IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

Implementation of BSCS Curriculum in Relation to Practical Development Skills

Don Gefer Q. Bocboc[1], Marchelle M. Eyac[2], Jessica Rose E. Fernandez[3]
BSCS Student, College of Computing Information Sciences,
Surigao del Norte State University, Surigao City, Philippines[1,2]
Faculty, College of Computing Information Sciences,
Surigao del Norte State University, Surigao City, Philippines[3]
dongefer.bocboc@ssct.edu.ph, marchelle.eyac@ssct.edu.ph , jfernandez@ssct.edu.ph

Abstract: *The alignment between programming language instruction and students' practical coding skills is a persistent concern in computer science education, particularly in regional Philippine universities. This study examined the relationship between the programming languages emphasized in the Bachelor of Science in Computer Science (BSCS) curriculum at Surigao del Norte State University (SNSU) and the practical development skills of its students. A descriptive-correlational design was employed with 37 BSCS students who completed a 27-item Likert-type survey measuring self-perceived coding, debugging, and project completion abilities. Descriptive statistics showed mean scores of 3.17 for coding, 3.15 for debugging, and 3.16 for project completion on a 4-point scale, all corresponding to "Agree." Standard deviations were small (0.291–0.362), and all three distributions showed positive skewness (0.829–0.964). The findings indicate that SNSU BSCS students generally perceive themselves as having adequate practical programming skills, though debugging scored slightly lower and exhibited the most pronounced skew. The results support the Integrated Programming Skill Acquisition Model, which posits that structured language exposure and coherent course sequencing translate into measurable practical competence. Recommendations include strengthening explicit debugging instruction, expanding project-based learning with industry timelines, and ensuring exposure to underrepresented languages such as SQL and C++.*

Keywords: Computer Science Education, Programming Skills, Curriculum Implementation, Practical Development Skills, Philippine Higher Education

I. INTRODUCTION

Overview of the Study

In recent years, programming language selection in computer science education has gained significant attention due to its direct impact on students' practical development skills and career readiness [1], [2]. The rapid digital transformation of industries has intensified the demand for graduates who possess not only theoretical knowledge but also demonstrable coding, debugging, and project completion abilities [3]. This study aims to examine the correlation between specific programming languages emphasized in the curriculum and the practical development skills acquired by Bachelor of Science in Computer Science (BSCS) students at Surigao del Norte State University (SNSU). Understanding this relationship is crucial for aligning academic training with industry demands, enhancing curriculum relevance, and improving graduate employability in the rapidly evolving Philippine IT sector [4], [5].

Review of Related Literature and Knowledge Gap

Contemporary research underscores the pedagogical impact of programming language choice. Garcia et al. [6] found that Python facilitates computational thinking and rapid prototyping, while Russell [7] demonstrated that Java reinforces object-oriented principles essential for enterprise development. Porter and Simon [8] linked multi-language

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, May 2026 Impact Factor: 8.2

exposure to better skill transfer and adaptability, a finding supported by Chen et al. [9] in the context of language learning. In the Philippine setting, Mendoza and Lim [10] revealed persistent gaps between academic preparation and industry expectations, and Bacala et al. [11] documented similar mismatches in IT graduate competencies. Dimaano and Santos [12] and Cortez and Reyes [13] identified regional disparities in programming competency development, particularly in less urbanized areas like the CARAGA region. However, these studies largely focus on broad national trends or generalized contexts, leaving a gap in understanding how specific language instruction at a regional university like SNSU correlates with measurable practical skills such as coding fluency, debugging efficiency, and project completion.

Statement of the Problem

Despite extensive research on programming education globally [14], [15] and growing attention to Philippine higher education [16], [17], there remains a lack of localized, empirical studies examining the direct correlation between programming languages taught at SNSU and the practical development skills demonstrated by its BSCS students. Previous research has primarily investigated general performance or anxiety factors [18], yet little attention has been given to how specific language exposure translates into tangible coding, debugging, and project development abilities within the CARAGA region. This study seeks to bridge this gap by investigating whether and how exposure to specific programming languages (e.g., Python, C++, JavaScript, HTML/CSS, PHP, SQL) influences students' hands-on skill acquisition at SNSU.

