International Journal of Education, Vocational and Social Science
https://e-journal.citakonsulindo.or.id/index.php/IJEVSS
E-ISSN : 2963 4954
IJEVSS

An Analysis of Learning Difficulties Among First-Year Students in Basic Computer Programming

Dian Atmasani[1]
[1]Universitas Negeri Makassar, Indonesia

Corresponding Author: dian.atmasani@unm.ac.id

Abstract

Learning difficulties in introductory programming are a common phenomenon experienced by first-year vocational students, particularly in the fields of engineering and mechatronics, which require mastery of conceptual, algorithmic logic, syntactic, and technical competencies. This study aims to analyze the level of learning difficulties in the Basic Computer Programming course using a descriptive quantitative approach involving 45 first-year students of the Vocational Mechatronics Education Study Program in the 2025/2026 Academic Year. Data were collected using a five-point Likert scale questionnaire and analyzed through descriptive statistics. The findings indicate that all difficulty indicators fall within the moderate category, with mean values ranging from 2.302 to 2.769. The highest level of difficulty was reported in algorithmic logic (mean = 2.769), reflecting students' limited ability to develop programming logic and understand conditional and looping structures. Syntactic difficulties (mean = 2.711) were observed in interpreting error messages and debugging, while conceptual difficulties (mean = 2.573) indicate that students' mental models of variables, data types, and input–process–output structures have not been fully established. Technical difficulties recorded the lowest mean (2.302), although challenges were still present in using IDEs and configuring directories. The findings also demonstrate interdependencies among indicators, where weak conceptual understanding leads to poor algorithmic reasoning and increased syntactic errors. Based on the results, the implementation of project-based learning strategies, conceptual reinforcement through visualization and structured practice, gradual algorithmic training, and technical mentoring for programming software usage is recommended. This research provides theoretical contributions to literature on programming education within the vocational context and offers practical foundations for designing adaptive and contextual pedagogical interventions to enhance students' learning achievement.

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

Keywords: learning difficulties, basic computer programming, algorithmic logic.

INTRODUCTION

Vocational education in the era of the Industrial Revolution 4.0 demands graduates who not only master theoretical knowledge but also possess strong technical competencies, particularly in the fields of automation, robotics, and mechatronics systems. One of the fundamental competencies required of vocational students is computer programming skills, as programming serves as the foundation for designing control systems, integrating sensors and actuators, and developing embedded system-based devices (Yahaya et al., 2020). Therefore, the Basic Computer Programming course holds a strategic position in supporting students' success in more advanced courses such as robotics, control systems, and industrial automation.

However, various studies have shown that first-year students in technical and vocational programs frequently encounter challenges in understanding basic programming, particularly with regard to conceptual aspects, algorithmic logic, syntax application, and the use of programming software (Qian & Lehman, 2017; Robins, 2019). Conceptual difficulties include understanding variables, data types, and basic algorithms, which serve as the foundation for program development. In the aspect of algorithmic logic, novice students often struggle to construct logical sequences, understand conditional structures (if–else), and master looping concepts (Saputro & Salam, 2021).

Students' success in understanding programming is strongly influenced by contextual instructional design and pedagogical support that is tailored to learner characteristics (Koulouri et al., 2015). In vocational contexts, instructional approaches emphasizing direct practice, mechatronics system simulation, and project-based programming have been shown to be more effective than conventional methods that focus solely on theory. Research by (Gomes & Mendes, 2022) demonstrated that project-based learning enhances both conceptual understanding and students' technical skills, particularly when supported by visualization and hardware integration.

Non-cognitive factors also play a significant role. The perception that programming is a difficult subject often generates negative psychological effects on students' learning motivation (Pratama & Yuliana, 2020). This condition is exacerbated by limited computer laboratory facilities, insufficient internet access, and inadequate experience in using programming software (IDE). In some institutions, these constraints impact students' participation in practical sessions (Fitriani & Sari, 2023).

