Paper ID #45197

Gendered Impacts of Code Critiquers on Self-Efficacy in First-Year Engineering Students.

Mary Benjamin, Michigan Technological University

PhD Student in the Dept. of Civil, Environmental, & Geospatial Engineering at Michigan Technological university.

Laura Albrant, Michigan Technological University

After completing a bachelor's degree in computer science, Laura Albrant decided to challenge how she viewed software development, by switching departments. Currently working towards a master's degree in human factors at Michigan Technological University, Laura pursues interests on both sides of the fence through education research.

Dr. Michelle E Jarvie-Eggart P.E., Michigan Technological University

Dr. Jarvie-Eggart is a registered professional engineer with over a decade of experience as an environmental engineer. She is an Assistant Professor of Engineering Fundamentals at Michigan Technological University. Her research interests include technology adoption, problem based and service learning, and sustainability.

Dr. Leo C. Ureel II, Michigan Technological University

Leo C. Ureel II is an Assistant Professor in Computer Science and in Cognitive and Learning Sciences at Michigan Technological University. He has worked extensively in the field of educational software development. His research interests include intelligent learning environments, computer science education, and Artificial Intelligence

Dr. Laura E Brown, Michigan Technological University
Dr. Jon Sticklen, Michigan Technological University

Jon Sticklen is an Associate Professor with the Engineering Fundamentals Department (EF) and Affiliated Faculty with the Department of Cognitive and Learning Sciences (CLS). He served as Chair of EF from 2014-2020, leading a successful effort to design a

Dr. AJ Hamlin, Michigan Technological University

AJ Hamlin is a Principle Lecturer in the Department of Engineering Fundamentals at Michigan Technological University, where she teaches first-year engineering courses. Her research interests include engineering ethics, spatial visualization, and educatio

Gendered Impacts of Code Critiquers on Self-Efficacy in First-Year Engineering Students

Introduction

The demand for skilled programmers in industry continues to grow. However, research has shown that women often face challenges in developing programming skills, primarily due to lower levels of programming self-efficacy [1], [2]. This discrepancy has contributed to persistent gender gaps in technology-related fields. There remains a critical research gap regarding the differential impact of educational interventions on programming self-efficacy across genders, particularly in engineering. Addressing this gap is essential to promote gender equity in programming and to support the success of all students in the field.

Self-efficacy refers to an individual's belief in their ability to succeed in specific tasks or challenges [5]. This belief can be both general and domain-specific, meaning that a person might feel confident in one area, such as math, but not in another, such as programming. Domain-specific self-efficacy, in this context, refers to the belief in one's ability to successfully perform programming tasks [6]. Developing programming self-efficacy is particularly important for first-year engineering students as it correlates with deeper engagement in programming tasks, greater persistence through challenges, and better academic performance [6], [7].

Conversely, low programming self-efficacy can lead to disengagement and attrition from engineering programs [7], [9]. This issue is particularly critical for female students, who often start with lower programming self-efficacy compared to their male peers, affecting their engagement and persistence in the field [10], [11]. Addressing this gap is essential for promoting gender equity in engineering education [12].

However, there is a notable research gap regarding the differential impact of educational interventions on programming self-efficacy by gender. Existing studies rarely disaggregate data by gender, missing the opportunity to tailor interventions to diverse needs [8], [13].

The primary goal of this study is to evaluate the effectiveness of the WebTA intervention in improving programming self-efficacy among first-year engineering students. In particular, the study aims to investigate whether the use of WebTA can help address known gender disparities in programming self-efficacy, with a focus on enhancing female students' confidence and skills in programming.

To achieve these objectives, the study is guided by the following research questions:

1. How does the WebTA intervention affect programming self-efficacy in first-year engineering students?
2. Are there gender-specific impacts of the WebTA intervention on programming self-efficacy, particularly for female students?

By answering these questions, the study seeks to provide insights into the potential of immediate feedback tools like WebTA to improve programming outcomes and contribute to reducing gender disparities in the field.

Literature Review

Self-efficacy, a concept central to Bandura's social cognitive theory, significantly influences how individuals think, behave, and feel. In education, self-efficacy predicts students' motivation, learning, and academic achievement [5]. Specifically, in programming, self-efficacy refers to a student's confidence in their ability to perform coding tasks. High programming self-efficacy is linked to greater engagement with challenging programming tasks and better performance in programming courses [14]. Conversely, low programming self-efficacy can lead to avoidance behaviors, poorer outcomes, and reduced interest in pursuing further studies in computer science [8]. In engineering education, engineering self-efficacy (ESE) is crucial for student persistence and success. ESE is shaped by mastery experiences, vicarious learning, and supportive feedback, all of which can help students build confidence in their engineering abilities [16]. Similarly, programming self-efficacy (PSE) is domain-specific to coding tasks, and high PSE is linked to greater engagement and improved performance in programming courses [14]. Conversely, low PSE can lead to avoidance behaviors, poor academic outcomes, and reduced interest in pursuing further studies in computer science [7]. By fostering positive experiences and providing constructive feedback, educators can enhance both engineering and programming self-efficacy, leading to improved learning outcomes and persistence in the field.

Gender Differences in Programming Self-Efficacy

In this paper, we use the terms "male" and "female" to refer to gender, although these terms are typically used for sex. This choice was made because our survey asked participants to self-identify as male, female, or non-binary.

Research consistently shows that female students often report lower programming self-efficacy than their male peers, despite achieving similar academic outcomes [8], [10]. This disparity is influenced by societal stereotypes portraying computing as male-dominated, leading to stereotype threat and further reducing self-efficacy [17], [18]. Female students also tend to have less prior exposure to programming, which contributes to lower confidence levels [7]. Classroom environments that emphasize competition may exacerbate these differences, whereas collaborative and supportive settings can help close the self-efficacy gap by building confidence through peer learning [11]. Addressing these disparities is essential for promoting gender equity in programming.

Immediate Feedback and Self-Efficacy
Immediate feedback is crucial for enhancing self-efficacy by providing learners with timely information that helps them achieve mastery experiences—the most powerful source of self-efficacy [5]. In programming, immediate feedback helps students quickly identify and correct mistakes, preventing the accumulation of misconceptions and promoting a more accurate understanding of coding concepts [22]. Research shows that immediate feedback significantly enhances learning outcomes by reinforcing correct responses and quickly rectifying errors, leading to increased confidence and better performance [23]. In programming education, immediate feedback reduces anxiety and promotes self-regulation, which are critical for building self-efficacy [24].