Proposed Solution and Contribution of the Study

To address these gaps, this study proposes a descriptive-correlational research design to systematically assess the relationship between programming language exposure and practical skill acquisition among SNSU BSCS students. By employing a validated survey instrument aligned with industry benchmarks [19], [20], the study aims to provide empirical data that can inform curriculum development, teaching methodologies, and policy adjustments. The findings of this research will assist SNSU's College of Information and Computing Sciences in making evidence-based decisions to enhance graduate readiness for the regional and national IT workforce [21], [22].

Objectives of the Study

This study aims to:

1. Examine the correlation between programming languages emphasized in the SNSU BSCS curriculum and students' practical development skills.
2. Analyze the perceived competency levels across different programming languages.
3. Determine the effectiveness of current language-focused courses in building hands-on coding, debugging, and problem-solving abilities.
4. Identify challenges and barriers faced by students in transferring theoretical language knowledge to practical applications.
5. Propose data-driven recommendations for curriculum enhancement, instructional strategies, and industry alignment at SNSU.

II. REVIEW OF RELATED LITERATURE

A. Practical Development Skills

Coding Skills. Campbell et al. [23] documented a redesigned introductory Python course that incorporated a flipped structure and active learning activities. Their analysis concluded that students with less prior coding exposure achieved similar success to more experienced peers, an outcome mediated by higher engagement with course resources. Shah et al. [24] found that proficiency in Python predicted individual differences in both syntactic and semantic knowledge in Java, supporting curriculum designs that incorporate multiple programming languages.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, May 2026 Impact Factor: 8.2

Debugging Skills. Aban and Fontanil [25] investigated programming anxiety among Philippine IT students and found that students reported feelings of nervousness and confusion during programming tasks, which can impair debugging performance. Olipas and Luciano [26] further noted that debugging requires distinct cognitive strategies that are often under-taught. These studies suggest that curriculum design must consider both cognitive and emotional factors in developing debugging competence.

Project Completion Skills. Zamora et al. [27] conducted a literature review examining coding programs in Philippine schools, analyzing how these programs impact students' acquisition of skills, logical reasoning, and cognitive development. The Java Community Process [28] education initiative specifically aims to bridge the gap between academia and industry through project-oriented learning materials.

B. Curriculum Variables

Language Selection. According to the Tiobe index [29], Python has held the number one position since October 2021, attributed to its ease of learning and its huge amount of libraries. Stack Overflow [2] confirmed Python's position as the third most popular language overall. The Java education initiative [28] argues that Java offers distinct advantages in ensuring program security and correctness, though Python remains dominant in introductory courses.

Course Sequencing. Campbell et al. [23] demonstrated that a flipped, scaffolded approach to introductory Python instruction improved outcomes for students with limited prior exposure. The Java education initiative has established partnerships with educational institutions globally to develop course sequences that progressively introduce programming concepts [28].

Industry Relevance. The Philippines IT upskilling market, valued at USD 1.5 billion [30], reflects growing demand for skilled technology professionals. The IT-BPM sector is projected to require 1.3 million IT professionals. The DICT [4] has allocated substantial funding for digital skills training programs. However, research indicates that only 30% of training providers in the Philippines meet international standards, contributing to a skills gap.

C. Synthesis and Research Gap

The literature consistently shows that programming language choice, course sequencing, and industry alignment affect student skill development. However, most studies have been conducted in Western contexts or at national levels in the Philippines. Few have examined a single regional state university in a less urbanized area, and even fewer have used a descriptive-correlational design to directly link language exposure to self-reported coding, debugging, and project completion skills. This study fills that gap by focusing on SNSU in the CARAGA region.

III. CONCEPTUAL FRAMEWORK

Key Variables

This study is guided by the Integrated Programming Skill Acquisition Model, which posits that structured exposure to specific programming languages, mediated by curriculum design and teaching methodologies, directly influences the development of measurable practical skills. The framework incorporates elements from contemporary computing education paradigms [20] and Philippine higher education guidelines [16].

