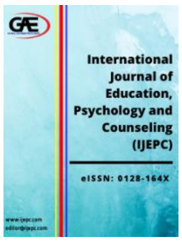
International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052


International Journal of Education, Psychology and Counseling (IJEPC)
eISSN: 0128-164X

INTERNATIONAL JOURNAL OF EDUCATION, PSYCHOLOGY AND COUNSELLING (IJEPC)
www.ijepc.com

PAIR PROGRAMMING FOR LEARNING PROGRAMMING SUBJECT IN HIGHER EDUCATION: A SYSTEMATIC REVIEW

Nor Zalina Ismail[1*], Mohd Rizal Razak[2]

[1] School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Cawangan Pahang Kampus Raub, Malaysia
Email: nza1601@uitm.edu.my
[2] School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Cawangan Pahang Kampus Raub, Malaysia
Email: dragon_admire007@uitm.edu.my
[*] Corresponding Author

Article Info:

Article history:
Received date: 30.06.2024
Revised date: 17.07.2024
Accepted date: 25.09.2024
Published date: 30.09.2024

To cite this document:

Ismail, N. Z., & Razak, M. R. (2024). Pair Programming For Learning Programming Subject In Higher Education: A Systematic Review. .*International Journal of Education, Psychology and Counseling, 9* (55), 770-789.

DOI: 10.35631/IJEPC.955052.

Abstract:

This Systematic Literature Review (SLR) investigates the role of pair programming in higher education, focusing on its impact on students' learning outcomes, collaboration, and programming skill development. As programming courses become increasingly complex, the demand for effective teaching methods has led to the growing popularity of collaborative learning techniques, such as pair programming. However, questions remain regarding its scalability, adaptability, and long-term benefits in educational settings. A thorough search of academic publications from 2021 to 2024 was conducted to explore these issues using databases such as Scopus and Web of Science (WoS). The study followed the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework, starting with 42 articles, of which 22 met the inclusion criteria for further analysis. Articles were excluded for irrelevance, mismatched titles, lack of empirical data, or unavailability of full texts. The review categorized its findings into three key themes: (1) Remote and distributed Pair Programming (RPP), (2) psychological and social dynamics, and (3) pedagogical approaches and tools. The results suggest pair programming enhances students' learning performance, particularly in introductory programming courses, by improving problem-solving skills and coding proficiency. However, challenges such as unequal participation and student resistance persist. The success of pair programming largely depends on proper implementation and teacher facilitation. While the review indicates that pair programming holds promise as an instructional strategy in higher education, it emphasizes the need for further research to assess its long-term impact and address its implementation challenges.

Keywords:

Higher Education, Impact, Learning, Pair Programming, Programming

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

Introduction

Pair programming is a collaborative approach to software development where two programmers work together at one workstation (Nagappan et al., 2003; Prabhakar, 2011). It has gained considerable attention in educational contexts. In this environment, students often face the dual challenge of mastering complex programming concepts and developing the collaborative skills essential in professional software development. Notably, pair programming offers a practical solution to these challenges, fostering a deeper understanding of coding principles through collaboration and peer learning. The significance of pair programming in higher education is multifaceted. First, it aligns with contemporary educational theories emphasizing active learning and social constructivism (Batten & Ross, 2021). These theories suggest that knowledge is constructed through interaction and dialogue, making pair programming a natural fit for environments where students learn through doing. In addition, this collaborative learning environment can also help reduce the cognitive load on individual students (Janssen & Kirschner, 2020; Wang et al., 2020), as tasks are divided and peer support is readily available.

Moreover, pair programming addresses some of the inherent challenges in teaching programming, notably the isolation often experienced by novice programmers (Fitzgerald et al., 2008; Pantic et al., 2016). Note that learning to code can be a solitary endeavor, leading to frustration and disengagement for some students (Forrester et al., 2022; McSorley et al., 2019; Plonka et al., 2012). Hence, pair programming mitigates these issues by creating a support system where students can rely on each other to navigate difficulties. This aspect is particularly relevant in introductory programming courses, where the dropout rate is often high due to the steep learning curve. Furthermore, by fostering a collaborative environment, pair programming can help students feel more connected and supported, which may improve retention rates and overall success in programming courses. The practice of pair programming also prepares students for real-world software development, where teamwork and collaboration are essential. In industry, software development is rarely a solitary activity; it often involves working in teams (Akgün, 2020; Defranco & Laplante, 2017; Endriulaitienė & Cirtautienė, 2021) where communication, collaboration, and collective problem-solving are key to success. Therefore, by incorporating pair programming into the curriculum, educational institutions can provide students with a more authentic learning experience that mirrors professional practices. This enhances their technical skills and develops their ability to work effectively in a team, an invaluable skill in the software industry.

Despite its benefits, the implementation of pair programming in higher education is not without challenges. Differences in skill levels between partners can lead to an imbalance in participation, where one student dominates the coding process while the other remains passive (Werner et al., 2004). This issue highlights the importance of careful pair selection and the need for instructors to monitor and guide the process to ensure that both students are actively engaged. Additionally, some students may initially resist pair programming due to a preference for working independently or concerns about peer evaluation (Beasley & Johnson, 2022; Van Der Meulen & Aivaloglou, 2021). Overcoming these challenges requires a well-structured approach, including clear guidelines, effective pairing strategies, and ongoing support from instructors. Pair programming represents a valuable pedagogical approach to learning programming in higher education. It also promotes active learning, reduces student isolation, and prepares students for the collaborative nature of professional software development. While challenges exist, they can be mitigated through thoughtful implementation and support. As such, pair programming offers a promising strategy for improving learning outcomes in programming education.

Literature Review

Pair programming, a collaborative learning method, has garnered significant attention in computer science education for its potential to enhance students' learning experiences and outcomes in higher education. Various studies have explored the efficacy of this pedagogical approach, yielding insights into its benefits and challenges. Research by Xu et al. (2023) it was revealed that pair programming fosters a collaborative problem-solving environment, which is crucial for computer programming education. The study identified four distinct collaborative patterns: consensus-achieved, argumentation-driven, individual-oriented, and trial-and-error, each associated with varying levels of process and summative performances. Notably, the study underscored the significance of examining these collaborative patterns from a multimodal perspective to optimize educational outcomes in programming courses. Similarly, Rong et al. (2012) they conducted an empirical study on the integration of pair programming into Personal Software Process (PSP) education. The study discovered that paired students exhibited improved program quality and exam scores compared to their solo counterparts. The study also highlighted the benefits of pairing in maintaining students' involvement and conformity to process discipline despite the experiment not confirming the expected improvement in performance. Smrtic and Grinstein (2004) also examined the use of pair programming within an academic environment, noting that it was effective for knowledge sharing and learning among graduate students. However, initial awkwardness and limitations in interpretation were observed.

