Shaking The Silos: Impact of Sequential Live Coding on Students' Performance and Perceptions

Kwansun Cho, University of Florida

Kwansun Cho is an Instructional Assistant Professor of the Department of Engineering Education, in the UF Herbert Wertheim College of Engineering. She has been teaching introductory computer programming courses for engineers. She holds two Masters' degrees in Electrical and Computer Engineering from the University of Florida and Yonsei University, specializing in speech signal processing. Her educational research interests include improved flipped classroom teaching/learning for students, and computer- or web-assisted personalized learning.

Syeda Fizza Ali, Texas A&M University

Syeda Fizza Ali is currently pursuing her PhD in Interdisciplinary Engineering (emphasis in Engineering Education) at Texas A&M University. She works as a graduate research assistant at the Department of Multidisciplinary Engineering. Her work focuses on instructional strategies in engineering, and educational technology. She is also passionate about student mental health and broadening participation in engineering.

Sung Je Bang, Texas A&M University

Sung Je Bang is a PhD student in the Department of Multidisciplinary Engineering at Texas A&M University. He holds a Bachelor of Science and a Master of Science in Computer Engineering from the Department of Computer Science and Engineering at Texas A&M University. During his studies, Sung Je gained industry experience as a software engineering intern. Currently, he serves as a Graduate Research Assistant in the Department of Multidisciplinary Engineering. His research interests include large language models, identity theory, and engineering education.

Dr. Saira Anwar, Texas A&M University

Saira Anwar is an Assistant Professor at the Department of Multidisciplinary Engineering, Texas A and M University, College Station. She received her Ph.D. in Engineering Education from the School of Engineering Education, Purdue University, USA. The Department of Energy, National Science Foundation, and industry sponsors fund her research. Her research potential and the implication of her work are recognized through national and international awards, including the 2023 NSTA/NARST Research Worth Reading award for her publication in the Journal of Research in Science Teaching, 2023 New Faculty Fellow award by IEEE ASEE Frontiers in Education Conference, 2022 Apprentice Faculty Grant award by the ERM Division, ASEE, and 2020 outstanding researcher award by the School of Engineering Education, Purdue University. Dr. Anwar has over 20 years of teaching experience at various national and international universities, including the Texas A and M University - USA, University of Florida - USA, and Forman Christian College University - Pakistan. She also received outstanding teacher awards in 2013 and 2006. Also she received the "President of Pakistan Merit and Talent Scholarship" for her undergraduate studies.

Shaking The Silos: Impact of Sequential Live Coding on Students' Performance and Perceptions

Abstract

In today's era, computer programming is a fundamental skill required of all undergraduate students, especially those in computing and engineering disciplines. Due to the conceptually challenging nature of programming courses, efforts have been made to improve student learning outcomes, and multiple instructional mechanisms that provide hands-on experiences have been proposed. One commonly used mechanism has been dynamic live coding. Although live coding by instructors is an invaluable source of learning, it has certain disadvantages, such as passive attention and limited hands-on experience. Keeping the essence of live coding, we examine the impact of a newly introduced "Sequential Live Coding" strategy on students' performance. "Sequential Live Coding" differs from traditional live coding in four main aspects: 1) multiple students are selected for each program coding session, 2) live coding is done by the students, where they take turns to complete the program, 3) the students explain their work to the class, and 4) instructor uses the backward lecture style (the completed program is used to lecture) to highlight and expand on the key points of the program in a step by step manner. This paper examines the effectiveness of this approach, focusing on two research questions: 1) Does performance in exams differ between the students who participated in "Sequential Live Coding" and those who did not participate? and 2) What are students' perceptions regarding "Sequential Live Coding"? The data were collected from 70 students enrolled in two programming courses, i.e., Python and C++. Using convergent parallel mixed methods research design, the study presents the results after triangulating qualitative (end-of-semester questionnaire of students' perceptions) and quantitative data (students' exam scores). It provides the convergence and divergence of using such activity in two programming courses as part of a real classroom investigation.

Introduction

In today's world, technology-enabled environments are taking over. These environments are guided by automation and digitization [1]. Such environments ease the lives of humans and provide them with a mechanism to control the interaction between humans and machines. However, these digitized and automated environments depend on algorithms and programs for efficiency and accuracy. Thus, computer programming is a fundamental skill to learn for all students, especially in the undergraduate computing and engineering disciplines [2].