Studies on Code Critiquers and Immediate Feedback Mechanisms
Code critiquers and immediate feedback tools are effective in enhancing programming education by providing real-time evaluations of students' code [25]. found that students receiving immediate feedback through an automated code critiquer showed higher engagement and learning gains. Similarly, [26] reported that automated feedback tools helped students identify and correct errors early, reducing frustration and deepening their understanding of programming concepts.

Leinonen and Vihavainen [27] demonstrated the positive effects of AI-driven automated feedback systems on students' self-efficacy in large-scale programming courses. The work by [28] highlighted the role of formative feedback in online coding platforms, particularly in maintaining engagement and retention during remote learning caused by the COVID-19 pandemic.

A study by [29] emphasized that code critiquers tailored to novice programmers can significantly boost programming self-efficacy, which is critical to student success in engineering education. This aligns with findings from the RICA project, which focused on immediate feedback regarding antipatterns in student code, reinforcing the role of feedback in improving students' mental models and coding practices [30].

Rivers and Koedinger [32] emphasized that immediate feedback fosters self-regulation, leading to better learning outcomes and higher self-efficacy. These studies suggest that immediate feedback mechanisms are critical for improving both the technical skills and confidence of programming students.

WebTA and Programming Self-Efficacy

WebTA, a code critiquing tool that provides immediate, formative feedback on programming tasks, shows promise in reducing the self-efficacy gap between genders in programming. By offering personalized, immediate feedback in a low-pressure setting, WebTA can help female students build confidence in their programming abilities, which is often lower than that of their male peers [30]. The tool's private, tailored feedback allows students to learn from mistakes without the fear of public scrutiny, supporting incremental learning and mastery experiences. WebTA's ability to reduce anxiety and support gradual skill development can enhance programming self-efficacy, particularly for those who initially struggle with coding tasks.

Methodology

WebTA is a web-based tool designed to improve programming self-efficacy by providing immediate feedback on students' coding practices. WebTA, a code critiquing tool that provides immediate, formative feedback on programming tasks, was initially developed for use with Java programming. The research team was directly involved in transitioning WebTA from Java to MATLAB, aligning with the course's focus on MATLAB programming. The rationale for selecting WebTA over other tools lies in its proven ability to provide real-time feedback and its proven effectiveness in Java-based environments motivated its selection for this study. This study represents the first comprehensive evaluation of WebTA's effectiveness in a MATLAB programming environment.

It identifies common coding mistakes (antipatterns) and reinforces effective coding techniques (good patterns). Using a traffic light system, WebTA categorizes feedback as:

- Green Light: Good practices that should be continued.
- Yellow Light: Potential issues that could lead to errors if not addressed.
- Red Light: Critical mistakes that must be corrected for the code to run properly.

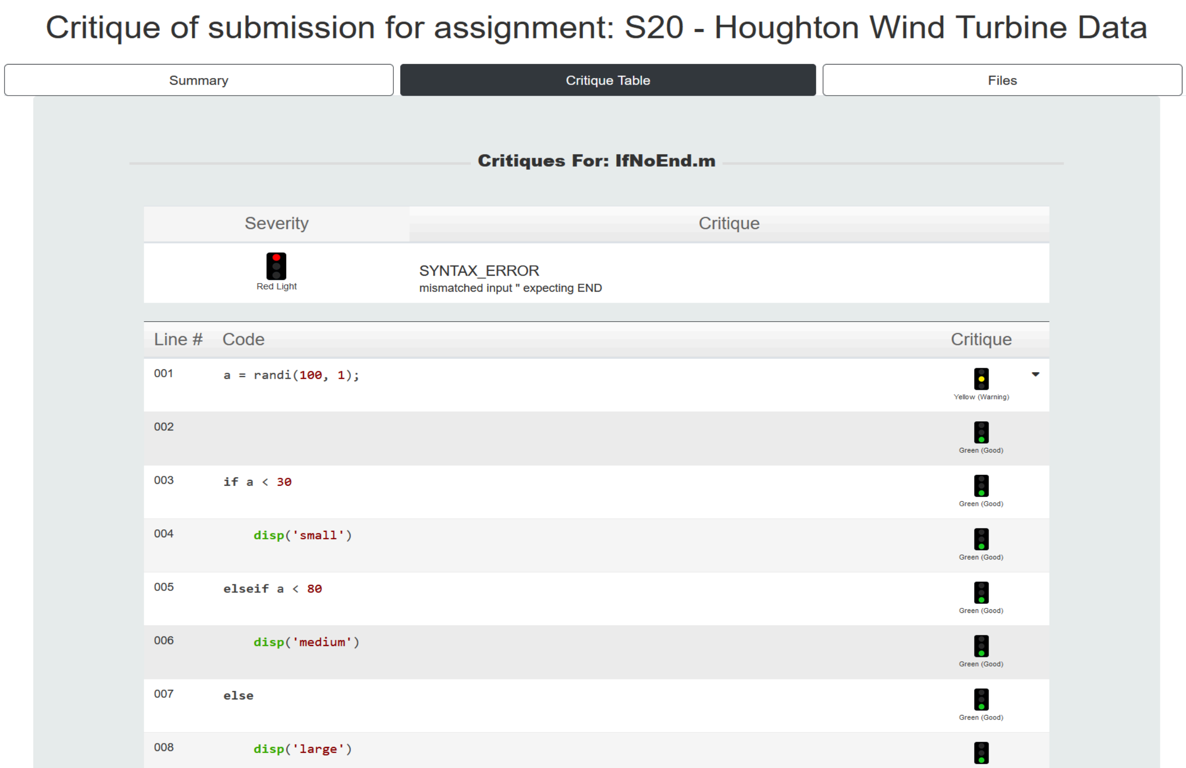

Figure 1: Traffic light system

Figure 2: Example feedback students received

This system offers clear, actionable feedback, helping students quickly identify and correct mistakes while reinforcing positive coding behaviors, which is especially beneficial for beginners.

WebTA has been partially implemented into the ENG1101 classroom for three semesters. In Spring of 2023, students were expected to submit three of their MATLAB assignments to WebTA, which included tasks such as writing functions and implementing conditional statements. After receiving feedback from WebTA, students were required to address the feedback and revise their code before submitting their final versions to the Learning

Management System (Canvas). Assignments included taking a screenshot of a green traffic light as proof of successful completion. Similarly, four assignments were assigned for Fall of 2023 and three for Spring of 2024. This system provides clear, actionable feedback, helping students quickly identify and correct mistakes while reinforcing positive coding behaviors, particularly in areas such as functions and conditional logic, which is especially beneficial for beginners.

Positionalities Statement

The research team for this project comprised faculty from CS, Psychology and Human Factors, and Engineering Fundamentals, as well as grad student researchers in Computer Science, Engineering, and Human Factors. The team had varying levels of programming experience and training, ranging from individuals with extensive programming expertise, those with formal training in how to teach programming, those who teach programming with little formal training in it, and to those new to MATLAB programming. This diversity in programming experience allowed the team to approach the project from multiple perspectives, ensuring that the WebTA tool was accessible to a broad range of students and effectively integrated into the educational setting.