Incorporating pair programming into various educational contexts has been proven to influence students' attitudes toward programming positively. Marimuthu et al. (2020) explored the attitudinal factors influencing the success of Commerce and Computer Science students in programming courses at a South African university. Consequently, the study reported that while both groups recognized the significance of programming, Computer Science students demonstrated a higher overall positive attitude. The research suggested that incorporating pair programming and problem-solving approaches could enhance the confidence and motivation of non-technical students. This, ultimately, improved their performance in programming courses. Similarly, Moore (2014) examined methods to support students in music technology-related higher education in learning computer programming. The study highlighted the effectiveness of collaborative activities, including pair programming, in promoting deeper learning and engagement among novices in programming.

The adoption of pair programming in computer science education is also linked to improving student retention and engagement, particularly among underrepresented groups. Slaten et al. (2005) conducted a case study on the impact of pair programming and agile software methodologies on undergraduate students' perceptions in a software engineering course. The findings suggested that these collaborative pedagogies contributed to more effective learning opportunities. They also discovered that it was particularly beneficial for female and minority students, who often perceive computer science as a solitary activity. Berenson et al. (2004) further explored the experiences of female students in a software engineering course, emphasizing the role of pair programming in enhancing confidence, productivity, and interest in Information Technology (IT) careers. These studies indicate that pair programming not only improves educational outcomes but also addresses issues of diversity and inclusion in computer science education.

In the context of online and remote learning, pair programming has been adapted to meet the challenges posed by the COVID-19 pandemic. Liu and Woo (2021) introduced CodeHelper, a web-based Integrated Development Environment (IDE) designed to facilitate e-mentoring in online programming courses through real-time pair programming. The study discovered that this tool effectively supported teacher-student interactions, reduced the time and cost of distance learning, and allowed students to check the correctness of their programs in a collaborative environment. Lal and Mourya (2022) also highlighted the importance of pair programming in online teaching, identifying it as a key challenge for computer science educators during the pandemic. The study recommended strategies to address these challenges, including the use of feedback mechanisms and collaborative tools to enhance the quality of online education.

The effectiveness of pair programming in higher education is further supported by studies examining its impact on student engagement and performance in programming courses. Mohamed (2019) reported on the successful design of a CS1 programming course at the University of British Columbia-Okanagan, which incorporated pair programming and a partially flipped classroom model. The study reported that these strategies improved class average grades, increased pass rates, and enhanced student engagement and satisfaction. Similarly, Sim et al. (2023) evaluated the experiences of students in software engineering subjects, finding that pair programming and other collaborative methods promoted teamwork and improved the outcomes of real-world software projects. These findings aligned with the broader literature on the benefits of collaborative learning in computer science education.

Despite the numerous advantages of pair programming, some studies have identified potential challenges and limitations. Lytle et al. (2020) investigated the use of pair programming in block-based environments, noting that it could lead to inequitable learning environments and negatively affect novice learners in certain contexts. The study introduced alternative collaboration modes, such as Pair-Separate and Partner Puzzles, to address these issues and enhance the collaborative experience. In addition, Nickel and Barnes (2010) also explored the use of collaborative educational games to complement traditional pair programming. They suggested that such games could address the challenges of evaluating student performance in collaborative settings. Based on the study, pair programming has been demonstrated to be a valuable pedagogical tool in higher education, particularly in computer science courses. It enhances students' collaborative problem-solving skills, improves educational outcomes, and addresses issues of diversity and inclusion. However, its implementation must be carefully designed to address potential challenges and ensure equitable learning opportunities for all students.

Research Question

Based on an advanced searching strategy on the Scopus database for the title "Pair Programming for Learning Programming Subject in Higher Education: A Systematic Review," we discovered and developed three themes. This includes Remote and Distributed Pair Programming (RPP), Psychological and Social Dynamics, and Pedagogical Approaches and Tools. Based on themes and using mnemonics style by Population, Interest, and Context (PICo), two Research Questions (RQs) are identified for this paper:

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

1. How does the implementation of RPP influence learning outcomes and collaborative skills among higher education students in computing programs in online and blended learning environments?

2. What are the psychological and social factors that influence the effectiveness of pair programming among undergraduate and graduate computer science students in in-class and project-based learning scenarios?

RQs are crucial in a Systematic Literature Review (SLR) since they provide the foundation and direction for the entire review process. They guide the scope and focus of the SLR, helping to determine which studies to include or exclude, ensuring that the review remains relevant and specific to the topic of interest. A well-defined RQ ensures that the literature search is exhaustive and systematic, covering all relevant studies that address key aspects of the topic. This minimizes the risk of bias and ensures a complete overview of the existing evidence. Additionally, RQs facilitate the categorization and organization of data from included studies, providing a framework for analyzing findings and synthesizing results to draw meaningful conclusions. They also enhance clarity and focus, avoiding ambiguity and keeping the review concentrated on specific issues, making the findings more actionable and relevant. Furthermore, well-formulated RQs contribute to the transparency and reproducibility of the review, allowing other researchers to follow the same process to verify findings or extend the review to related areas. Ultimately, RQs ensure that the review aligns with the overall objectives of the study, whether it is to identify gaps in the literature, evaluate the effectiveness of interventions, or explore trends in a specific field, making them the backbone of a rigorous, focused, and relevant SLR. Specifying the RQs is the most crucial activity at the planning stage; it is also the most essential part of any SLR as it drives the entire review methodology (Kitchenham, 2007).

RQs are often formulated using the PICo framework, a mnemonic technique that is especially useful in qualitative research. Population, Interest, and Context are referred to as PICo. The meaning of each part is as follows:

1. Population (P):
- This denotes the study's target group or participants. It indicates the target audience for the study, which may be a community, patient group, or particular demographic.

2. Interest (I):
- This stands for the primary subject of interest or phenomenon in the research. It might be a specific experience, behavior, action, or problem that the study seeks to investigate or comprehend.

3. Context (Co):
- This describes the location, place, or particular context in which the population and topic are located. It could allude to a place's physical location, societal or cultural contexts, or any other pertinent background information for the study.

By dividing the main components of the study into these three parts, the PICo framework facilitates the precise and methodical structuring of RQs. It is simpler to search for pertinent literature or create a study when using this method, which guarantees that the research is narrowly focused and the questions are clearly stated.

