PANTAO
The International Journal of the Humanities and Social Sciences
Volume 5, Issue 1
ISSN 3028-0877 | DOI 10.69651/2022PIJHSS

Integrated analysis between prerequisite subject proficiency and advanced programming course performance of computer engineering students at Bulacan State University

Arvin N. Migallos*
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: arvinmigallos05@gmail.com

Dionel SM. De Guzman
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: diondeguzman0@gmail.com

Lorence N. Hernandez
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: lorencehernande07@gmail.com

Coleen DC. Deliguer
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: deliguercoleensuexx@gmail.com

Engr. Alexander M. Aquino
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: alexander.aquino@bulsu.edu.ph

Dr. Lech Walesa M. Navarra
Bulacan State University – Main Campus
Malolos, Bulacan, Philippines.
Email: lechwalesa.navarra@ms.bulsu.edu.ph

ABSTRACT

The study analyzes the relationship between prerequisite subject proficiency and academic performance in advanced programming courses among fourth-year Computer Engineering students at Bulacan State University. The study employs a mixed-methods explanatory sequential design to systematically compare quantitative data from students' academic records with qualitative insights obtained through self-evaluation surveys and their responses to open-

*Corresponding author
DOI: http://doi.org/10.69651/PIJHSS0501864

Recommended citation:
Migallos, A. N., De Guzman, D. S., Hernandez, L. N., Deliguer, C. D., Aquino, A. M., & Navarra, L. W. M. (2026). Integrated analysis between prerequisite subject proficiency and advanced programming course performance of computer engineering students at Bulacan State University. Pantao (The International Journal of the Humanities and Social Sciences) 5 (1), 9724-9731. http://doi.org/10.69651/PIJHSS0501864

ended questionnaires. This integrated data gathering allows the researchers to investigate the connection between the respondents' academic trends. The quantitative analysis found a strong, significant positive correlation ($r = 0.74$, $p < 0.05$) between students' grades in foundational programming courses and their success in advanced programming subjects. These findings mean that the level of understanding and application in prerequisite subjects reflects and does have an impact on their advanced subject performances. The researchers also found out that most of the students identified "Data Structures and Algorithms" as a particularly challenging subject that requires them a deeper mastery of abstract reasoning and algorithmic thinking. Furthermore, the qualitative findings highlight structural and instructional factors, including frequent switching of programming languages, inconsistent teaching methods, and outdated laboratory resources, as additional barriers impacting learning outcomes of students. Through these results, the researchers conclude that curricular changes and reforms are essential to enhance vertical alignment and depth in prerequisite programming knowledge. Recommendations include intensified focus on algorithmic logic development, improved faculty coordination, early identification of at-risk students, and investment in laboratory infrastructure. The findings of the research provide actionable insights for students, educators, and administrators to strengthen programming competence and academic success within the Computer Engineering program.

Keywords: Prerequisites, subject proficiency, academic performance, programming subjects.

Date Submitted: January 30, 2026
Date Accepted: February 10, 2026
Date Published: February 28, 2026

INTRODUCTION

Prerequisites are courses required to be passed before taking advanced subjects in universities. These requirements make sure that students have a basic understanding of what will be taught in the succeeding subjects, using the skills they acquired from preceding ones. However, some courses have prerequisites that either demands knowledge yet to be instructed or completely unrelated to the current subjects that the students take, especially in computer programming as there is not one language to study and master.

According to Valstar et al. (2019), while prerequisites layout a foundation for future lessons, less is known about how it sets up students for success and affect their performance in later courses. It is unclear if a student's proficiency in prior topics influences their performance in the subsequent subjects. For computer engineers, mastering only one programming language may not translate to equal understanding when introduced to another style of coding.

The study from Dita and Velasco (2025) pointed out performance gaps within students who do not have an equal depth in their foundational subjects. While grades are a quantitative measure of understanding, these does not predict that students will yield the succeeding subjects aligned with prior courses. Along with hindering factors, this creates a performance gap among students where they struggle to apply what they have learned from their prerequisites.

The researchers seek to find out a definitive relationship between computer engineering students' prerequisite course proficiency to their success in advanced programming subjects while addressing challenges and various hindrances in learning. The study seeks to provide insights and recommendations to potentially fill this gap by examining the relationship between prerequisite subject proficiency and advanced programming course outcomes among Computer Engineering students at Bulacan State University.

Statement of the problem

1. To be able to determine the grades of the fourth-year Computer Engineering students in their specific prerequisite and advanced programming subjects.
2. To be able to analyze if there is a significant relationship between the students' grades in prerequisite subjects and their performance in advanced programming courses.
3. To be able to assess the proficiency level of the fourth-year Computer Engineering students in prerequisite subjects based on their academic marks and self-assessment.
4. To be able to identify the technical challenges students face in advanced programming linked to potential gaps in foundational knowledge.

