive thinking skills. In pursuit of higher-level reasoning and conceptual understanding, prior works advocated for group learning pedagogies. Acquiring teamwork and collaboration skills for programming students is also essential because real-life software projects demand the coordinated efforts of a team. However, the methodological gaps in existing implementations warrant further research. This study conducted a three-armed cluster-randomized controlled trial to comparatively evaluate group learning pedagogies in computer programming. In all course deliverables, the PP group received the lowest mean scores. Meanwhile, no significant difference was found between the PLTL and PAGL groups. Except for psychological safety, social factors such as task cohesion, interdependence, and group potency were significantly different between the groups. Both PLTL and PAGL groups reported a significant increase in social factors after 14 weeks of intervention. These findings provide a rationale for educational leaders and teachers to formulate curricular plans that integrate PLTL and PAGL in computer programming education. Overall, this study contributes to the literature on group learning, expands the pedagogies in computer programming, and serves as additional empirical evidence on cognitive and sociocultural perspectives of learning.

REFERENCES

Ala, O. G., Yang, H., & Ala, B. K. (2021). Characteristics and Comparison of Peer-Assisted Learning Interactions Among University Students in Harbin, China. *Social Sciences & Humanities Open*, *4*(1), 1-8. https://doi.org/10.1016/j.ssaho.2021.100164

Altintas, T., Gunes, A., & Sayan, H. (2016). A Peer-Assisted Learning Experience in Computer Programming Language Learning and Developing Computer Programming Skills. *Innovations in Education and Teaching International*, *53*(3), 329-337. https://doi.org/10.1080/14703297.2014.993418

Bandura, A. (1977). Self-Efficacy: Toward a Unifying Theory of Behavioral Change. *Psychological Review*, *84*(2), 191-215. https://doi.org/10.1037/0033-295X.84.2.191

Berssanette, J. H., & de Francisco, A. C. (2021). Active Learning in the Context of the Teaching/Learning of Computer Programming: A Systematic Review. *Journal of Information Technology Education: Research*, *20*, 201-220. https://doi.org/10.28945/4767

Bodaker, L., & Rosenberg-Kima, R. B. (2022). Online Pair-Programming: Elementary School Children Learning Scratch Together Online. *Journal of Research on Technology in Education*, 1-18. https://doi.org/10.1080/15391523.2022.2036653

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

Chan, J. Y. K., & Bauer, C. F. (2015). Effect of Peer-Led Team Learning (PLTL) on Student Achievement, Attitude, and Self-Concept in College General Chemistry In Randomized And Quasi Experimental Designs. *Journal of Research in Science Teaching*, *52*(3), 319-346. https://doi.org/10.1002/tea.21197

Demir, Ö., & Seferoglu, S. S. (2020). The Effect of Determining Pair Programming Groups According to Various Individual Difference Variables on Group Compatibility, Flow, and Coding Performance. *Journal of Educational Computing Research*, *59*(1), 41-70. https://doi.org/10.1177/0735633120949787

Eteng, I., Akpotuzor, S., Akinola, S. O., & Agbonlahor, I. (2022). A Review on Effective Approach to Teaching Computer Programming to Undergraduates in Developing Countries. *Scientific African*, *16*, 1-18. https://doi.org/10.1016/j.sciaf.2022.e01240

Fung, C. Y., Su, S. I., Perry, E. J., & Garcia, M. B. (2022). Development of a Socioeconomic Inclusive Assessment Framework for Online Learning in Higher Education. In M. B. Garcia (Ed.), *Socioeconomic Inclusion During an Era of Online Education* (pp. 23-46). IGI Global. https://doi.org/10.4018/978-1-6684-4364-4.ch002

Gafney, L., & Varma-Nelson, P. (2008). *Peer-Led Team Learning: Evaluation, Dissemination, and Institutionalization of a College Level Initiative*. Springer. https://doi.org/10.1007/978-1-4020-6186-8

Gamage, L. N. (2021). A Bottom-Up Approach for Computer Programming Education. *Journal of Computing Sciences in Colleges*, *36*(7), 66-75. https://dl.acm.org/doi/abs/10.5555/3469581.3469588

Garcia, M. B. (2021). Cooperative Learning in Computer Programming: A Quasi-Experimental Evaluation of Jigsaw Teaching Strategy with Novice Programmers. *Education and Information Technologies*, *26*(4), 4839-4856. https://doi.org/10.1007/s10639-021-10502-6

Garcia, M. B. (2022). Hackathons as Extracurricular Activities: Unraveling the Motivational Orientation Behind Student Participation. *Computer Applications in Engineering Education*, *30*(6), 1903-1918. https://doi.org/10.1002/cae.22564