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

Material And Methods

For performing SLRs, the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) technique is a generally accepted standard that guarantees transparency, completeness, and consistency throughout the procedure. Researchers can improve the accuracy and rigor of their analysis by adhering to PRISMA standards, which guide how to systematically find, filter, and include papers in their review. The method also highlights the significance of randomized studies, acknowledging their ability to lessen bias and provide strong evidence for the review.

Due to their robustness and wide coverage, two important databases—Scopus and Web of Science (WoS) were used in this investigation. WoS and Scopus provide a thorough index of peer-reviewed literature on various subjects. It is understood that no database is flawless; each has drawbacks, such as varying degrees of detail or coverage gaps, which must be considered during the review process.

The four key sub-sections of the PRISMA technique are identification, screening, eligibility, and data abstraction. Finding all pertinent studies requires examining databases for identification. After that, studies are screened by evaluating them against predetermined standards to weed out irrelevant or subpar research. The remaining studies are further assessed during the eligibility phase to ensure they satisfy the inclusion requirements. Ultimately, data abstraction is the process of gathering and combining information from the included studies to create findings that are trustworthy and relevant. This methodical methodology guarantees a high degree of rigor in the systematic review, yielding dependable data that can guide future study and practice.

Identification

This study employed essential steps of the systematic review process to collect a significant body of relevant literature. The process started with selecting keywords, followed by searching for related terms using dictionaries, thesauri, encyclopedias, and existing research. All relevant terms were identified, and search strings were constructed for the WoS and Scopus databases (see Table 1). This initial phase of the systematic review identified 276 publications relevant to the study topic from the two databases.

Table 1: The Search String(Date of Access: August 2024)

Scopus	Web of Science
TITLE-ABS-KEY (("pair programming") AND ("high* institution" OR university OR "high* education")) AND (LIMIT-TO (PUBYEAR, 2021) OR LIMIT-TO (PUBYEAR, 2022) OR LIMIT-TO (PUBYEAR, 2023) OR LIMIT-TO (PUBYEAR, 2024)) AND (LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE, "cp")) AND (LIMIT-TO (LANGUAGE, "English")) AND (LIMIT-TO (PUBSTAGE, "final"))	(("pair programming") AND ("high* institution" OR university OR "high* education")) (Topic) and 2021 or 2022 or 2023 or 2024 (Publication Years) and Article or Proceeding Paper (Document Types) and Early Access (Exclude – Document Types) and English (Languages) and 2021 or 2022 or 2023 or 2024 (Final Publication Year)

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

Screening
During the screening step, potentially relevant research items are assessed to ensure they align with the predefined RQ(s). This phase typically involves selecting studies based on the Pair Programming Strategy in Learning Programming, with duplicate papers being removed at this stage. Initially, 221 publications were excluded, resulting in 55 papers for further examination according to specific inclusion and exclusion criteria (see Table 2). The first criterion focused on literature, as it serves as the primary source of practical recommendations, including reviews, meta-syntheses, meta-analyses, books, book series, chapters, and conference proceedings not covered in the most recent study. The review was restricted to English-language publications from 2021 to 2024. In total, 13 publications were excluded due to duplication.

Table 2: The Searching Selection Criterion

Criterion	Language	Timeline	Literature type	Publication Stage
Inclusion	English	2021 – 2024	Journal (Article), Conference Proceedings	Final
Exclusion	Non-English	< 2021	Book, Review	In Press

Eligibility
Forty-two articles were ready for assessment in the eligibility phase, the third step. At this phase, every article's title and essential material were thoroughly scrutinized to ensure they satisfied the inclusion requirements and complemented the ongoing research goals. As a result, 20 papers were disqualified since their abstracts had nothing to do with the study's goal, their titles were not relevant, they were not in the field, and they lacked full-text access based on empirical data. Thus, there are now 22 papers remaining for the next review.

Data Abstraction and Analysis
An integrative analysis was employed as one of the assessment strategies in this study to examine and synthesize a variety of research designs (quantitative methods). The goal of the competent study was to identify relevant topics and subtopics. The data collection stage was the first step in developing the theme. Figure 2 displays how the authors meticulously analyzed a compilation of 50 publications for assertions or material relevant to the topics of the current study. Correspondingly, the authors evaluated the current significant studies related to pair programming strategy. The methodology used in all studies, as well as the research results, are being investigated. Next, the author collaborated with other co-authors to develop themes based on the evidence in this study’s context. In addition, a log was kept throughout the data analysis process to record any analyses, viewpoints, riddles, or other thoughts relevant to the data interpretation. Finally, the authors compared the results to see if there were any inconsistencies in the theme design process. It is worth noting that if there are any disagreements between the concepts, the authors discuss them amongst themselves.

Quality of Appraisal
According to the guidelines proposed by Kitchenham and Charters (Kitchenham, 2007), once we have selected primary resources we have to assess the quality of the research they present and quantitatively compare them. In this study, we apply Quality Assessment (QA) from Abouzahra et al. (2020), which consists of six QAs for our SLR. The scoring procedure for

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

evaluating each criterion involves three possible ratings: "Yes" with a score of 1 if the criterion is fully met, "Partly" with a score of 0.5 if the criterion is somewhat met but contains some gaps or shortcomings, and "No" with a score of 0 if the criterion is not met at all.

Table 3: Quality Assessment by Experts

Quality Assessment	**Is the purpose of the study clearly stated?**	**Is the interest and the usefulness of the work clearly presented?**	**Is the study methodology clearly established?**	**Are the concepts of the approach clearly defined?**	**Is the work compared and measured with other similar work?**
Expert 1	Yes	Yes	Yes	Yes	Yes
Expert 1	Yes	Yes	Yes	Yes	Yes
Expert 1	Yes	Yes	Yes	Yes	Yes
Total Mark	3	3	3	3	3

A QA procedure that is used to assess a study according to predetermined standards is provided in the table. Each of the following criteria is evaluated by three experts, and the results are reported as "Yes," "Partly," or "No." The following is a thorough explanation:

1. Is the study's goal clearly stated?
- This criterion verifies that the goals of the investigation are precisely stated and specified. A well-defined aim aids in determining the course and extent of the investigation.

2. Is the work's interest and value clearly presented?
- This criterion assesses how well-explained the study's importance and possible contributions are. It gauges the significance and effect of the study.

3. Is the methodology for the study well-defined?
- This evaluates if the research approach is clear and suitable for accomplishing the goals of the investigation. The validity and reproducibility of the study depend heavily on the methodology's clarity.

4. Do the approach's concepts have a clear definition?
- The clarity of the theoretical framework and important concepts is examined in this criterion. Definitions must be understood to comprehend the study's methodology.