The faculty team was responsible for overseeing the project, ensuring ethical standards, and coordinating data collection and analysis. The CS and Psychology and Human Factors graduate students, focused on the technical implementation of WebTA and its integration into the educational setting.

Study Setting and Participants

ENG1101 is an introductory engineering course that includes a MATLAB programming component. It is a core requirement for first-year engineering students and is designed to introduce fundamental programming concepts using MATLAB. The course is offered in multiple sections, but for the purposes of this study, we focused on a single section with 70 students. This decision was made to ensure consistency and to capture the unique experiences of a single cohort without introducing variations across multiple sections.

All students in the selected section received the WebTA intervention as part of the course curriculum. Participation in the study was voluntary, with 63 of the 70 students consenting to participate, resulting in a 90% participation rate.

No control group was used in this study, as only one offering of the class was available in the spring semester. The lack of a control group is acknowledged as a limitation, and this is discussed further in the limitations section. The fall semester includes several offerings of ENG1101. Current research is repeating this work in the fall semester with control groups (class offerings in which the intervention is not implemented).

Participants

The gender distribution of the participants was as follows: 37 (58.7%) identified as male, 26 students(41.3%) identified as female, and one student (1.6%) identified as non-binary. This breakdown reflects a slightly higher proportion of male students, which is consistent with broader trends in engineering education where male students often outnumber their female counterparts. The inclusion of both male and female students was crucial for examining the gender-specific impacts of the WebTA intervention on self-efficacy in programming and engineering. By analyzing the experiences of these students, the study aimed to gain insights into how such educational tools can support all learners, particularly those from underrepresented groups in engineering and computer science. For the purpose of maintaining confidentiality while reporting results, women and non-binary students are reported as a group.

Instruments Used for Assessment

Longitudinal Assessment of Engineering Self-Efficacy (LAESE)

The LAESE scale was employed to measure students' self-efficacy in engineering tasks. This instrument assesses various dimensions of engineering self-efficacy, including students' confidence in their ability to succeed in engineering courses, solve technical problems, and persist in their engineering studies [15]. The assessment of engineering self-efficacy among students will be focused on several constructs, each measured through specific items that provide a comprehensive understanding of students' confidence and perceived abilities within the field of engineering, including Factor1: Engineering Self-Efficacy, Factor 2: Engineering Career Expectations, Factor 3: Sense of Belonging, and Factor 4: Coping Self-Efficacy.

Computer Programming Self-Efficacy Scale (CPSES)

The CPSES was used to evaluate students' self-efficacy specifically related to programming. This scale measures students' beliefs in their ability to complete programming tasks, debug code, and learn new programming concepts [13]. The assessment of programming self-efficacy among first-year engineering students will focus on several key constructs, including Factor 1: Independence and Persistence; Factor 2: Complex Programming Tasks; Factor 3: Self-Regulation; and Factor 4: Simple Programming Tasks.

Data Collection Process

Prior to the implementation of WebTA, all participants completed baseline assessments using the LAESE and CPSES scales. These assessments were designed to capture the students' initial levels of self-efficacy in both engineering and programming [15], [13]. After several weeks of using WebTA, participants were reassessed using the same LAESE and CPSES instruments. The post-intervention assessments were aimed at identifying any changes in self-efficacy that occurred as a result of the intervention.

Statistical Analysis Methods

Paired t-tests were conducted to compare pre- and post-intervention scores on the LAESE and CPSES scales. These tests were used to determine whether the observed changes in self-efficacy scores were statistically significant [34]. This analysis provided insights into how the intervention differentially affected male and female students in terms of their engineering and programming self-efficacy [35].

Results

This section presents a summary of the key findings from the study. The analysis explores the impact of the code critiquer intervention on students' self-efficacy across different constructs, measured by both the CPSES and LAESE factors. To evaluate the overall impact of the intervention, paired t-tests were conducted to compare pre- and post-intervention self-efficacy scores within each group. Additionally, independent t-tests were used to examine potential gender differences, assessing whether the intervention had differential effects on male and female students' self-efficacy.

Pre and Post Intervention Scores

In this study, both paired and independent samples t-tests were conducted to analyze the data. A paired sample t-test was used to examine the means of two related groups—pre- and post-intervention scores on the CPSES and LAESE scales—assessing whether the mean difference between these scores was significantly different from zero and thus indicative of an intervention effect. Additionally, an independent sample t-test was employed to compare mean scores between distinct groups, examining potential differences unrelated to paired conditions. A significance level (alpha) of 0.05 was applied to both tests, consistent with common practice in educational research where 0.05 is frequently used as the threshold for significance [40]. However, some studies may adopt a more stringent level, such as 0.01, depending on the research context and objectives.

