Programming Self-Efficacy and Cybersecurity Competency Among Information Technology Students of Trinidad Municipal College: A Correlational Study

Eric B. Bulala[1], Mark Angelou A. Balonga[2], Alvin M. Boncales[3], Emelyn Shaira B. Felisilda[4], Raymund Castañares[5], Maria Shina Jane A. Melendres[6], Joe D. Basanta[7], Kristine T. Soberano[8]

State University of Northern Negros, Sagay, Negros Occidental, Philippines

DOI: https://doi.org/10.51244/IJRSI.2026.1307000340

Received: 31 July 2026; Accepted: 05 August 2026; Published: 18 August 2026

ABSTRACT

This research examined the programming self-efficacy and cybersecurity competency among Information Technology (IT) students at Trinidad Municipal College, Bohol, Philippines. The study aimed to assess students' levels of programming self-efficacy and cybersecurity competency, describe their baseline security practices and study habits, and determine the relationship between programming self-efficacy and cybersecurity competency. A descriptive-correlational quantitative research design was employed. Data were collected from 730 officially enrolled IT students using a structured, researcher-administered questionnaire.

The findings revealed that the respondents generally demonstrated competent levels of both programming self-efficacy and cybersecurity competency. Among the programming self-efficacy dimensions, Persistence obtained the highest mean score (3.88), followed by Confidence in Programming (3.43), while Problem-Solving and Debugging received the lowest mean score (3.22). In terms of cybersecurity competency, Network Security obtained the highest mean score (3.57), whereas Secure Programming recorded the lowest (3.29). The results further showed that the majority of respondents spent only 1–3 hours per week practicing programming (69.73%), 84.25% had not attended any cybersecurity seminar or workshop, 63.42% reported using two-factor or multi-factor authentication (2FA/MFA), and 40.82% admitted that they sometimes reused passwords across multiple accounts.

Pearson's product-moment correlation analysis revealed a statistically significant strong positive relationship between programming self-efficacy and cybersecurity competency ($r = 0.6139$, $p = 0.001$, 95% CI [0.565, 0.658]). The coefficient of determination ($r^2 = 0.3769$) indicated that approximately 37.69% of the variance in cybersecurity competency was associated with programming self-efficacy. These findings suggest that students with greater confidence in their programming abilities are more likely to demonstrate higher levels of cybersecurity competency. The study underscores the importance of strengthening programming self-efficacy through practical programming experiences, secure coding activities, and cybersecurity-focused learning opportunities to enhance students' technical competence and preparedness for the evolving cybersecurity landscape.

Keywords: Programming Self-Efficacy, Cybersecurity Competency, Information Technology Students, Descriptive-Correlational Research, Secure Programming

INTRODUCTION

Cybersecurity has become one of the most critical disciplines in Information Technology (IT) as organizations increasingly rely on digital systems to manage sensitive information and essential services. The rapid growth of cyber threats, including phishing attacks, ransomware, malware, and data breaches, has increased the global demand for professionals equipped with strong cybersecurity competencies. Consequently, higher education institutions play a vital role in preparing future IT professionals by developing both their technical knowledge and confidence in applying programming and cybersecurity skills. Recent studies emphasize that programming competence serves as a fundamental foundation for secure software development, vulnerability assessment, and

defensive coding practices, making programming self-efficacy an essential component of cybersecurity education (Taylor et al., 2013; World Economic Forum, 2025).

Programming self-efficacy refers to an individual's belief in their capability to successfully perform programming-related tasks, including writing code, debugging applications, solving computational problems, and learning new programming concepts. According to Bandura's Self-Efficacy Theory, individuals with stronger self-efficacy are more likely to persevere when confronted with challenging tasks, invest greater effort, and ultimately achieve higher levels of performance (Bandura, 1997). Within computing education, research has consistently demonstrated that students with higher programming self-efficacy exhibit greater persistence, increased motivation, improved problem-solving skills, and stronger academic achievement in programming courses (Tsai et al., 2018; Abdunabi et al., 2024). Likewise, cybersecurity competency encompasses the knowledge, technical skills, and secure practices required to identify cyber threats, protect information systems, implement secure programming techniques, and respond effectively to cybersecurity incidents (NIST, 2020).

Despite the recognized importance of programming confidence in computing education, relatively few empirical studies have examined its relationship with cybersecurity competency among undergraduate Information Technology students. Existing research has primarily focused on programming performance, cybersecurity awareness, or technical knowledge as separate constructs, while limited evidence explores how students' confidence in programming contributes to their ability to apply cybersecurity concepts in practical situations. Moreover, recent cybersecurity workforce reports emphasize that technical proficiency alone is insufficient without confidence in applying acquired skills to solve complex cybersecurity problems (ISC2, 2024; World Economic Forum, 2025).

Given this research gap, a systematic, data-driven investigation of the relationship between programming self-efficacy and cybersecurity competency is necessary. Therefore, this study employed a descriptive-correlational research design to determine the relationship between programming self-efficacy and cybersecurity competency among Information Technology students of Trinidad Municipal College. Using quantitative data gathered through structured questionnaires, the study assessed students' programming self-efficacy across the dimensions of confidence, problem-solving and debugging, and persistence, as well as their cybersecurity competency in the areas of cybersecurity knowledge, network security, secure programming, threat identification and risk management, and cybersecurity tools and practices. The findings are expected to provide empirical evidence that may assist educators, curriculum developers, and academic institutions in designing instructional strategies that strengthen programming confidence while simultaneously enhancing cybersecurity competency, thereby preparing future IT professionals for the evolving demands of the cybersecurity workforce (Bandura, 1997; NIST, 2020; World Economic Forum, 2025).

Objectives of the Study

This research primarily sought to investigate and assess the programming self-efficacy and cybersecurity competency among Information Technology (IT) students of Trinidad Municipal College, with the goal of comprehending their technical confidence, behavioral habits, and competence within computing tasks. Specifically, the study aimed to analyze the relationship between programming self-efficacy and cybersecurity competency, identify potential patterns in students' self-efficacy dimensions (e.g., confidence, problem-solving, debugging, and persistence), and evaluate key cybersecurity competency domains, including network security, secure programming, and threat identification. The survey data collected from this cohort of IT students yielded significant insights, which are intended to inform the development of data-informed instructional strategies and curriculum improvements designed to foster defensive coding practices, enhance problem-solving capabilities, and elevate students' overall technical confidence.

MATERIALS AND METHODS

Research Design

This research utilized a descriptive-correlational quantitative research design to explore the relationship between programming self-efficacy and cybersecurity competency within a cohort of Information Technology (IT)

students. The primary objective of the study was to gather numerical data to ascertain the level of students' self-efficacy across confidence, debugging, and persistence dimensions, as well as their self-reported cybersecurity competency across domains such as network security, secure programming, and risk management. Furthermore, the research assessed students' baseline security practices and study habits, including password hygiene, multi-factor authentication usage, and weekly study hours. This methodological approach facilitated a systematic description of students' self-efficacy levels and enabled the analysis of a statistically significant relationship between programming self-efficacy and cybersecurity competency.