5. Is the work evaluated and compared to other comparable works?
- This assesses if the study has been compared to earlier studies. The work's contributions are highlighted, and its place in the larger academic context is helped by comparisons with other studies.

These criteria are used by each expert to independently evaluate the study; the total score is then calculated by adding the scores of all the experts. A study's overall mark, which results from adding the ratings from each of the three experts, must be higher than 3.0 to move on to the next step. This cutoff ensures that only research that fulfills a particular caliber requirement continues.

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

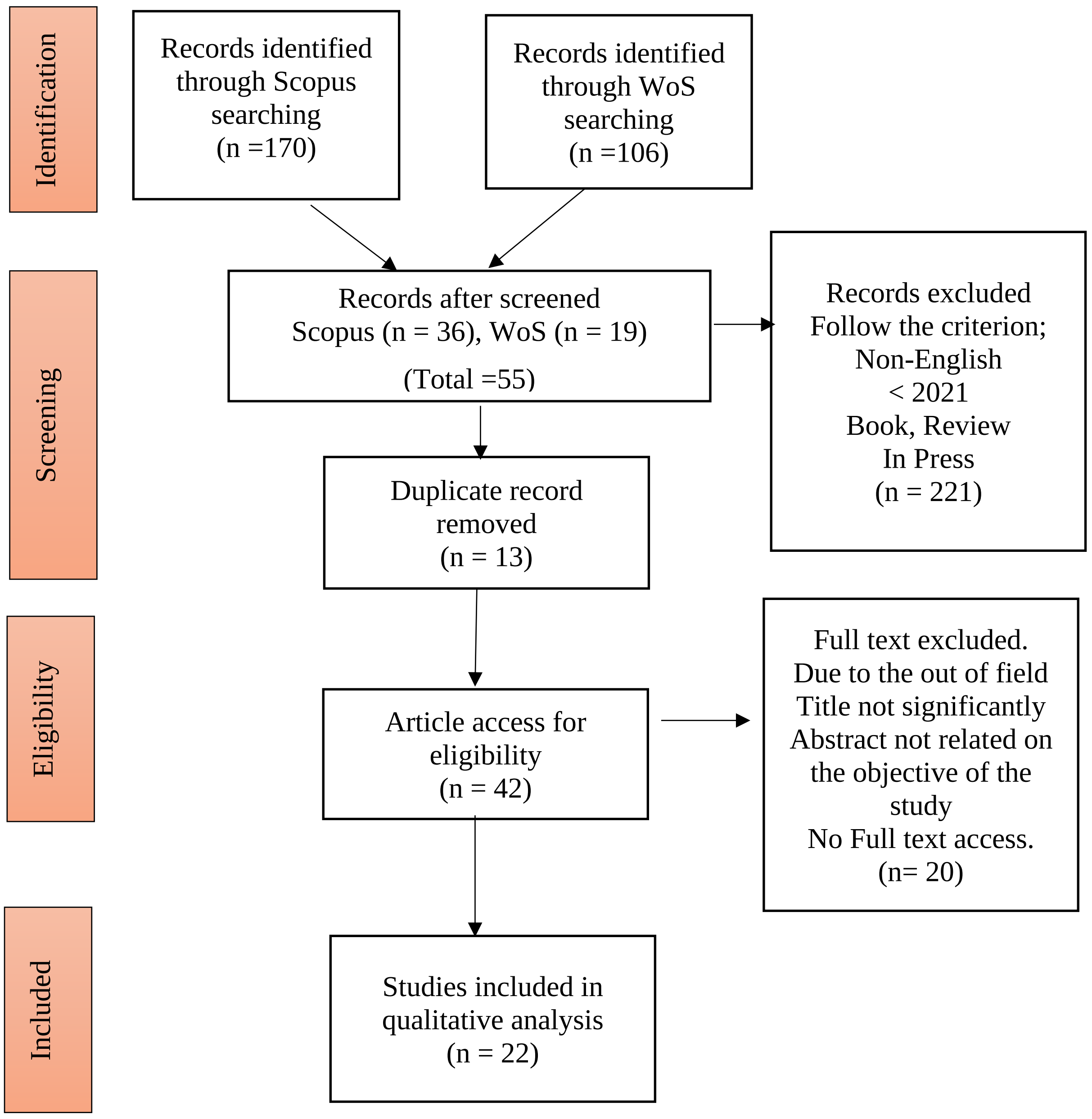

Identification
Screening
Eligibility
Included
Records identified through Scopus searching (n =170)
Records identified through WoS searching (n =106)
Records after screened Scopus (n = 36), WoS (n = 19) (Total =55)
Records excluded Follow the criterion; Non-English < 2021 Book, Review In Press (n = 221)
Duplicate record removed (n = 13)
Article access for eligibility (n = 42)
Full text excluded. Due to the out of field Title not significantly Abstract not related on the objective of the study No Full text access. (n= 20)
Studies included in qualitative analysis (n = 22)

Figure 1: Flow Diagram of The Proposed Searching Study

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

Result and Finding
Table 4 summarizes the background of the selected study.

Table 4: Detail of Studies Included in Qualitative Analysis (n = 22)

No	Authors and Year	Title	Journal	Scopus	Web of Sciences
1	(Adeliyi, Wermelinger, et al., 2021)	Investigating Remote Pair Programming in Part-Time Distance Education	ACM International Conference Proceeding Series	/	/
2	(Durán Toro et al., 2024)	Exploring Gender Bias In Remote Pair Programming Among Software Engineering Students: The twincode Original Study And First External Replication	Empirical Software Engineering	/	/
3	(Tsai et al., 2023)	The effects of online peer-facilitated learning and distributed pair programming on students' learning	Computers and Education	/	
4	(Valovy, 2023)	Psychological aspects of pair programming	ACM International Conference Proceeding Series	/	/
5	(Demir & Seferoglu, 2021b)	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	/	
6	(Weidmann et al., 2023)	Proposal for a Peer-to-Peer Coding Platform for Teaching Introductory Programming to Large Classes of Novice Students	Lecture Notes in Networks and Systems	/	