METHODOLOGY

For the research design of the study, it utilized a mixed-methods explanatory sequential design which allowed the researchers to analyze both quantitative and qualitative data more extensively to come up with deeper insights and concrete conclusions. Additionally, according to Oranga (2025), using this method allows the researchers to have further understanding on their study as it uses the traits of both qualitative and quantitative research designs providing a better understanding of the research problem, comprehensive data validation, and attain more contextualized data.

The target population of the study mainly focused on the two hundred thirty (230) fourth- year Computer Engineering students enrolled in advanced programming courses during academic years 2022–2023 through 2024–2025 at Bulacan State University. From this population, the researchers selected fifty (50) students, using purposive sampling technique, to serve as the sample size of the study. The selected respondents then participated in a combination of quantitative and qualitative assessments which includes self-rating survey and open-ended questions about technical challenges that they experience during their studies.

The researchers administered questionnaires that consists of several structured sections designed to track the academic history and the lived experiences of the fourth-year Computer Engineering students. The first section required participants to access the university's official student portal to ensure the accuracy of the grades provided for both prerequisite and advanced programming subjects. The second part featured a 5-point Likert scale which was used to determine the level of self-perceived proficiency in core learning areas such as logic formulation, syntax, and algorithmic thinking. The questionnaire underwent content validation by two (2) experts: (1) a research professional and licensed teacher, and (2) a certified engineer.

The researchers began the process of collecting data by distributing the surveys to the participants face-to-face to maintain a high level of response integrity. All data gathered from these questionnaires were tallied and computed for interpretation, serving as the basis to determine the correlation between foundational knowledge and advanced subject performance.

For the quantitative data analysis, the researchers used descriptive statistics to calculate the average grades and distributions of the participant. The researchers then used the weighted mean for each subject and tested the degree of association between prerequisite and advanced course grades using the Pearson correlation coefficient. Statistical significance was assessed at the alpha level of 0.05.

On the other hand, for the qualitative data, the researchers used thematic analysis to identify recurring technical challenges and contextual factors influencing the participants'

academic performance. The researchers categorized their responses into separate themes such as logic formulation, syntax complexity, debugging difficulties, and time management.

The survey did not require the students to put their names, sections, and/or other private information irrelevant to the study to ensure that their identities remain anonymous. The researchers also briefed and informed the respondents, that they participation in the survey is voluntary and that the information that they write will only be used solely for the purpose of the study.

RESULTS AND DISCUSSION

The present study involved a total of 50 fourth-year Computer Engineering students whose academic performance and learning experiences were examined using a mixed-methods approach that integrated both quantitative and qualitative data. The quantitative component utilized weighted mean computations and Pearson product–moment correlation analysis, while the qualitative component employed thematic analysis to identify recurring challenges. Data were collected through academic records and a structured self-assessment instrument, ensuring that the interpretation of results remained grounded in the data gathered and aligned directly with the objectives of determining the relationship between prerequisite subject proficiency and performance in advanced programming courses.

The analysis of academic performance in prerequisite subjects revealed that the respondents demonstrated a consistently high level of achievement, with an overall weighted mean of 1.86 interpreted as Very Satisfactory. Specifically, the weighted mean for CPE 102L/Computer Fundamentals and Programming was 1.87, while CPE 104L/Programming Logic and Design obtained a weighted mean of 1.85, both interpreted as Very Satisfactory. These results indicate that the students possessed a strong foundational understanding of programming concepts prior to enrolling in advanced courses. The closeness of the weighted mean values suggests uniformity in performance across foundational subjects, reflecting effective curriculum delivery and alignment with expected learning outcomes. This finding supports the premise that a well-structured foundational curriculum plays a critical role in preparing students for higher-level programming tasks, consistent with established principles in engineering education that emphasize the importance of prerequisite mastery.

In contrast, the analysis of academic performance in advanced programming subjects revealed a slight decline in overall achievement, with an overall weighted mean of 1.97, still interpreted as Very Satisfactory. Among the subjects, CPE 304L/Software Design Lab recorded the highest performance with a weighted mean of 1.59, interpreted as Very Satisfactory, while CPE 202L/Object Oriented Programming (2.12), CPE 206L/Data Structures and Algorithms (2.13), and CPE 209L/Database Management System (2.02) were all interpreted as Satisfactory. The highest weighted mean value of 2.13 in Data Structures and Algorithms indicates relatively lower performance compared to other subjects, suggesting that it posed the greatest level of difficulty among the advanced courses. The observed variation in performance reflects the increasing complexity of advanced programming subjects, where students are required to apply abstract reasoning, algorithmic thinking, and multi-layered problem-solving skills. This pattern is consistent with Cognitive Load Theory, which posits that learners may experience cognitive overload when task complexity exceeds their working memory capacity, particularly when foundational knowledge is not fully automated.

