International Journal of Research Studies in Education
2023 Volume 12 Number 8, 101-112

Examining student experiences: Challenges and perception in computer programming

Napalit, Francisco ✉
Information Communications Technology Academy (iACADEMY), Philippines (kiko.napalit@iacademy.edu.ph)
Tanyag, Bennett
Information Communications Technology Academy (iACADEMY), Philippines (bennett.tanyag@iacademy.edu.ph)
So, Carl Louie
Information Communications Technology Academy (iACADEMY), Philippines (carl.so@iacademy.edu.ph)
Sy, Cecilia
Information Communications Technology Academy (iACADEMY), Philippines (cecilia.sy@iacademy.edu.ph)
San Pedro, Jay R.
Information Communications Technology Academy (iACADEMY), Philippines (jay.sanpedro@iacademy.edu.ph)

ISSN: 2243-7703
Online ISSN: 2243-7711

OPEN ACCESS

Received: 1 November 2023 **Revised**: 13 November 2023 **Accepted**: 25 November 2023
Available Online: 25 November 2023 **DOI**: 10.5861/ijrse.2023.71

Abstract

This study investigates practical issues students face in learning computer programming, exploring potential differences based on gender, degree program, and Senior High School (SHS) track. In addition, this study examines and evaluates students' understanding of computer programming courses, difficulties, and situations in learning computer programming. The research investigates students' knowledge, difficulties, and experiences in computer programming education. Results indicate that gender and program enrollment do not significantly impact students' perceptions of programming difficulties or preferences for practical learning contexts. Despite robust comprehension of fundamental concepts, students consistently encounter design, syntax, and problem-solving challenges across gender and program-based categories. However, significant variations in perceived understanding among different programs highlight the need for tailored support. SHS track does not substantially affect students' perceptions of programming difficulty. Recommendations include prioritizing inclusivity in programming courses and offering tailored support based on program-specific needs. Future research should explore additional factors influencing programming experiences, such as teaching methods and individual learning styles, to inform effective pedagogical strategies and resources. The overarching goal is to create a diverse and inclusive programming education environment, ensuring equal opportunities for all students to excel in computer programming.

Keywords: computer programming education, gender differences, program enrollment, senior high school track, challenges in programming

Examining student experiences: Challenges and perception in computer programming

1. Introduction

In the ever-evolving technological landscape, including computer programming courses in primary computing curricula offers significant advantages. These courses are particularly crucial for computer science and information technology students who must acquire programming skills. Mastery of programming languages is essential, necessitating a comprehensive understanding of the discipline (Cheah, 2020). Therefore, students enrolled in programming courses must demonstrate proficiency in programming and logical principles, forming a critical foundation for future computer science and information technology pursuits. Additionally, as competition for computer programming roles intensifies, skills development among IT professionals becomes increasingly vital for enhancing job performance (Vaidyanathan, 2020). Computer programming is recognized as a creative endeavor that fosters problem-solving and innovation by leveraging interconnected algorithms in specific languages to create customized computer programs. Through continuous refinement of their programming skills, IT professionals can navigate the dynamic programming landscape, ensuring they remain competitive in this evolving industry (Vaidyanathan, 2020).

Programming languages offer various approaches, each with distinct strengths and methodologies. After acquiring fundamental programming skills, students must progress to a higher level of proficiency to enhance their problem-solving abilities in tackling intricate computer-related issues. Achieving this skill level demands an integrated strategy that combines critical thinking and computational prowess, enabling students to expand their basic programming skills into advanced, refined problem-solving capabilities, resulting in significant success in their programming pursuits (Harimurti et al., 2019).

Nevertheless, novice programmers often face many challenges and difficulties throughout their programming studies, leading to elevated failure rates, unfavorable perceptions of programming courses, and fear and apprehension. In response to these challenges, educators and institutions have introduced various strategies to address the issues inherent in computer programming courses. However, despite the pivotal role of programming as a skill for students in computing-related programs, performance hindrances stemming from the difficulties encountered during the learning process may persist (Piwek & Savage, 2020). Furthermore, students engaging in programming must cultivate a specific thought process with numerous programming concepts and structures, which can be particularly challenging for many learners. Winslow (1996) outlines a four-step process for identifying and resolving programming problems, involving problem evaluation, potential solution determination, the transformation of the solution into a coded computer program, and the subsequent stages of testing, review, and debugging.