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

7	(Valový, 2023)	Effects of Pilot, Navigator, and Solo Programming Roles on Motivation: An Experimental Study	Lecture Notes in Networks and Systems	/	
8	(Scott et al., 2023)	Retention in First Stage Undergraduate Computing: Lessons Learned from a Collaborative Learning Intervention	Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE	/	/
9	(Choi, 2021)	'Better Communication Leads to a Higher Output?' An Analysis of Pair Communication on Pair Programming Productivity	IEEE Transactions on Professional Communication	/	
10	(Colin et al., 2024)	Design and Evaluation of a Web-based Distributed Pair Programming Tool for Novice Programmers	Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE	/	
11	(Hawlitschek et al., 2023)	Exploring students' and lecturers' views on collaboration and cooperation in computer science courses - a qualitative analysis	Computer Science Education	/	
12	(Adeliyi, Hughes, et al., 2021)	Remote Pair Programming	SIGCSE 2021 - Proceedings of the 52nd ACM Technical Symposium on Computer Science Education	/	
13	(Sim et al., 2023)	Experiences and Lessons Learned from Real-World Projects in Software Engineering Subject	Software Engineering Education Conference, Proceedings	/	
14	(Liu & Woo, 2021)	CodeHelper: A Web-Based Lightweight IDE for E-Mentoring in Online Programming Courses	2021 3rd International Conference on Computer Communication and the Internet, ICCCI 2021	/	
15	(Küng et al., 2022)	Gender and pair programming–Effects of the gender composition of pairs on collaboration in a robotics workshop	Frontiers in Education	/	

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

16	(Izhikevich et al., 2022)	Exploring Group Dynamics in a Group-Structured Computing Undergraduate Research Experience	ICER 2022 - Proceedings of the 2022 ACM Conference on International Computing Education Research	/	/
17	(Demir & Seferoglu, 2021a)	A Comparison of Solo and Pair Programming in Terms of Flow Experience, Coding Quality, and Coding Achievement	Journal of Educational Computing Research	/	
18	(Bowman et al., 2021)	The Impact of Pair Programming on College Students' Interest, Perceptions, and Achievement in Computer Science	ACM Transactions on Computing Education	/	/
19	(Roque-Hernández et al., 2021)	Acceptance and Assessment in Student Pair-Programming: A Case Study	International Journal of Emerging Technologies in Learning	/	/
20	(Sobral, 2021)	Pair Programming and the Level of Knowledge in the Formation of Pairs	Advances in Intelligent Systems and Computing	/	
21	(Beasley & Johnson, 2022)	The Impact of Remote Pair Programming in an Upper-Level CS Course	Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE	/	/
22	(Bjorn et al., 2022)	It's Okay Because I Worked Really Hard! - Student Justifications for Questionable Collaboration while Solving Computer Labs	Proceedings - Frontiers in Education Conference, FIE	/	

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

To maintain uniformity, the themes that were created were eventually modified. Three programming lecturers with more than five years of experience conducted the analytical selection to ascertain the validity of the difficulties. The domain validity is established during the expert review process, which guarantees the significance, appropriateness, and clarity of each subtheme. To address any disparities in the theme creation procedure, the writers additionally compared their findings. It should be noted that the authors address any discrepancies that may arise regarding the themes with one another. Ultimately, minor adjustments were made to the generated concepts to guarantee coherence. Three specialists with backgrounds in programming instruction conducted the assessments to guarantee the legitimacy of the problems. Notably, by demonstrating domain validity, the expert review stage ensured the sufficiency, clarity, and significance of each sub-theme. Accordingly, the author has made changes at their discretion in response to expert opinions and input.

Remote and Distributed Pair Programming

RPP has emerged as a pivotal strategy in enhancing programming education, particularly in the context of higher education. Moreover, various studies have focused on the impact of RPP on student learning outcomes, engagement, and the psychological and social dynamics within pairs. A synthesis of recent research reveals several key findings and insights critical to understanding the effectiveness of RPP in educational settings.

Remote pair programming has been reported to be an effective pedagogical tool, positively influencing student learning outcomes and engagement. Adeliyi, Wermelinger, et al. (2021) conducted a study on part-time distance education students and concluded that most students perceived RPP as beneficial despite the additional demands on their limited study time. This positive perception was echoed in the work of Beasley and Johnson (2022), who reported that students engaged in remote pair programming scored higher on assignments and exams compared to those who worked individually. Furthermore, these students, particularly females, exhibited an increase in confidence and persistence in their major, highlighting the potential of RPP to improve educational outcomes in a remote setting. Similarly, Tsai et al. (2023) investigated the effects of Distributed Pair Programming (DPP) combined with peer-facilitated learning. They discovered that while the latter significantly enhanced programming skills, enjoyment, and intention to learn, the expected benefits of DPP were not observed. This suggests that while RPP can be beneficial, its effectiveness may depend on the specific educational context and the combination with other pedagogical strategies.

The social and psychological aspects of RPP have been scrutinized to understand how they influence the learning process, particularly concerning gender biases and communication styles. The study by Durán Toro et al. (2024) explored the presence of gender bias in remote pair programming among software engineering students. The original study reported no statistically significant effects related to gender bias in RPP. However, the replication study observed some gender-related differences in behavior, such as more informal communication when students perceived their partner as male and a tendency to delete more source code characters when paired with a female partner. Despite these findings, the results were considered inconclusive due to the small sample size, indicating the need for further research to confirm or refute these initial observations. Adeliyi, Hughes, et al. (2021) also highlighted the importance of the social dynamics in RPP, noting that the success of the method depended on whether student pairs could 'jell' quickly and effectively collaborate, which was influenced by their existing programming experience and the nature of their interaction.

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

The implementation of RPP in higher education is not without challenges. Key barriers include the availability and suitability of communication tools and the need for students to quickly adapt to new collaborative dynamics in a remote setting. Adeliyi, Hughes, et al. (2021) emphasized that generic communication tools were sufficient for facilitating RPP, although the study also identified specific challenges related to the adaptation process. This was particularly relevant in the context of distance education, where students often have less interaction and are required to manage their study time independently. In contrast, the study by Beasley and Johnson (2022) indicated that RPP could perform as effectively as in-person pair programming when these barriers are addressed. This suggests that with proper support and tools, the potential benefits of RPP can be fully realized.

In conclusion, the literature on RPP in higher education presents a complex picture with mixed results. While RPP has been demonstrated to enhance learning outcomes and student engagement, particularly for female students, its effectiveness can vary depending on the context and the presence of supportive tools and pedagogical strategies. The social and psychological dynamics within pairs, including the potential for gender bias, also play a significant role in the success of RPP. However, further research is required to fully understand these effects. Furthermore, addressing the barriers to implementation is crucial for maximizing the benefits of RPP. This suggests that educators must carefully consider the specific needs and circumstances of their students when adopting this approach.

Psychological and Social Dynamics in Pair Programming

Recent studies emphasize the psychological benefits of pair programming, particularly in enhancing student motivation and engagement in software engineering education. Valovy (2023) highlighted that students demonstrate higher intrinsic motivation when engaged in pilot-navigator roles than in solo programming. This is corroborated by findings from Demir and Seferoglu (2021b), who reported increased flow and coding performance in homogeneous pairs when similarity in learning style and friendship is present. These results indicate that effective pair formation, considering personality traits and individual differences, significantly impacts the motivation levels and performance outcomes in pair programming settings.