Garcia, M. B., Enriquez, J. B. R., Adao, R. T., & Happonen, A. (2022). "Hey IDE, Display Hello World": Integrating a Voice Coding Approach in Hands-on Computer Programming Activities. *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*. https://manuelgarcia.info/publication/voice-coding-approach

Garcia, M. B., Juanatas, I. C., & Juanatas, R. A. (2022). TikTok as a Knowledge Source for Programming Learners: A New Form of Nanolearning? *2022 10th International Conference on Information and Education Technology (ICIET 2022)*. https://doi.org/10.1109/ICIET55102.2022.9779004

Garcia, M. B., & Revano, T. F. (2021). Assessing the Role of Python Programming Gamified Course on Students' Knowledge, Skills Performance, Attitude, and Self-Efficacy. *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, 1-5. https://doi.org/10.1109/HNICEM54116.2021.9731935

Garcia, M. B., Rull, V. M. A., Gunawardana, S. S. J. D., Bias, D. J. M., Chua, R. C. C., Cruz, J. E. C., Raguro, M. C. F., & Perez, M. R. L. (2022). Promoting Social Relationships Using a Couch Cooperative Video Game: An Empirical Experiment With Unacquainted Players. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, *14*(1), 1-18. https://doi.org/10.4018/IJGCMS.303106

Garcia, M. B., & Yousef, A. M. F. (2022). Cognitive and Affective Effects of Teachers' Annotations and Talking Heads on Asynchronous Video Lectures in a Web Development Course. *Research and Practice in Technology Enhanced Learning*, *18*(20), 1-23. https://doi.org/10.58459/rptel.2023.18020

Gladstone, J. R., & Cimpian, A. (2021). Which role models are effective for which students? A Systematic Review and Four Recommendations for Maximizing the Effectiveness of Role Models in STEM. *International Journal of STEM Education*, *8*(1), 59. https://doi.org/10.1186/s40594-021-00315-x

Goñi, J., Cortázar, C., Alvares, D., Donoso, U., & Miranda, C. (2020). Is Teamwork Different Online Versus Face-to-Face? A Case in Engineering Education. *Sustainability*, *12*(24), 1-18. https://doi.org/10.3390/su122410444

Gosser, D. K., Cracolice, M. S., Kampmeier, J. A., Roth, V., Strozak, V. S., & Varma-Nelson, P. (2001). *Peer-Led Team Learning: A Guidebook*. Prentice Hall.

Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair Programming in Education: A Literature Review. *Computer Science Education*, *21*(2), 135-173. https://doi.org/10.1080/08993408.2011.579808

Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. K. (2009). The Effectiveness of Pair Programming: A Meta-Analysis. *Information and Software Technology*, *51*(7), 1110-1122. https://doi.org/10.1016/j.infsof.2009.02.001

Hawlitschek, A., Berndt, S., & Schulz, S. (2022). Empirical Research on Pair Programming in Higher Education: a Literature Review. *Computer Science Education*, 1-29. https://doi.org/10.1080/08993408.2022.2039504

Hayashi, Y., Fukamachi, K., & Komatsugawa, H. (2015). Collaborative Learning in Computer Programming Courses That Adopted the Flipped Classroom. *2015 International Conference on Learning and Teaching in Computing and Engineering*, 209-212. https://doi.org/10.1109/LaTiCE.2015.43

Hennessy, S. (1993). Situated Cognition and Cognitive Apprenticeship: Implications for Classroom Learning. *Studies in Science Education*, *22*(1), 1-41. https://doi.org/10.1080/03057269308560019

Honicke, T., & Broadbent, J. (2016). The Influence of Academic Self-efficacy on Academic Performance: A Systematic Review. *Educational Research Review*, *17*, 63-84. https://doi.org/10.1016/j.edurev.2015.11.002

Jacobs, C. T., Gorman, G. J., Rees, H. E., & Craig, L. E. (2016). Experiences With Efficient Methodologies for Teaching Computer Programming to Geoscientists. *Journal of Geoscience Education*, *64*(3), 183-198. https://doi.org/10.5408/15-101.1

Järvelä, S., Kirschner, P. A., Hadwin, A., Järvenoja, H., Malmberg, J., Miller, M., & Laru, J. (2016). Socially Shared Regulation of Learning in CSCL: Understanding and Prompting Individual- and Group-Level Shared Regulatory Activities. *International Journal of Computer-Supported Collaborative Learning*, *11*(3), 263-280. https://doi.org/10.1007/s11412-016-9238-2

Khan, I. A., Iftikhar, M., Hussain, S. S., Rehman, A., Gul, N., Jadoon, W., & Nazir, B. (2020). Redesign and validation of a computer programming course using Inductive Teaching Method. *PLOS ONE*, *15*(6), e0233716. https://doi.org/10.1371/journal.pone.0233716