At iACADEMY's School of Computing, first-year college students from various programs can enroll in fundamental computing programming courses: Fundamentals of Programming (FUNDPROG) and Advanced Programming (ADVAPROG). FUNDPROG, offered in the first trimester of their first year, focuses on instilling core concepts such as process flowcharting, algorithm processes, and basic programming. In the second trimester of their first year, students can progress to ADVAPROG, which builds upon the foundational knowledge from FUNDPROG and explores more advanced programming concepts. Both courses are structured as 3-unit classes, comprising one unit of laboratory sessions and two units of lecture classes. Success in these courses relies on students' ability to develop a strong grasp of programming language syntax and parameters and skills in planning, development, testing, and debugging.

Students face several challenges in programming courses, such as the steep difficulty level of learning, especially for newcomers, and transitioning from fundamental concepts to more advanced programming can be daunting. Students must master programming language syntax, and parameters can be tricky, potentially leading

to code errors. Comprehending abstract programming concepts, such as process flowcharting and algorithm processes, can be a hurdle for those with prior programming experience. Planning, developing, and testing code is complex and error-prone, while debugging, though essential, can be time-consuming and frustrating. Maintaining motivation and perseverance throughout demanding programming courses can be an issue, particularly for those finding the subject challenging. Students' diverse backgrounds and levels of prior programming experience can lead to differing preparedness, and the class structure, with lecture and laboratory sessions, may only accommodate some learning styles.

Acquiring proficiency in computer programming entails the acquisition of both academic knowledge and hands-on experience in program development. Instructors typically design programming exercises to enhance students' capabilities in evaluating, applying techniques, and solving programming problems (Thinakaran & Ali, 2019). For many students, grasping computer programming is challenging due to the dual requirement for theoretical understanding and practical learning competence. Recognizing the symbiotic relationship between theoretical learning and practical application, particularly during laboratory sessions, becomes pivotal in shaping these activities (Thuné & Eckerdal, 2018). The intricacies of computer programming, coupled with the challenges in comprehending program logic, may result in frustration, diminished motivation, and waning enthusiasm for programming education. Consequently, some students might encounter setbacks or withdraw from programming classes or programs.

College is a level of education where students will pursue their preferred goals in the future, so they must be adequately prepared before enrolling (Bonquin, et al., 2021). The high school strand or track has a significant impact on students' college program selection; it will prepare them for the course they want to pursue and give them an idea about the course they want to follow (Ouano et al., 2019). Freshmen students with a good understanding before the course will adjust to the environment and achieve excellent grades. Still, students with low academic background and knowledge of the chosen course will be suppressed and fall behind in their academic track. (Alipio,2020). In university education, the first year often witnesses the highest student attrition rate, with many students either failing or dropping out. Within the domain of computer programming, students' struggles during their initial year of study can be attributed to the challenging transition from high school to university (Barlow-Jone, 2019). This transition period presents a significant contributing factor to the difficulties experienced by computer programming students. The increased academic demands, unfamiliar learning environments, and higher expectations can create obstacles for students as they adapt to the rigorous nature of university-level programming courses.

Baist and Pamungkas (2017) studied programming challenges students face. They highlighted three critical aspects students should master’s in computer programming: programming structure, design principles, and programming language syntax. Cabo (2019) delved into students' difficulties in creating functional computer programs using the Python language, with a common starting point being a need for more comprehension of the problem at hand. To address this, supporting students in developing effective problem-solving methods is crucial to enable the creation of viable computer programs. Among the array of challenges encountered by students in programming, difficulties were most prominently observed in comprehending syntax, debugging, and crafting algorithms. These areas consistently emerged as the most problematic for students. In particular, grasping the correct usage and arrangement of programming language elements, which falls under the domain of syntax, was notably challenging (Islam et al., 2019).

Inexperienced programmers often grapple with various programming challenges, encompassing program structures, code comprehension, debugging, and code navigation. These findings underscore the need for educational program designers to reevaluate and enhance their methodologies in addressing these challenges effectively. An important implication lies in integrating real-world problem-solving scenarios within programming courses, facilitating students in honing their coding, design, comprehension, and debugging skills (Muller et al., 2019). Students' early encounters with programming significantly influence their self-perception regarding programming proficiency and their overall perception of the field. These initial experiences can

exhaust students' interest in computing domains, prompting many educators to embrace project-based learning as an effective pedagogical approach to teaching computer programming (Atta, 2021).