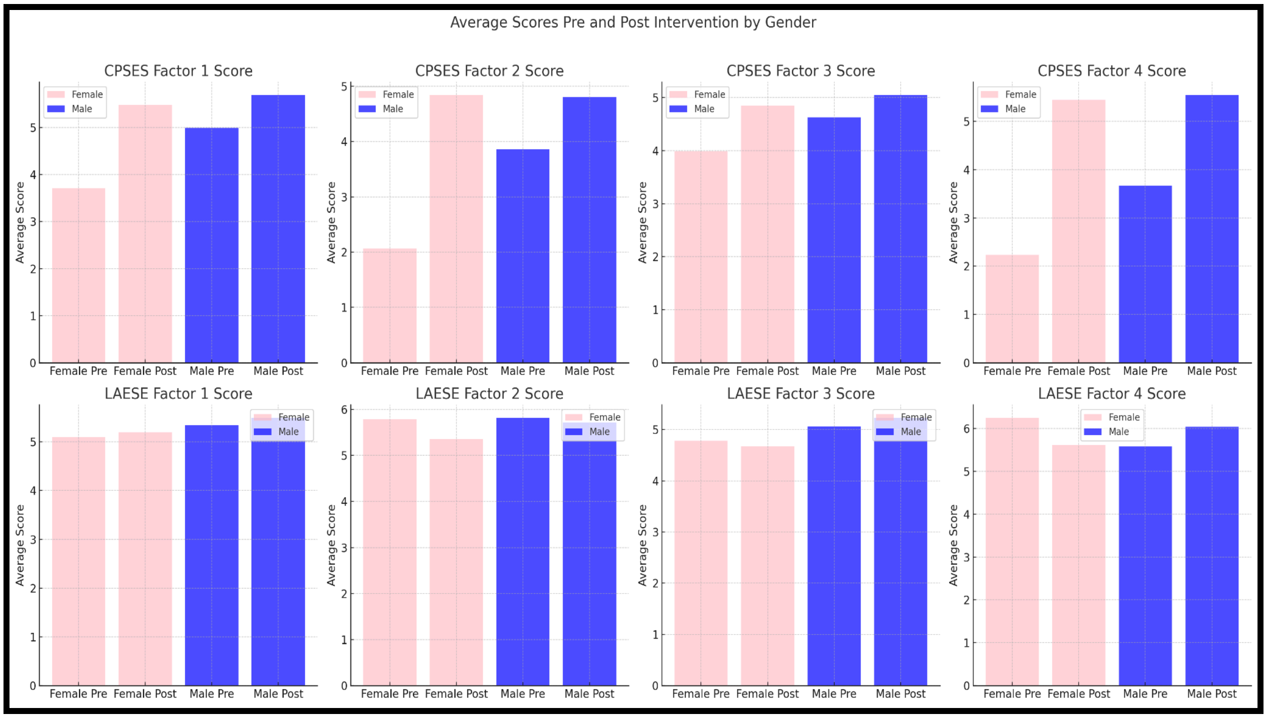

Figure 3: Pre and Post mean Intervention Score

Recall the CPSES factors (or constructs) of Independence and Persistence; Complex Programming Tasks, Self-Regulation, and Simple Programming Tasks. Figure 3 illustrates the comparison of pre- and post-intervention average scores for both male and female participants across four factors of the CPSES and LAESE scales. In the CPSES factors, both genders demonstrate an increase in average scores post-intervention, with some gender differences noted in the degree of change. For example, in CPSES Factors 1 (Independence and Persistence), and 4 (Simple Programming Tasks), females show a more pronounced improvement, closing the initial gap with males. Conversely, in CPSES Factor 2, males show a smaller pre-intervention baseline but achieve comparable scores post-intervention.

Recall the LAESE factors (or constructs) of Engineering Self-Efficacy, Engineering Career Expectations, Sense of Belonging, and Coping Self-Efficacy. The LAESE factors, however, exhibit minimal changes between pre- and post-intervention scores for both genders. Across all four LAESE factors, the scores remain relatively stable, with only minor fluctuations. In LAESE Factor 4, Coping Self-Efficacy, there is a slight decline in post-intervention scores for females, while male scores remain relatively consistent.

These trends set the stage for the subsequent statistical analysis, where paired and independent samples t-tests were conducted to determine the significance of these observed differences.

Paired T-TEST

To assess the impact of the code critiquer intervention on students' self-efficacy, statistical analyses were conducted on pre- and post-intervention scores. The focus of the analysis was to determine whether significant changes occurred in self-efficacy levels after the intervention and whether these changes differed by gender. Paired t-tests were employed to evaluate within-group differences in self-efficacy over time, while independent t-tests were used to compare gender-based differences in the intervention's effect. The following sections detail the results of these analyses.

Gender	Factor	Pre-intervention Mean (SD)	Post-interventio n Mean (SD)	t-statistic s	p-value
Female	CPSES Factor 1	3.71 (1.41)	5.48 (1.15)	4.95	<0.05*
Female	CPSES Factor 2	2.07 (1.03)	4.84 (1.07)	9.52	< 0.05*
Female	CPSES Factor 3	3.99 (1.46)	4.85 (1.16)	2.34	< 0.05
Female	CPSES Factor 4	2.24 (1.38)	5.45 (1.14)	9.13	< 0.05*
Female	LAESE Factor 1	5.10 (1.11)	5.20 (1.34)	0.29	0.773
Female	LAESE Factor 2	5.78 (1.48)	5.36 (1.63)	-0.98	0.332
Female	LAESE Factor 3	4.78 (1.09)	4.68 (1.43)	-0.29	0.773
Female	LAESE Factor 4	6.24 (0.66)	5.61 (1.00)	-2.02	< 0.05
Male	CPSES Factor 1	5.00 (1.37)	5.69 (1.00)	2.48	< 0.05
Male	CPSES Factor 2	3.86 (1.42)	4.80 (1.28)	3.01	< 0.05*
Male	CPSES Factor 3	4.63 (0.90)	5.05 ((1.18)	1.72	0.09
Male	CPSES	3.67 (1.80)	5.54 (1.28)	5.17	< 0.05*

	Factor 4				
Male	LAESE Factor 1	5.34 (1.28)	5.49 (1.14)	0.52	0.605
Male	LAESE Factor 2	5.81 (1.47)	5.72 (1.29)	-0.28	0.78
Male	LAESE Factor 3	5.06 (1.22)	5.23 (1.12)	0.6	0.554
Male	LAESE Factor 4	5.57 (1.65)	6.04 (0.93)	1.5	0.139

Table 2: Paired T-test
*denotes p-values also <0.01

CPSES

The paired t-test results for the CPSES (Computer Programming Self-Efficacy Scale) reveal significant improvements in self-efficacy across multiple factors for both male and female participants. For female participants, the mean score for CPSES Factor 1, Independence and Persistence, increased from 3.71 (SD = 1.41) to 5.48 (SD = 1.15), with the t-test indicating statistical significance ($t = 4.95$, $p < 0.01$) and a large effect size (Cohen's $d = 1.37$). This suggests a notable enhancement in programming-related self-efficacy. Similarly, CPSES Factor 2, Complex Programming Tasks, scores rose from 2.07 (SD = 1.03) to 4.84 (SD = 1.07) ($t = 9.52$, $p < 0.01$), with a very large effect size (Cohen's $d = 2.64$), pointing to a substantial increase in confidence in this specific area.

CPSES Factor 3, Self-Regulation, also showed significant improvement, with mean scores increasing from 3.99 (SD = 1.46) to 4.85 (SD = 1.16) ($t = 2.34$, $p = 0.023$) and a moderate effect size (Cohen's $d = 0.65$). The largest shift was observed in CPSES Factor 4, Simple Programming Tasks, where mean scores for female participants increased from 2.24 (SD = 1.38) to 5.45 (SD = 1.14) ($t = 9.13$, $p < 0.01$), with a large effect size (Cohen's $d = 2.53$), suggesting a considerable improvement in self-efficacy related to this factor.

For male participants, similar patterns were observed, with increases across the CPSES factors. However, the degree of improvement varied slightly between genders, suggesting the possibility of differential responses. These changes point to an overall enhancement in programming self-efficacy, though it is essential to note that the absence of a control group prevents definitive conclusions about the source of these changes. Other factors, such as external influences or additional support mechanisms, may also have contributed to the observed improvements.