Watson (Watson et al., 2020) further highlight that first-year students also experience

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

academic transition challenges, shifting from structured learning in secondary schools to autonomous learning in higher education, often resulting in difficulty managing time and interpreting task instructions. In the context of electronics and mechatronics education, programming learning difficulties should be viewed as a multidimensional phenomenon. These challenges do not merely reflect individual weaknesses, but also indicate misalignment between instructional approaches, students' learning readiness, and the academic environment. Therefore, systematic analysis is required to identify factors contributing to learning difficulties to design more effective and contextual learning strategies ((Saeli et al., 2020).

Based on these conditions, this study was conducted to comprehensively analyze the learning difficulties experienced by first-year students in the Basic Computer Programming course of the Vocational Mechatronics Education Study Program during the odd semester of the 2025–2026 academic year. The analysis focused on four main indicators, consistent with the questionnaire instrument: (1) conceptual difficulties, (2) algorithmic logic difficulties, (3) syntactic difficulties, and (4) technical difficulties. The findings are expected to serve as a basis for lecturers in designing instructional approaches, strengthening supporting materials, and formulating relevant interventions to enhance student achievement in basic programming courses.

The benefits of this research are expected to be both theoretical and practical. Theoretically, the results may enrich literature on programming learning difficulties in vocational technical education contexts, particularly concerning conceptual, algorithmic logic, syntactic, and technical aspects. Practically, the findings can guide lecturers in designing learning strategies that are more adaptive to vocational learners' characteristics, including strengthening foundational concepts, assisting in programming logic development, and improving technical competence in the use of programming software. Furthermore, this research may support educational institutions in developing instructional policies that encourage first-year students' success in mastering basic computer programming. Accordingly, this study holds strong urgency in enhancing the quality of learning in the Basic Computer Programming course and fostering graduate competencies relevant to industry needs and 21st-century education.

RESEARCH METHOD

This study employed a descriptive quantitative approach, which aims to describe the phenomenon of learning difficulties experienced by first-year students in the Basic Computer Programming course based on numerical data obtained from the survey. This approach was selected as it aligns with the research objective, which focuses on identifying and analyzing the

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

level of learning difficulties without manipulating the variables being studied (Creswell & Creswell, 2018)The descriptive approach is also widely used in vocational education research to map student learning problems and serve as the basis for the formulation of instructional strategies (Sugiyono, 2021)

The research design used was a survey research design, in which data were collected using a questionnaire developed based on indicators of students' learning difficulties in programming. According to (Cohen et al., 2018), the survey design is suitable for obtaining general insights into respondents' perceptions, attitudes, and experiences on a larger scale. In the context of this study, the survey was used to identify the types and levels of learning difficulties experienced by first-year students based on four main indicators aligned with the questionnaire instrument, namely conceptual difficulties, algorithmic logic difficulties, syntactic difficulties, and technical difficulties. The research subjects were first-year students of the Vocational Mechatronics Education Study Program, Department of Electronics Engineering Education, at a state university in Indonesia. A total of 45 students enrolled in the 2025/2026 academic year were involved as respondents, considering that the population size is still within an acceptable range to be quantitatively analyzed using a population study approach (Arikunto, 2019).

The research was conducted during the odd semester of the 2024/2025 academic year within the Faculty of Engineering. The primary research instrument was a closed-ended questionnaire using a five-point Likert scale, developed based on indicators of programming learning difficulties derived from theoretical references, including the programming learning difficulty model by (Qian & Lehman, 2021) and common indicators of difficulties in introductory programming as outlined in Robins' study (Robins, 2019). The questionnaire was designed to measure the following four main indicators:

1. Conceptual Difficulties, which include understanding variables, data types, algorithms, input–process–output (IPO), and the operational mechanisms of program instructions.
2. Algorithmic Logic Difficulties, which encompass structuring logical steps, understanding conditional statements (if–else), loops, and the ability to relate logical sequences to expected outputs.
3. Syntactic Difficulties, which relate to errors in code writing, interpretation of error messages, understanding of basic code structure, punctuation usage, and the ability to perform debugging of minor coding errors.

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

4. Technical Difficulties, which include the ability to use an Integrated Development Environment (IDE), run programs, manage directories or project files, install or configure software, and confidence in using computers for independent practice.