The compatibility of roles within pair programming setups is strongly influenced by individual personality traits, which can either enhance or diminish the effectiveness of the pairing. Valový's (2023) study on the impact of different programming roles on motivation discovered that specific personality traits, such as openness and extraversion, are better suited to certain roles (pilot and navigator, respectively). Hence, this finding suggests that aligning personality traits with specific roles within pair programming can lead to better motivational outcomes and overall satisfaction with the programming process.

The impact of gender composition on collaboration quality and performance in pair programming was explored by Küng et al. (2022). Their research did not discover significant differences in collaboration outcomes based on gender composition among younger students. However, it noted that homogeneous male pairs in older groups tended to violate collaborative norms more frequently. This suggests gender dynamics may influence collaboration quality in pair programming, particularly as students grow older. Correspondingly, this highlights the need for tailored approaches in mixed and same-gender pair programming to optimize learning and performance outcomes.

Communication within pairs, an essential component of pair programming, was discovered to have varied impacts on productivity. Choi (2021) determined that while communication competency levels significantly affect the perceived communication quality, they do not necessarily correlate with the programming output. This highlights the complexity of communication dynamics in pair programming, suggesting that effective communication is crucial for a satisfying collaborative experience but does not always translate directly into higher productivity.

Pedagogical Approaches and Tools for Pair Programming

The integration of sophisticated tools and platforms significantly enhances the learning outcomes in pair programming scenarios. Weidmann et al. (2023) developed a novel web-based platform embedding an IDE that facilitated live pair programming and demonstrated increased student grades in introductory programming courses. Similarly, Colin et al. (2024) introduced a DPP tool designed specifically for novice programmers, incorporating features to support blended learning scenarios effectively. Both studies suggested that well-integrated platforms can bridge the gap between traditional and online learning environments by enhancing student interaction and collaboration. Collaborative learning interventions, particularly in computing education, have revealed promising results in improving student retention and performance. Scott et al. (2023) observed an improvement in retention rates at Falmouth University's Games Academy following the implementation of pair programming and peer instruction. This aligns with the findings from Sim et al. (2023), who noted that real-world projects involving pair programming enhanced teamwork and project outcomes in software engineering education. In particular, these interventions highlight the critical role of collaborative learning in retaining students and improving their academic and practical skills.

Exploring the differences between pair programming and solo programming, Demir and Seferoglu (2021a) highlighted that pair programming significantly enhanced the flow experience and coding quality over solo programming. This study supported the idea that collaborative coding improves technical skills and enhances psychological engagement among students, which is crucial in educational settings. Additionally, Bowman et al. (2021) examined the broader impacts of pair programming on students' interest and achievement in computer science. The study suggested that while pair programming might not alter course performance significantly, it influences engagement and perception of the subject.

The effective implementation of pair programming faces several challenges, including the need for proper instructional design and alignment with learning objectives. Hawlitschek et al. (2023) highlighted that the lack of structured guidance and assessment of collaborative activities often undermines their potential benefits. Thus, addressing these challenges involves meticulously designing instructional activities and integrating comprehensive assessment strategies to fully leverage the advantages of pair programming in educational settings.

Discussion and Conclusion

RPP is increasingly recognized as a vital tool in programming education, especially within the higher education sector. This approach has demonstrated positive effects on learning outcomes and student engagement. Studies have indicated that students involved in RPP tend to perform better academically and report higher levels of confidence and persistence, particularly among female students. However, the effectiveness of RPP varies across different educational settings and when combined with other pedagogical strategies. In terms of the social and psychological dynamics, the impact of RPP on these aspects is still not fully understood, with studies yielding inconclusive results, especially concerning gender biases. Furthermore, the ability of students to quickly form effective working relationships in RPP settings is influenced by factors such as prior programming experience and communication styles. Challenges to RPP include the adequacy of communication tools and the need for students to adapt to collaborative dynamics remotely. While generic communication tools have been deemed sufficient, specific challenges related to adaptation need to be addressed to harness the full benefits of RPP. Overall, while RPP offers significant potential to enhance educational outcomes in programming, its success depends on contextual factors, the use of effective pedagogical strategies, and overcoming implementation barriers. Hence, further research is necessary to better understand the social dynamics and optimize the integration of RPP in educational practices.

Recent research underscores the psychological advantages of pair programming, particularly in boosting motivation and engagement among software engineering students. Findings indicate that students demonstrate greater intrinsic motivation when engaged in pilot-navigator roles rather than solo programming. This enhanced motivation and performance are evident when pairs are formed based on similarities in learning styles and interpersonal relationships. Other than that, it underscores the importance of effective pair formation, considering personality traits and individual differences. In addition, role compatibility within pair programming is greatly affected by individual personality traits. In particular, specific traits such as openness and extraversion are more suitable for certain roles, suggesting that aligning personality traits with specific programming roles can optimize motivational outcomes and satisfaction in the programming process. Studies exploring the impact of gender on collaboration quality in pair programming revealed nuanced results. While no significant differences were reported among younger students, older male groups tended to deviate from collaboration norms more frequently. This indicates that gender dynamics can influence collaboration quality, especially as students age, necessitating customized strategies for both mixed and same-gender pairing to enhance learning and performance. Therefore, communication within pairs, while crucial for a satisfying collaborative experience, does not directly correlate with productivity. Although high communication competency impacts perceived communication quality, it does not necessarily lead to better programming output. Nevertheless, this separation between communication effectiveness and productivity highlights the complex dynamics of communication in pair programming and emphasizes that while good communication is essential, it is not the sole determinant of productive output.

Conflicts of Interest

The authors declare no conflicts of interest to report regarding the present study.

Acknowledgments

I would like to express my sincere gratitude to the administration of UiTM Cawangan Pahang Kampus Raub for granting me the time off to attend the SLR Seminar. Their support has been invaluable in allowing me to deepen my understanding of the subject.