Computer programming has become a required and critical 21st-century skill [3] and provides many additional benefits. For example, researchers have associated learning to program with students' critical thinking and problem-solving abilities [4]. However, developing these skills through programming is not easy [5], given that introductory programming courses have a failure rate of 28% [6]. Due to the conceptually challenging nature of programming courses, static modes of instruction, such as code examples, are not ideal since they fail to convey the logical flow behind coding to the students [7]. Instead, more hands-on approaches to coding are beneficial for improved learning outcomes [8].

Several instructional mechanisms have been proposed to provide students with a more hands-on programming experience. One commonly used approach is live coding. Live coding is an instructional activity where the instructor thinks aloud as they write code in real-time in front of the students [9], [10]. Live coding facilitates students' understanding of coding and allows

them to learn debugging a good programming practice from the instructor [11]. Prior literature has found that most students in introductory programming courses view live coding positively and often prefer it over static instructional activities [12], [13]. However, depending on how it is conducted, live coding can become a passive activity for students [9]. Previous research findings report that during passive live coding, students may disengage, feel disoriented, or struggle to keep up with the instructor [14], [15].

To overcome the passive attention limitation of live coding, we propose "Sequential Live Coding" as an alternative approach that engages students at both peer and instructor levels. Four main characteristics set "Sequential Live Coding" apart from traditional live coding:

1) The students, not the instructor, lead the process, so students are the ones coding,
2) Multiple students are selected for each coding exercise,
3) The students talk out loud as they code and share their thought processes with their peers, and
4) The instructor uses a backward lecture style to review the code step-by-step, highlighting critical aspects and closing the loop. The instructor may expand on the solution when needed.

We hypothesize that such coding exercises will help the students in multiple ways: 1) As students see their peers' code, they may feel more connected to the program and approach. 2) Due to peer coding, they may find the pace more suited to their understanding, and 3) As the instructor will provide systematic feedback, the students have multiple chances to clarify any confusion. We posit that these factors may contribute to improved student performance.

Considering the potential benefits, we investigate the effectiveness of "Sequential Live Coding" in introductory programming courses in this paper. More specifically, we examine the impact of the activity on students' performance and report students' perceptions of this instructional activity. The following research questions guide the research

1) Does exam performance differ between the students who participated in "Sequential Live Coding" and those who did not participate?
2) What are students' perceptions regarding "Sequential Live Coding"?

Literature Review

While computer programming is considered a fundamental skill in response to the increased need for computing and IT professionals by various national standards (e.g., Bureau of Labor Statistics [16], NASEM [17]), the reported attrition rates are also high [18], [19]. One potential reason for such attrition could be that programming coursework is difficult [20]. The other reason could be associated with how programming is taught to students [21]. To remedy the situation, researchers have examined various approaches to introduce programming to undergraduate students, such as providing hands-on experiences [8], working in groups or with a peer (pair programming [22]), and live coding [9].

Researchers have examined how the introduced approaches improved students' cognitive and non-cognitive aspects. For example, Wu and colleagues [23] found that engaging students in hands-on practices in contrast to passive approaches for programming courses improved students' learning experience, lessened the stress they experienced during programming, and increased their interest in coding for the future. Similarly, Canfield and colleagues [24] used microcontroller units to teach programming units. The authors noted that lessons utilizing

hands-on activities to teach programming allowed students to become more engaged with their work and actively exposed to programming concepts that they will learn in more detail later in their careers.

Additionally, researchers have explained the importance of group work in programming. For example, Williams and Kessler [22] found that students working in groups experience an effective way to learn programming because of the ability to get immediate peer feedback learning directly from each other's coding. In addition, if students are asked to explain their work to the rest of the class, it could further enhance their abilities and spark inquiries among other students, which can help them more effectively build a conceptual understanding of concepts being taught in class [25].

Besides the added benefits of many approaches, in some cases, the strategies introduced to improve student outcomes have mixed results. Researchers have found that live coding has benefits as well as drawbacks [9]. From a positive perspective, live coding has been shown to help undergraduate students improve their debugging skills [26], [27], [11] and learn programming concepts more effectively [28], [29]. For example, Rubin [30] noted that live coding is an effective teaching approach due to students preferring it over code examples shown in PowerPoint slides, having a better learning experience with live coding than with static coding, and being able to see instructors unintentionally create bugs in their live coding, which helps them understand better ways to code themselves. This type of activity can also cause the listening students to be attentive to possible design choice errors within the presenting students' works [31].