LAESE

The paired t-test results for the LAESE (Longitudinal Assessment of Engineering Self-Efficacy) factors indicate more stability in self-efficacy scores over time, with fewer significant changes observed compared to the CPSES factors. For female participants, the mean score for LAESE Factor 1, Engineering Self-Efficacy, changed only slightly from 5.10 (SD = 1.11) to 5.20 (SD = 1.34), and the t-test results showed no significant difference ($t = 0.29$, $p = 0.773$) with a negligible effect size (Cohen's $d = 0.08$). This suggests that self-efficacy related to this factor remained relatively stable over the study period.

Similar patterns were found for the other LAESE factors, with both male and female participants showing minimal changes in their self-efficacy scores pre- and post-intervention. This stability suggests that the changes observed in programming-specific self-efficacy did not extend to broader engineering self-efficacy. However, given the lack of a control group, it is important to interpret these results cautiously. The stability in scores might reflect external influences or a natural progression in participants' self-efficacy unrelated to the intervention.

In summary, while the paired t-test results show significant improvements in programming self-efficacy (as measured by the CPSES) for both genders, particularly in female participants, the broader engineering self-efficacy (measured by the LAESE) remained largely unchanged. Due to the absence of a control group, caution is required in attributing these changes directly to the intervention, as other variables may have played a role in influencing the outcomes.

Following the analysis of within-group differences through paired t-tests, it is also important to examine whether the observed changes in self-efficacy differ between male and female participants. To explore potential gender-based differences in the impact on self-efficacy scores, independent t-tests were conducted. These tests compare the mean differences between male and female participants' pre- and post-intervention scores, providing insight into whether there were significant variations in how each gender responded. The independent t-tests allow for a more nuanced understanding of any differential effects between genders, though, as previously noted, the lack of a control group warrants caution in interpreting these results.

Independent t-test

The independent t-tests conducted to compare post-intervention self-efficacy scores between male and female participants across the CPSES and LAESE factors revealed no statistically significant differences between the two groups. For CPSES Factor 1, Independence and Persistence, the mean score for males was 5.69 (SD = 1.00) compared to 5.48 (SD = 1.15) for females, with a t-statistic of 0.76 and a p-value of 0.449. The effect size (Cohen's $d = 0.20$) was small, suggesting only a minor variation in self-efficacy levels between genders for this factor.

#	Construct	Male Mean (SD)	Female Mean (SD)	t-statistics	p-value
1	CPSES Factor 1	5.69 (1.15)	5.48 (1.15)	0.76	0.449
2	CPSES Factor 2	4.80 (1.28)	4.84 (1.07)	-0.12	0.905
3	CPSES Factor 3	5.05 (1.18)	4.85 (1.16)	0.67	0.503
4	CPSES Factor 4	5.45 (1.28)	5.45 (1.14)	0.32	0.754
5	LAESE Factor 1	5.49 (1.14)	5.20 (1.34)	0.89	0.376
6	LAESE Factor 2	5.72 (1.29)	5.36 (1.63)	0.94	0.350
7	LAESE Factor 3	5.23 (1.12)	4.68 (1.43)	1.62	0.111
8	LAESE Factor 4	6.04 (0.93)	5.61 (1.46)	1.34	0.187

Table 3: independent t-test

Similarly, CPSES Factor 2, complex Programming Tasks, showed comparable mean scores for males (4.80, SD = 1.28) and females (4.84, SD = 1.07), with a t-statistic of -0.12 and a p-value of 0.905, indicating no significant difference. The negligible effect size (Cohen's d = -0.03) further supports the lack of variation in self-efficacy between genders for this factor.

For CPSES Factor 3, Self-regulation, males had a mean score of 5.05 (SD = 1.18) compared to 4.85 (SD = 1.16) for females. The t-statistic was 0.67, with a p-value of 0.503, indicating no significant gender difference. The small effect size (Cohen's d = 0.17) points to minimal differences in self-efficacy levels.

In CPSES Factor 4, Simple Programming Tasks, male participants had a mean score of 5.54 (SD = 1.28), while females scored 5.45 (SD = 1.14), with a t-statistic of 0.32 and a p-value of 0.754. The effect size (Cohen's d = 0.08) was negligible, indicating that self-efficacy levels were virtually the same between the genders.

The LAESE Factor 1, Engineering Self-efficacy, scores showed similar results, with males scoring 5.49 (SD = 1.14) and females scoring 5.20 (SD = 1.34). The t-statistic of 0.89 and p-value of 0.376 suggest no significant difference, with a small effect size (Cohen's d = 0.23) pointing to a minor, non-significant difference in self-efficacy.

Overall, the results indicate that male and female participants exhibited similar self-efficacy levels across all measured factors, with no significant differences between the groups. However, due to the absence of a control group, these findings should be interpreted cautiously. It is possible that factors outside the intervention may have influenced the results, and without a control condition, it remains difficult to isolate the cause of any observed changes or similarities in self-efficacy between genders.

Data Normality

Prior to conducting the t-tests, it was necessary to assess the normality of the data, as both paired and independent t-tests assume normally distributed data. While normality becomes less critical with larger sample sizes, typically over 30 participants per group, due to the Central Limit Theorem (Field, 2013), it remains essential to formally test for normality, particularly when sample sizes are smaller or the data distribution is uncertain. The Central Limit Theorem suggests that, as sample sizes increase, the distribution of the sample mean differences approaches normality, even when the underlying data is not perfectly normal (Ghasemi & Zahediasl, 2012).

To ensure the data met this assumption, the Shapiro-Wilk test was employed to assess normality. The Shapiro-Wilk test is frequently used in real-world applications across various fields, including educational and psychological research, to evaluate whether data significantly deviates from a normal distribution (Razali & Wah, 2011). This approach helped ensure the validity of the subsequent t-tests, providing confidence that the assumptions of the statistical models were adequately met.