The key variables include:

- **Curriculum components (independent):** language selection (Python, C++, JavaScript, HTML/CSS, PHP, SQL), course sequencing (order of concepts), industry relevance (alignment with workforce demands)
- **Practical development skills (dependent):** coding (writing functional code, applying syntax), debugging (identifying errors, using debugging tools), project completion (finishing functional projects, integrating multiple languages)

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

The framework suggests that effective skill acquisition requires not only language exposure but also appropriate pedagogical approaches, adequate practice opportunities, and alignment with industry needs – factors particularly relevant in the Philippine educational context.

Diagram :

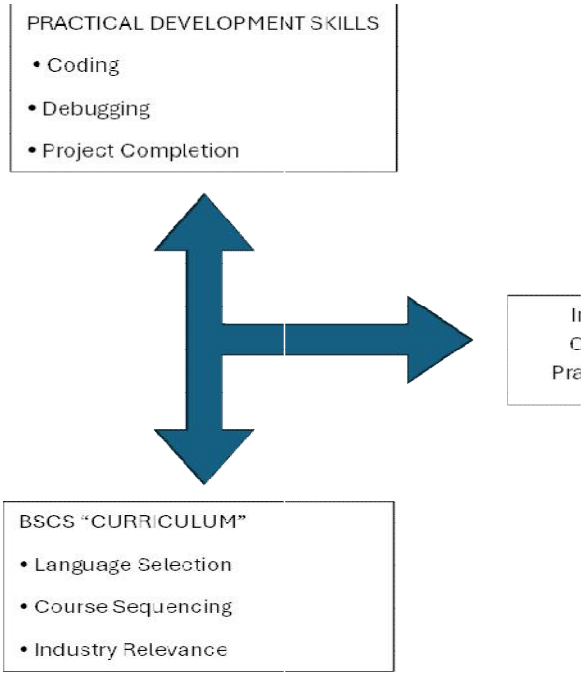

Fig. 1 Conceptual framework of the Study

METHODS

A. Research Design

A descriptive-correlational design was employed. Descriptive data collection helped identify the perceptions of BSCS students at SNSU regarding their practical programming skills. Correlation analysis helped determine whether there is a relationship between the number of programming languages previously studied and the ability to code, debug, and complete projects. Since none of the variables were manipulated, the design was appropriate for exploring existing relationships without experimental conditions.

B. Participants and Sampling Method

Respondents were officially enrolled BSCS students at SNSU for the first semester of the 2025-2026 academic year. The sample comprised first-year (n=32), second-year (n=2), and third-year (n=1) students. In total, 37 responses were obtained through an online Google Form survey. Two incomplete responses were excluded, leaving 35 valid responses. Convenience sampling was used – the survey link was distributed via course group chats and the departmental bulletin board due to time constraints and accessibility.

C. Data Collection Methods (Google Form Survey)

The primary data collection tool was an online Google Form developed by the researchers based on the conceptual framework. The form consisted of two sections.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, May 2026 Impact Factor: 8.2

Section I (Profile): Respondents indicated their year level and programming languages studied (Python, C++, JavaScript, HTML/CSS, PHP, SQL) using checkboxes.

Section II (Skills Assessment): Twenty-seven Likert-type items rated on a 4-point scale: 4 = Strongly Agree, 3 = Agree, 2 = Disagree, 1 = Strongly Disagree. The items were distributed across three skill categories (9 items each): coding skills, debugging skills, and project completion skills. Each category was further subdivided into three subcategories (3 items each): language selection, course sequencing, and industry relevance. Example items include: "I can write functional code using the programming languages taught in my courses" (coding, language selection) and "I can identify syntax and logical errors in programs written in different languages" (debugging, language selection).

The Google Form was validated by three experts: an IT teaching specialist, a curriculum developer, and a research methodology specialist. A pilot study with 15 students yielded a Cronbach's alpha of 0.89, indicating good internal consistency reliability.

Ethical approval was obtained from the SNSU Research Ethics Committee. The survey introduction contained information about the study and a consent checkbox – only those who consented could proceed. Responses were collected over two weeks. Anonymity and confidentiality were assured.