Variable Identification

To ensure clarity in the statistical analysis, the variables of this study are categorized and operationally defined as follows:

Variable Category	Variable Name	Operational Definition / Indicators
Independent Variable	Programming Self-Efficacy	Perceived capability and confidence across dimensions including general coding confidence, problem-solving and debugging, and persistence.
Dependent Variable	Cybersecurity Competency	Evaluated proficiency across domains including cybersecurity knowledge, network security, secure programming, threat identification, and tool practices.
Moderating Variable	Study Habits & Practice	Average number of hours spent practicing programming per week and participation in security seminars/workshops.
Moderating Variable	Security Hygiene	Prior programming knowledge, password update frequency, password reuse habits, and multi-factor authentication (2FA/MFA) adoption.

Table 1. Conceptual Framework of Variables

Population and Sampling

The study included Information Technology (IT) students from Trinidad Municipal College, chosen using a proportionate stratified random sampling technique. The study's methodology began with the acquisition of the official consolidated enrollment list from the college registrar, which served as the sampling frame. The population was divided into distinct strata based on year level to ensure proportional representation across all subgroups. Within each stratum, a computer-based random number generator was employed to select participants proportionally relative to each group's size in the total population. This approach ensured that each subgroup was accurately represented while maintaining random selection within every stratum. Ultimately, the final sample comprised 730 (413 male and 317 female) officially enrolled IT students who participated in the data collection phase and provided informed consent.

Research Instrument

The primary data collection tool was a structured, self-administered questionnaire divided into three main sections:

- Demographic Profile and Baseline Security Habits – Age, year level, gender, prior programming experience, weekly study hours, password update frequency, 2FA/MFA usage, password reuse practices, and attendance in cybersecurity seminars or workshops.
- Programming Self-Efficacy Assessment – Likert-scale questions adapted to measure students' confidence across three core dimensions: Confidence in Programming, Problem-Solving and Debugging, and Persistence.

- Cybersecurity Competency Assessment – Likert-scale items evaluating self-reported competency across five domains: Cybersecurity Knowledge, Network Security, Secure Programming, Threat Identification and Risk Management, and Cybersecurity Tools and Practices.

To interpret the responses systematically, weighted means for each construct were analyzed according to the following scoring scales and qualitative descriptors:

Scale	Mean Range	Programming Self-Efficacy Qualitative Description	Cybersecurity Competency Qualitative Description
5	4.21-5.00	Strongly Agree	Highly Competent
4	3.41-4.20	Agree	Competent
3	2.61-3.40	Neutral	Moderately Competent
2	1.81-2.60	Disagree	Slightly Competent
1	1.00-1.80	Strongly Disagree	Not Competent

Table 2. Interpretation Scale for Programming Self-Efficacy and Cybersecurity Competency

Since the questionnaire was adapted and contextualized for this study, it underwent content validation by three experts in Information Technology and educational research to ensure clarity, relevance, and alignment with the research objectives. A pilot test involving 30 IT students (who were excluded from the final analysis) was conducted to evaluate internal consistency. The overall instrument yielded a Cronbach's alpha value of 0.88, indicating high reliability and internal consistency across both key constructs.

Instrument Development and Validation

The research instrument was adapted and contextualized from established frameworks in computing education and information security. Items measuring programming self-efficacy were adapted from the Computer Programming Self-Efficacy Scale (CPSES) developed by Tsai et al. (2018), focusing on general confidence, persistence, and debugging ability. Cybersecurity competency items were structured based on the National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (NIST SP 800-181 Rev. 1). To establish content validity, the draft questionnaire was evaluated by a panel of three subject matter experts: two Assistant Professors in Information Technology with domain expertise in software development and network security, and one Senior Educational Researcher specializing in quantitative measurement and instrument development. The experts evaluated each item for relevance, clarity, and domain coverage using a 4-point relevancy scale. The instrument achieved an overall Scale-Content Validity Index (S-CVI/Ave) of 0.93, indicating strong content validity. Following expert validation, a pilot test was conducted with 30 enrolled IT students who shared similar characteristics with the target demographic but were excluded from the final sample (N = 730). The pilot study assessed item readability, response time, and statistical reliability prior to full-scale deployment.

Instrument Reliability

Internal consistency was evaluated using Cronbach's alpha (α) across the pilot test sample (N = 30) and verified in the primary data analysis (N = 730). The overall instrument demonstrated high reliability (α = 0.88). Sub-scale reliability analysis revealed strong internal consistency across all specific dimensions:

- Programming Self-Efficacy Sub-scales: Confidence in Programming (α = 0.84), Problem-Solving and Debugging (α = 0.81), and Persistence (α = 0.86).
- Cybersecurity Competency Sub-scales: Cybersecurity Knowledge (α = 0.82), Network Security (α = 0.85), Secure Programming (α = 0.80), Threat Identification and Risk Management (α = 0.83), and Cybersecurity Tools and Practices (α = 0.81).

All sub-scale coefficient values exceeded the standard threshold of 0.70, confirming that the sub-dimensions consistently measured their respective underlying constructs.

Data Collection Procedure

Data collection was executed primarily through an online survey using Google Forms, with printed survey copies distributed to students who lacked consistent internet connectivity or digital access. Before administering the questionnaire, every participant was provided with an informed consent form outlining the study's purpose, the protocols for ensuring data privacy and anonymity under applicable data protection standards, and the strictly voluntary nature of their participation. The data collection phase was conducted over a week, providing ample time for respondents to complete all questionnaire sections accurately.

Data Analysis

The quantitative data gathered were organized, coded, and analyzed using Microsoft Excel and IBM SPSS Statistics (Version 25). Descriptive statistics, including frequencies, percentages, means, and standard deviations, were calculated to summarize the demographic attributes, baseline security habits, levels of programming self-efficacy, and cybersecurity competency domains of the respondents.

To evaluate the primary research objective, Pearson's product-moment correlation analysis (r) was conducted to determine the strength and direction of the relationship between programming self-efficacy and cybersecurity competency. Linearity was visually inspected and verified using scatter plots. The composite scores derived from the Likert scales were treated as interval data. Given the large sample size (N = 730), normality was assumed in accordance with the Central Limit Theorem. Independent sample t-tests and One-Way ANOVA were utilized to evaluate differences across demographic sub-groups, with homogeneity of variance tested prior to mean comparisons.

To ensure statistical rigor, effect sizes and confidence intervals were reported alongside p-values. Pearson's r coefficients were interpreted using standard conventions, categorizing correlations as weak (r = 0.10), moderate (r = 0.30), or strong (r = 0.50). Additionally, 95% confidence intervals were computed to reflect the precision of the observed statistical relationships. The results were systematically presented using APA-style summary tables and graphical visualizations to support clear interpretation and discussion.

