A Structured Pair/Trio Model for Collaborative Learning in Computer Programming: A Novel Approach for Adult Learners

Urvi Maniar* [a] and Kah How Koh [a]

[a] Republic Polytechnic/School of Engineering, Senior Lecturer, Singapore

Urvi Maniar* (urvi_maniar@rp.edu.sg)

This study explores the impact of a structured pair/trio programming model on adult learners in Computer Programming Continuing Education and Training (CET) lessons, using a social constructivist approach. The model was designed based on Dillenbourg's (1999) requirements for collaborative learning, with structured pair/trio programming exercises created to enhance adult learners' programming skill acquisition. The study employed a mixed-method approach to evaluate the efficacy of the model. Mid-Semester Assessment (MSA) scores between control and experimental groups were compared while quantitative and qualitative insights through survey questionnaire, interviews, and independent lesson observation were also gathered. Although MSA score comparisons showed no statistically significant difference between groups, survey responses from the experimental group highlighted the benefits of structured pair/trio programming. Learners reported improved error detection, enhanced peer learning, and increased confidence in programming. Motivation and engagement were high, though some challenges emerged, such as group compatibility and varying participation levels. Notably, most respondents were willing to recommend this approach to others. In the student interviews conducted, further benefits of structured pair/trio learning such as real-time feedback, reduced anxiety, and deeper conceptual understanding through structured discussions were uncovered. Staff interviews supported these observations, noting increased peer problem-solving and engagement in CET lessons. An independent lesson observation confirmed high student participation and the application of social constructivist principles. Despite these positive outcomes, challenges like skill mismatches and participation inconsistencies indicate areas for improvement. The study's limited sample size and institutional setting constrain generalizability. Future research should explore best practices for group formation, rotation strategies, and long-term effects on student retention and post-course learning. These insights provide valuable guidance for educators seeking to enhance collaborative programming education.

Keywords: *Structured pair/trio, Social constructivism, Programming education, Mixed-method research*

Introduction

Computer programming is a difficult field that requires a deep understanding of programming practices and concepts. Teaching programming is a complex process which is difficult for novice learners (Gomes & Mendes, 2007). The high failure and drop-out rates associated with introductory programming courses are a common issue, prompting many researchers to suggest various tools and methodologies aimed at improving the learning experience for students in programming (Ma, Ferguson, Roper, & Wood, 2011). Recent educational research has focused a great deal of attention on the study of successful computer programming teaching methods, especially pair programming. Traditional pair programming, particularly the driver-navigator model, has proven effective in enhancing performance, teamwork, and learning outcomes in computer science courses (Cliburn, 2003; McDowell et al., 2002). Incorporating pair programming in computer science classrooms has been shown to enhance student learning, increase satisfaction, and reduce common frustrations. This approach also alleviates the burden on educators, as students begin to tap on their peers for technical support rather than depending solely on the teaching staff (Williams & Upchurch, 2001). Studies show that students who work in pairs on programming create better programs, have increased motivation toward programming and have more confidence in their coding skills than those who work alone (McDowell et al., 2002). Despite its benefits, pair programming presents limitations like the failure to foster genuine collaboration, with students sharing a machine without engaging in meaningful interaction (Bevan et al., 2002).

Goel and Kathuria (2010) introduced a framework for pair programming that integrates Dillenbourg's four conditions for collaborative learning. Their approach focuses on ensuring both students in a pair contribute equally to the task, thus promoting collaboration through clearly defined roles. Their study found that this method significantly improved problem-solving skills, the quality of work, trust, and teamwork, particularly helping inexperienced programmers perform at the same level as their more experienced peers. The framework highlights the importance of structured collaboration in enhancing both technical and soft skills in programming education. The study was conducted with undergraduate students, specifically tested in an introductory computer programming course offered to first-year engineering students.

The effectiveness of these collaborative methods remains underexplored for adult learners. Our study aims to fill that gap by providing empirical insights that can inform educators and practitioners about the potential advantages of structured collaborative learning environments in developing essential programming skills of adult learners.