In terms of disadvantages, live coding is described as too time-consuming in class [32], and students have little to no time to take notes [33]. Another aspect to consider is that different lecturers execute live coding differently, causing the practice to not be active learning in some cases [9]. For instance, some lecturers consider live coding exercises as having students solve in-class coding exercises and lecturers offer direct feedback when asked [34]. This practice cannot be considered an active learning example because it lacks live demonstrations or student feedback [9]. Despite its strengths, the weaknesses of the current live coding strategy indicate the need to update and improve.

Considering the strategies' advantages, we propose a novel approach, "Sequential Live Coding," based on what worked in existing studies. The first two principles of the approach are based on giving students autonomy to work and emphasizing the feedback process [35]. Also, this approach addresses both student and instructor-level aspects, where the instructor uses a backward lecture style. Research indicates that such integration is helpful for students in computer science as it promotes active engagement in two ways [36]. One, it makes students actively participate in lectures as they teach programming concepts to the class. Second, this type of learning helps teach how computer science concepts can be practically applied to assignments and any tasks related to the professional field [36].

Research Design and Methods

The study employs a convergent parallel mixed methods research design to answer the research questions. The data were collected and analyzed for both quantitative and qualitative aspects. We chose the mixed methods research design to help understand students' perceptions based on their frequency of participation.

Site and Participants

We collected data from 70 undergraduate engineering students enrolled in one of two elective programming courses for non-computer science (CS) majors taught by the same instructor and offered at the University of Florida, a large R1 university in Fall 2023. Table 1 provides information about the demographics of the participants. In these 15-week introductory courses, the common topics covered in class are variables, inputs, outputs, data types, flow of control, functions, types of data structures, and basic object-oriented programming principles.

Table 1. Student demographics

	Number of students	Percentage
Gender		
Male	51	72.9
Female	19	27.1
Race/Ethnicity		
Hispanic or Latino, or Spanish Origin of any race	19	27.1
Asian	11	15.7
Black or African American	2	2.9
White	34	48.6
Two or more races	3	4.3
Prefer not to disclose	1	1.4
Academic Year		
Freshman	13	18.6
Sophomore	24	34.3
Junior	12	17.1
Senior	17	24.3
Other	4	5.7
Programming Experience		
Some Experience	44	62.9
No Experience	26	37.1

Course Design for “Sequential Live Coding”

Two introductory programming courses (C++ and Python) for non-CS major engineering students used a flipped classroom design. Each class was divided into three phases (before, during, and after class). Each student was expected to be familiar with the main concepts of the weekly class by reviewing and understanding the provided study material before class.

Students were expected to be prepared to practice coding assignments during class. An in-class live coding assignment would be unlocked at the start of the class. Students were given about five minutes at the beginning of the class to individually design a high-level flow of solving the assigned problem as a pseudocode. Next, “Sequential Live Coding” would be initiated by the students. Several students (usually five or six) would voluntarily do “Sequential Live Coding” while the instructor was present. With shared screens, students would see each step of the way to complete the assigned coding problem together. For example, the first student would set the stage for the live coding by starting a supported IDE (integrated development environment), such as Visual Studio, and creating a source code with initial comments that included a short description of a program, inputs, outputs of the program, skeleton code, and pseudocode (a highlighted flow of the program in plain English

as comments). The next student would start to convert the pseudocode written by the first student into a programming language (C++ or Python). During the process, students also answered other students' questions. If a student doing live coding made mistakes, the rest of the class would help debug the code and identify the mistakes together. While students were doing a live coding assignment, the instructor would take note of the common mistakes students made. After running the program successfully, the instructor would conduct a mini-backward lecture with the completed code done by students to reinforce some important concepts for the week.

Data Collection and Measures

We collected data on 1) students' perceptions and 2) performance. The students' perceptions were collected in an end-of-semester survey questionnaire with open-ended questions designed to understand the participants' experiences with the newly introduced "Sequential Live Coding." In this study, we present the results of two questions: 1) What are your perceptions about your and your peers' engagement during the "Sequential Live Coding" sessions? 2) How did the "Sequential Live Coding" sessions influence your learning? Additionally, we used students' total scores in the course to measure their performance.