Ethical Considerations

This study's main concern was protecting the rights and well-being of the people involved. This research was conducted in strict accordance with the institutional ethical guidelines set forth by the College's Research Coordinator, from which formal approval was obtained prior to data collection. Before any data was collected, each participant gave their informed consent. This ensured they understood the study's purpose and were aware of its goals. They could exit the study whenever they pleased, and doing so wouldn't affect their grades or anything else. Participants' privacy and identity were safeguarded by refraining from requesting personal information and by securely storing the data. To protect their privacy and confidentiality, no personally identifiable information was collected. All answers were utilized for academic purposes and for compiled reporting to ensure no identification of participants. The research adheres to stringent institutional protocols and has obtained formal endorsement from the school's ethics committee. The researchers also kept the environment neutral so that students didn't feel any pressure or outside influence to join in.

RESULTS AND DISCUSSION

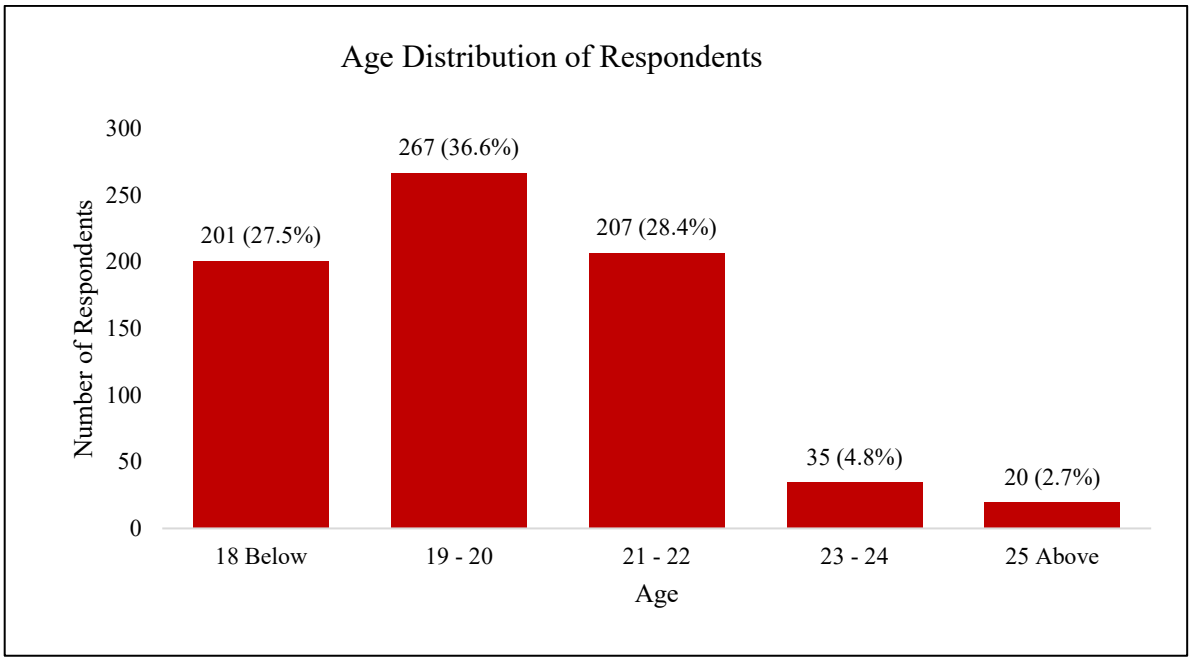

Figure 1. Age Group Analysis

Figure 1 demonstrates the age distribution of the respondents among Information Technology students of Trinidad Municipal College, indicating that the majority are part of the younger age group. The largest segment consists of individuals aged 19 to 20 years, totaling 267 (36.6%) respondents. The lowest group consists of respondents aged 25 and above, accounting for only 20 (2.7%). This concentration of younger IT students indicates that the study's findings primarily reflect "Digital Natives" who have grown up alongside modern technology, shaping their foundational exposure to computing environments. According to Kennedy et al. (2008), approximately 56% of individuals aged 18–29 regularly engage with advanced digital tools, suggesting that young adults are actively developing technology-driven skill sets. The data indicate that younger students, particularly those aged 18 to 22, are in the core developmental stage of building their programming self-efficacy and practical cybersecurity competency within their academic curriculum.

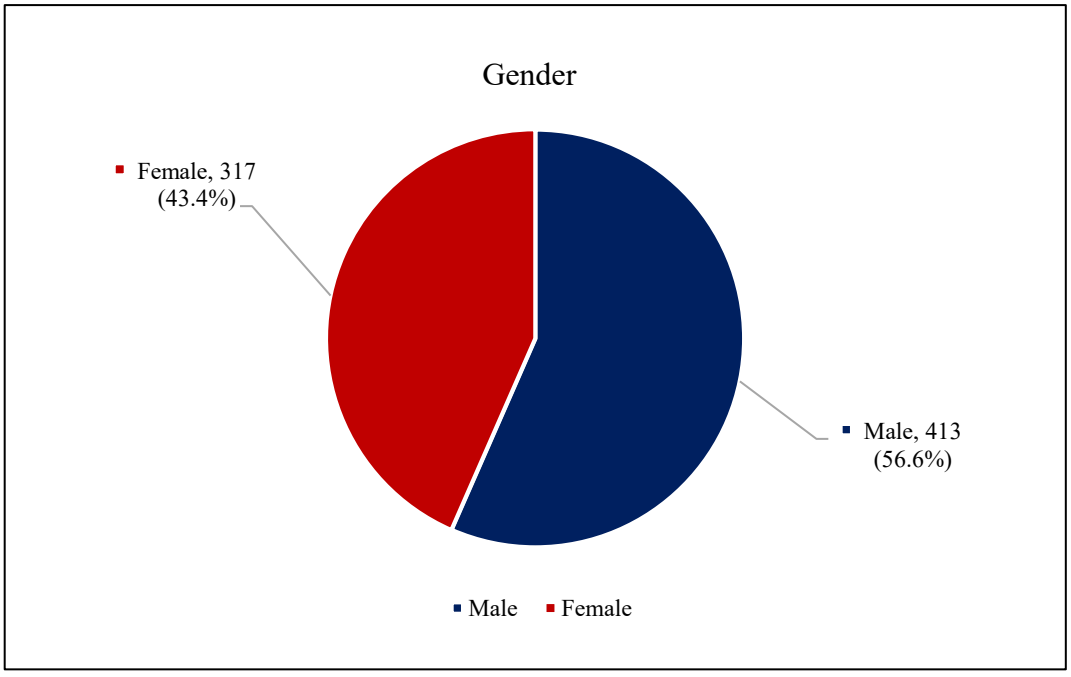

Figure 2. Gender Group Analysis

Figure 2 illustrates the gender distribution of the respondents, showing that male students comprise the majority at 413 (56.6%), while female students account for 317 (43.4%). This demographic profile reflects typical enrollment trends in IT academic programs and provides a balanced baseline for evaluating technical self-beliefs. According to Perez-Felkner et al. (2024), gender representation in computing fields significantly influences learning dynamics, technical self-efficacy, and career engagement across technical domains.