Each item was measured using a five-point Likert scale (1 = strongly disagree to 5 = strongly agree). A total of 20 statement items were developed based on literature review and tailored to the context of vocational students. According to (Hair et al., 2022), a five-point Likert scale is effective in measuring perceptions and attitudes in technical education research. Data collection was carried out online using Google Forms to facilitate completion and optimize time efficiency. Respondents were instructed to answer honestly based on their learning experiences in the Basic Computer Programming course. Online instrument administration has proven effective in vocational education research, particularly in improving time efficiency and accuracy in digital data processing (Basar, 2021).

The collected data were analyzed using descriptive statistics. The mean value of each item was calculated to determine the level of learning difficulty in each dimension. This analysis aimed to identify which dimension of difficulty was most dominantly experienced by first-year students (Herdiana & Supriatna, 2023). The results were presented in tables and charts to facilitate interpretation and enhance the clarity of conclusions regarding the learning difficulty situation in the Basic Computer Programming course (Sugiyono, 2021).

DISCUSSION

The analysis results indicate that the level of learning difficulties in the Basic Computer Programming course is distributed across four main indicators: conceptual difficulties, algorithmic logic difficulties, syntactic difficulties, and technical difficulties. Overall, all indicators fall within the moderate category, with mean values ranging from 2.302 to 2.769, suggesting that students encounter challenges in several fundamental aspects of programming without exhibiting severe difficulty.

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

International Journal of Education, Vocational and Social Science
https://e-journal.citakonsulindo.or.id/index.php/IJEVSS
E-ISSN : 2963 4954
IJEVSS

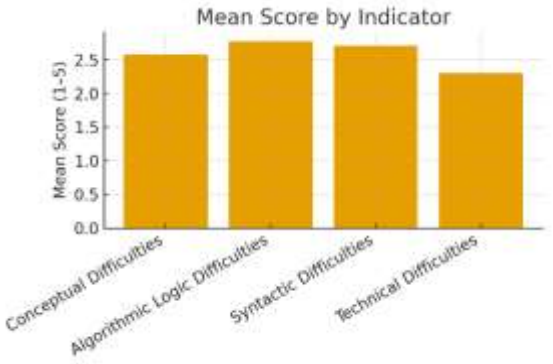


Figure 1. Indicator Charts

The analysis of the conceptual difficulty indicator shows a mean score of 2.573, indicating that a portion of first-year students have not yet fully understood fundamental concepts such as variables, data types, the input–process–output structure, and the role of instructions within a program. Item-wise analysis reveals that the mean scores for each statement range from 2.48 to 2.64, with the item "understanding data types" obtaining the highest mean of 2.644. In terms of frequency, 38% of respondents stated "Agree" regarding their difficulty in understanding data types, while 33%–42% fell into the "Neutral" category for the other items under the same indicator. This distribution suggests that students have not yet developed a stable mental model regarding the basic structure of programming. This condition aligns with the findings of (Sorva et al., 2011), which explain that conceptual misconceptions are an initial barrier commonly encountered by novice learners in introductory programming.

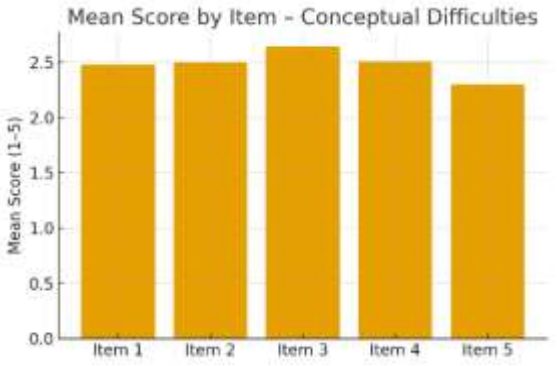


Figure 2. Conceptual Difficulty Chart

The difficulty related to the algorithmic logic indicator obtained the highest mean score of

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

2.769, indicating that students encountered the most significant challenges in structuring programming logic. All items under this indicator recorded mean values ranging from 2.71 to 2.82. Frequency data show that 40–47% of respondents selected “Agree” in experiencing difficulties in designing conditional and looping structures, while 30–38% chose the “Neutral” category. The item with the highest score was the ability to construct conditional logic, with a mean of 2.822, suggesting that nearly half of the students experienced difficulties in understanding if–else structures. This quantitative pattern indicates that students have not yet mastered algorithmic reasoning skills, which involve decomposing problems and determining a sequential solution process. These findings are consistent with (Whalley et al., 2006), who reported that logical challenges often arise because students tend to memorize program structures rather than understand the underlying algorithmic flow.