In addition to the challenges of learning computer programming, gender disparities among students' attitudes and performance toward programming are a noteworthy consideration. Du and Wimmer's study (2019) uncovered that males typically have more exposure to computer programming, yet women outperform men in problem understanding, hinting at women's potentially higher aptitude for computer programming. Duran, Haaranen, and Hellas (2019) delved into gender-based differences in determination, confidence, concentration, prior experience, and performance. They found that males tend to have earlier and more extensive experience and are more inclined toward Computer Science degrees. This often translates into a perception that male students find computer programming more manageable, exhibit greater intentions to pursue it in the future and achieve superior learning outcomes compared to female students. In contrast, males tend to view programming through a Computer Science lens. These collective studies illuminate the challenges students encounter in computer programming courses, influenced by their prior academic experiences, skills, and readiness for their chosen college programs. Additionally, research by Atta and Atta (2021) centered on Computer Science and Information Technology students at Kwame Nkrumah University of Science and Technology and revealed a gender disparity in programming interest and enrollment, with more males displaying an interest and enrolling in programming courses. McAdams (2021) investigated why many girls discontinued computer programming studies during the transition from middle school to senior school and found that girls exhibited significantly higher programming abilities than boys. Notably, there was no gender difference among the most skilled programmers, suggesting that addressing gender attitudes and providing engaging programming instruction can enhance girls' programming abilities and narrow the gender gap in programming.

A study is conducted to mitigate the dropout rate among undergraduate computer science students, focusing on identifying their primary challenges and evaluating current initiatives to address these issues. Understanding these issues is pivotal in proposing tailored solutions and raising awareness among educators and peers. Furthermore, it underscores the importance of future research in this domain, urging graduate STEM researchers to contribute to creating a more supportive environment for female undergraduate students in the field. The ultimate aim is to reduce the dropout rate among female computer science students (Silva et al., 2021). Su et al. (2022) introduced a problem-based learning approach integrated with an online programming system in a STEM programming course. Simplified assignments and teacher guidance facilitated programming task solutions, while problem-based learning enhanced students' understanding, planning, execution, reflection, and debugging. The study reported a substantial difference between students' prior knowledge and learning outcomes, suggesting the positive impact of problem-based learning and online programming systems on students' perceptions, outcomes, and behaviors in STEM programming, enriching their learning experiences. An analysis of existing literature on gender, program, and STEM tracks provided valuable insights, revealing gaps, and limitations, the need to address gender disparities in programming, explore program-specific dynamics in diverse STEM fields, and conduct a more thorough investigation into the unique challenges faced by students in various tracks. While prior research identified common programming challenges, further exploration is needed to uncover effective interventions for first-year students. By addressing these gaps, the proposed study aims to enhance the existing body of knowledge by developing targeted interventions to address gender disparities, exploring effective strategies for overcoming programming challenges and providing practical recommendations to enhance the learning experience.

The study aims to identify the practical problems of the students in learning—computer programming. Specifically, to establish whether there is a significant difference among students in their computer programming learning when will group with their programs, gender, and senior high school track. In addition, this study examines to evaluate students' understanding of computer programming courses, difficulties, and situations in learning computer programming.

1.1 Research Questions and Hypotheses

This study aimed to answer the following questions.

- How may the respondents be described as to? Gender, Program (Major) and Senior High School Track.
- Are there differences in students' level of understanding, difficulties, and learning computer programming based on gender, degree program, and senior high school track?

Null Hypothesis

- There is no significant difference between the student's gender in learning computer programming.
- There is no significant difference between the student's program and senior high school track in learning computer programming.

2. Methods

This study solely utilized a quantitative research method to explore the problem statement. The research design focused on gathering data at a single time to summarize the behaviors or attributes of a specific group. The objective of this method was to gain a deeper understanding of the present condition of the variables pertinent to the problem statement, relying solely on numerical data. By employing a quantitative method, the study provided a comprehensive overview of the variables under investigation, emphasizing measurable aspects of the phenomenon. The chosen research design facilitated participant data collection at a specific moment, enabling statistical analysis of their experiences or viewpoints on the research subject. This approach allowed for examining the data's relationships, patterns, and trends, providing quantitative insights into the research topic. Using statistical analysis techniques further enhanced the study's ability to draw objective conclusions based on the numerical data collected. By employing a quantitative method, the study aimed to provide a comprehensive understanding of the studied variables, focusing on the phenomenon's measurable aspects. This research design aligned with exploring the problem statement using numerical data, allowing for rigorous analysis and objective interpretations. Additionally, referencing previous studies that had successfully employed a quantitative approach supported the effectiveness and relevance of this chosen research design.