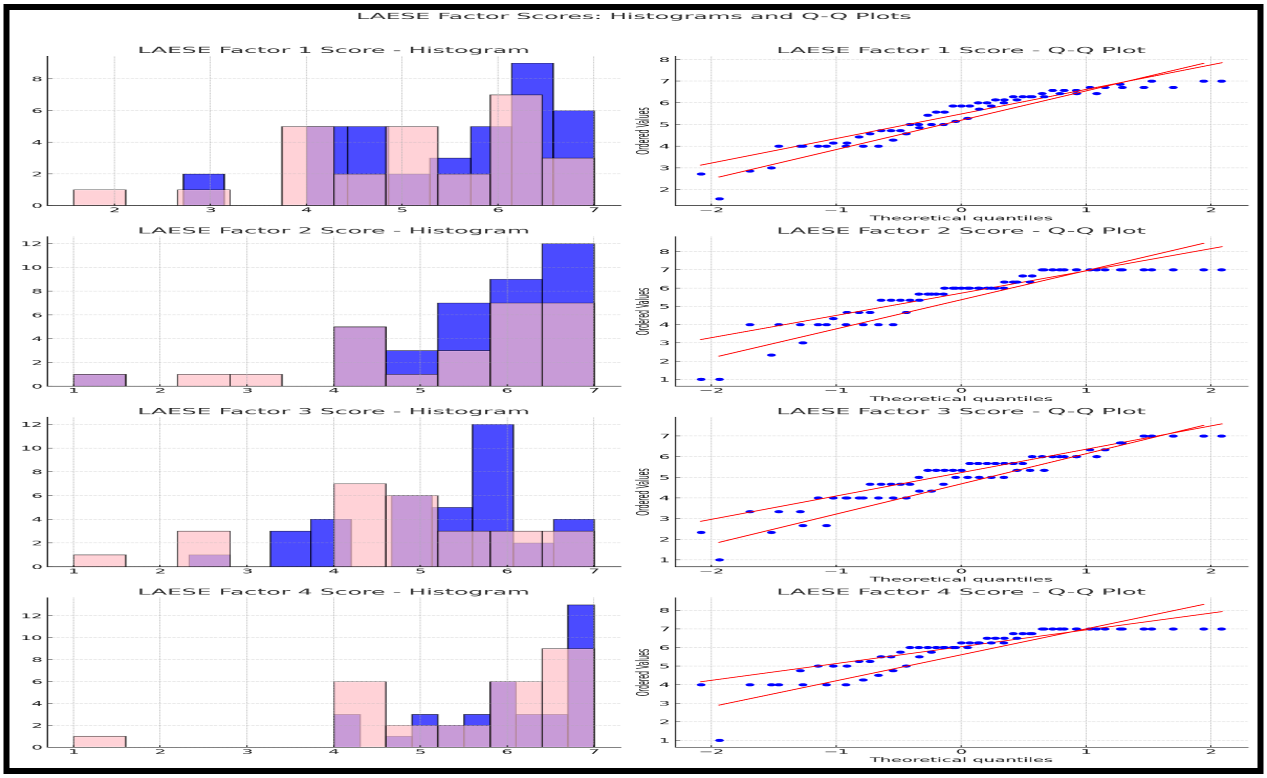

Figure 4: LAESE Factor scores - Histograms and Q-Q plots

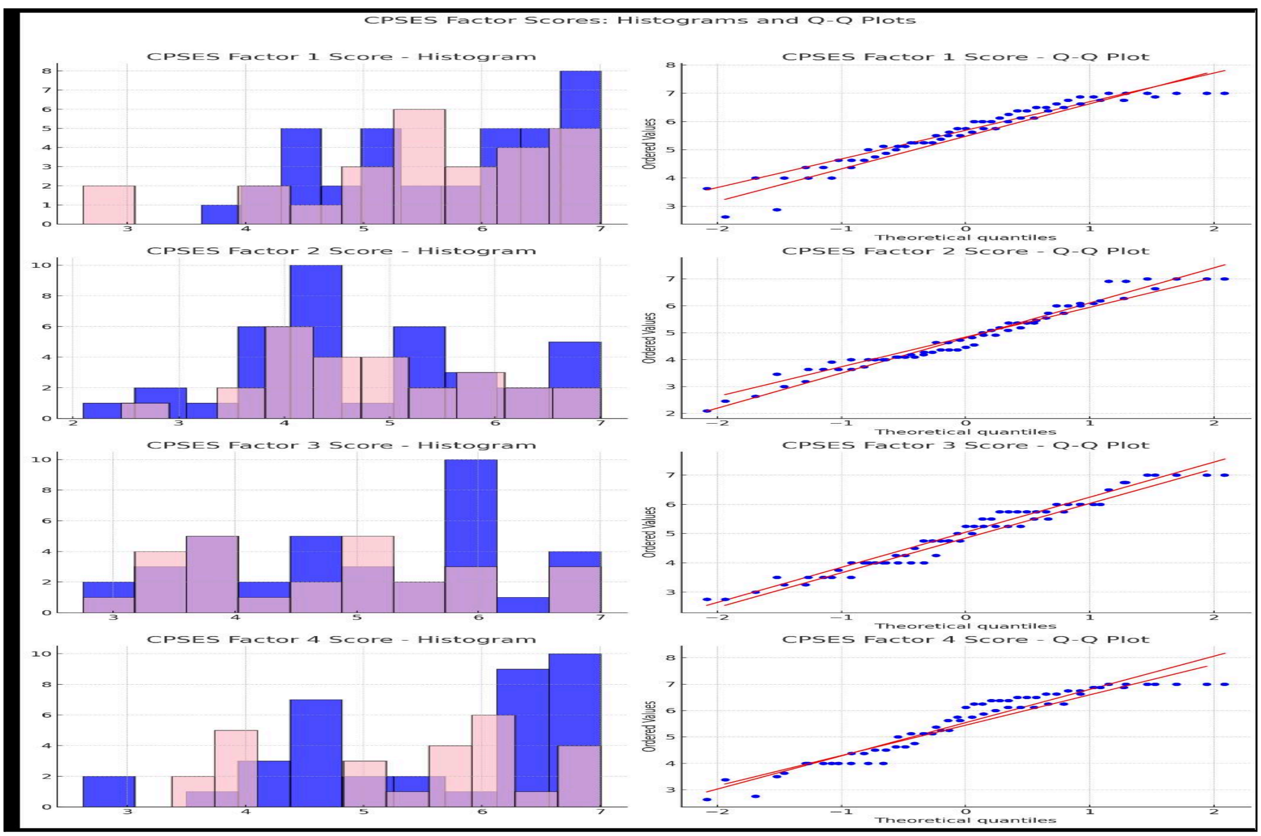

figure 5: CPSES Factor scores - Histograms and Q-Q plots

Figures 4 and 5 provide visual assessments of the data distribution for the CPSES and LAESE factors, respectively, through histograms and Q-Q plots. These figures are essential in evaluating the assumption of normality, which underpins the use of t-tests in this analysis.

For the CPSES factors in Figure 4, the histograms suggest that the majority of participants' scores cluster around the mid-to-high range (4-7). Although the Q-Q plots demonstrate that the data points largely follow the expected normal distribution line, slight deviations are present, particularly in the tails. These deviations indicate that while the assumption of normality holds reasonably well, the presence of some skewness or kurtosis in the data, especially at the extremes, may influence the paired and independent t-tests' outcomes. Despite these deviations, the Central Limit Theorem mitigates concerns for large sample sizes, supporting the robustness of the t-tests. However, caution should still be applied in interpreting any significant results, given the small deviations from normality and the absence of a control group.