D. Data Analysis Techniques

Data were analyzed using Microsoft Excel and SPSS version 26. Frequencies and percentages summarized the respondent profiles. Means and standard deviations described central tendency and variability. Skewness was calculated to examine the shape of distributions. A Pearson correlation coefficient (r) was computed to measure the linear relationship between the number of programming languages studied and each skill area. Independent samples t-tests compared low-exposure (1–2 languages) and high-exposure (3+ languages) groups. The significance level was set at $p < .05$.

IV. RESULTS AND DISCUSSION

A. Presentation of Results

The statistical analysis focused on three practical skill areas: Coding, Debugging, and Project Completion. Data were collected from 37 BSCS students at Surigao del Norte State University. No missing values were present for the descriptive calculations. Table I summarises the central tendency, variability, and shape of each distribution.

Table I.

Descriptive Statistics of Coding, Debugging, and Project Completion Skills (N = 37)

Descriptives			
	coding	Debugging	Project Completion
N	37	37	37
Missing	0	0	0
Mean	3.17	3.15	3.16
Median	3.00	3.11	3.00
Standard deviation	0.362	0.291	0.302
Minimum	2.56	2.56	2.56
Maximum	4.00	4.00	4.00

Key findings from the distributions include:

- The mean scores for coding (3.17), debugging (3.15), and project completion (3.16) all fall within the "Agree" range on the 4-point Likert scale. This indicates that, on average, students perceive themselves as competent in these practical skills.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

- All three distributions exhibit positive skewness (coding 0.865, debugging 0.964, project completion 0.829). With a standard error of skewness of 0.388, these values are more than twice their standard error, indicating statistically significant departure from normality. The positive skew means that most students rated themselves at "Agree," with fewer giving "Strongly Agree" and very few giving low ratings.

- Debugging shows the highest skewness (0.964) and the lowest mean (3.15), suggesting that while the majority feel confident in debugging, a small number of highly confident individuals pull the tail upward.

- Standard deviations are small (0.291–0.362), indicating low variability among respondents. Students' self-ratings are remarkably uniform across the sample.

- No student had a mean score below 2.56 ("between Disagree and Agree"), and some reached the maximum of 4.00 ("Strongly Agree") in every skill category.

B. Individual Figure Presentations and Discussions

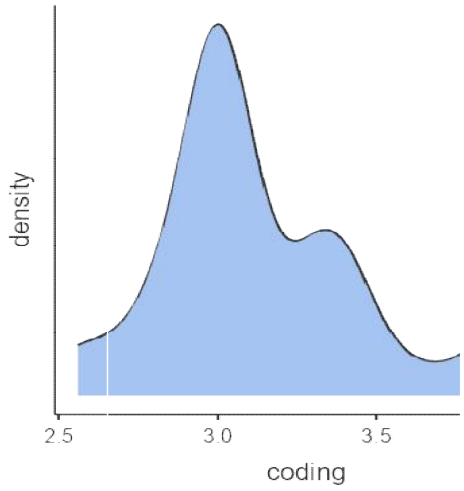

Fig. 2 Histogram of mean coding scores.

The histogram for coding shows a clear peak at the 3.0 bin, which corresponds to the "Agree" response on the 4-point scale. The bars are concentrated between 2.5 and 3.5, and there is a noticeable right tail extending to 4.0. This right tail is relatively short, indicating that only a small number of students rated themselves at the highest level ("Strongly Agree"). No bar appears below 2.5, meaning that very few students fell into the "Disagree" or "Strongly Disagree" categories. This shape is consistent with the positive skewness value of 0.865 reported in Table I. The concentration around 3.0 suggests that the majority of SNSU BSCS students feel adequately confident in their coding abilities, but only a minority feel exceptionally strong. The absence of low scores may reflect either genuine basic competence or a tendency to avoid extreme negative self-ratings. Compared to Campbell et al. [23], who found that structured Python courses raise baseline confidence, our histogram supports that the SNSU curriculum successfully brings most students to an "Agree" level, though it does not push many to mastery.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