The study used purposive sampling and convenience methods to select participants. The target participants were students enrolled at the School of Computing (SOC) from the Academic Year 2019 to 2022. A power analysis tool was conducted using a 95% confidence level and a 5% confidence interval to determine the minimum target respondents required for the study. This analysis ensured an adequate sample size to achieve statistically significant results. While the study aimed to meet the minimum target respondents required, it is important to note that responses from participants beyond the minimum target were still accepted. This approach provided an opportunity to obtain a more comprehensive understanding of the topic under investigation and enhanced the generalizability of the study's findings. By including responses from a larger sample, the study captured a broader range of perspectives and experiences, contributing to a more robust analysis.

The study adopted the survey research instrument developed by Rosminah et al. (2012). The survey instrument covered the demographic profile of the participants and their experience in programming while learning the concepts and techniques for computer programming courses. The first section of the instrument gathered the demographic profile of the participants, such as gender, program, and senior high school track. The second section of the instrument focused on the participant's level of understanding of the different topics in the programming course. The items were designed using a 5-point Likert scale from 5 (understand clearly) to 1 (not understand). The third section of the instrument concentrated on the participant's difficulty level while learning to program, using a 5-point Likert scale from 5 (Strongly Agree) to 1 (Strongly Disagree). The fourth section of the instrument centered on the situations that would help the participants in programming; this section also used

a 5-point Likert scale, the same as in Section 3. Lastly, the last section of the instrument focused on the participant's level of agreement with the factors that made them perform poorly or fairly while learning a computer programming course; this section also used a 5-point Likert scale, the same as in Section 3. Table 1 and Table 2 show the Likert scale used in the study.

Table 1

Descriptive ratings for five-point scale for the second section of the instrument

Weight	Mean Range	Verbal Interpretation
4	3.00-4.00	Understand Clearly
3	2.00-2.99	Understand
2	1.00-1.99	Somehow Understand
1	0-0.99	Not Understand at all

Table 2

Descriptive ratings for five-point scale for the third and fourth section of the instrument

Weight	Mean Range	Verbal Interpretation
4	3.00-4.00	Strongly Agree
3	2.00-2.99	Agree
2	1.00-1.99	Disagree
1	0-0.99	Strongly Disagree

The survey instrument for this study was created using Google Forms, an online platform known for its effectiveness in electronic survey administration. An email containing the survey form link was sent to the respective chairpersons of each program to ensure efficient access for the target participants. Once the survey responses were collected through Google Forms, they were exported in CSV format and organized, summarized, and analyzed using spreadsheet applications. This approach prioritized the privacy and anonymity of the participants while facilitating efficient data handling. In conjunction with spreadsheet applications, the CSV format enabled convenient data management and further analysis of the collected responses.

In the quantitative method, data obtained from the demographic profile of the participants were analyzed using percentage and frequency to provide a comprehensive understanding of the sample characteristics. This analysis helped determine the distribution of participants based on their gender, degree program, and senior high school track, allowing for a comparison of demographic profiles to identify any significant differences. A comparison was conducted among participants based on their demographic profiles to assess programming difficulties. Specifically, gender, degree program, and senior high school track were examined to determine if there were any notable variations in the challenges faced. The mastery of programming structure was assessed using mean scores and verbal interpretations. This analysis provided an understanding of participants' understanding and proficiency in these programming courses. Furthermore, the significant difference in knowledge and mastery of the topics among participants based on gender was examined using the chi-square test. Moreover, to explore potential variations in programming difficulties based on participants' programs and senior high school track, the analysis of variance (ANOVA) was employed. This statistical test allowed for the examination of significant differences in the challenges faced by participants from different programs and senior high school tracks.

Before implementing the research protocol, the researchers sought approval from the authors of the instruments used in their study titled "Difficulties in learning programming: Views of students." This step ensured the instruments were appropriate and aligned with the research objectives. The researchers contacted the Program chairpersons for assistance distributing the survey link to the target participants, who were students at the School of Computing (SOC). Participants were fully informed of the research objective and the significance of their participation through informed consent. They were provided with clear explanations regarding the purpose and scope of the study. Moreover, participants could withdraw from the study at any point while completing the survey questionnaire. The research did not involve sensitive or risky information that could potentially harm participants, and their privacy was protected throughout the process of completing the survey

questionnaire. The researchers securely stored the data obtained from the online survey form. Appropriate measures were taken to ensure the confidentiality and privacy of participants' information. The collected data was retained for approximately two months and was protected against unauthorized access. Once all the data was collated and the study was completed, the data was deleted properly and securely. The informed consent procedure was thoroughly reviewed to ensure it aligned with ethical guidelines and safeguarded the participants' rights and well-being throughout their involvement as subjects in the research study. This approach aimed to address ethical considerations and protect the rights and privacy of the participants in the study, ensuring a responsible and ethical research process.