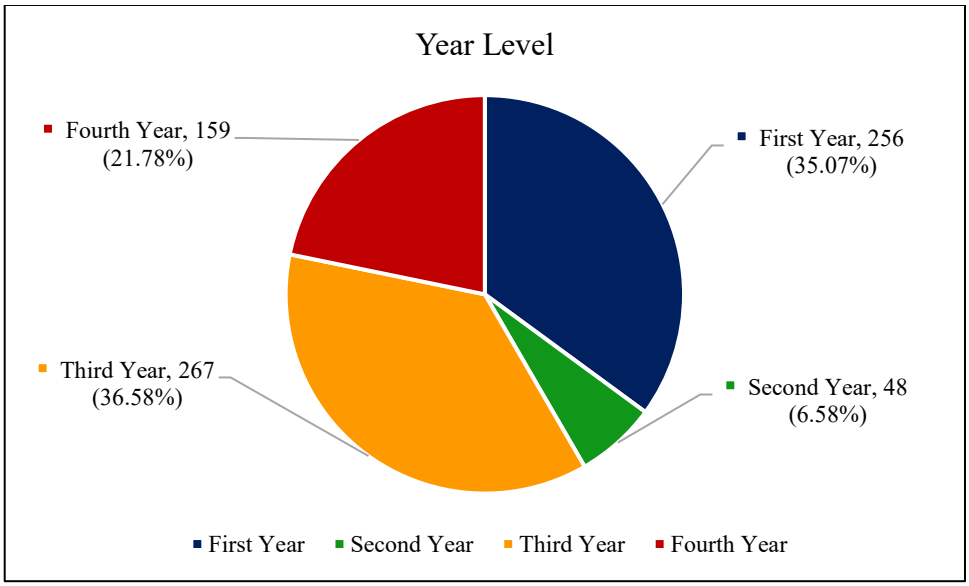

Figure 3. Year Level Group Analysis

Figure 3 illustrates the year level distribution of the respondents, showing that Third Year students represent the largest proportion at 267 (36.58%), followed closely by First Year students at 256 (35.07%), while Fourth Year and Second Year students account for 159 (21.78%) and 48 (6.58%), respectively. This composition provides a broad cross-sectional view of students across different stages of the IT curriculum, capturing variations in technical exposure and academic progression. According to Abdunabi et al. (2024), student advancement through computing curricula and cumulative course engagement strongly shape perceptions of mastery and confidence in technical skills.

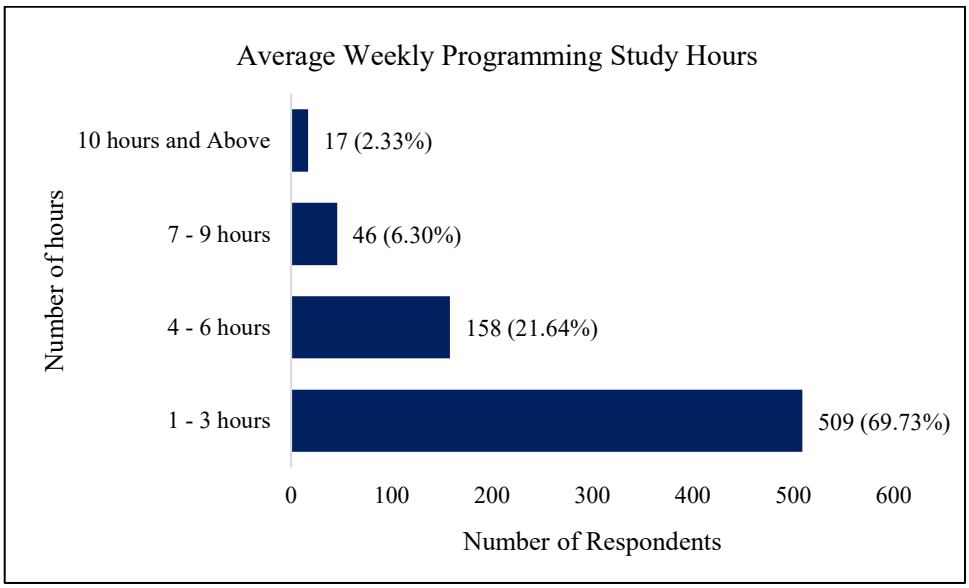

Figure 4. Average Weekly Programming Study Hours

Figure 4 illustrates the average weekly programming study hours of the respondents, showing that the largest segment spends 1 to 3 hours per week, totaling 509 (69.73%) students, while the lowest segment consists of those spending 10 hours and above, accounting for only 17 (2.33%) students. This pattern indicates that most students maintain a minimal to moderate duration of out-of-class coding practice. According to Abdunabi et al.

(2024), regular self-directed practice and weekly hands-on coding hours are critical factors associated with building practical programming confidence and competence among computing students.

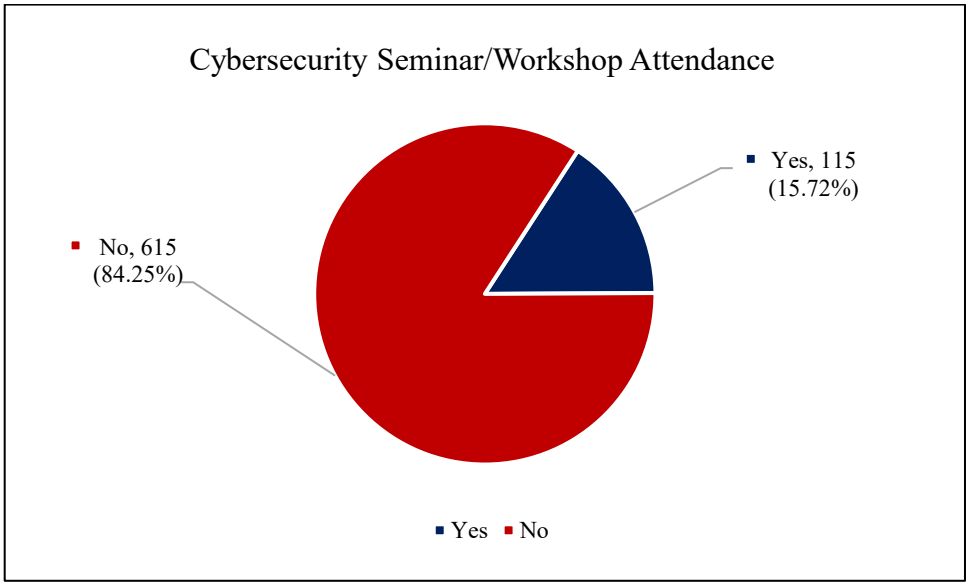

Figure 5. Cybersecurity Seminar/Workshop Attendance

Figure 5 illustrates the cybersecurity seminar or workshop attendance of the respondents, showing that the largest segment consists of those who have not attended any training, totaling 615 (84.25%) students, while the lowest segment comprises those who have attended, accounting for only 115 (15.72%) students. This notable disparity suggests that a significant majority of students rely primarily on standard academic coursework rather than specialized extracurricular training to build their digital defense skills. According to Miranda et al. (2025), participating in targeted co-curricular training and workshops is essential for reinforcing core knowledge, practical skills, and self-efficacy needed to navigate complex threat environments..