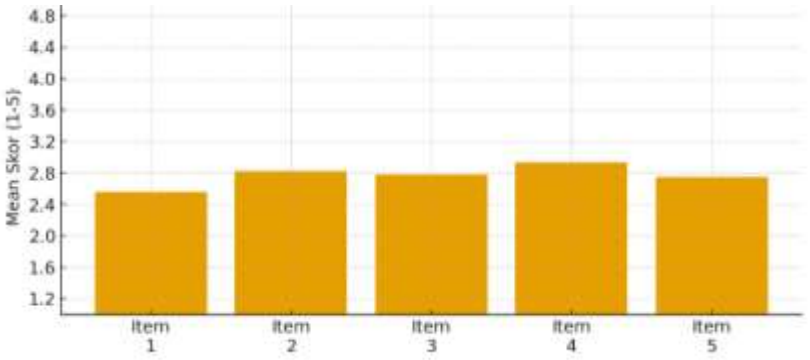


Figure 3. Algorithmic Logic Difficulty Graph

The syntactic difficulty indicator recorded a mean value of 2.711, indicating that students experienced a considerable level of difficulty in understanding coding conventions, particularly related to punctuation placement, debugging, and interpretation of error messages. Frequency distribution data show that 39–49% of respondents selected “Agree” regarding experiencing syntactic difficulties, with the item “understanding error messages” obtaining the highest mean score of 2.756. This figure suggests that nearly half of the students were unable to independently identify the causes of programming errors. These difficulties are reinforced by the recurring patterns of mistakes observed during practical sessions, such as incorrect use of brackets, semicolons, or calling undeclared variables. This phenomenon is consistent with the findings of (Fitzgerald et al., 2008), who stated that understanding error messages is a core challenge in introductory programming learning.

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

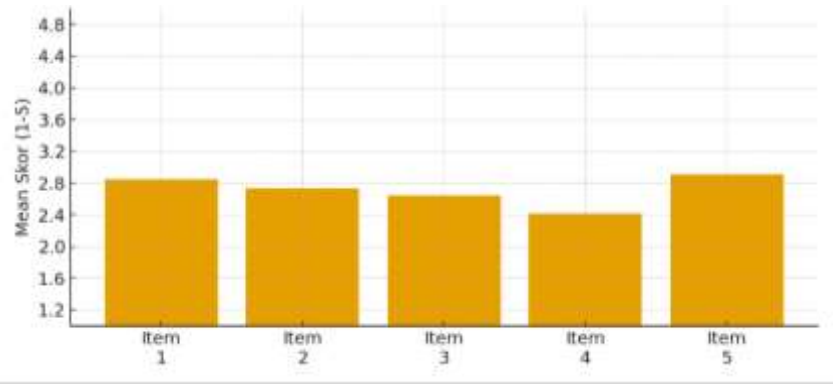


Figure 4. Syntactic Difficulty Chart

The indicator with the lowest level of difficulty was technical difficulties, with a mean score of 2.302. However, technical challenges were still notable in several items, particularly those related to directory management and program execution within the IDE. The mean values for the technical indicator ranged from 2.16 to 2.42, with the item "directory and file configuration" obtaining a mean of 2.422, where 27% of respondents selected "Agree" in experiencing difficulties and 12% selected "Strongly Disagree", indicating the presence of a group of students who were entirely unfamiliar with using development software. These results reinforce the findings of (Lahtinen et al., 2005), who stated that technical obstacles in using IDEs may reduce the effectiveness of programming learning, particularly when students lack prior experience.