3. Results

Table 3 depicted a significant gender imbalance among the respondents, with males forming the majority and females representing a minority in the survey sample. This gender disparity is an important observation and may have implications for understanding the experiences of male and female students in computer programming courses. Furthermore, the data revealed the preferred senior high school tracks among the 103 surveyed students. Notably, many students opted for STEM-related tracks, indicating a strong interest in science, technology, engineering, and mathematics. Additionally, some students pursued specialized tracks, which could provide insights into their diverse academic interests and career aspirations. This diversity in academic backgrounds and aspirations among the surveyed students is a valuable aspect to consider in discussions regarding computer programming education. Analyzing the demographic profile of the surveyed group based on their academic programs, it became evident that most students belonged to the Software Engineering program. However, there were also students enrolled in specialized programs. This variation in program choices highlights the dynamic nature of technology education and career interests in the constantly evolving technical job market.

Table 3

Demographic profile of the respondents

Profile	Frequency	Percentage
Gender		
Male	83	80.58
Female	20	19.42
Senior High School Track		
General Academic (GAS)	10	9.71
Accountancy, Business, Management (ABM)	2	1.94
Arts and Design	5	4.85
Humanities and Social Sciences (HUMMS)	8	6.79
Science, Technology and Mathematics (STEM)	47	45.63
Sports	2	1.94
Technical-Vocational Livelihood (TVL)	30	29.13
Program		
Data Science	3	2.91
Cloud Computing	5	4.85
Game Development	17	16.50
Software Engineering	59	57.28
Web Development	19	18.45

Table 4 analyzes students' perceived difficulties and challenges in their computer programming course, focusing on their understanding of different programming topics. The data indicated that, overall, students had a solid grasp of core programming concepts. Concepts such as variables, constants, data types, input/output statements, functions, selection statements, iteration statements, and array structures received favorable mean ratings. These ratings suggest that students generally found these topics to be easily comprehensible. However, the topic of multiple/dynamic array structures received a lower rating, indicating that students might encounter more challenges or uncertainty when dealing with this subject. Further discussion could explore the potential reasons behind this variation in student perception and ways to address their difficulties in comprehending

multiple/dynamic array structures.

Table 4

Level of student's understanding on different topic on programming course1

Criteria	Mean	Verbal Interpretation
Variables, constants, and data type	3.60	Understand Clearly
Input/output statement	3.71	Understand Clearly
Functions	3.50	Understand Clearly
Selection Statement	3.38	Understand Clearly
Iteration Statement	3.38	Understand Clearly
Array Structure	3.11	Understand Clearly
Multiple/Dynamic Array Structure	2.82	Understand
Mean	3.36	Understand Clearly

This study aimed to uncover the challenges students face when engaging in programming activities, and the data reflects their responses on a scale indicating agreement levels. Table 5 presents that students generally agree that they encounter difficulties grasping fundamental programming concepts, designing programs to address specific tasks, mastering programming language syntax, gaining access to a computer, using program development environments, and identifying and resolving bugs in their code. These findings highlight a range of hindrances students may encounter during their programming course, from conceptual understanding to technical and practical aspects.

Table 5

Difficulty while programming

Criteria	Mean	Verbal Interpretation
Understanding the basic concepts of programming structure	2.91	Agree
Designing a program to solve certain tasks	2.99	Agree
Learning the programming language syntax	2.97	Agree
Gaining access to a computer	2.90	Agree
Using program development environment	2.93	Agree
Finding bugs from my own program	2.87	Agree
Mean	2.99	Agree

This study sought to uncover the situations that students find most helpful for learning programming, with the responses presented in Table 6 reflecting their answers on a scale indicating their level of agreement. Notably, students generally agree on the efficacy of practical or laboratory sessions, where hands-on experiences enable them to apply theoretical concepts and gain a deeper understanding of programming. Additionally, they highly value consultations and discussions with instructors, peers, or seniors, emphasizing the importance of collaboration and mentorship. Small group exercise sessions, often called tutorials, are also perceived as beneficial for interactive learning and problem-solving. While the rating is relatively lower, there is still agreement regarding the value of independent study, allowing some students to focus on individual projects and assignments.

Table 6

Situations that would help to learn programming.