In this study, a model based on Dillenbourg's (1999) set of four conditions was adapted to establish an active learning context for a CET class in Computer Programming. The four conditions include setting up initial conditions such as group formation, over-specifying roles within the collaboration, scaffolding effective interactions through structured activities, and monitoring and regulating interactions to ensure engagement and focus throughout the learning process. The study was conducted from a social constructivist perspective in CET, rooted in the four social constructive principles of knowledge construction, zone of proximal development, collaborative learning and contextual learning. The social constructivist approach emphasizes the value of social interaction and collaboration in the learning process. According to Vygotsky's theory, learning occurs most effectively through social engagement and active participation within a community of practice (Vygotsky, 1978). In the context of this study, structured pair/trio programming activities were designed to foster knowledge construction through individual and peer interaction, supporting both cognitive development and the acquisition of programming skills in adult learners. Table 1 illustrates how Dillenbourg's (1999) four conditions for collaborative learning were applied in the study, aligning each condition with the corresponding social constructivist principle to guide the implementation of structured pair/trio programming in the CET Computer Programming class.

Table 1: Dillenbourg's Set of Conditions (1999) and Social Constructivist Principles

Dillenbourg's Set of Conditions (1999)	Social Constructivist Principles
Set up the initial conditions: Pair/Trio arrangement fixed at the start of the semester and followed for all lessons in the module	*Knowledge construction*: Learners engage in collaborative programming tasks, constructing knowledge together through discussion and problem-solving
Over-specify the collaboration contract with scenario based on roles: Mixture of academically strong and not-so-strong members in each Pair/Trio	*Zone of proximal development*: Pair/trio programming offers opportunities for learners to work within their zone, challenge themselves, and learn from peers
Scaffold productive interactions by encompassing interaction in medium: Deliberate activities for individual, Pair/Trio, and Team discussions	*Collaborative Learning*: Pair/trio settings foster collaboration and communication between learners
Monitor and regulate the interactions: Trainer monitors and regulates the interaction face-to-face as well as with EdTech tools like Padlet and Kahoot	*Contextual learning*: Learners engage in relevant programming tasks, allowing them to construct knowledge within meaningful, real-world contexts.

The purpose of this study is to investigate the effects of a structured pair/trio approach on the adult learners' computer programming skills based on a social constructivist perspective in CET.

The research questions that drive this investigation are:

- What are the perceived effects of structured pair/trio programming on adult learners' computer programming skills?
- How does the social constructivist approach influence the learning outcomes of adult learners in structured pair/trio programming?

By answering these questions, this study seeks to enhance understanding of how social constructivist teaching methods might improve CET learning outcomes.

Methods

The study involved forty-four learners from Part-time Diploma in Engineering (Electrical and Electronics) who enrolled on a foundational Computer Programming course. These learners were split into two classes: Class A and Class B. Nineteen learners from Class A were placed in the control group, while twenty-five learners from Class B were placed in the experimental group. Based on their cumulative Grade-Point Average (cGPA), learners in the experimental group were grouped in a structured team of pair or trio to work on the individual and group work in the class. The grouping ensured a deliberate mix of academically strong and weak learners. The class was conducted in a lab with 5 desks, each equipped with 5 chairs, where each desk accommodated one pair and one trio for collaborative learning. Learners in the control group followed the usual classroom setting. In the first lesson, learners were briefed about the seating arrangement and instructed on how to interpret the various icons, as shown in Figure 1, used in the lesson materials, which indicated when to work individually, collaborate in pairs/trios, or engage in class discussions. The arrangement was continued for all lessons in the module. Mid-Semester Assessment (MSA) results for both groups were compared to evaluate the effectiveness of the collaborative learning approach. Additionally, interviews and surveys were conducted with participants from the experimental group, as well as staff, to gather insights into the teaching and learning experiences, providing a deeper understanding of the impact of the structured pair/trio setting. An independent staff was invited to observe the learners during one of the lessons

to provide an unbiased, external perspective on the effectiveness of structured pair/trio approach. and identify challenges, assess student engagement, and ensure that the teaching methods aligned with the intended learning outcomes.