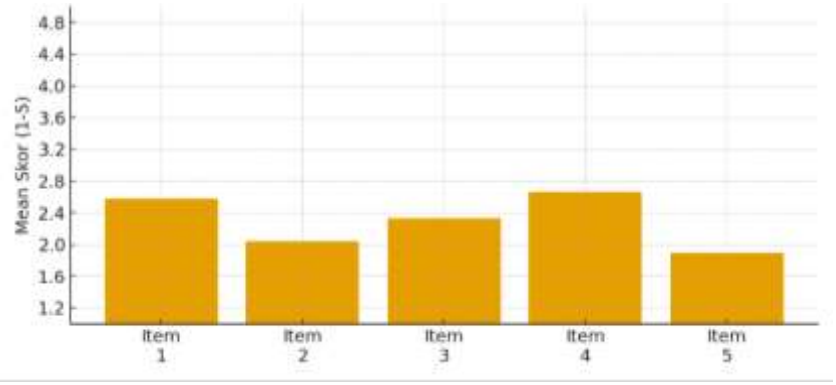


Figure 5. Technical Difficulty Chart

Overall, the inter-indicator relationship pattern shows that conceptual difficulties have a direct impact on algorithmic logic difficulties, which subsequently influence the frequency of syntactic errors. Students who do not fully grasp fundamental concepts tend to struggle in

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

constructing logical flow and, as a result, make more frequent mistakes when writing code. This aligns with the findings of (Zhong & Xia, 2020), which indicate that the use of scaffolding in learning can reduce both logical and syntactic errors. The quantitative pattern of this study also shows that students tended to select the "Neutral" category for several items, suggesting uncertainty in self-assessment, consistent with the illusory of competence phenomenon commonly reported in programming studies (Brown & Kölling, 2015).

From a comparative perspective, the quantitative findings of this study are consistent with international research conducted over the past five years. For example, Whalley et al. (2006) reported that failure to understand program execution (code tracing) is a major source of logical and syntactic challenges. (Rivers & Koedinger, 2017) further found that initial misconceptions about code structure serve as strong predictors of long-term programming performance. Thus, the results of this study are not only locally relevant but also align with global empirical patterns regarding introductory programming learning. By integrating quantitative findings with theoretical support, it can be concluded that first-year students require targeted instructional interventions focused on strengthening conceptual understanding, practicing algorithmic reasoning, structured debugging, and technical support related to the use of IDEs.

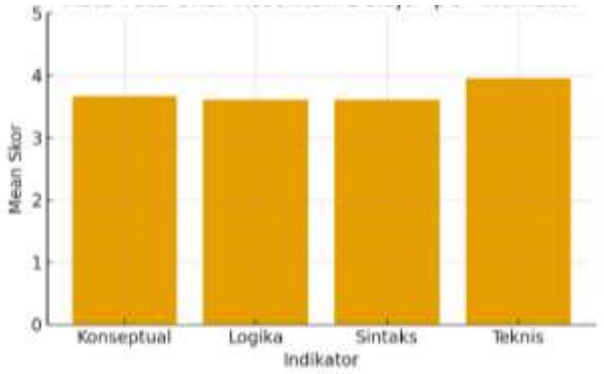


Figure 6. Average Learning Difficulty Score Graph

CONCLUSION

This study aimed to analyze the level of learning difficulties experienced by first-year students in the Basic Computer Programming course based on four main indicators: conceptual difficulties, algorithmic logic difficulties, syntactic difficulties, and technical difficulties. Based on the analysis of responses from 45 participants, it can be concluded that all indicators fall within the moderate difficulty category, with mean values ranging between 2.302 and 2.769. These

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

findings indicate that students encounter various challenges across fundamental aspects of programming, although the level of difficulty has not yet reached a severe category.

The conceptual difficulty indicator, with a mean of 2.573, suggests that some students have not yet fully understood basic concepts such as variables, data types, and the input–process–output structure. This condition indicates that students' initial understanding of programming structure has not yet developed consistently and therefore needs reinforcement through visual learning approaches, contextual examples, and repeated practice. This result is consistent with the findings of (Putri & Sari, 2024), who emphasized that vocational students tend to encounter conceptual challenges when abstract concepts are not delivered using contextual and demonstrative approaches.