Data Analysis

In this convergent parallel mixed methods study, the data were analyzed using the statistical software SPSS V 29.0, and Microsoft Excel was used for basic frequency calculations.

To answer the first research question, we examined the mean difference between students who participated in "Sequential Live Coding" throughout the semester and those who did not. We first determined the equality of variances using Levene's test and found both variances to be equal. We also verified the data normality with skewness, kurtosis, and box plots. Secondly, we divided the data into three groups based on the frequency of participation where students who participated in one or two live coding sessions were placed in the infrequent participation groups, and students with greater or equal to three participations were placed in frequent participation groups. Due to sample size limitation, we used the Kruskal-Wallis H test to examine whether all three samples had equal or varying means.

To answer the second research question, we initially used qualitative content analysis to determine the general perception of students on engagement and learning after being engaged in "Sequential Live Coding." We also used in vivo coding to understand students' perceptions and identify the key insights using students' direct quotes.

Further, we triangulated the results of students' perceptions of engagement with their frequency of participation and perceptions about learning with the frequency of participation and the results of research question 1.

Results

Differences in Students' Performance

To answer the first research question and identify the difference between students' performance based on who participated in "Sequential Live Coding," we used an independent samples t-test. The results of the t-test are presented in Table 2.

Table 2. Student performance difference based on students' participation in "Sequential Live Coding"

	Participated N=42		Not participated N=28					
	M	SD	M	SD	df	*t*	*p*	Cohen's d
Total Score	95.01	5.78	89.83	6.79	68	3.424	.001**	.835

**p<0.05, **p<0.01*

The results indicate a significant performance difference between the students who participated in the "Sequential Live Coding" compared to those who did not participate. With $t(68) = 3.424$, $p = .001$. Cohen's d= 0.835 shows a large effect size [37]. These results indicate that students who were engaged in the "Sequential Live Coding" performed better than the students who did not.

We additionally examined the differences between the frequencies of participation. We divided students into three groups, i.e., "No Participation," "Infrequent Participation," and "Frequent Participation" groups. The students who did not participate in a "Sequential Live Coding" session were placed in the "No Participation" group. The students who participated in a "Sequential Live Coding" session only once or twice were placed in the "Infrequent Participation" group, and students who participated in more than three "Sequential Live Coding" sessions were placed in the "Frequent Participation" group. Due to sample size constraints, we conducted the Kruskal-Wallis H test. We assessed the significant difference between total scores and the frequency of students' participation in the live coding session. Table 3 presents the results.

Table 3. Student performance difference based on the frequency of participation

	No Participation N=28		Infrequent Participation N=24		Frequent Participation N=18				
	M	SD	M	SD	M	SD	Df	*H*	*p*
Total Score	89.83	6.79	93.81	6.01	96.61	5.20	2	15.507	<.001 **

**p<0.05, **p<0.01*

The results indicate a significant difference in students' performance based on their frequency of participation in a "Sequential Live Coding" session.

Students' Perceptions and Triangulation

To answer research question 2, we first examined students' perceptions and later triangulated the results with their frequency of participation. The students' perceptions of the "Sequential Live Coding" experience included their viewpoints on engagement and influence on learning.

Regarding engagement, most students found "Sequential Live Coding" a constructive, helpful, and engaging experience. Approximately 74% of students described classroom engagement as high and very good, while approximately 26% found it to be low, where some students participated more than others. However, no students described the activity as not engaging. For example, a student explained "Sequential Live Coding" as engaging and stated:

"*I think almost all of my peers were attentive, engaged, and focused on creating functional solutions to live coding problems. Me personally, I was engaged almost every time as well,*

consistently saving my live coding files and actively participating during any live coding demonstration."

Notably, students who found the activity less engaging attributed the experience to hard concepts, reluctance due to performance fear, or lack of programming experience. For example, a student stated:

"It seemed for the most part that my peers and I both shared a reluctance to participate in the live coding. I suspect this has to do with a lack of familiarity with the material for the given [in-class assignment]. I only felt comfortable participating in the [in-class assignment] when I was confident in my knowledge of the material."

We also triangulated these results with students' participation. For this purpose, we examined how the perceptions of engagement varied based on students' participation levels. Table 4 summarizes the triangulation on engagement and participation.