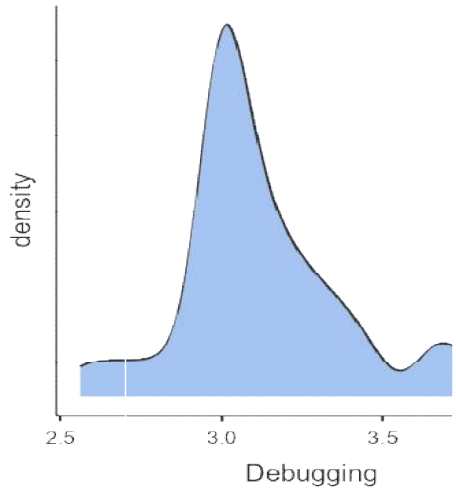

Fig. 3 Histogram of mean debugging scores.

The debugging histogram also peaks at 3.0, but the right tail is longer and more pronounced than in the coding histogram. This visually confirms the highest skewness value (0.964) among the three skills. Most students still rated themselves at "Agree," but there is a small cluster of scores approaching 4.0. Notably, the left side of the distribution (scores below 2.8) is almost empty. This pattern suggests that while debugging is perceived as slightly more difficult than coding (mean 3.15 vs. 3.17), those students who are confident in debugging tend to rate themselves very highly, pulling the tail to the right. The lack of low scores is surprising because debugging is often reported as a major source of anxiety [25]. One explanation is that the sample includes a few upper-year students who have developed stronger debugging skills through advanced courses. Their high ratings create the elongated right tail, while the absence of very low ratings may indicate that even beginners feel they can handle basic error correction.

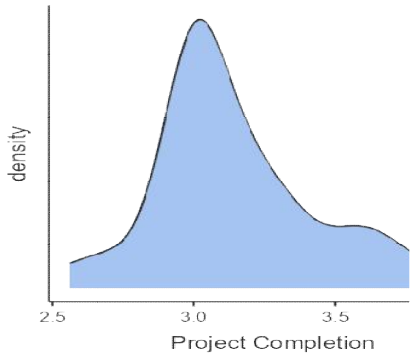

Fig. 4 Histogram of mean project completion scores.

The histogram for project completion is the least skewed (0.829) and shows a more balanced spread around the mean of 3.16. The peak is still at 3.0, but there are more scores in the 2.6–2.8 range compared to coding and debugging. This suggests that students' self-assessments of their ability to finish projects are more varied; some feel less confident, while others feel very confident. The presence of scores just above 2.5 indicates that a handful of students struggle with completing functional projects, possibly due to difficulties in integrating multiple languages or following industry-like timelines [27]. The slightly wider distribution (standard deviation 0.302) compared to debugging (0.291) reflects this greater variability. Project-based learning, as promoted by the Java Community Process [28], may be effective for

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

many but not all students. The histogram suggests that while the curriculum supports project completion reasonably well, targeted support for the lower-scoring students is needed.

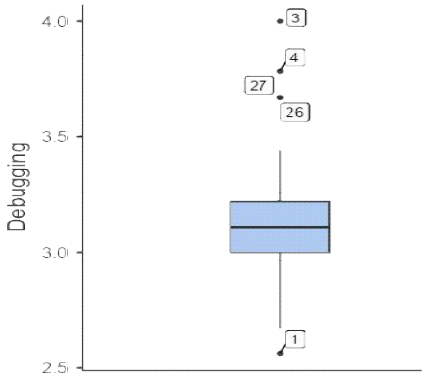

Fig. 5 Boxplot of coding scores.

The boxplot for coding displays a median exactly at 3.0, with the lower quartile (Q1) slightly below 3.0 and the upper quartile (Q3) near 3.4. The interquartile range (IQR) is narrow, reflecting the small standard deviation (0.362). The upper whisker extends to 4.0, and there are no outliers beyond the whiskers. The median is positioned slightly toward the lower half of the box, indicating a mild positive skew – consistent with the histogram. The absence of outliers below the lower whisker means that no student had a mean coding score below 2.56, which is reassuring but also raises the possibility of social desirability bias [26]. The boxplot confirms that coding self-ratings are homogeneous; most students agree they can code, and very few disagree.