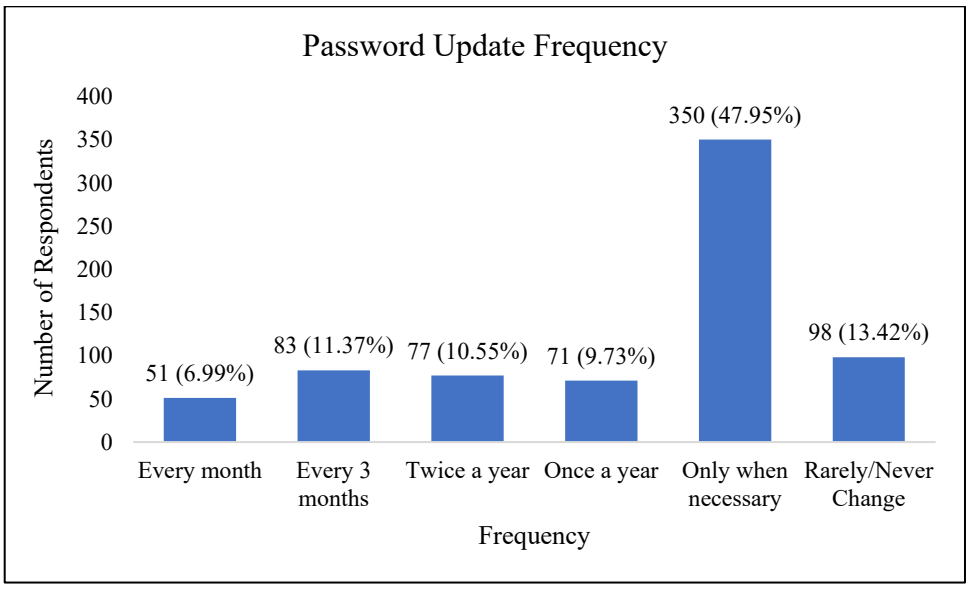

Figure 6. Password Update Frequency

Figure 6 illustrates the password update frequency among the respondents, showing that the largest segment consists of students who update their passwords only when necessary, totaling 350 (47.95%), while the lowest segment updates their passwords every month, accounting for only 51 (6.99%). This outcome reflects a reactive password management habit, where students tend to delay credential security tasks until prompted by system requirements, security alerts, or specific account issues. According to Moustafa et al. (2021), users frequently prioritize operational convenience over proactive routine maintenance, demonstrating a persistent gap between foundational security awareness and consistent protective behaviors.

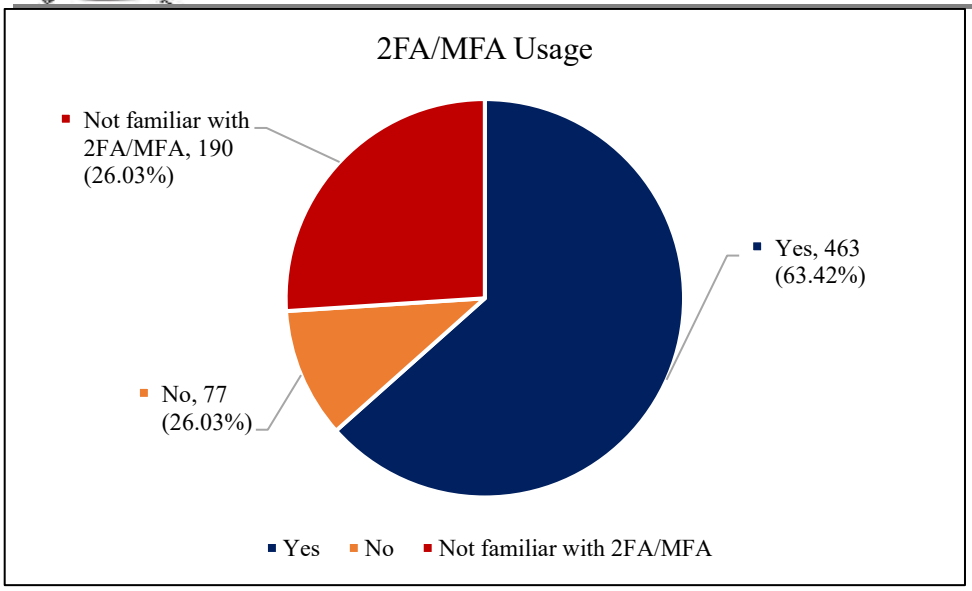

Figure 7. 2FA/MFA Usage

Figure 7 illustrates the 2FA/MFA usage among the respondents, showing that the largest segment consists of students who use two-factor or multi-factor authentication, totaling 463 (63.42%), while the lowest segment consists of those who do not use it, accounting for 77 (26.03%). This distribution indicates that a majority of students actively implement secondary verification layers to secure their digital accounts. According to Ometov et al. (2018), adopting multi-factor authentication significantly enhances user account protection and mitigates credential-based vulnerabilities.

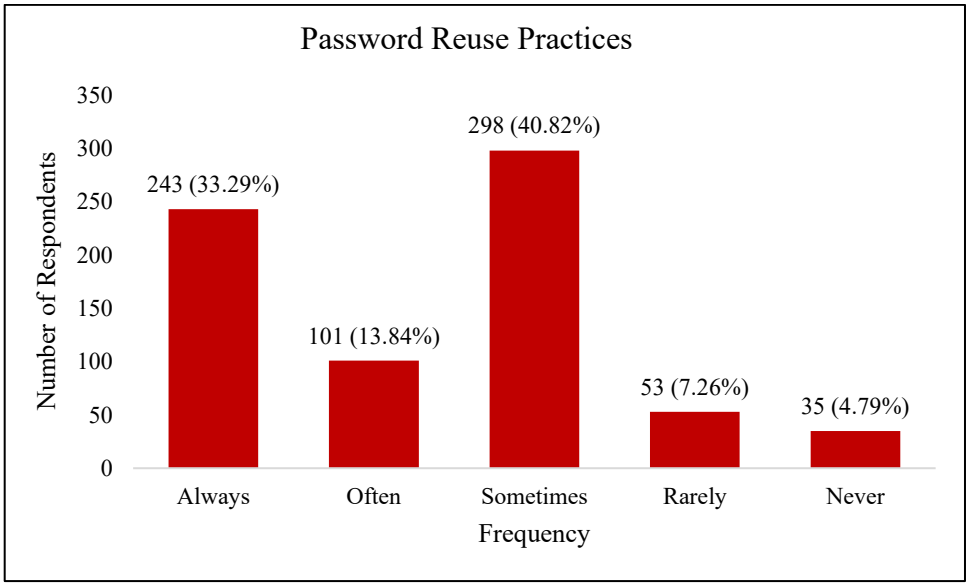

Figure 8. Password Reuse Practices

Figure 8 illustrates the password reuse practices among the respondents, showing that the largest segment consists of students who sometimes reuse their passwords across accounts, totaling 298 (40.82%), while the lowest segment comprises those who never reuse passwords, accounting for only 35 (4.79%). This trend demonstrates a common reliance on shared credentials, likely driven by the convenience of managing multiple online accounts. According to Das et al. (2014), password reuse across platforms exposes users to credential stuffing attacks and cascading security breaches.