Criteria	Mean	Verbal Interpretation
In practical or laboratory sessions	3.49	Agree
Consultation or discussion with lectures, seniors or friend	3.60	Agree
In small group exercises sessions (tutorials)	3.39	Agree
While working alone one programming coursework	3.11	Agree
Mean	3.40	Agree

The data presented in Table 7 reveal no significant gender-based disparities in students' perceptions of computer programming. This lack of distinction is consistent across multiple aspects of programming education, including students' understanding of various programming topics, their perceived difficulty when working on programming tasks, and their evaluation of the effectiveness of different learning environments. These findings

collectively indicate that gender does not appear to substantially influence students' perceptions and experiences related to the challenges and learning opportunities associated with computer programming. Notably, these results support the notion that computer programming can provide an equitable and inclusive educational experience for students, irrespective of their gender. The discussion may further explore the potential reasons behind this gender-neutral perspective and whether this observation holds across different educational settings or institutions. Additionally, it could address how educators and institutions can leverage this finding to promote diversity and inclusivity in computer programming courses.

Table 7

Significant difference in perceived difficulties and challenges in computer programming in terms of gender

Profile Indicators	Pearson on Chi-Square	p-value	Decision	Remarks
Level of student's understanding on different topic on programming course	14.01	0.36	Failed to reject	Not Significant
Difficulty while programming	5.03	0.21	Failed to reject	Not Significant
Situations that would help to learn programming	6.51	0.45	Failed to reject	Not Significant

Note: "If p-value is less than or equal to 0.05 level of significance, reject ho, otherwise, failed to reject ho"

As illustrated in Table 8, the results investigate potential variations in perceived programming challenges associated with students' program choices. Three essential profile indicators, namely "Level of student's understanding of different topics in the programming course," "Difficulty while programming," and "Situations conducive to learning programming," were evaluated across different programs. The findings reveal significant variations in students' perceived understanding of programming concepts among these programs, highlighting that program enrollment plays a pivotal role in shaping how students perceive their grasp of programming topics. Conversely, there were no substantial distinctions in students' perceived programming difficulties or preferences for effective learning environments based on their program enrollment. This data suggests tailoring programming education to meet the unique needs of students within various programs, especially when it comes to understanding programming concepts.

Table 8

Significant difference in perceived difficulties and challenges in computer programming in terms of program

Profile Indicators	Program	Mean	f-value	p-value	Remarks
Level of student's understanding on different topic on programming course	SE	3.54	3.28	0.04	Not significant
	WD	3.06			
	GD	3.33			
	CC	2.95			
	DS	2.90			
Difficulty while programming	SE	3.10	1.80	0.22	Not significant
	WD	2.78			
	GD	2.75			
	CC	2.27			
	DS	2.83			
Situations that would help to learn programming	SE	3.37	.093	0.49	Not significant
	WD	3.41			
	GD	3.48			
	CC	3.45			
	DS	3.25			

Note: “If the p-value is less than or equal to the level of significance (0.05), reject Ho, otherwise failed to reject Ho."

Table 9 provides the results to identify whether significant variations exist in students' perceived difficulties and challenges in computer programming based on the Senior High School (SHS) tracks they pursued. This analysis encompasses three vital profile indicators: "Level of student's understanding of different topics in the programming course," "Difficulty while programming," and "Situations that would help to learn to program" across different SHS tracks. The findings indicate that, according to the dataset and variables examined in this study, there are no substantial differences among SHS tracks regarding students' perceptions of programming challenges, their understanding of programming concepts, or their preferences for effective learning

environments in computer programming. These results emphasize the importance of adapting programming education to cater to students' diverse backgrounds and experiences across various SHS tracks.

Table 9

Difference in perceived difficulties and challenges in computer programming in terms of high school track

Profile Indicators	Program	Mean	f-value	p-value	Remarks
Level of student's understanding on different topic on programming course	STEM	3.55	2.04	0.26	Not Significant
	TVL	3.68			
	GA	3.70			
	HUMMS	3.57			
	AD	3.50			
	ABM	3.38			
	SPORTS	2.50			
Difficulty while programming	STEM	2.88	0.62	0.73	Not Significant
	TVL	2.93			
	GA	3.17			
	HUMMS	2.77			
	AD	3.25			
	ABM	3.33			
	SPORTS	2.83			
Situations that would help to learn programming	STEM	3.44	0.69	0.63	Not Significant
	TVL	3.38			
	GA	3.23			
	HUMMS	3.38			
	AD	3.44			
	ABM	3.63			
	SPORTS	3.50			

Note: "If the p-value is less than or equal to the level of significance (0.05), reject Ho, otherwise failed to reject Ho."

The data analyzed in this study reveals several significant insights into computer programming education. Firstly, it underscores a noticeable gender imbalance among respondents, with males constituting a majority and females a minority. This gender disparity calls for a deeper examination of the factors contributing to it and the implementation of measures to encourage more gender diversity in computer programming courses. The prominence of STEM tracks among the surveyed students signifies a strong inclination toward technology-related fields, offering valuable guidance for institutions and educators to align their programs with students' interests.