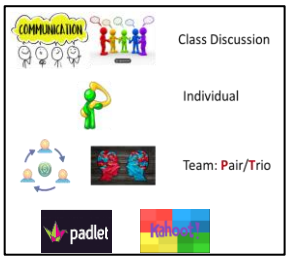

Figure 1: Icons used for activities

The lesson flow for the computer programming lessons is depicted in Figure 2. The learning outcomes and sequence of the key activities were identical for both classes. The hands-on activities for Class B were conducted at various levels—individual, pair/trio, and whole-class sharing—each represented by distinct icons to indicate the corresponding level. Padlet served as a collaboration platform for every lesson. Quiz activities at the end of each lesson were conducted using team-based Kahoots.

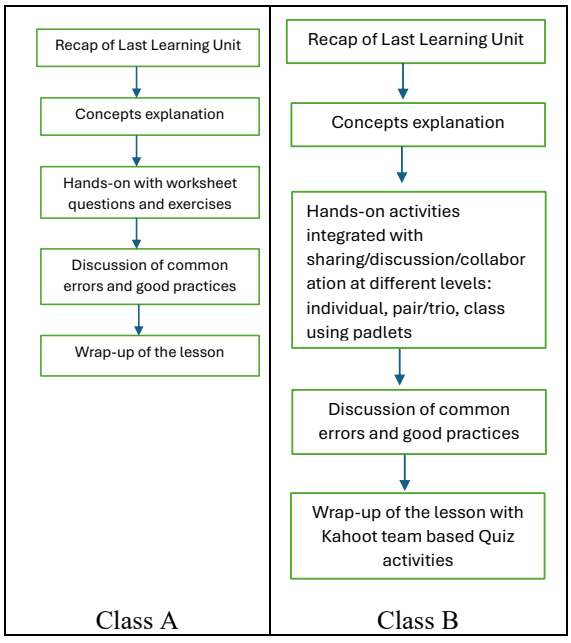

Figure 2: Lesson flow

The study adopted a mixed-method approach comprising of surveys to measure the perceived effects of structured pair/trio programming on learners' computer programming skills, interviews of learners and lecturers as well as class observation by an independent lecturer to gain deeper understanding of the adult learners' experiences. The MSA scores for the two classes were compared at the middle of the course using a two-sample t-test assuming unequal variance. The survey questionnaire was administered to the learners of experimental class B. Three learners and two staff members were interviewed to gather insights on the perceived effects of structured pair/trio programming. The interviews followed a semi-structured format, allowing participants to elaborate on their experiences and perspectives. The learner interviews focused on their experiences with collaboration, challenges, and learning outcomes, while the staff interviews explored their observations of student engagement and effectiveness of the approach. An independent lecturer was invited to observe a lesson and provide an unbiased view on the class dynamics. The qualitative data was analyzed using inductive thematic analysis adopting a semantic approach (Caulfield 2023).

Results and Discussion

Quantitative Statistical Insights

Independent samples t-test was conducted to investigate the effect of a structured pair/trio learning approach on adult learners' computer programming skills, as measured by MSA scores. The results of this analysis are presented in tabular format in Figure 3.

t-Test: Two-Sample Assuming Unequal Variances		
	MSA Scores	
	Class A	Class B
Mean	30.05263	30.92
Variance	77.16374	93.91
Observations	19	25
Hypothesized Mean Difference	0	
df	41	
t Stat	-0.31022	
P(T<=t) one-tail	0.378984	
t Critical one-tail	1.682878	
P(T<=t) two-tail	0.757967	
t Critical two-tail	2.019541	

Figure 3. Independent samples t-Test to compare MSA scores

The mean MSA score for Class A (control group) was 30.05 (SD = 8.78), while the mean MSA score for Class B (pair/trio learning) was 30.92 (SD = 9.69). There is a slight trend towards experimental group (Class B) having higher grades than control group (Class A). However, the independent samples t-test revealed no statistically significant difference between the mean MSA scores of the two groups ($p = 0.76$). This suggests that, in this sample, the structured pair/trio learning approach adopted in class B did not lead to a statistically significant improvement in MSA scores compared to the class A learning approach. This non-significant finding may be attributed to the relatively small sample size, which could have limited the power to detect a statistically significant difference, even if one existed. Furthermore, the wide range of student scores in both groups, as indicated by the standard deviations, highlights the potential influence of other factors, such as prior programming experience, learning styles, or the dynamics within the learning pairs and trios. Future research with larger samples and more detailed measures of these potential moderating factors is warranted.