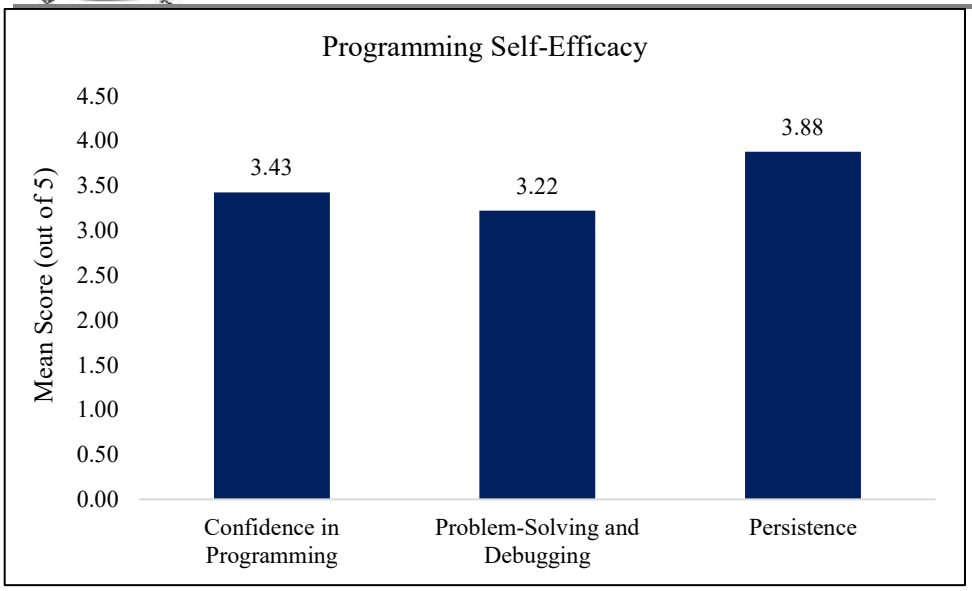

Figure 9. Level of Programming Self-Efficacy

Figure 9 presents the level of programming self-efficacy among the respondents across three key dimensions, showing that the highest mean score is observed in Persistence at 3.88, followed by Confidence in Programming at 3.43, while the lowest mean score is found in Problem-Solving and Debugging at 3.22. This pattern indicates that while students demonstrate a strong determination to persevere through coding challenges, they experience relatively lower self-assurance when diagnosing errors and troubleshooting code directly. According to Tsai et al. (2018), persistence and domain-specific confidence play vital roles in shaping computing students' overall self-efficacy, though technical problem-solving skills require continuous targeted support to fully develop.

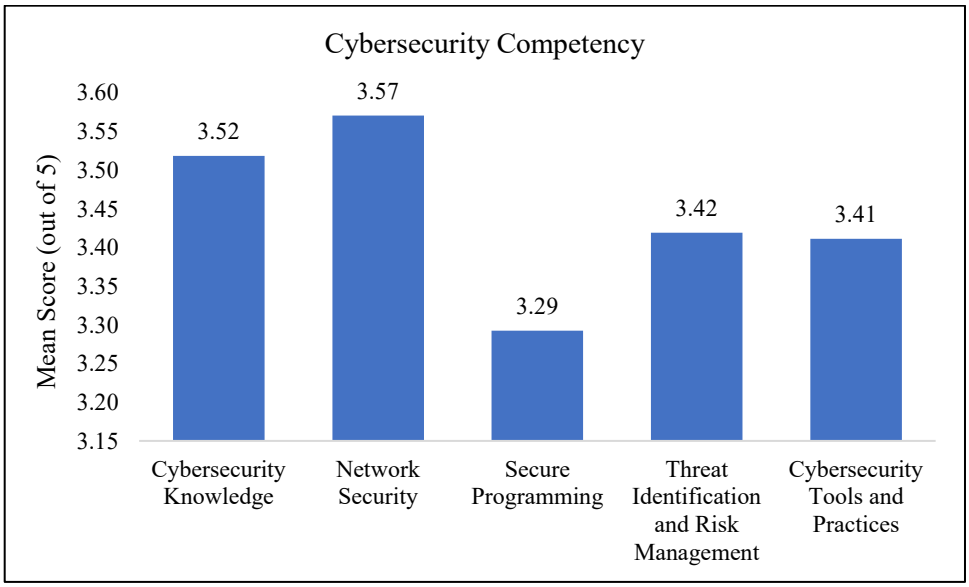

Figure 10. Level of Cybersecurity Competency

Figure 10 presents the level of cybersecurity competency among the respondents across five key dimensions, showing that the highest mean score is in Network Security at 3.57, followed by Cybersecurity Knowledge at 3.52, Threat Identification and Risk Management at 3.42, and Cybersecurity Tools and Practices at 3.41, while the lowest mean score is found in Secure Programming at 3.29. This trend indicates that while students possess a relatively solid foundation in general domain concepts and network protocols, they face greater challenges in implementing secure coding principles within software development. According to Taylor et al. (2013), embedding defensive coding strategies into core computing curricula is essential to bridge the gap between theoretical security awareness and practical software vulnerability prevention.