Kreijns, K., Kirschner, P. A., & Jochems, W. (2003). Identifying the Pitfalls for Social Interaction in Computer-Supported Collaborative Learning Environments: A Review of the Research. *Computers in Human Behavior*, *19*(3), 335-353. https://doi.org/10.1016/S0747-5632(02)00057-2

Lawler, Edward J., Thye, Shane R., & Yoon, J. (2000). Emotion and Group Cohesion in Productive Exchange. *American Journal of Sociology*, *106*(3), 616-657. https://doi.org/10.1086/318965

Lockwood, P., & Kunda, Z. (1997). Superstars and Me: Predicting the Impact of Role Models on the Self. *Journal of Personality and Social Psychology*, *73*(1), 91-103. https://doi.org/10.1037/0022-3514.73.1.91

Lombardi, D., & Shipley, T. F. (2021). The Curious Construct of Active Learning. *Psychological Science in the Public Interest*, *22*(1), 8-43. https://doi.org/10.1177/1529100620973974

Lou, Y., Abrami, P. C., & d'Apollonia, S. (2001). Small Group and Individual Learning with Technology: A Meta-Analysis. *Review of Educational Research*, *71*(3), 449-521. https://doi.org/10.3102/00346543071003449

Machanick, P. (2007). A Social Construction Approach to Computer Science Education. *Computer Science Education*, *17*(1), 1-20. https://doi.org/10.1080/08993400600971067

Major, C. (2020). Collaborative Learning: A Tried and True Active Learning Method for the College Classroom. *New Directions for Teaching and Learning*, *2020*(164), 19-28. https://doi.org/10.1002/tl.20420

Malik, S. I., & Coldwell-Neilson, J. (2017). Comparison of Traditional and ADRI Based Teaching Approaches in an Introductory Programming Course. *Journal of Information Technology Education: Research*, *16*, 267-283. https://doi.org/10.28945/3793

Margulieux, L. E., Morrison, B. B., & Decker, A. (2020). Reducing withdrawal and failure rates in introductory programming with subgoal labeled worked examples. *International Journal of STEM Education*, *7*(1), 19. https://doi.org/10.1186/s40594-020-00222-7

Marx, D. M., & Ko, S. J. (2012). Superstars "like" me: The Effect of Role Model Similarity on Performance Under Threat. *European Journal of Social Psychology*, *42*(7), 807-812. https://doi.org/10.1002/ejsp.1907

Mills, N. M., Blackmon, A. T., McKayle, C., Stolz, R., & Romano, S. (2020). Peer-Led Team Learning and its Effect on Mathematics Self-Efficacy and Anxiety in a Developmental Mathematics Course. In O. Ortega, E. D. Lawrence, & E. H. Goins (Eds.), *The Golden Anniversary Celebration of the National Association of Mathematicians* (Vol. 759, pp. 93-102). American Mathematical Society. https://doi.org/10.1090/conm/759

Morgenroth, T., Ryan, M. K., & Peters, K. (2015). The Motivational Theory of Role Modeling: How Role Models Influence Role Aspirants' Goals. *Review of General Psychology*, *19*(4), 465-483. https://doi.org/10.1037/gpr0000059

Muller, O., Shacham, M., & Herscovitz, O. (2018). Peer-Led Team Learning in a College of Engineering: First-Year Students' Achievements and Peer Leaders' Gains. *Innovations in Education and Teaching International*, *55*(6), 660-671. https://doi.org/10.1080/14703297.2017.1285714

Nguyen, K. A., Borrego, M., Finelli, C. J., DeMonbrun, M., Crockett, C., Tharayil, S., Shekhar, P., Waters, C., & Rosenberg, R. (2021). Instructor Strategies to Aid Implementation of Active Learning: A Systematic Literature Review. *International Journal of STEM Education*, *8*(1), 1-18. https://doi.org/10.1186/s40594-021-00270-7

Nokes-Malach, T. J., Richey, J. E., & Gadgil, S. (2015). When Is It Better to Learn Together? Insights from Research on Collaborative Learning. *Educational Psychology Review*, *27*(4), 645-656. https://doi.org/10.1007/s10648-015-9312-8

Olivera, F., & Straus, S. G. (2004). Group-to-Individual Transfer of Learning: Cognitive and Social Factors. *Small Group Research*, *35*(4), 440-465. https://doi.org/10.1177/1046496404263765

Padberg, F., & Muller, M. M. (2003). Analyzing the Cost and Benefit of Pair Programming. *5th International Workshop on Enterprise Networking and Computing in Healthcare Industry*, 166-177. https://doi.org/10.1109/METRIC.2003.1232465

Polly, D., Allman, B., Casto, A. R., & Norwood, J. (2017). Sociocultural Perspectives of Learning. In R. E. West (Ed.), *Foundations of Learning and Instructional Design Technology*. EdTech Books. https://edtechbooks.org/lidtfoundations/sociocultural_perspectives_of_learning