A survey questionnaire, based on 5-point Likert scale, was administered to the learners of experimental class B in the middle of the course before the mid-semester assessment to measure the perceived effects of structured pair/trio programming on learners' computer

programming skills. Table 2 shows the descriptive statistics for the survey items, which highlight both the strengths and challenges of structured pair/trio programming in enhancing programming skills among adult learners. The survey results indicate that learners had moderately positive perceptions of structured pair/trio programming in their computer programming course. The highest-rated category was Recommending This Approach (M = 3.57, SD = 0.79), suggesting that most learners found it effective and would endorse its use. Identifying and Correcting Mistakes (M = 3.52, SD = 0.85) was also highly rated, reinforcing the role of collaborative approach in debugging and problem-solving. However, learners encountered Challenges (M = 3.17, SD = 0.83), indicating that while beneficial, the approach posed difficulties for some learners. Motivation (M = 3.35, SD = 0.83) and overall Learning Experience (M = 3.35, SD = 0.78) suggest a generally positive impact, but with variations across individuals.

Table 2: Summary table (M and SD of each question)

Survey Item	M (Mean)	SD (Standard Deviation)
Learning Experience	3.35	0.78
Motivation	3.35	0.83
Identifying and Correcting Mistakes	3.52	0.85
Contributing to the Learning of Others	3.48	0.95
Encountering Challenges or Difficulties	3.17	0.83
Recommending This Approach	3.57	0.79

In addition to the statistical analysis, qualitative insights were gathered from learner and staff interviews to provide a deeper understanding of their experiences. The lecturer's independent observations further enrich the findings, offering additional context on the structured pair/trio programming approach. Next, we discuss the qualitative analysis of the data gathered through interviews and lesson observation.

Qualitative insights – Thematic analysis of interviews

Thematic analysis of learner and staff interviews revealed four key themes regarding their experiences with structured pair/trio learning, with a snapshot of the analysis shown in Table 3. The benefits of pair/trio learning emerged as a significant theme in the student interviews where learners highlighted how peer discussions helped in debugging errors, reinforcing understanding, reducing anxiety and increasing confidence in programming. Learners valued the opportunity to see multiple perspectives, learn from and support one another, particularly when encountering challenges that they might not have resolved independently. However, some learners also expressed challenges in pair/trio learning, noting that differences in skill levels and communication issues sometimes slowed their progress, and stronger learners occasionally felt burdened by excessive questioning. Another recurring theme was the potential to apply the pair/trio approach in other modules, particularly in subjects requiring complex calculations or conceptual understanding, such as Circuit Analysis and Control and Mathematics. Learners emphasized that peer verification and structured collaboration could reduce errors and improve accuracy in these modules. Additionally, classroom engagement was identified as a crucial factor, where tools like Kahoot and Padlet were seen as valuable resources to recap key learning points, facilitate discussion, and extend collaborative learning beyond immediate group members. Finally, the theme of learning preferences and group size considerations surfaced, where learners reflected on their preferred working styles, with some favouring independent learning while others found the structured environment more beneficial. Learners also suggested that pair/trio sizes of two to three members reviewed at regular intervals were optimal, whereas larger groups could become less effective due to limited participation. These insights collectively underscore the value and limitations of structured peer collaboration in programming education and highlight the role of interactive engagement strategies in enhancing the learning experience.

Thematic analysis of staff interviews revealed key perspectives on the effectiveness and classroom dynamics of structured pair/trio in programming lessons. Staff emphasized that pair/trio settings enhanced classroom interactions, providing a structured platform for learner-learner and learner-trainer engagement. They noted that this format encouraged peer discussion, immediate clarification of doubts, and collaborative problem-solving, allowing learners to address misconceptions in real time. Additionally, staff identified time efficiency as a major benefit, particularly in CET lessons, where structured collaboration helped learners manage their learning within limited class hours. Pair/trio learning was perceived as a natural and effective approach, promoting deeper discussions and increased participation compared to independent learning. Staff also highlighted that some learners naturally extended their collaboration beyond assigned groups, engaging with peers across teams to solve problems collectively. While no specific challenges were mentioned, staff acknowledged that factors such as workload, module content, and logistical considerations influence whether a pair/trio or a larger group of 4-5 learners would be more suitable for other modules. These insights suggest that structured pair/trio arrangement is a valuable approach that can be adapted to different learning contexts based on instructional needs and logistical factors.