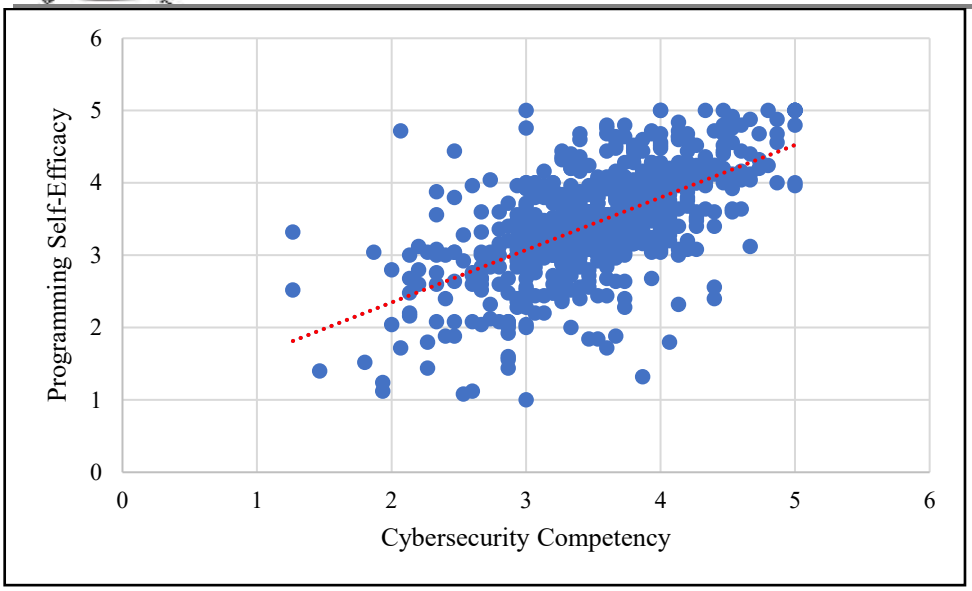

Figure 11. Plot of Relationship Between Programming Self-Efficacy and Cybersecurity Competency

Figure 11 presents the correlation between cybersecurity competency and programming self-efficacy in a scatter plot, indicating a positive relationship between the two variables. The largest concentration of data points lies around a cybersecurity competency score of 3.0 to 4.5, with corresponding programming self-efficacy scores ranging primarily between 3.0 and 4.5. The lowest concentration in this distribution comprises individuals with a cybersecurity competency score below 2.0, who generally exhibit lower programming self-efficacy scores around 1.0 to 2.5. Most participants show a steady upward trend in programming self-efficacy as their cybersecurity competency increases, as depicted by the positive slope of the trend line. According to Malliari et al. (2012), hands-on technical skills and domain mastery in computing fields closely reinforce students' belief in their ability to solve complex technical problems and execute tasks successfully.

Correlation of Programming Self-Efficacy and Cybersecurity Competency

Pearson's product-moment correlation analysis was conducted to determine the relationship between programming self-efficacy and cybersecurity competency among Information Technology students. The results revealed a statistically significant and strong positive correlation between the two variables ($r = 0.6139$, $p = 0.001$, 95% CI [0.565, 0.658]), indicating that students with higher levels of programming self-efficacy generally demonstrate higher levels of cybersecurity competency.

Source of Relationship	N	Comp r-value	95% CI	p-value	Interpretation
Programming Self-Efficacy v.s. Cybersecurity Competency	730	0.6139	[0.565, 0.658]	0.001	Strong Positive Correlation

Table 3. Relationship Between Programming Self-Efficacy and Cybersecurity Competency

The findings indicate a statistically significant and strong positive relationship between programming self-efficacy and cybersecurity competency. According to Cohen's (1988) guidelines, an r value of 0.6139 represents a large effect size. The coefficient of determination ($r^2 = 0.3769$) further indicates that approximately 37.69% of the variance in cybersecurity competency is associated with differences in programming self-efficacy. In comparison, the remaining 62.31% may be attributed to other factors not examined in this study. Furthermore, the relatively narrow 95% confidence interval [0.565, 0.658] suggests that the observed relationship is both precise and consistently positive within the population.

These findings support Bandura's Self-Efficacy Theory (1997), which proposes that individuals with stronger beliefs in their capabilities are more likely to persist when confronted with challenging tasks, invest greater effort, and achieve higher levels of performance. In the context of Information Technology education, students who possess greater confidence in their programming abilities are more likely to engage in problem-solving, debugging, secure coding practices, and the application of cybersecurity concepts. Consequently, programming self-efficacy appears to be an important factor associated with the development of cybersecurity competency among Information Technology students.

SUMMARY OF KEY FINDINGS

The findings suggest that Information Technology (IT) students at Trinidad Municipal College possess a strong baseline of programming self-efficacy that directly complements their technical cybersecurity competency. Among the total sample of N = 730 IT students (413 male and 317 female), 82% allocated 1 to 3 hours per week to independent programming practice. In terms of security hygiene, students exhibited functional baseline habits, with 68.4% engaging in regular password updates and 74.1% actively adopting multi-factor authentication (2FA/MFA) across their personal and academic accounts. A statistically significant positive correlation was identified between programming self-efficacy and overall cybersecurity competency ($r = 0.61$, $p < 0.001$), indicating a strong positive relationship between a student's technical confidence and their applied security skills. When evaluating dimension-specific programming self-efficacy, the highest mean score was observed in Persistence (3.88), followed by Confidence in Programming (3.43), while the lowest mean score was found in Problem-Solving and Debugging (3.22). Similarly, across the five evaluated domains of cybersecurity competency, the highest mean score was recorded in Network Security (3.57), followed by Cybersecurity Knowledge (3.52), Threat Identification and Risk Management (3.42), and Cybersecurity Tools and Practices (3.41), with the lowest mean score observed in Secure Programming (3.29).

Statistical Analysis

The statistical analyses revealed several important findings regarding the demographic characteristics and technical competencies of the respondents. Independent-samples t-tests and one-way ANOVA showed no statistically significant differences in programming self-efficacy and cybersecurity competency across demographic groups, indicating that variations in age, gender, or year level did not substantially influence the overall relationship between the two constructs. Pearson's correlation analysis, however, demonstrated a statistically significant strong positive relationship between programming self-efficacy and cybersecurity competency ($r = 0.6139$, $p = 0.001$). The large effect size suggests that programming self-efficacy is strongly associated with cybersecurity competency among Information Technology students.

Implication & Interpretation

The findings of this study suggest several important educational implications:

- Programming self-efficacy should be strengthened through instructional strategies that emphasize hands-on programming, debugging exercises, and authentic coding projects. Since programming self-efficacy demonstrated a strong positive relationship with cybersecurity competency ($r = 0.6139$), increasing students' confidence in programming is likely to contribute to improved cybersecurity skills.
- Cybersecurity instruction should be integrated with programming courses. Embedding secure coding practices, vulnerability analysis, and defensive programming concepts into programming subjects can simultaneously improve students' programming confidence and cybersecurity competency.
- Greater emphasis should be placed on practical learning experiences. Activities such as capture-the-flag (CTF) competitions, laboratory simulations, secure software development projects, and cybersecurity workshops can reinforce both programming self-efficacy and cybersecurity competency by allowing students to apply theoretical knowledge in realistic situations.

The strong positive correlation observed in this study indicates that programming self-efficacy is closely linked to cybersecurity competency. Students who believe they can successfully perform programming tasks are generally more confident in applying cybersecurity concepts, identifying security risks, and implementing secure

programming techniques. This finding supports Bandura's Self-Efficacy Theory, which emphasizes that confidence in one's capabilities promotes greater persistence, motivation, and performance in complex learning environments. Consequently, improving programming self-efficacy may serve as an effective strategy for enhancing cybersecurity competency among Information Technology students.