Moreover, the data demonstrates that students, in general, have a robust understanding of core programming concepts, with lower ratings primarily observed in the context of multiple/dynamic array structures. Students also express challenges encompassing conceptual understanding and practical implementation, highlighting the multifaceted nature of programming education. Despite these challenges, there are no substantial gender-based or program-based differences in students' perceptions of computer programming. This suggests that the challenges and learning opportunities are experienced similarly across these groups. These findings emphasize the importance of accommodating diverse student backgrounds and experiences in programming education and providing tailored support to address their unique needs.

4. Conclusion

The results examining differences in students' perceived difficulties and challenges in computer programming across gender, program enrollment, and Senior High School (SHS) tracks have provided valuable insights into the dynamics of programming education. In summary, the analysis reveals that, within the context of the data and variables considered in this study, gender and program enrollment do not significantly impact students' perceptions of programming difficulties or their preferences for effective learning contexts. These findings suggest that computer programming can offer an inclusive and equitable educational experience for students from diverse backgrounds and program choices. Furthermore, the study thoroughly investigated differences in students' levels of understanding, difficulties, and learning experiences in computer programming

based on these demographic factors. While students displayed robust comprehension of fundamental programming concepts, they encountered challenges in various aspects of programming, from design and syntax to problem-solving. Notably, the results revealed that these challenges were experienced consistently across gender and program-based categories, refuting the null hypotheses. Moreover, the results indicate that program enrollment influences students' perceived understanding of programming concepts, with significant variations observed among different programs. This underscores the importance of tailoring programming education to meet the unique needs of students within various programs. However, the SHS track students does not substantially affect their perceptions of programming difficulty or their preferences for effective learning contexts within computer programming.

The study's implications for computer programming education are multifaceted. Gender and program enrollment were found to have minimal impact on students' perceptions of programming difficulties and their preferences for learning contexts, promoting inclusivity and equality in programming education. Educators are encouraged to adopt gender-neutral and program-agnostic teaching approaches. Additionally, the study identified students' strong grasp of fundamental programming concepts but also highlighted challenges in areas like design and syntax. It emphasizes the need for tailored support and resources to address these challenges, promoting an equitable learning environment accessible to all students. Lastly, variations in students' perceived understanding based on program enrollment underline the importance of personalized programming education, urging educators to adapt teaching methods and resources to meet the diverse needs of students in various programs.

Based on these findings, it is recommended that programming educators and institutions prioritize inclusivity in their programming courses, ensuring that all students, regardless of gender or program enrollment, have equal opportunities to excel. Additionally, recognizing the variations in perceived understanding of programming concepts among different programs, educators should consider offering tailored support and resources to address the unique needs of students in each program. Further research is encouraged to explore additional factors influencing students' programming experiences, such as teaching methods, prior programming exposure, and individual learning styles. A more comprehensive understanding of these factors can guide the development of effective pedagogical strategies and resources to enhance the quality of programming education and promote student success. Ultimately, the goal should be to create a diverse and inclusive programming education environment that empowers all students to thrive in computer programming.

5. References

Alipio, M. (2020). Academic Adjustment and Performance among Filipino Freshmen College Students in the Health Sciences: Does Senior High School Strand Matter? *EdArXiv Preprints,* 1–12. https://doi.org/10.35542/osf.io/xq4pk

Alshahrani, S., Ahmed, E., & Ward, R. (2017). The influence of online resources on student–lecturer relationship in higher education: a comparison study. *Journal of Computers in Education*, *4*(2), 87–106. https://doi.org/10.1007/s40692-017-0083-8

Atta, I. A., Senior. (2021). Mediating Effect of Students' Perception in Programming on the Relationship Between Project Work and College Students' Interest in Programming. *Research Square (Research Square).* https://doi.org/10.21203/rs.3.rs-935163/v2

Atta, I. A., Senior, & Atta, I. A. (2021). College students' interest in programming. *Research Square (Research Square).* https://doi.org/10.21203/rs.3.rs-1099754/v1

Baist, A., & Pamungkas, A. S. (2017). Analysis of student difficulties in computer programming. *Volt: Jurnal Ilmiah Pendidikan Teknik Elektro*, *2*(2), 81. https://doi.org/10.30870/volt.v2i2.2211

Barlow-Jones, G. (2019). The Struggles Experienced by First Year Computer Programming Students at a University in South Africa. *Diamond Scientific Publishing*. https://doi.org/10.33422/2nd.icre.2019.12.952