Table 4. Triangulation of participation frequency, perceptions on engagement, and its interpretation

Frequency of Participation	Perceptions about Engagement– Direct Quotes	Interpretation
No participation	*"The live coding in class helped me see examples of the new material being used in a wide range of different types of applications."* *"Some students were very eager to participate in live coding, which appeared to be those who may have had some experience with Python. These students were very engaged. Myself and others, however, seemed nervous to go up and live code in front of everyone but we were also engaged."*	75% of the 28 students who did not participate in the activity found "Sequential Live Coding" as an engaging experience. Such students attributed their lack of participation to their nervousness and fear of performance. However, those who found it low engaging (25%) indicated it was a nerve-wracking experience.
Infrequent participation	*"I enjoyed this learning concept greatly as I could see various ways to solve the same problem."* *"I feel that people felt reluctant to go up and perform, most likely because they are scared of doing the wrong thing or aren't sure what to do. I personally had one experience where I wasn't sure what to do during the live-coding, but I still went up to get some guidance."*	83% of the 24 students who infrequently participated found "Sequential Live Coding" helpful as it could help them see how others approach the problem. The students who found it less helpful (17%) attributed the lack of engagement to reluctance and performance fear
Frequent Participation	*"I think the live coding is works well for increasing the engagement with the material. It helps to address questions you may have after reading the textbook, as*	67% of the 18 frequently participated students found that "Sequential Live Coding" supported reinforcement and helped address questions about

	well as reinforces everything you learned about." *"I tried to engage with the live coding when I felt confident, and I noticed that others often did not, with some barely ever participating, perhaps because they are shy or just do not know the material."*	the material. The students who perceived lower engagement, although they frequently participated (33%), attributed it to students' lack of understanding or personality traits.

The results indicate that most students in each participation group felt engaged with the "Sequential Live Coding" process. However, the students with no and infrequent participation attributed less engagement to reluctance, nervousness to code in front of others, and performance fears. The students who frequently participated attributed it to a lack of preparation and knowledge.

Regarding students' perceptions about the influence of "Sequential Live Coding" on their learning experience, almost 80% of the students found it influential and helpful to learning while 20% felt that the activity did not do much towards their learning experience. Also, when triangulated with the frequency of most students in all participation groups, it indicated a high influence on learning (please see Table 5).

Table 5. Co-examination of students' perceptions of learning and participation frequency

	No Participation N=28	Infrequent participation N=24	Frequent participation N=18
High influence on learning	85.7%	79.2%	83.3%
Low influence on learning	14.3%	20.8%	16.7%

The students who found the "Sequential Live Coding" activity influential for learning praised that activity's ability to provide hands-on experience, variation in solutions, and availability of examples. For example, a student stated:

"I think watching other people work through problems and by having my own ideas that were different from other peoples and watching how the code differed was really interesting, and it really helped my process later on when working on the somewhat harder assignments."
The students who found the sequential approach less influential suggested that this approach was similar to any other approach (e.g., teacher doing coding). For example, a student stated:

"Just a little. It was helpful to see how tasks are solved and to be encouraged to think along. However, if the tasks had been solved by the professor, I would probably have learned just as much."

These results diverge from results of research question 1, which indicated that students who participated more performed better than those who did not participate or participated infrequently. Specifically, there was a higher performance mean difference between students who participated frequently and those who did not.

Discussion and Conclusion

This study discusses the design, introduction, and role of a novel approach to teaching programming to non-CS major students: “Sequential Live Coding.” As an instructional activity, “Sequential Live Coding” engaged students in a live coding process, where students were given autonomy to code and share their thought processes with their peers. Additionally, the instructor used a backward approach to highlight the key aspects of the course after students successfully implemented the solution. To examine the impact of this approach, this study answered two research questions; the first examined the impact of this approach on students' performance, and the second provided insights on students' perceptions of the approach from two aspects, i.e., engagement and learning. Considering the mixed methods approach, the results also present the triangulation for interpretation; students' perceptions of engagement were examined with the frequency of participation, and perceptions about learning were triangulated with the frequency of participation and impact on performance.