The algorithmic logic indicator recorded the highest mean value of 2.769. Nearly half of the respondents reported experiencing difficulties in designing conditional and looping structures, indicating that students' algorithmic reasoning skills are still low. These findings align with research by (Sari & Efendi, 2025), who stated that novice students often face difficulties with control structures because their algorithmic thinking processes have not yet been systematically formed.

Syntactic difficulties, with a mean of 2.711, were reflected in frequent struggles related to interpreting error messages, debugging, and the use of punctuation. These challenges arise because students have not yet been able to link syntactic errors with the logical flow of the program. This is reinforced by findings from (Efendi & Cahyono, 2024), who reported that error messages are a major source of confusion for students in introductory programming courses, particularly when they lack initial experience in debugging.

Meanwhile, the technical difficulty indicator showed the lowest mean value of 2.302, yet still indicated significant challenges related to the use of IDEs, directory configuration, and program execution. Although the difficulty level is relatively lower compared to other indicators, basic technical competence remains essential to ensure students are not hindered during programming practice. These findings are consistent with the research by (Nugroho & Wibawa, 2025), which confirmed that students' technical readiness significantly influences the success of programming practice in vocational education.

Overall, the inter-indicator pattern demonstrates that conceptual difficulties directly affect algorithmic logic difficulties, which subsequently influence the occurrence of syntactic errors. Students who do not understand fundamental concepts tend to fail in constructing logical flow and ultimately make more frequent coding errors. This is supported by the findings of (Rahardja

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

& Abdullah, 2025), which demonstrated that conceptual and algorithmic competence strongly correlates with syntactic accuracy among novice programming learners. The quantitative pattern of this study also shows that students frequently selected the "Neutral" option, indicating uncertainty in self-assessment, similar to the illusory of competence phenomenon among programming learners. Combining quantitative findings with theoretical support, it can be concluded that first-year students require instructional interventions focused on strengthening conceptual understanding, practicing algorithmic reasoning, structured debugging, and technical support in IDE usage.

FUTURE DIRECTIONS

Based on the findings of this study, future research is recommended to adopt intervention-oriented and experimental designs to evaluate the effectiveness of specific pedagogical approaches such as project-based learning, structured algorithmic reasoning exercises, and visualization-based instruction in reducing learning difficulties among novice programming students. Longitudinal studies could be conducted to monitor how students' conceptual understanding and programming abilities develop over time, particularly as they progress through more advanced courses. Further research should also explore affective and non-cognitive factors, including learning motivation, programming anxiety, and self-efficacy, which may influence students' persistence in solving programming problems. Considering the prominent difficulties related to debugging and error interpretation, it is suggested that future studies investigate the integration of structured debugging training models and the development of IDE-assisted learning tools designed to support error analysis, directory configuration, and logic structuring. Additionally, comparative research across different educational levels or study programs may provide broader empirical insights into programming learning challenges. Advanced methods such as learning analytics, process mining, or eye-tracking could also be employed to capture real-time cognitive processing patterns during code tracing and debugging. Overall, researchers are encouraged to design and empirically validate multidimensional instructional models that emphasize conceptual reinforcement, algorithmic reasoning practice, and technical support to optimize programming learning outcomes among first-year vocational students.

REFERENCES

Arikunto, S. (2019). Prosedur Penelitian: Suatu Pendekatan Praktik (Revisi). Rineka Cipta.
Basar, A. M. (2021). The effectiveness of online learning in vocational education during pandemic

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

COVID-19. International Journal of Education and Learning, 3(3), 185–195. https://doi.org/10.31763/ijele.v3i3.370

Brown, N. C. C., & Kölling, M. (2015). Highlighting key words in a code editor can help novices read code better. Proceedings of the 46th ACM Technical Symposium on Computer Science Education, 573–578. https://doi.org/10.1145/2676723.2677266

Cohen, L., Manion, L., & Morrison, K. (2018). Research Methods in Education (8th ed.). Routledge.

Creswell, J. W., & Creswell, J. D. (2018). Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (5th ed.). Sage Publications.