Schulz, S., Berndt, S., & Hawlitschek, A. (2022). Exploring Students' and Lecturers' Views on Collaboration and Cooperation in Computer Science Courses - A Qualitative Analysis. *Computer Science Education*, 1-24. https://doi.org/10.1080/08993408.2021.2022361

Sheard, J., & Hagan, D. (1998). Our Failing Students: A Study of a Repeat Group. *6th Annual Conference on the Teaching of Computing and the 3rd Annual Conference on Integrating Technology into Computer Science Education*, 223–227. https://doi.org/10.1145/282991.283550

Taber, K. S. (2019). Experimental Research into Teaching Innovations: Responding to Methodological and Ethical Challenges. *Studies in Science Education*, *55*(1), 69-119. https://doi.org/10.1080/03057267.2019.1658058

Tennyson, M. F., Casteele, J., & Morena, A. R. P. (2018). A Study of Peer-Assisted Learning in Introductory Programming Courses. *Journal of Computing Sciences in Colleges*, *33*(5), 55-62. https://dl.acm.org/doi/abs/10.5555/3204979.3204991

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), 1-13. https://doi.org/10.1145/2996201

Van den Bossche, P., Gijselaers, W. H., Segers, M., & Kirschner, P. A. (2006). Social and Cognitive Factors Driving Teamwork in Collaborative Learning Environments: Team Learning Beliefs and Behaviors. *Small Group Research*, *37*(5), 490-521. https://doi.org/10.1177/1046496406292938

Vihavainen, A., Airaksinen, J., & Watson, C. (2014). A Systematic Review of Approaches for Teaching Introductory Programming and their Influence on Success. *10th Annual Conference on International Computing Education Research*, 19–26. https://doi.org/10.1145/2632320.2632349

Vygotsky, L. S. (1978). *Mind in Society: Development of Higher Psychological Processes*. Harvard University Press. https://doi.org/10.2307/j.ctvjf9vz4

Waite, W. M., Jackson, M. H., Diwan, A., & Leonardi, P. M. (2004). Student Culture vs Group Work in Computer Science. *35th SIGCSE Technical Symposium on Computer Science Education*, 12–16. https://doi.org/10.1145/971300.971308

Wilson, S. B., & Varma-Nelson, P. (2016). Small Groups, Significant Impact: A Review of Peer-Led Team Learning Research with Implications for STEM Education Researchers and Faculty. *Journal of Chemical Education*, *93*(10), 1686-1702. https://doi.org/10.1021/acs.jchemed.5b00862

Winterton, C. I., Dunk, R. D. P., & Wiles, J. R. (2020). Peer-Led Team Learning for Introductory Biology: Relationships Between Peer-Leader Relatability, Perceived Role Model Status, and the Potential Influences of these Variables on Student Learning Gains. *Disciplinary and Interdisciplinary Science Education Research*, *2*(1), 3-11. https://doi.org/10.1186/s43031-020-00020-9

Young, J. D., & Lewis, S. E. (2022). Evaluating Peer-Led Team Learning Integrated into Online Instruction in Promoting General Chemistry Student Success. *Journal of Chemical Education*, *99*(3), 1392-1399. https://doi.org/10.1021/acs.jchemed.1c01118

RELATED RESEARCH

Research Article

Cooperative Learning in Computer Programming: A Quasi-Experimental Evaluation of Jigsaw Teaching Strategy with Novice Programmers

Manuel B. Garcia (2021). *Education and Information Technologies*. https://manuelgarcia.info/publication/cooperative-learning-computer-programming

Conference Paper

TikTok as a Knowledge Source for Programming Learners: A New Form of Nanolearning?

Manuel B. Garcia, Irish C. Juanatas, and Roben A. Juanatas (2022). *2022 10th International Conference on Information and Education Technology*. https://manuelgarcia.info/publication/tiktok-programming-learners

Conference Paper

Assessing the Role of Python Programming Gamified Course on Students' Knowledge, Skills Performance, Attitude, and Self-Efficacy

Manuel B. Garcia and Teodoro F. Revano, Jr. (2021). *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management*. https://manuelgarcia.info/publication/python-programming-gamified-course

LET'S COLLABORATE!

If you are looking for research collaborators, please do not hesitate to contact me at mbgarcia@feutech.edu.ph.

ABOUT THE CORRESPONDING AUTHOR:

Manuel B. Garcia is a professor of information technology and the founding director of the Educational Innovation and Technology Hub (EdITH) at FEU Institute of Technology, Manila, Philippines. His interdisciplinary research interest includes topics that, individually or collectively, cover the disciplines of education and information technology. He is a licensed professional teacher and a proud member of the National Research Council of the Philippines – an attached agency to the country's Department of Science and Technology (DOST-NRCP).