Cabo, C. (2019). Fostering Problem Understanding as a Precursor to Problem-Solving in Computer Programming. *IEEE Frontiers in Education Conference (FIE).* https://doi.org/10.1109/fie43999.2019.9028664

Cheah, C. S. (2020). Factors Contributing to the Difficulties in teaching and learning of computer Programming: a literature review. *Contemporary Educational Technology*, *12*(2), ep272. https://doi.org/10.30935/cedtech/8247

Du, J., & Wimmer, H. (2019). Hour of Code: A Study of Gender Differences in Computing. *Information Systems Education Journal (ISEDJ)*, *17*(4), 91–100. https://files.eric.ed.gov/fulltext/EJ1219536.pdf

Duran,R., Haaranen,L., & Hellas, A.,(2020). Gender Differences in Introductory Programming: Comparing MOOCs and Local Courses. *Technical Symposium on Computer Science Education (SIGCSE '20).* Association for Computing Machinery (pp. 692–698). USA. https://doi.org/10.1145/3328778.3366852

Harimurti, R., Ekohariadi, E., Munoto, M., & Winanti, E. T. (2019). Analysis of Programming Skills Concept in Developing Problem-Solving Skills. *Jurnal Pendidikan Teknologi Dan Kejuruan*, *25*(1), 43–51. https://doi.org/10.21831/jptk.v25i1.22638

Islam, N., Sheikh, G. S., Fatima, R., & Alvi, F. S. (2019). A study of difficulties of students in learning programming. *Journal of Education & Social Sciences*, *7*(2), 38–46. https://doi.org/10.20547/jess0721907203

McAdams, Terence (2018) *Gender and computer programming: teaching and learning strategies designed to increase the engagement of girls.* EdD thesis, University of Reading.

Muharam, L. O., Ihjon, I., Hijrah, W. O., & Samiruddin, T. (2019). The effect of teaching style on students' motivation and academic achievement: Empirical evidence from public senior high school in Konawe Selatan Regency. *International Journal of Scientific & Technology Research*, *8*(9), 1934–1938.

Müller, L., Silveira, M. S., & De Souza, C. S. (2019). Source Code Comprehension and Appropriation by Novice Programmers: Understanding Novice Programmers' Perception about Source Code Reuse. *SBC Journal on Interactive Systems*, *10*, 96. https://doi.org/10.5753/jis.2019.556

Ouano, J. J. G., Torre, J. F. D. L., Japitan, W. I., & Moneva, J. C. (2019). Factors Influencing on Grade 12 Students Chosen Courses in Jagobiao National High School– Senior High School Department. *International Journal of Scientific and Research Publications*, *9*(1), p8555. https://doi.org/10.29322/ijsrp.9.01.2019.p8555

Piwek, P., & Simon Savage, S., (2020). Challenges with Learning to Program and Problem Solve: An Analysis of Student Online Discussions. *Technical Symposium on Computer Science Education (SIGCSE '20).* Association for Computing Machinery (pp. 494–499). USA. https://doi.org/10.1145/3328778.3366838

Rosminah, S. Mohamad A., Ahmad Z. & Ali, M. (2012). Difficulties in learning programming: Views of students. https://doi.org/10.13140/2.1.1055.7441

Silva, U. F., Ferreira, D. J., Ambrósio, A. P., & De Oliveira, J. L. C. (2022). Problems faced by female computer science undergraduates: a systematic review. *Educação E Pesquisa*, *48*. https://doi.org/10.1590/s1678-4634202248236643eng

Su, Y., Chang, C., Wang, C., & Lai, C. (2022). A study of students' learning perceptions and behaviors in remote STEM programming education. *Frontiers in Psychology*, *13*. https://doi.org/10.3389/fpsyg.2022.962984

Thuné, M., & Eckerdal, A. (2018). Analysis of Students' learning of computer programming in a computer laboratory context. *European Journal of Engineering Education*, *44*(5), 769–786. https://doi.org/10.1080/03043797.2018.1544609

Tondeur, J., Scherer, R., Baran, E., Siddiq, F., Valtonen, T., & Sointu, E. (2019). Teacher educators as gatekeepers: Preparing the next generation of teachers for technology integration in education. *British Journal of Educational Technology*, *50*(3), 1189–1209. https://doi.org/10.1111/bjet.12748

Thinakaran, R., & Ali, R. (2015). Work in progress: An initial review in programming tutoring tools. *2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*. pp. 1-4, https://doi.org/10.1109/TALE.2015.7386006