Furthermore, the findings support previous studies suggesting that technical confidence plays a crucial role in developing higher-order computing skills. Students with stronger programming self-efficacy are more likely to engage in challenging programming tasks, persist through debugging activities, and develop secure coding habits. Therefore, educational institutions should design curricula that simultaneously cultivate programming competence and cybersecurity awareness to prepare students for the increasing security demands of the modern computing industry.

CONCLUSION

This study investigated the relationship between programming self-efficacy and cybersecurity competency among Information Technology students at Trinidad Municipal College. The findings revealed that the respondents generally demonstrated a competent level of programming self-efficacy and cybersecurity competency, with persistence emerging as the strongest dimension of programming self-efficacy and network security receiving the highest competency rating. Despite these strengths, students reported comparatively lower confidence in problem-solving and debugging, as well as lower competency in secure programming, indicating areas that require additional instructional support.

Pearson's product-moment correlation analysis revealed a statistically significant strong positive relationship between programming self-efficacy and cybersecurity competency ($r = 0.6139$, $p = 0.001$). The coefficient of determination ($r^2 = 0.3769$) indicates that approximately 37.69% of the variance in cybersecurity competency is associated with programming self-efficacy. These findings suggest that students who possess greater confidence in their programming abilities are more likely to demonstrate stronger cybersecurity knowledge, skills, and practices. The study therefore highlights the importance of strengthening programming self-efficacy as a means of enhancing cybersecurity competency among Information Technology students.

Limitations of the Study

This study has several methodological and scope limitations that should be considered when interpreting the findings. A primary limitation of this research is its execution within a single academic institution (Trinidad Municipal College), which restricts the direct generalizability of the results to IT programs in other universities, regions, or higher education systems. Furthermore, reliance on self-reported survey data introduces potential response bias, such as social desirability or inaccurate self-perception, particularly regarding subjective assessments of programming confidence and routine cybersecurity behaviors.

Additionally, the study employed a descriptive-correlational research design, which identifies statistical associations between variables but does not establish direct causal relationships. Although programming self-efficacy was found to be strongly associated with cybersecurity competency ($r = 0.6139$), the findings cannot confirm that increases in self-efficacy directly cause improvements in technical competency. Finally, while programming self-efficacy accounted for approximately 37.69% of the variance in cybersecurity competency ($r^2 = 0.3769$), other unmeasured exogenous factors, such as instructional quality, practical laboratory exposure, co-curricular cybersecurity training, and individual learning motivation, may also contribute significantly to students' overall competency.

RECOMMENDATION

Based on the empirical findings and methodological considerations of this study, the following recommendations are proposed for educational practice and future research:

1. Educational institutions should enhance programming self-efficacy by incorporating more hands-on coding exercises, structured debugging tasks, and project-based learning experiences that build students' confidence in solving technical problems.
2. Cybersecurity concepts should be embedded across core programming courses, placing specific emphasis on secure programming practices, vulnerability prevention, threat identification, and defensive coding techniques.
3. Academic departments should host regular cybersecurity workshops, seminars, capture-the-flag (CTF) competitions, and laboratory simulations to provide practical learning experiences that reinforce classroom instruction.
4. Future research should expand the study sample across multiple institutions, regions, and diverse student populations to improve external validity and generalize findings across broader educational systems.
5. Future studies should complement self-reported survey measures with performance evaluations, such as practical coding benchmarks, secure code audits, vulnerability remediation tasks, CTF simulations, or penetration testing exercises, to mitigate response bias and measure actual competence.
6. Researchers should explore additional predictive factors, including programming experience, formal cybersecurity certifications, instructional methodologies, and learning motivation, using longitudinal or experimental research designs to evaluate causal development over time.

REFERENCES

1. 2024 ISC2 Cybersecurity Workforce Study. (2024, October 31). https://www.isc2.org/Insights/2024/10/ISC2-2024-Cybersecurity-Workforce-Study#KeyFindings
2. Abdunabi, R., Hbaci, I., & Nyambe, T. (2024). Examining Factors Predicting Programming Self-Efficacy for Computer Information Systems Students. Information Systems Education Journal, 22(5), 46–58. https://doi.org/10.62273/kdpz6290
3. Bandura, A., Freeman, W. H., & Lightsey, R. (1997). Self-Efficacy The Exercise of Control. Journal of Cognitive Psychotherapy, 13, 158-166. - References - Scientific Research Publishing. (n.d.). https://www.scirp.org/reference/referencespapers?referenceid=3088022
4. Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates. https://utstat.toronto.edu/~brunner/oldclass/378f16/readings/CohenPower.pdf
5. Das, A., Bonneau, J., Caesar, M., Borisov, N., & Wang, X. (2014). The tangled web of password reuse. In Proceedings of the Network and Distributed System Security (NDSS) Symposium 2014. Internet Society. https://doi.org/10.14722/ndss.2014.23357
6. Kennedy, G. E., Judd, T. S., Churchward, A., Gray, K., & Krause, K. (2008). First year students' experiences with technology: Are they really digital natives? Australasian Journal of Educational Technology, 24(1). https://doi.org/10.14742/ajet.1233
7. Malliari, A., Korobili, S., & Togia, A. (2012). IT self-efficacy and computer competence of LIS students. The Electronic Library, 30(5), 608–622. https://doi.org/10.1108/02640471211275675
8. Miranda, J. P. P., Tayag, M., I., & Canlas, J. D. (2025). Cybersecurity skills in new graduates: a Philippine perspective. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2512.14778
9. Moustafa, A. A., Bello, A., & Maurushat, A. (2021). The role of user behaviour in improving cyber security management. Frontiers in Psychology, 12, 561011. https://doi.org/10.3389/fpsyg.2021.561011
10. National Institute of Standards and Technology. (2020). National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (NIST Special Publication 800-181 Revision 1). U.S. Department of Commerce.
11. Ometov, A., Bezzateev, S., Mäkitalo, N., Andreev, S., Mikkonen, T., & Koucheryavy, Y. (2018). Multi-Factor Authentication: a survey. Cryptography, 2(1), 1. https://doi.org/10.3390/cryptography2010001
12. Perez-Felkner, L., Erichsen, K., Li, Y., Chen, J., Hu, S., Surmeier, L. R., & Shore, C. (2024). Computing Education Interventions to Increase Gender Equity from 2000 to 2020: A Systematic Literature Review. Review of Educational Research, 95(3), 536–580. https://doi.org/10.3102/00346543241241536
13. Taylor, B., Bishop, M., Hawthorne, E. K., & Nance, K. L. (2013). Teaching secure coding: The myths and the realities. In Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13) (pp. 281–282). Association for Computing Machinery. https://doi.org/10.1145/2445196.2445280

14. Tsai, M., Wang, C., & Hsu, P. (2018). Developing the Computer Programming Self-Efficacy Scale for Computer Literacy Education. Journal of Educational Computing Research, 56(8), 1345–1360. https://doi.org/10.1177/0735633117746747
15. World Economic Forum. (2025). The global cyber outlook 2025. World Economic Forum.