Table 3: *Snapshot of how the thematic analysis was done on learner, and staff interviews to generate the themes*

Raw data (excerpts)	Open code	Themes	Cluster themes
Working in pairs/trio helps as we can counter more problems, more people so we can discuss (student)	Collaborative problem-solving	Effectiveness of Pair/Trio Learning	Benefits of Structured Pair/Trio
CET lesson is limited time so having a Pair/Trio setting helps in the learning. (staff)	Pair/Trio is beneficial for time-constrained lessons	Structured Peer Support in Time-Limited Sessions	Benefits of Structured Pair/Trio
I take the analogy of say bubblegum, see the bubble gum you put in your mouth, it sticks everywhere, right. So this pair/trio is something so unique that it can connect the facilitator to the student. (staff)	Encourages discussion and resolving contradictions instantly	Real-Time Clarification of Concepts	Benefits of Structured Pair/Trio
Sometimes we are afraid to ask a question or also if it is the right time to ask a question, but in pair/trio we are more open and freely ask questions (student)	Pair/trio format encourages open questioning	Confidence in Asking Questions	Classroom Engagement
In pair/trio setting students work more closely as they know that this is a pair so more discussion instead of independently. (staff)	Pair/Trio promotes deeper engagement in discussions	Collaboration and Active Participation	Classroom Engagement
If more knowledgeable, then it may hinder as friends might keep asking questions. (student)	More knowledgeable students may feel burdened by constant questions	Unequal Cognitive Load in Groups	Challenges of Structured Pair/Trio
Sometimes pair/trio are not communicating well. (student)	Communication issues in pair/trio settings	Challenges in Peer Interaction	Challenges of Structured Pair/Trio

Lesson observation and triangulation of findings

To gain non-biased insights into the delivery of a lesson using a structured pair/trio setting in a programming module, an independent lecturer was invited to observe a session on Strings. The observer noted that the class was highly interactive, with learners actively participating in team-based activities and whole-class discussions, particularly during exercises involving hexadecimal and ASCII values. Collaborative learning was evident, as learners engaged in peer discussions, worked together on coding problems, and supported one another in debugging and troubleshooting. The observer also highlighted that the learning environment was safe, allowing learners—especially the more vocal ones—to freely share ideas, contribute to coding activities on the whiteboard, and experiment with different coding approaches. Real-world coding exercises in Visual Studio, where learners independently coded, sought clarification from their peers, and engaged in cross-team discussions to compare solutions reinforced the authenticity in learning. The observer further noted that the lesson promoted self-directed learning, as learners were encouraged to attempt exercises independently before engaging in team discussions. Additionally, reflective learning was incorporated, with learners summarizing key learning points at the end of the session. The observer suggested that encouraging peer movement across teams could further enhance collaboration and emphasized the importance of ensuring logistical readiness, such as reminding learners to bring laptops for coding activities.

The findings from the lesson observation helped to triangulate the insights gathered from student and staff interviews, reinforcing the benefits of structured pair/trio learning in fostering peer collaboration, engagement, and knowledge-sharing. Like the perspectives shared by learners and staff, the observer noted that learners leveraged peer interactions to clarify doubts, troubleshoot coding errors, and reinforce their understanding of key concepts. This alignment across multiple data sources strengthens the validity of the study's findings on the effectiveness of structured peer collaboration in programming education.

Recommended Next Steps

Educators should consider a structured process for introducing and sustaining pair/trio learning. We propose the following five-step approach, as illustrated through a flowchart in Figure 4 below.

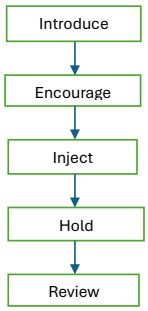

Figure 4: Flow-chart on recommended next steps

The process begins with deliberately introducing structured Pair/Trio settings in the class instead of letting it happen by chance. Once implemented, it is essential to encourage learners to work in their pairs and trios during their lessons. Worksheet Questions/ Exercise Questions are recommended to be injected in the lessons for

pair/trio and class discussions to reinforce engagement and peer exchange. The learners are to be held accountable to work with their pair/trio for their submissions. Finally, the grouping should be reviewed periodically, allowing educators to reassess and adjust groupings to maximize learning effectiveness and accommodate student compatibility. This structured approach provides a practical framework for integrating pair/trio learning into the curriculum, ensuring that collaboration enhances student engagement, knowledge construction, and problem-solving skills.