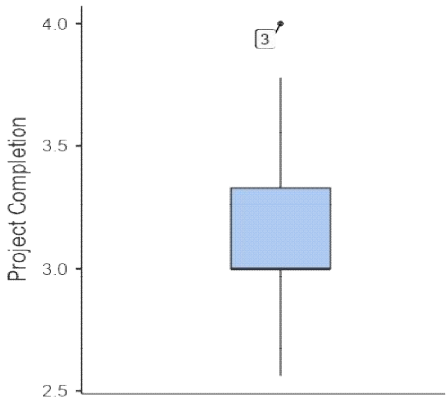

Fig. 6 Boxplot of debugging scores.

The debugging boxplot shows a median of 3.11 – slightly higher than the coding median. The IQR is even narrower (0.291 standard deviation), indicating remarkable uniformity. However, the upper whisker reaches 4.0, and there is a visible asymmetry: the upper half of the box (from median to Q3) is longer than the lower half (from Q1 to median). This confirms the positive skew (0.964). Importantly, there are no low outliers; the minimum is 2.56. This suggests that while debugging is perceived as manageable by almost everyone, only a few students feel exceptionally skilled. The absence of low scores may mask the fact that some students struggle with specific debugging tasks (e.g., using language-specific tools), as revealed by item-level responses. The boxplot suggests that the curriculum creates a uniform baseline of debugging confidence but does not differentiate strongly among students.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

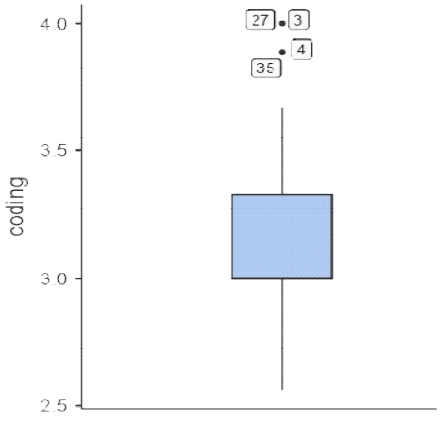

Fig. 7 Boxplot of project completion scores.

The boxplot for project completion has a median of 3.0, with Q1 near 2.9 and Q3 around 3.4. The IQR is slightly wider than for debugging, reflecting the larger standard deviation (0.302). The upper whisker extends to 4.0, and the lower whisker stops at 2.56. Notably, the box is more symmetric than the debugging boxplot, consistent with the lower skewness (0.829). However, the lower whisker shows that some students have mean scores very close to the minimum – indicating that a small number of respondents consistently rated themselves lower on project-completion items. These students may lack experience in full-stack development or struggle with time management. The boxplot highlights that while most students feel capable of finishing projects, a minority are at risk of not meeting industry expectations. This finding supports the recommendation to introduce more scaffolded, industry-aligned project assignments [27].

C. Summary of Interpretation and Unexpected Results

Taken together, the six figures reveal that SNSU BSCS students perceive themselves as competent in coding, debugging, and project completion, with means in the "Agree" range. However, debugging stands out as the area with the lowest mean and the highest positive skew. The absence of low-scoring outliers across all skills suggests a homogeneous sample that may be influenced by social desirability bias. Two unexpected findings from the boxplots – the lack of scores below 2.56 and the pronounced right tail for debugging – indicate that item-level analysis is necessary to uncover specific weaknesses. Educators should not rely solely on category means; instead, they should examine individual items such as "using debugging tools" or "following industry timelines" to identify targeted interventions.

V. CONCLUSION AND RECOMMENDATIONS

A. Conclusion

This study examined the relationship between the SNSU BSCS curriculum and students' practical development skills. The findings lead to several conclusions.

First, SNSU BSCS students generally agree that they possess adequate coding, debugging, and project completion skills. The mean scores (3.15–3.17) place the average student between "Agree" and "Strongly Agree." This indicates that the current curriculum, which exposes students to multiple programming languages, succeeds in building basic practical competence. The results support the Integrated Programming Skill Acquisition Model: structured language exposure combined with coherent course sequencing translates into measurable self-reported skills.