The results confirm the study’s hypothesis that students who participated, or participated more frequently showed better performance than those who did not participate. These results align with existing research that emphasizes the importance of preparation [38] or indicates that giving students autonomy [35], or engaging them in hands-on activities promotes learning and performance [23]. The results of the second research question indicate that most students found the activity engaging for them and their peers. However, students also noted that they face peer and performance pressure, which can negatively impact their engagement. Also, some students indicated that lack of prior programming experience reflected inversely on their engagement, which aligns with existing literature on the importance of prior programming experience [39].

The triangulation of results highlights some important insights. First, the students, regardless of their participation group, highlighted similar factors that could be the reason for lower or higher engagement, such as personal learning style, prior experience, fear of performance, shyness, or nervousness. Aligned to existing research (e.g., [40]), these traits indicated the importance of students' emotions, prior experience, and learning styles while examining programming learning. Further, similar to existing literature, these results also highlight the importance of use of different instructional strategies to improve students’ engagement and reduce distraction [15]. Second, regardless of the course performance or participation group, most students found the activity helpful. These results could be due to any of the three hypothesized reasons for introducing this approach where students felt connected as their peers were coding, found the pace more appropriate, or found the instructor feedback helpful for reinforcing concepts. However, the conclusive reasons require further investigation.

The study's results must be viewed in the light of some limitations and future directions. First, the study was based on one measure of students' performance, and future studies could consider other real-time measures of performance or impact on other programming skills, such as debugging. Second, the study has a smaller sample size of 70 students, with one semester of data, two courses, and one instructor. Future studies can expand on a larger sample size for more conclusive results. Third, the study's question 1 followed a cross-sectional analysis. The absence of randomized control could affect the generalizability of the claims, which could be rectified in future studies. Fourth, the data on students' perceptions uses only student-reported evidence on engagement. Future studies could rely on other process data and supplementary information, such as using classroom observations as a secondary source of data, which may help to see researchers' perspectives on students’ engagement and learning [41]. Fifth, this study did not consider students' demographic

information for understanding the variations. Future studies could account for such variables. Sixth, the students' participation in the live coding was voluntary, resulting in some students who never participated. Future studies could utilize models proposed in existing research (e.g. [42]) to encourage participation from all students.

These results provide insight into how students perceived the novel instructional activity and how it influenced their performance. In particular, the factors revealed in our findings related to students' reluctance to live code present opportunities for further investigation of the approach and its effectiveness.

References

[1] J. Wajcman, "Automation: Is it really different this time?" *Br. J. Sociol*., vol. 68, no. 1, pp. 119-127, 2017. doi: 10.1111/1468-4446.12239.

[2] T. Koulouri, S. Lauria, and R. D. Macredie, "Teaching introductory programming: A quantitative evaluation of different approaches," *ACM Trans. on Comput. Educ. (TOCE)*, vol. 14, no. 4, pp. 1-28, 2014. doi: 10.1145/2662412.

[3] J. A. Rios, G. Ling, R. Pugh, D. Becker, and A. Bacall, "Identifying critical 21st-century skills for workplace success: A content analysis of job advertisements," *Educ. Res.,* vol. 49, no. 2, pp. 80-89, 2020. doi: 10.3102/0013189x19890600.

[4] Y. C. Liao and G. W. Bright, "Effects of computer programming on cognitive outcomes: A meta-analysis," *J. Educ. Comp. Res*., vol. 7, no. 3, pp. 251-268, 1991. doi: 10.2190/e53g-hh8k-ajrr-k69m.

[5] S. Anwar, "Role of different instructional strategies on engineering students' academic performance and motivational constructs" Ph.D. dissertation, Purdue Univ., West Lafayette, IN, USA, 2020.

[6] J. Bennedsen and M. E. Caspersen, "Failure rates in introductory programming: 12 years later," *ACM Inroads*, vol. 10, no. 2, pp. 30-36, 2019. doi: 10.1145/3324888.

[7] C. S. Cheah, "Factors contributing to the difficulties in teaching and learning of computer programming: A literature review," *Contemp. Educ. Technol*., vol. 12, no. 2, 2020, Art no. ep272. doi: 10.30935/cedtech/8247.

[8] K. Von Hausswolff, A. Eckerdal, and M. Thuné, "Learning to program hands-on: A controlled study," in *Proc. 20th Koli Calling Int. Conf. on Comput. Educ. Res*., 2020, pp. 1-10. doi: 10.1145/3428029.3428058.