The relationship between prerequisite and advanced programming performance was further examined using Pearson product–moment correlation. The results revealed a Pearson r value of 0.74 with a corresponding p-value of 0.000, indicating a strong positive correlation that is statistically significant at the 0.05 level of significance ($p<0.05$). This finding leads to

the rejection of the null hypothesis and confirms that there is a significant relationship between students' academic performance in prerequisite subjects and their performance in advanced programming courses. The positive direction of the correlation implies that higher achievement in foundational programming subjects is associated with higher performance in advanced courses. This result underscores the predictive value of prerequisite subject proficiency and highlights the importance of ensuring strong foundational competencies as a determinant of success in more complex programming domains.

The self-assessment of programming mastery further supports the quantitative findings, with an overall weighted mean of 3.66 interpreted as Proficient. Among the assessment items, Problem-Solving and Algorithmic Thinking obtained the highest weighted mean of 3.79, followed by Logic Formulation and Flowcharting at 3.61, and Syntax and Basic Commands at 3.58, all interpreted as Proficient. These results indicate that while students perceive themselves as capable in conceptual and analytical aspects of programming, they experience relatively lower confidence in technical syntax and command execution. The disparity between conceptual understanding and technical execution suggests that students may possess higher-order thinking skills but encounter challenges in translating these into precise programming language constructs. This aligns with existing literature that differentiates between declarative and procedural knowledge in programming, where learners may understand what needs to be done but struggle with how to implement it in code.

The qualitative findings provide deeper insight into the specific challenges encountered by students in advanced programming courses. Thematic analysis revealed that Logic Formulation was the most frequently cited challenge, with a frequency of 28 out of 50 respondents, equivalent to 56%. This indicates that more than half of the participants experienced difficulty in translating problems into algorithms and constructing appropriate flowcharts. Syntax Complexity was identified by 14 respondents, representing 28%, highlighting challenges related to advanced library functions and language-specific rules. Debugging Errors were reported by 5 respondents or 10%, reflecting difficulties in identifying and resolving logical and runtime errors, while Time Management was cited by 3 respondents or 6%, indicating constraints in completing complex programming tasks within given time frames. These findings demonstrate that the primary difficulties lie not only in technical execution but also in the cognitive processes involved in problem decomposition and algorithm design.

Furthermore, the qualitative data suggest that external and contextual factors also influence learning outcomes. The presence of frequent switching between programming languages, inconsistent teaching methodologies, and outdated laboratory resources were identified as contributing factors that affect students' ability to consolidate learning. Additionally, the reported over dependency on AI tools for code generation indicates a potential gap in conceptual understanding, as reliance on automated solutions may limit opportunities for active problem-solving and deep learning. These contextual elements provide a broader perspective on the observed performance patterns, indicating that while the curriculum structure is fundamentally sound, its implementation and the learning environment require reinforcement to ensure that foundational knowledge is effectively sustained and applied in advanced contexts.

In synthesis, the findings of this study demonstrate that the 50 fourth-year Computer Engineering students exhibited strong foundational proficiency in prerequisite subjects, as evidenced by an overall weighted mean of 1.86, which significantly influenced their performance in advanced programming courses, reflected by an overall weighted mean of 1.97. The statistically significant strong positive correlation ($r=0.74$, $p=0.000$, $p<0.05$) confirms that prerequisite subject proficiency is a key predictor of success in advanced programming. While

students perceive themselves as proficient in problem-solving and algorithmic thinking (3.79) and overall programming mastery (3.66), they continue to encounter challenges in logic formulation (56%), syntax complexity (28%), debugging (10%), and time management (6%). These results highlight the interplay between foundational knowledge, cognitive demands, and contextual learning conditions. The study contributes to the field by providing empirical evidence on the importance of strengthening foundational programming competencies and addressing both cognitive and environmental barriers to learning. The findings provide a clear basis for developing targeted instructional strategies and curriculum enhancements, thereby setting the stage for subsequent discussions on recommendations and practical interventions to improve student outcomes in advanced programming education.

CONCLUSION

The findings of the study demonstrate that the academic performance of the 50 fourth year Computer Engineering students is strongly influenced by their level of proficiency in prerequisite programming subjects. The results revealed a strong positive correlation of $r = 0.74$ with a corresponding p value of 0.000, indicating that students who performed well in foundational courses such as Computer Fundamentals and Programming Logic and Design were more likely to achieve higher performance in advanced programming subjects including Data Structures and Algorithms and Object Oriented Programming. This statistically significant relationship confirms that prerequisite knowledge serves as a critical determinant of success in higher level programming courses and underscores the essential role of foundational learning in shaping academic outcomes. The consistency between descriptive and inferential findings further validates that strong academic preparation in earlier courses contributes meaningfully to students' ability to manage the increased complexity of advanced programming tasks.