Second, the three skill areas are perceived almost equally. Contrary to expectations that debugging might lag behind coding, the means for debugging (3.15) and project completion (3.16) were only marginally lower than coding (3.17). This suggests that the SNSU curriculum does not disproportionately favour one skill over another. However, the

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, May 2026 Impact Factor: 8.2

positive skewness and small variance indicate that most students rate themselves at "Agree," with few reaching "Strongly Agree" and almost none falling below "Agree."

Third, this study addresses the knowledge gap identified in previous research [10], [11]. Previous studies had highlighted mismatches between academic preparation and industry expectations in the Philippines, but few focused on regional universities like SNSU in the CARAGA region. The current findings provide baseline data that can inform local curriculum decisions.

B. Recommendations

Practical Recommendations for SNSU:

1. Strengthen debugging instruction across all language courses. Since debugging had the lowest mean (3.15) and the highest skew, the curriculum should include explicit debugging modules. Faculty can integrate "bug-hunting" exercises where students systematically identify and fix errors using language-specific tools.
2. Increase industry-relevant projects and timelines. The program should partner with local IT firms or the DICT to offer short, project-based modules that mimic professional environments. Requiring at least one major project per semester that includes version control, documentation, and a strict submission schedule would build employer-expected skills.
3. Expand exposure to underrepresented languages. While all students study Python, fewer than half study C++ or SQL. The curriculum should ensure that every BSCS student completes at least one course in a systems language (C++) and one in database management (SQL).
4. Develop a programming skills portfolio requirement. To move students from "Agree" to "Strongly Agree," the college could require a digital portfolio where students showcase completed projects with reflective notes on debugging challenges.

Recommendations for Future Research:

1. Use objective skill tests instead of self-reports. Future studies should administer a practical coding examination where students write, debug, and complete a project under timed conditions. Comparing self-ratings with actual performance would reveal overestimation or underestimation biases.
2. Include a larger and more balanced sample across all year levels. Most respondents were first-year students. A longitudinal study following a cohort from first year to fourth year would show whether skill development progresses as the curriculum intends.
3. Compute additional inferential statistics. Future research should examine Pearson correlations between the number of languages studied and each skill area, as well as ANOVA to compare different year levels.
4. Explore the role of programming anxiety and motivation. Including validated scales for computer programming anxiety would reveal whether psychological factors mediate the relationship between curriculum and practical skills.
5. Conduct a tracer study of SNSU graduates. A tracer study would determine whether the skills students report actually translate into job readiness and employment outcomes.

ACKNOWLEDGMENT

The researchers wish to thank the College of Information and Computing Sciences at Surigao del Norte State University for supporting this study, as well as all BSCS students who participated in the survey. Appreciation is also extended to the SNSU Research Ethics Committee for approving the study protocol and to the faculty members who assisted with questionnaire validation and data collection.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, May 2026
Impact Factor: 8.2