Similarly, Figure 5 shows the distribution of scores for the LAESE factors. The histograms suggest more pronounced skewness, particularly for Factors 3 and 4, where a substantial portion of the data is clustered in the higher score ranges. The Q-Q plots for these factors reveal a greater departure from the normal distribution line at both tails, indicating a potential violation of the normality assumption. This departure could impact the t-test results for the LAESE factors, particularly for smaller sample sizes. The observed skewness might lead to a reduction in the sensitivity of the t-tests, increasing the risk of Type I or Type II errors.

In conclusion, while the data largely approximates normality, as evidenced by the visualizations in Figures 4 and 5, some deviations exist. These deviations, especially in the LAESE factors, should be considered when interpreting the results of both the paired and independent t-tests. Without a control group and with slight deviations from normality, the findings should be approached with caution, acknowledging that other unmeasured factors may have influenced the data distribution and the statistical outcomes.

Discussion

Significance of Changes in Self-Efficacy Constructs

The results of this study indicated significant changes in self-efficacy across both CPSES and LAESE constructs after the intervention. Paired t-tests revealed substantial improvements in programming self-efficacy, particularly in CPSES factors where both male and female students experienced notable gains (Factor 1: Independence and Persistence; Factor 2: Complex Programming Tasks; and Factor 4: Simple Programming Tasks). The increases observed in the CPSES factors suggest that participants felt more confident in their ability to tackle programming-related tasks. This enhancement in self-efficacy is important, as it is known to positively influence students' motivation, persistence, and performance in programming-related coursework [5]. The significant improvements in areas such as problem-solving and coding complexity imply that students felt better equipped to manage the demands of programming tasks after the intervention [14] .

Differential Impact on Male Versus Female Students

The results of the independent t-tests highlight that the intervention had a relatively balanced effect on both male and female students, with no statistically significant gender differences in post-intervention self-efficacy scores across both CPSES and LAESE constructs. However, the effect sizes for female students across all CPSES factors were larger, suggesting that female students might have experienced more pronounced improvements in self-efficacy, particularly in areas where their initial scores were lower. Additionally, the pre-post changes were significant for female students across all CPSES factors. These findings align with previous research suggesting that targeted interventions can help narrow the self-efficacy gap between male and female students in technical domains such as programming [11, 6]. The increase in female students' self-efficacy levels is encouraging, as it may help to address the gender disparity in confidence often observed in computing fields.

Implications for Educational Practice

The findings underscore the importance of integrating interventions that support self-efficacy development, particularly for underrepresented groups in technical fields. Although the independent t-tests did not reveal significant gender differences, the overall gains in self-efficacy, especially among female students, highlight the potential of targeted educational tools to address confidence disparities [16]. Educators should consider incorporating similar interventions that offer structured feedback and guidance to students, as these tools may be effective in enhancing self-efficacy and ultimately improving student outcomes in programming and engineering education [19].

Enhancing Programming Self-Efficacy

The improvements seen in programming self-efficacy across all participants point to the value of focusing on this construct in programming and engineering curricula. As programming proficiency becomes increasingly important in engineering and technology fields, it is crucial that students feel confident in their coding abilities [4]. The intervention, by improving self-efficacy, likely contributed to students feeling more capable in their programming tasks. Such interventions, which provide immediate and constructive feedback [21], could be expanded in future courses to support students in building essential technical skills and maintaining confidence in their abilities.

Limitations of the Study

One of the primary limitations of this study is the absence of a control group, which restricts the ability to isolate the effects of the intervention from other factors that could have influenced the observed improvements in self-efficacy. While the paired t-tests provided insights into changes within the same cohort over time, future studies would benefit from including a control group to establish more robust causal claims [34]. Furthermore, although the sample size was sufficient to detect significant changes, larger and more diverse samples would enhance the generalizability of the findings. Expanding the study across different institutions and educational contexts would provide a clearer picture of how similar interventions impact a broader range of students.

Generalizability of Findings

The generalizability of these findings is somewhat limited, given that the study was conducted with a specific cohort of students within a single institution. Future research should seek to replicate these results across multiple institutions and disciplines to determine the broader applicability of the intervention [38]. Additionally, longitudinal studies tracking the long-term effects of improved programming self-efficacy on academic and career success would provide valuable insights into the lasting impact of such interventions on students in programming and engineering fields [39].

Conclusion

This study demonstrated that the WebTA code critiquing tool significantly improved programming self-efficacy among first-year engineering students, with particularly notable gains for female students. These findings underscore the effectiveness of targeted educational interventions in boosting self-efficacy, which is essential for success in programming. Tools like WebTA, which provide immediate feedback, can be instrumental in addressing confidence disparities and promoting gender equity in programming fields.

Recommendations for Future Research

Future research should focus on larger and more diverse samples, including underrepresented genders, to validate these findings and explore the broader impact of interventions like WebTA. Additionally, examining other demographic factors, such as race and socioeconomic status, could provide deeper insights into how different groups benefit from educational tools. Longitudinal studies would further help understand the long-term effects of such interventions on academic and career outcomes in programming.

References

1. Beyer, S. (2014). Why Are Women Underrepresented in Computer Science? Gender Differences in Stereotypes, Self-Efficacy, Values, and Interests and Predictors of Future CS Course-Taking and Grades. Computer Science Education, 24(2), 153-192.

2. Cohoon, J. M. (2006). Just Get Over It or Just Get On with It: Retaining Women in Undergraduate Computing. In Women and Information Technology: Research on Underrepresentation, MIT Press.

3. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35. https://doi.org/10.1145/1118178.1118215

4. Guzdial, M. (2009). Programming for engineers: A multidisciplinary approach. IEEE Transactions on Education, 52(3), 386-390. https://doi.org/10.1109/TE.2008.924215

5. Bandura, A. (1997). Self-efficacy: The exercise of control. W.H. Freeman

6. Margolis, J., & Fisher, A. (2002). Unlocking the clubhouse: Women in computing. MIT Press.

7. Denner, J., Werner, L., & Ortiz, E. (2014). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? Computers & Education, 69, 28-37. https://doi.org/10.1016/j.compedu.2013.06.005

8. Barker, L. J., McDowell, C., & Kalahar, K. (2009). Exploring factors that influence computer science introductory course students to persist in the major. ACM SIGCSE Bulletin, 41(1), 153-157. https://doi.org/10.1145/1539024.1508923

9. Cohoon, J. M. (2001). Toward improving female retention in the computer science major. Communications of the ACM, 44(5), 108-114. https://doi.org/10.1145/374308.374367