References

Abouzahra, A., Sabraoui, A., & Afdel, K. (2020). Model composition in Model Driven Engineering: A systematic literature review. *Information and Software Technology, 125*(May), 106316. https://doi.org/10.1016/j.infsof.2020.106316

International Journal of Education, Psychology and Counseling IJEPC
EISSN : 0128-164X
Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

Adeliyi, A., Hughes, J., Kear, K., Law, B., Murphy, B., Rosewell, J., Walshe, A., & Wermelinger, M. (2021). Remote Pair Programming. *SIGCSE 2021 - Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, 1289. https://doi.org/10.1145/3408877.3439681

Adeliyi, A., Wermelinger, M., Kear, K., & Rosewell, J. (2021). Investigating Remote Pair Programming in Part-Time Distance Education. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3481282.3481290

Akgün, A. E. (2020). Team wisdom in software development projects and its impact on project performance. *International Journal of Information Management*. https://doi.org/10.1016/j.ijinfomgt.2019.05.019

Batten, J. S., & Ross, M. S. (2021). A Systematic Review of Social Constructivist Pedagogies in Computing and their Effects on Broadening Participation for Women in Undergraduate Computing (Research). *ASEE Annual Conference and Exposition, Conference Proceedings*. https://doi.org/10.18260/1-2--36622

Beasley, Z. J., & Johnson, A. R. (2022). The Impact of Remote Pair Programming in an Upper-Level CS Course. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, *1*, 235–240. https://doi.org/10.1145/3502718.3524772

Berenson, S. B., Slaten, K. M., Williams, L., & Ho, C.-W. (2004). Voices of women in a software engineering course: Reflections on collaboration. *ACM Journal on Educational Resources in Computing*, *4*(1), 3. https://doi.org/10.1145/1060071.1060074

Bjorn, C., Haglund, P., Munz, K., & Stromback, F. (2022). It's Okay Because I Worked Really Hard! - Student Justifications for Questionable Collaboration while Solving Computer Labs. *Proceedings - Frontiers in Education Conference, FIE*, *2022-Octob*. https://doi.org/10.1109/FIE56618.2022.9962546

Bowman, N. A., Jarratt, L., Culver, K. C., & Segre, A. M. (2021). The Impact of Pair Programming on College Students' Interest, Perceptions, and Achievement in Computer Science. *ACM Transactions on Computing Education*, *21*(3). https://doi.org/10.1145/3440759

Choi, S. (2021). "Better Communication Leads to a Higher Output?" An Analysis of Pair Communication on Pair Programming Productivity. *IEEE Transactions on Professional Communication*, *64*(4), 338–353. https://doi.org/10.1109/TPC.2021.3110399

Colin, J., Hoarau, S., Declercq, C., & Broisin, J. (2024). Design and Evaluation of a Web-based Distributed Pair Programming Tool for Novice Programmers. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, *1*, 527–533. https://doi.org/10.1145/3649217.3653571

de Raadt, M., Lai, D., & Watson, R. (2007). Incorporating programming strategies explicitly into curricula. *Seventh Baltic Sea Conference on Computing Education Research (Koli Calling 2007)*.

Defranco, J. F., & Laplante, P. A. (2017). Review and analysis of software development team communication research. In *IEEE Transactions on Professional Communication*. https://doi.org/10.1109/TPC.2017.2656626

Demir, Ö., & Seferoglu, S. S. (2021a). A Comparison of Solo and Pair Programming in Terms of Flow Experience, Coding Quality, and Coding Achievement. *Journal of Educational Computing Research*, *58*(8), 1448–1466. https://doi.org/10.1177/0735633120949788

Demir, Ö., & Seferoglu, S. S. (2021b). 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

Durán Toro, A., Fernández, P., Bernárdez, B., Weinman, N., Akalın, A., & Fox, A. (2024). Exploring Gender Bias In Remote Pair Programming Among Software Engineering Students: The twincode Original Study And First External Replication. *Empirical Software Engineering*, *29*(2). https://doi.org/10.1007/s10664-023-10416-6

Endriulaitienė, A., & Cirtautienė, L. (2021). Team effectiveness in software development: The role of personality and work factors. *Business: Theory and Practice*. https://doi.org/10.3846/btp.2021.12824

Fitzgerald, S., Lewandowski, G., McCauley, R., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. *Computer Science Education*. https://doi.org/10.1080/08993400802114508

Forrester, C., Schwikert, S., Foster, J., & Corwin, L. (2022). Undergraduate R Programming Anxiety in Ecology: Persistent Gender Gaps and Coping Strategies. *CBE Life Sciences Education*. https://doi.org/10.1187/CBE.21-05-0133

Hawlitschek, A., Berndt, S., & Schulz, S. (2023). Empirical research on pair programming in higher education: a literature review. *Computer Science Education*, *33*(3), 400–428. https://doi.org/10.1080/08993408.2022.2039504

Izhikevich, K., Ong, K., & Alvarado, C. (2022). Exploring Group Dynamics in a Group-Structured Computing Undergraduate Research Experience. *ICER 2022 - Proceedings of the 2022 ACM Conference on International Computing Education Research*, *1*, 135–148. https://doi.org/10.1145/3501385.3543959

Janssen, J., & Kirschner, P. A. (2020). Applying collaborative cognitive load theory to computer-supported collaborative learning: towards a research agenda. *Educational Technology Research and Development*. https://doi.org/10.1007/s11423-019-09729-5

Kitchenham, B. (2007). Guidelines for performing systematic literature reviews in software engineering. *Technical Report, Ver. 2.3 EBSE Technical Report. EBSE*.

Küng, J., Schmid, A. M., & Brovelli, D. (2022). Gender and pair programming–Effects of the gender composition of pairs on collaboration in a robotics workshop. *Frontiers in Education*, *7*. https://doi.org/10.3389/feduc.2022.973674

Lal, S., & Mourya, R. (2022). For CS Educators, by CS Educators: An Exploratory Analysis of Issues and Recommendations for Online Teaching in Computer Science. *Societies*, *12*(4). https://doi.org/10.3390/soc12040116

Liu, X., & Woo, G. (2021). CodeHelper: A Web-Based Lightweight IDE for E-Mentoring in Online Programming Courses. *2021 3rd International Conference on Computer Communication and the Internet, ICCCI 2021*, 220–224. https://doi.org/10.1109/ICCCI51764.2021.9486772

Lytle, N., Milliken, A., Catete, V., & Barnes, T. (2020). Investigating different assignment designs to promote collaboration in block-based environments. *SIGCSE 2020 - Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 832–838. https://doi.org/10.1145/3328778.3366943

Marimuthu, M., Kumar, D., & Chhagan, M. (2020). The Difference Between Attitudinal Factors Influencing Success of Commerce and Computer Science Students at a South African University. *African Journal of Research in Mathematics, Science and Technology Education*, *24*(3), 321–332. https://doi.org/10.1080/18117295.2020.1833544

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052

McSorley, E., Gilchrist, I. D., & McCloy, R. (2019). The role of fixation disengagement in the parallel programming of sequences of saccades. *Experimental Brain Research*. https://doi.org/10.1007/s00221-019-05641-9