[9] A. Selvaraj, E. Zhang, L. Porter, and A. G. Soosai Raj, "Live coding: A review of the literature," in *Proc. 26th ACM Conf. on Innov. and Technol. in Comput. Sci. Educ*., vol. 1, 2021, pp. 164-170. doi: 10.1145/3430665.3456382.

[10] K. Cho, D. Bang, S. Anwar, "A blended approach to design an introductory programming course for non-CS majors - Students' feedback," *in 2023 ASEE Annu. Conf. & Expo.,* 2023. doi: 10.18260/1-2--42349.

[11] A. G. S. Raj, J. M. Patel, R. Halverson, and E. R. Halverson, "Role of live-coding in learning introductory programming," in *Proc. 18th Koli Calling Int. Conf. Comput. Educ. Res.,* 2018, pp. 1-8. doi: 10.1145/3279720.3279725.

[12] J. Paxton, "Live programming as a lecture technique," *J. Comput. Sci. Coll*., vol. 18, no. 2, pp. 51-56, 2002.

[13] A. Gaspar and S. Langevin, "Restoring "coding with intention" in introductory programming courses," in *Proc. 8th ACM SIGITE Conf. on Inf. Technol. Educ*., 2007, pp. 91-98. doi: 10.1145/1324302.1324323.

[14] T.-M. Grønli and S. Fagernes, "The live programming lecturing technique: A study of the student experience in introductory and advanced programming courses," in *Norsk IKT-konferanse for forskning og utdanning*, no. 4, 2020.
[15] D. Bang, S. Anwar, S. F. Ali, and A. Magana, "Relationship between instructional activities and students' distraction," in *2023 ASEE Gulf-Southwest Annu. Conf.,* 2023.
[16] U.S. Bureau of Labor Statistics (2023, Sep.) *Computer and Information Technology Occupations: Occupational Outlook Handbook*. Available: https://www.bls.gov/ooh/computer-and-information-technology/home.htm
[17] National Academies of Sciences, Engineering, and Medicine (NASEM), "Cultivating interest and competencies in computing: Authentic experiences and design factors," The Nat. Acad. Press, 2021. https://nap.nationalacademies.org/catalog/25912/cultivating-interest-and-competencies-in-computing-authentic-experiences-and-design
[18] J. Bennedsen and M. E. Caspersen, "Failure rates in introductory programming," *SIGCSE Bull.*, vol. 39, no. 2, pp. 32–36, Jun. 2007, doi: 10.1145/1272848.1272879.
[19] C. Watson and F. W. B. Li, "Failure rates in introductory programming revisited," in *Proc. 2014 Conf. Innov. and Technol. Comp. Sci. Educ. (ITiCSE '14)*, 2014, pp. 39–44. doi: 10.1145/2591708.2591749.
[20] T. W. Goodman, "Identifying students who may experience difficulty in an introductory computer science course," in *Proc. 28th Annu. Southeast Regional Conf. (ACM-SE 28)*, 1990, p. 182. doi: 10.1145/98949.98988.
[21] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion," *Comp. Sci. Educ.*, vol. 13, no. 2, pp. 137–172, Jun. 2003, doi: 10.1076/csed.13.2.137.14200.
[22] L. A. Williams and R. R. Kessler, "Experiments with industry's 'pair-programming' model in the computer science classroom," *Comp. Sci. Educ.*, vol. 11, no. 1, pp. 7–20, Jan. 2001, doi: 10.1076/csed.11.1.7.3846.
[23] H.-T. Wu, P.-C. Hsu, C.-Y. Lee, H.-J. Wang, and C.-K. Sun, "The impact of supplementary hands-on practice on learning in introductory computer science course for freshmen," *Comput. Educ.*, vol. 70, pp. 1–8, Jan. 2014, doi: 10.1016/j.compedu.2013.08.002.
[24] S. L. Canfield and M. Abdelrahman, "Enhancing the programming experience for first-year engineering students through hands-on integrated computer experiences," *J. STEM Educ.*, vol. 13, no. 4, Jun. 2012.
[25] C. Rasmussen and O. N. Kwon, "An inquiry-oriented approach to undergraduate mathematics," *J. Math. Behav.*, vol. 26, no. 3, pp. 189–194, Jan. 2007, doi: 10.1016/j.jmathb.2007.10.001.
[26] J. Bennedsen and M. E. Caspersen, "Revealing the programming process," *ACM SIGCSE Bull.*, vol. 37, no. 1, pp. 186–190, Feb. 2005, doi: 10.1145/1047124.1047413.
[27] N. R. Boyer, S. Langevin, and A. Gaspar, "Self direction and constructivism in programming education," in *Proc. 9th ACM SIGITE Conf. on Inf. Technol. Educ.*, pp. 89–94, Oct. 2008, doi: 10.1145/1414558.1414585.
[28] N. Giacaman, "Teaching by example: Using analogies and live coding demonstrations to teach parallel computing concepts to undergraduate students," in *2012 IEEE 26th Int. Parallel and Distrib. Process. Symp. Workshops & PhD Forum*, pp. 1295–1298, May 2012, doi: 10.1109/ipdpsw.2012.158.
[29] L. Johnston, M. Bonsma-Fisher, J. Ostblom, A. Hasan, J. Santangelo, L. Coome, L. Tran, E. de Andrade, and S. Mahallati, "A graduate student-led participatory live-coding quantitative methods course in R: Experiences on initiating, developing, and teaching," *J. Open Source Educ.*, vol. 2, no. 16, p. 49, Jun. 2019, doi: 10.21105/jose.00049.