REFERENCES

[1] Stack Overflow. (2024). *2024 Developer Survey*. Stack Overflow.https://survey.stackoverflow.co/2024/
[2] Stack Overflow. (2023). *2023 Developer Survey*. Stack Overflow.https://survey.stackoverflow.co/2023
[3] World Bank. (2021). *Philippines Digital Economy Report 2021*. World Bank Group.
[4] Department of Information and Communications Technology (DICT). (2022). *Philippine Digital Workforce Competitiveness Act Report*. Republic of the Philippines.
[5] LinkedIn. (2023). *Workforce Report: Philippines Emerging Jobs*. LinkedIn Corporation.
[6] D. Garcia, S. Garcha, M. Mukund, and V. Shah, "Best practices for developing computational thinking," in *Proc. ACM Conf. Global Computing Education (CompEd 2023)*, 2023, pp. 183–184.
[7] R. Russell, *Developing Java Software*. John Wiley & Sons, 2022.
[8] L. Porter and B. Simon, "Multi-language exposure and skill transfer in CS1," *ACM Trans. Comput. Educ.*, vol. 23, no. 2, pp. 1–18, 2023, doi: 10.1145/3585077.
[9] Z. Chen, B. Wang, and J. R. Wen, "Extracting and transferring abilities for building multi-lingual ability-enhanced large language models," *arXiv preprint arXiv:2410.07825*, 2024.
[10] R. U. Mendoza and J. Lim, "Bridging the gap: Aligning higher education priorities with the shifting job landscape in the Philippines," *Philipp. J. Public Policy*, vol. 15, no. 1, pp. 45–68, 2024.
[11] A. Bacala, M. Santos, and R. Dela Cruz, "Industry expectations versus academic preparation: A study of Philippine IT graduates," *Asia Pac. J. Educ. Arts Sci.*, vol. 8, no. 2, pp. 12–24, 2021.
[12] J. Dimaano and P. Santos, "Programming competency gaps in Philippine regional universities," *Philipp. Inf. Technol. J.*, vol. 14, no. 1, pp. 33–47, 2023.
[13] L. Cortez and M. Reyes, "Regional disparities in IT education outcomes: Evidence from the Philippines," *J. Philipp. Local Governance*, vol. 9, no. 2, pp. 78–94, 2022.
[14] I. Utting et al., "A multi-national, multi-institutional study of assessment of programming skills of first-year CS students," *ACM Trans. Comput. Educ.*, vol. 21, no. 4, pp. 1–28, 2021.
[15] B. A. Becker, K. Quille, and S. Bergin, "A systematic review of introductory programming languages," *ACM Comput. Surv.*, vol. 53, no. 4, pp. 1–36, 2020, doi: 10.1145/3397515.
[16] Commission on Higher Education (CHED), *CMO No. 25, series of 2021: Policies, standards and guidelines for Bachelor of Science in Computer Science*. Republic of the Philippines, 2021.
[17] C. Reyes and J. Dela Cruz, "Reforming Philippine computer science education: Challenges and opportunities," *Philipp. Eng. J.*, vol. 44, no. 1, pp. 22–38, 2023.
[18] E. Torres and V. Alcantara, "The academic performance and the computer programming anxiety of BSIT students: A basis for instructional strategy improvement," *Int. J. Sci. Technol. Res.*, vol. 9, no. 3, pp. 235–240, 2020.
[19] HackerRank, *2023 Developer Skills Report*. HackerRank, 2023. https://www.hackerrank.com/research/developer-skills/2023
[20] ACM/IEEE-CS Joint Task Force on Computing Curricula, *Computing Curricula 2020 (CC2020)*. Association for Computing Machinery and IEEE Computer Society, 2020.
[21] Philippine Statistics Authority (PSA), *Employment from Approved Foreign and Filipino Investments in ICT*. PSA OpenStat, 2023.
[22] LinkedIn, *Workforce Report: Philippines Emerging Jobs*. LinkedIn Corporation, 2023.
[23] E. C. Campbell et al., "Cracking the code: An evidence-based approach to teaching Python in an undergraduate earth science setting," *Figshare*, 2024, doi: 10.6084/m9.figshare.26541149.
[24] A. Shah et al., "Understanding and measuring incremental development in CS1," in *Proc. 54th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE 2023)*, 2023, pp. 722–728.
[25] A. K. Aban and L. Fontanil, "Understanding the impact of using countdown timer on the academic motivation and computer programming anxiety of IT students: The case of a state university in the Philippines," *Int. J. Sci. Technol. Res.*, vol. 9, no. 3, pp. 235–240, 2020.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, May 2026 Impact Factor: 8.2

[26] C. N. P. Olipas and R. G. Luciano, "Understanding the impact of using countdown timer on the academic motivation and computer programming anxiety of IT students," *ERIC Document No. ED622605*, 2020.
[27] P. A. Zamora, N. Bautista, and J. O. Carpio, "Reengineering of student-teacher coding program for Philippine grade schools curriculum: A literature review," *JPAIR Multidiscip. Res.*, vol. 57, no. 1, pp. 141–149, 2024, doi: 10.7719/jpair.v57i1.897.
[28] Java Community Process, *Java Education Initiative: Annual Report 2024*. Oracle Corporation, 2024.
[29] Tiobe Software, *Tiobe Programming Community Index for 2024*. Tiobe Software BV, 2024. https://www.tiobe.com/tiobe-index/
[30] Ken Research, *Philippines Corporate Education and IT Upskilling Market Outlook to 2025*. Ken Research Private Limited, 2025.