Mohamed, A. (2019). Designing a CS1 programming course for a mixed-ability class. *Proceedings of the 24th Western Canadian Conference on Computing Education, WCCCE 2019*. https://doi.org/10.1145/3314994.3325084

Moore, D. (2014). Supporting students in music technology higher education to learn computer programming. *Journal of Music, Technology and Education*, *7*(1), 75–92. https://doi.org/10.1386/jmte.7.1.75_1

Nagappan, N., Williams, L., Wiebe, E., Miller, C., Balik, S., Ferzli, M., & Petlick, J. (2003). Pair learning: With an eye toward future success. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2753*, 185–198. https://doi.org/10.1007/978-3-540-45122-8_21

Nickel, A., & Barnes, T. (2010). Games for CS education: Computer-supported collaborative learning and multiplayer games. *FDG 2010 - Proceedings of the 5th International Conference on the Foundations of Digital Games*, 274–276. https://doi.org/10.1145/1822348.1822391

Pantic, K., Fields, D. A., & Quirke, L. (2016). Studying situated learning in a constructionist programming camp: A multimethod microgenetic analysis of one girl's learning pathway. *Proceedings of IDC 2016 - The 15th International Conference on Interaction Design and Children*. https://doi.org/10.1145/2930674.2930725

Plonka, L., Sharp, H., & Van Der Linden, J. (2012). Disengagement in pair programming: Does it matter? *Proceedings - International Conference on Software Engineering*. https://doi.org/10.1109/ICSE.2012.6227166

Prabhakar, A. (2011). Applying pair programming for advanced Java course: A different approach. *SIGITE'11 - Proceedings of the 2011 ACM Special Interest Group for Information Technology Education Conference*, 319–320. https://doi.org/10.1145/2047594.2047682

Rong, G., Zhang, H., Xie, M., & Shao, D. (2012). Improving PSP education by pairing: An empirical study. *Proceedings - International Conference on Software Engineering*, 1245–1254. https://doi.org/10.1109/ICSE.2012.6227018

Roque-Hernández, R. V, Guerra-Moya, S. A., & Caballero-Rico, F. C. (2021). Acceptance and Assessment in Student Pair-Programming: A Case Study. *International Journal of Emerging Technologies in Learning*, *16*(9), 4–19. https://doi.org/10.3991/ijet.v16i09.18693

Scott, M. J., Mitchell, A., & Brown, D. (2023). Retention in First Stage Undergraduate Computing: Lessons Learned from a Collaborative Learning Intervention. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, *2*, 631. https://doi.org/10.1145/3587103.3594185

Sim, Y. H. R., Lua, Z. Z., Kelaver, K. K., Chua, J. Q., Lim, I. Z. J., Cao, Q., Keoh, S. L., & Lim, L. H. I. (2023). Experiences and Lessons Learned from Real-World Projects in Software Engineering Subject. *Software Engineering Education Conference, Proceedings*, *2023-Augus*, 173–177. https://doi.org/10.1109/CSEET58097.2023.00036

Slaten, K. M., Droujkova, M., Berenson, S. B., Williams, L., & Layman, L. (2005). Undergraduate student perceptions of pair programming and agile software methodologies: Verifying a model of social interaction. *Proceedings - AGILE Confernce 2005*, *2005*, 323–330. https://doi.org/10.1109/ADC.2005.48

Volume 9 Issue 55 (September 2024) PP. 770-789
DOI 10.35631/IJEPC.955052
Smrtic, M. B., & Grinstein, G. (2004). A case study in the use of Extreme Programming in an academic environment. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 3134*, 175–182. https://doi.org/10.1007/978-3-540-27777-4_18

Sobral, S. R. (2021). Pair Programming and the Level of Knowledge in the Formation of Pairs. In R. A., A. H., D. G., M. F., & C. A.M.R. (Eds.), *Advances in Intelligent Systems and Computing: Vol. 1367 AISC* (pp. 212–221). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-030-72660-7_21

Tsai, C.-W., Lin, M. Y.-C., Cheng, Y.-P., Lee, L.-Y., Chyr, W.-L., Lin, C.-H., Lin, J.-W., & Tsai, M.-C. (2023). The effects of online peer-facilitated learning and distributed pair programming on students' learning. *Computers and Education*, *203*. https://doi.org/10.1016/j.compedu.2023.104849

Valovy, M. (2023). Psychological aspects of pair programming. *ACM International Conference Proceeding Series*, 210–216. https://doi.org/10.1145/3593434.3593458

Valový, M. (2023). Effects of Pilot, Navigator, and Solo Programming Roles on Motivation: An Experimental Study. In M. J., M. M., R. A., & H.-N. V. (Eds.), *Lecture Notes in Networks and Systems: Vol. 576 LNNS* (pp. 84–98). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-031-20322-0_6

Van Der Meulen, A., & Aivaloglou, E. (2021). Who Does What? Work Division and Allocation Strategies of Computer Science Student Teams. *Proceedings - International Conference on Software Engineering*, 273–282. https://doi.org/10.1109/ICSE-SEET52601.2021.00037

Wang, C., Fang, T., & Gu, Y. (2020). Learning performance and behavioral patterns of online collaborative learning: Impact of cognitive load and affordances of different multimedia. *Computers and Education*. https://doi.org/10.1016/j.compedu.2019.103683

Weidmann, P., Gianinazzi, M., & Moccozet, L. (2023). Proposal for a Peer-to-Peer Coding Platform for Teaching Introductory Programming to Large Classes of Novice Students. In K. Z., C. F., K. T.-E., I. M., L. L., & P. M.A. (Eds.), *Lecture Notes in Networks and Systems: Vol. 769 LNNS* (pp. 163–173). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-031-42134-1_16

Werner, L. L., Denner, J., & Bean, S. (2004). Pair programming strategies for middle school girls. *Proceedings of the Seventh IASTED International Conference on Computers and Advanced Technology in Education*, 161–166. https://www.scopus.com/inward/record.uri?eid=2-s2.0-11144255756&partnerID=40&md5=ff90d8966b0f6384f6bc0fb83e342974

Xu, W., Wu, Y., & Ouyang, F. (2023). Multimodal learning analytics of collaborative patterns during pair programming in higher education. *International Journal of Educational Technology in Higher Education*, *20*(1). https://doi.org/10.1186/s41239-022-00377-z

Zhang, L., Wang, X., He, T., & Han, Z. (2022). A Data-Driven Optimized Mechanism for Improving Online Collaborative Learning: Taking Cognitive Load into Account. *International Journal of Environmental Research and Public Health*. https://doi.org/10.3390/ijerph19126984