However, the qualitative findings provide an important nuance to these results by revealing that high academic performance does not always equate to genuine mastery of programming concepts. Despite achieving Very Satisfactory grades, some students reported difficulties in fully understanding the underlying principles of programming, particularly in areas such as logic formulation and algorithm development. This suggests that surface level achievement may mask gaps in deeper conceptual understanding, which can become more evident as students progress to more complex coursework. Furthermore, the data from the open ended survey responses highlighted a growing reliance on artificial intelligence tools for code generation. While such tools may support task completion and contribute to satisfactory academic performance, they may also limit opportunities for active learning and critical thinking if used excessively without sufficient conceptual engagement. These findings indicate that while prerequisite mastery is strongly associated with success, the quality and depth of that mastery remain equally important in ensuring sustained academic development.

In light of these findings, the study emphasizes the need for students to prioritize a deeper understanding of programming logic rather than focusing solely on producing functional code. Developing strong conceptual foundations in how programming languages operate, beyond memorizing syntax, will enable students to adapt more effectively to different programming environments and languages. For instructors, it is recommended that they assess students' prior academic performance at the beginning of each semester to identify those who may require additional support. By doing so, instructors can implement targeted instructional strategies and provide appropriate interventions that address gaps in foundational knowledge. This approach allows for a more responsive and adaptive teaching process that aligns with the diverse learning needs of students.

At the institutional level, the findings highlight the importance of ensuring that the curriculum provides sufficient time and emphasis for mastering fundamental programming concepts. Enhancing laboratory facilities and updating software resources to reflect current industry standards are also essential in supporting effective learning experiences. Additionally, maintaining consistency in the programming languages used across different courses may help students build fluency and reduce cognitive overload associated with frequent transitions between languages. These improvements in curriculum design and learning environment can strengthen the connection between foundational knowledge and advanced application.

Finally, the study recognizes the need for further research to expand and deepen understanding of programming education. Future studies are encouraged to include a larger and more diverse sample drawn from multiple campuses to improve the generalizability of findings. Investigating the impact of artificial intelligence tools on students' learning processes and conceptual development is also recommended, as this emerging factor plays an increasingly significant role in programming education. Such research can provide valuable insights that inform more effective teaching strategies and support systems.

Overall, the study affirms that mastery of prerequisite programming subjects is a fundamental predictor of success in advanced coursework, as evidenced by the strong correlation of $r = 0.74$ and p value of 0.000. At the same time, it highlights the need to ensure that this mastery reflects genuine understanding rather than superficial performance. By addressing both cognitive and contextual factors, including instructional practices, curriculum design, and the responsible use of technological tools, educational institutions can enhance the quality of programming education and better prepare students for the demands of higher level learning and professional practice.

REFERENCES

Valstar, M., et al. (2019). On the role of prerequisites in student success in programming courses.

Dita, B. V., & Velasco, M. J. M. (2025). The relationship of senior high school strands and academic performance in college among computer engineering students: Basis for policy guideline development. International Journal of Innovative Science and Research Technology (IJISRT), 10(1), 176–183. https://doi.org/10.5281/zenodo.14631683

Oranga, J. (2025). Mixed methods research: Merits, applications and challenges. International Journal of Social Science, 5(2), 233–238. https://doi.org/10.53625/ijss.v5i2.11034

Gagné, R. M. (2018). The conditions of learning and theory of instruction. Routledge.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. Cognitive Science, 12(2), 257–285. https://doi.org/10.1207/s15516709cog1202_4

Quality Research International. (2025). Analytic quality glossary. https://www.qualityresearchinternational.com/glossary/

Association for Computing Machinery Committee for Computing Education in Community Colleges. (2023). Bloom's taxonomy adapted for computing disciplines. https://www.acm.org/

Llego, M. A. (2018). Philippine professional standards for teachers (PPST). https://www.teacherph.com/philippine-professional-standards-for-teachers-ppst/

United Nations Educational, Scientific and Cultural Organization. (2016). Education 2030: Incheon declaration and framework for action for the implementation of sustainable development goal 4. https://unesdoc.unesco.org/

Gao, P., & Liu, J. (2025). Predicting student performance in programming courses using educational data mining.

Mosia, M. (2025). Factors influencing success in advanced engineering mathematics using structural equation modeling.

Commission on Higher Education. (2017). CHED memorandum order no. 87, series of 2017: Policies, standards, and guidelines for bachelor of science in computer engineering (BS CpE). https://ched.gov.ph/.