10. Sax, L. J., Lehman, K. J., Jacobs, J. A., Kanny, M. A., Lim, G., Monje-Paulson, L., & Zimmerman, H. B. (2015). Anatomy of an enduring gender gap: The evolution of women's participation in computer science. Journal of Women and Minorities in Science and Engineering, 21(3), 229-249. https://doi.org/10.1615/JWomenMinorScienEng.2015012938

11. Master, A., Cheryan, S., & Meltzoff, A. N. (2016). Computing whether she belongs: Stereotypes undermine girls' interest and sense of belonging in computer science. Journal of Educational Psychology, 108(3), 424-437. https://doi.org/10.1037/edu0000061

12. Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? Computer Science Education, 21(2), 105-134. https://doi.org/10.1080/08993408.2011.579805

13. Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004). Self-efficacy and mental models in learning to program. ACM SIGCSE Bulletin, 36(3), 171-175. https://doi.org/10.1145/1026487.1008042

14. Lent, R. W., Brown, S. D., & Hackett, G. (2008). Social cognitive career theory and its applications to engineering education: An empirical review. Journal of Engineering Education, 97(4), 407-423. https://doi.org/10.1002/j.2168-9830.2008.tb00992.x

15. Marra, R. M., Rodgers, K. A., Shen, D., & Bogue, B. (2009). Women engineering students and self-efficacy: A multi-year, multi-institution study of women engineering student self-efficacy. Journal of Engineering Education, 98(1), 27-38. https://doi.org/10.1002/j.2168-9830.2009.tb01003.x

16. Cheryan, S., Plaut, V. C., Davies, P. G., & Steele, C. M. (2009). Ambient belonging: How stereotypical cues impact gender participation in computer science. Journal of Personality and Social Psychology, 97(6), 1045-1060. https://doi.org/10.1037/a0016239

17. Steele, C. M., & Aronson, J. (1995). Stereotype threat and the intellectual test performance of African Americans. Journal of Personality and Social Psychology, 69(5), 797-811. https://doi.org/10.1037/0022-3514.69.5.797

18. Yoder, B. L. (2017). Engineering by the numbers. American Society for Engineering Education. Retrieved from https://ira.asee.org/wp-content/uploads/2017/10/2017-Engineering-by-Numbers-Engineering-Statistics.pdf

19. Sax, L. J., Kanny, M. A., Riggers-Piehl, T. A., Whang, H., & Paulson, L. (2017). But I'm not good at math: The changing salience of mathematical self-concept in shaping women's and men's STEM aspirations. Research in Higher Education, 58(8), 823-852. https://doi.org/10.1007/s11162-017-9452-6

20. Hong, L., & Page, S. E. (2004). Groups of diverse problem solvers can outperform groups of high-ability problem solvers. Proceedings of the National Academy of Sciences, 101(46), 16385-16389. https://doi.org/10.1073/pnas.0403723101

21. Shute, V. J. (2008). Focus on formative feedback. Review of Educational Research, 78(1), 153-189. https://doi.org/10.3102/0034654307313795

22. Van der Kleij, F. M., Feskens, R. C. W., & Eggen, T. J. H. M. (2015). Effects of feedback in a computer-based learning environment on students' learning outcomes: A meta-analysis. Review of Educational Research, 85(4), 475-511. https://doi.org/10.3102/0034654314564881

23. Rivers, K. (2016). Immediate Feedback and the Effect of Feedback Timing on Learning Outcomes. PhD Dissertation, Carnegie Mellon University.

24. Edwards, S. H., Tilley, B. W., & Barnette, N. M. (2009). An automated grading system with immediate feedback. SIGCSE Bulletin, 41(1), 148-152. https://doi.org/10.1145/1539024.1508927

25. Ihantola, P., Ahadi, A., Karavirta, V., & Petersen, A. (2010). Educational data mining and learning analytics in programming: Literature review and case studies. Journal of Educational Data Mining, 2(1), 3-14.

26. Jadud, M. C. (2019). Programming feedback and problem-solving strategies in novice programmers. Computers & Education, 135, 14-29.

27. Leinonen, J., & Vihavainen, A. (2020). Artificial intelligence-driven feedback in programming education: Effects on self-efficacy. Journal of Computer Science Education, 28(4), 335-352.

28. Pettit, J. B., & Ariza, R. (2021). Formative feedback in online coding platforms: Engagement and retention during the COVID-19 pandemic. ACM Transactions on Computing Education, 21(2), Article 12. https://doi.org/10.1145/3454127

29. Benjamin, M. E., Brown, L. E., Sticklen, J., Ureel, L. C., & Jarvie-Eggart, M. (2023). Engaging Novice Programmers: A Literature Review of the Effect of Code Critiquers on Programming Self-efficacy. In IEEE Frontiers in Education Conference, FIE 2023.

30. Ureel, L. C., Brown, L. E., Jarvie-Eggart, M., Sticklen, J., & Albrant, L. (2023). Current Progress of Providing Rich Immediate Critique of Anti-patterns in Student Code. American Society for Engineering Education (ASEE).

31. Smith, P., Tahaei, M., & Khandelwal, A. (2022). Understanding Developer Feedback in IDEs: A Security Perspective. In Towards Immediate Feedback for Security-Relevant Code in Development Environments (pp. 215-232).

32. Rivers, K., & Koedinger, K. R. (2017). Immediate Feedback Promotes Self-Regulation: Effect of Feedback Timing on Learning Outcomes. Educational Technology Research and Development, 65(2), 325-348. https://doi.org/10.1007/s11423-016-9479-1

33. Secules, S., Gupta, A., Elby, A., & Tanu, G. (2021). Positionality Practices and Reflexivity in Engineering Education Research: Implications for Equity and Inclusion. Journal of Engineering Education, 110(1), 18-35. https://doi.org/10.1002/jee.20377

34. Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

35. Field, A. (2013). Discovering statistics using IBM SPSS statistics (4th ed.). SAGE Publications.

36. West, S. G., & Aiken, L. S. (2003). Multiple regression: Testing and interpreting interactions. SAGE Publications.

37. Walther, J., & Sochacka, N. W. (2014). Qualifying Qualitative Research Quality (The Q3 Project) – An interactive discourse around research quality in interpretive approaches to

engineering education research. In Frontiers in Education Conference (FIE), Madrid, Spain. IEEE.

38. Creswell, J. W., & Creswell, J. D. (2017). *Research design: Qualitative, quantitative, and mixed methods approaches* (5th ed.). SAGE Publications.

39. Tinto, V. (1993). Leaving college: Rethinking the causes and cures of student attrition (2nd ed.). University of Chicago Press.

40. Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (2011). *How to design and evaluate research in education* (8th ed.). McGraw-Hill.