[30] M. J. Rubin, “The effectiveness of live-coding to teach introductory programming,” in *Proc. 44th ACM Tech. Symp. Comp. Sci. Educ.*, Mar. 2013, pp. 651–656. doi: 10.1145/2445196.2445388.
[31] C. H. Crouch and E. Mazur, "Peer Instruction: Ten years of experience and results," *Am. J. Phys.*, vol. 69, no. 9, pp. 970–977, Sep. 2001, doi: 10.1119/1.1374249.
[32] R. E. Bruhn and P. J. Burton, "An approach to teaching Java using computers," *ACM SIGCSE Bull.*, vol. 35, no. 4, pp. 94–99, Dec. 2003, doi: 10.1145/960492.960537.
[33] B. Stephenson, "Coding demonstration videos for CS1," in *Proc. 50th ACM Tech. Symp. Comput. Sci. Educ.*, Feb. 2019, pp. 105–111. doi: 10.1145/3287324.3287445.
[34] A. Gaspar and S. Langevin, "Active learning in introductory programming courses through student-led 'live coding' and test-driven pair programming", in *Int. Conf. Educ. and Inf. Sys., Tech. and Appl*, 2007.
[35] D. T. Fokum, D. N. Coore, and C. Busby-Earle, “Learner autonomy as a means to improve pass rates among first-year computing students.” *The UWI Quality Education Forum*, 2016.
[36] S. Grissom, R. Mccauley, and L. Murphy, "How student centered is the computer science classroom? A survey of college faculty," *ACM Trans. Comput. Educ.*, vol. 18, no. 1, pp. 1–27, Nov. 2017, doi: 10.1145/3143200.
[37] J. Cohen, *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum Associates, 1988. doi: 10.4324/9780203771587.
[38] K. Cho and S. Anwar, "Work-in-progress: relationship of students' class preparation and learning in a flipped computer programming course," in *2022 ASEE Annu. Conf. & Expo*., 2022. doi: 10.18260/1-2--41246
[39] C. Wilcox and A. Lionelle, “Quantifying the benefits of prior programming experience in an introductory computer science course.” in *Proc. 49th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE '18), 2018,* pp. 80–85, doi: 10.1145/3159450.3159480 2018.
[40] S. F. Ali, D. Bang, A. J. Magana, and S. Anwar, "Impact of instructional activities on students' positivity, participation, and perceived value in a systems analysis and design course," *2023 IEEE Frontiers in Education Conference (FIE)*, College Station, TX, USA, 2023, pp. 1-7, doi: 10.1109/FIE58773.2023.10343012.
[41] S. Anwar and M. Menekse, "A systematic review of observation protocols used in postsecondary STEM classrooms," *Rev. Educ.,* vol. 9, no. 1, pp. 81-120, 2021. doi: 10.1002/rev3.3235.
[42] S. R. Jayasekaran, U. Farooq, and S. Anwar, “Impact of extra credit for practice questions on programming students’ participation and performance.” in *2023 ASEE Annu. Conf. & Expo.,* 2023. doi: 10.18260/1-2--43888