Conclusions

The qualitative analysis of student interviews, staff interviews, and lesson observations provided valuable insights into the perceived effects of structured pair/trio learning in a programming module. Findings from learners highlighted the benefits of peer collaboration, such as improved problem-solving, enhanced engagement, and increased confidence in programming. Staff interviews reinforced these perspectives, emphasizing how structured peer interactions enhanced classroom dynamics and time efficiency, particularly in time-constrained CET lessons. Additionally, the lesson observation by an independent lecturer validated these findings, showcasing active student participation, peer-driven knowledge-sharing, and self-directed learning in a socially constructivist environment. The triangulation of these data sources strengthens the study's conclusions, demonstrating that while structured pair/trio learning fosters engagement, collaboration, and knowledge construction, its effectiveness depends on group formation strategies, logistical considerations, and the nature of the learning tasks.

Limitations and Challenges

This study has certain limitations and challenges that must be considered. Skill level mismatches and compatibility issues in pair/trio were noted. The findings cannot be generalized due to the small sample size and learner-centric teaching context of the higher education institution. To draw more definitive conclusions, future research should collect data from a larger population of adult learners to ensure broader applicability of findings. Additionally, decisions regarding pair/trio work should consider individual needs, learning preferences, and course context to provide a more effective learning experience. Regular review and reorganization of pair/trio groups are recommended to ensure compatibility and sustained engagement among learners. Future research should also examine how structured Pair/Trio programming influences attrition rates in adult learning programs to assess its long-term impact. These insights offer practical implications for educators seeking to optimize collaborative learning structures in programming education and beyond.

Acknowledgements

The authors are thankful to the cooperation and support extended by Ms Shri Latha and Dr Jiang Lijun in conducting the lessons. The first author is grateful to Dr Abdul Kahlid and Dr Jeeva Periasamy for their insightful and enriching discussions. The inputs given by reviewers and Dr Jolly Parikh also need a special mention.

References

Bevan, J., Werner, L., & McDowell, C. (2002). *Guidelines for the use of pair programming in a freshman programming class*. Conference on Software Engineering Education and Training (pp. 100-107). KY: IEEE Computer Society.

Cliburn, D. C. (2003). *Experiences with pair programming at a small college*. Journal of Computing Sciences in Colleges, 19(1), 20-29.

Caulfield, J. (2023, June 22). *How to do thematic analysis | Step-by-step guide & examples.* Scribbr. Retrieved January 31, 2024, from https://www.scribbr.com/methodology/thematic-analysis/

Dillenbourg, P. (1999). What do you mean by collaborative learning? *Collaborative-learning: Cognitive and computational approaches.*, 1-19.

Goel, S., & Kathuria, V. (2010). *A novel approach for collaborative pair programming*. Journal of Information Technology Education Research, 9, 183–196. https://doi.org/10.28945/1290

Gomes, A., & Mendes, A. J. (2007). *Learning to program - difficulties and solution*. Conference paper in proceedings of International Conference on Engineering Education, Coimbra, Portugal.

Ma, L., Ferguson, J., Roper, M., & Wood, M. (2011). *Investigating and improving the models of programming concepts held by novice programmers*. Computer Science Education, 21(1), 57-80. doi:10.1080/08993408.2011.554722.

McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). *The effects of pair programming on performance in an introductory programming course*. Proceedings of the Thirty-Third Technical Symposium on Computer Science Education (SIGCSE 2002), pp. 38-42. ACM Press.

Vygotsky, L. (1978). *Mind in Society: The development of higher psychological processes*. https://ci.nii.ac.jp/ncid/BA03570814

Williams, L., & Upchurch, R. L. (2001). *In support of student pair-programming.* ACM SIGCSE Bulletin, 33(1), 327–331. https://doi.org/10.1145/366413

ISATE2025_Approval
_Certificate_UM.pdf