Efendi, H., & Cahyono, B. (2024). Examining the impact of block-based visual programming environments on motivation and learning outcomes of novice programming students in Indonesia. Indonesian Journal of Teaching in Science, 4(1), 45–58. https://ejournal.upi.edu/index.php/IJoTis/article/download/78732/29888

Fitriani, A., & Sari, Y. (2023). Analisis kesulitan belajar mahasiswa pada mata kuliah pemrograman dasar: Studi kasus di politeknik negeri. Jurnal Pendidikan Vokasi, 13(1), 45–56. https://doi.org/10.21831/jpv.v13i1.56789

Fitzgerald, S., Lewandowski, G., Eagan, R. E., Sandler, B., & Thomas, L. (2008). Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. ACM SIGCSE Bulletin, 40(1), 177–181. https://doi.org/10.1145/1352322.1352191

Gomes, A., & Mendes, A. J. (2022). Difficulties in learning programming: A review. Education and Information Technologies, 27(3), 3141–3162. https://doi.org/10.1007/s10639-021-10747-5

Hair, J. F., Page, M., & Brunsveld, N. (2022). Essentials of Business Research Methods (5th ed.). Routledge.

Herdiana, D., & Supriatna, A. (2023). Analisis kesulitan mahasiswa dalam mata kuliah pemrograman dasar menggunakan pendekatan survei. Jurnal Pendidikan Teknik Elektro, 4(1), 78–89. https://doi.org/10.24037/jpte.v4i1.567

Koulouri, T., Lauria, S., & Macredie, R. D. (2015). Teaching introductory programming: A quantitative evaluation of different approaches. ACM Transactions on Computing Education (TOCE), 14(4), 1–28. https://doi.org/10.1145/2662412

Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. ACM SIGCSE Bulletin, 37(3), 199–202. https://doi.org/10.1145/1151958.1151963

Nugroho, A., & Wibawa, S. (2025). An innovative learning approach for enhancing basic

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

programming skills in Indonesian vocational higher education. TEM Journal, 14(2), 1629–1641. https://www.temjournal.com/content/142/TEMJournalMay2025_1629_1641.pdf

Pratama, A., & Yuliana, D. (2020). Penggunaan media pembelajaran interaktif untuk meningkatkan pemahaman konsep pemrograman dasar mahasiswa. Jurnal Pendidikan Teknologi Dan Kejuruan, 27(1), 55–66. https://doi.org/10.21831/jptk.v27i1.30791

Putri, T., & Sari, R. P. (2024). A preliminary study: The cause of university students' difficulties in programming subject. Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika, 9(4), 123–135. https://www.researchgate.net/publication/378121219_A_preliminary_study_The_cause_of_university_students_difficulties_in_programming_subject

Qian, Y., & Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. ACM Transactions on Computing Education (TOCE), 18(1), 1–24. https://doi.org/10.1145/3077618

Qian, Y., & Lehman, J. D. (2021). Correlates of Success in Introductory Programming: A Study with Middle School Students. Journal of Research on Technology in Education, 53(4), 432–449. https://doi.org/10.1080/15391523.2020.1778856

Rahardja, U., & Abdullah, M. (2025). A systematic review of challenges in teaching and learning programming in higher education. International Journal of Computing Sciences, 3(1), 12–25. http://ijcs.net/ijcs/index.php/ijcs/article/view/4592

Rivers, K., & Koedinger, K. R. (2017). Data-driven hint generation in vast solution spaces: A self-improving python programming tutor. International Journal of Artificial Intelligence in Education, 27(1), 37–64. https://doi.org/10.1007/s40593-016-0129-y

Robins, A. (2019). Novice programmers and introductory programming. Computer Science Education, 29(2--3), 135–172. https://doi.org/10.1080/08993408.2019.1565235

Saeli, M., Perrenet, J., Jochems, W., & Zwaneveld, B. (2020). The role of learning analytics in improving computer programming education. Journal of Educational Technology & Society, 23(4), 45–60. https://www.jstor.org/stable/26921467

Saputro, R. E., & Salam, M. (2021). Analisis kesulitan mahasiswa dalam memahami logika pemrograman berbasis Python pada mata kuliah algoritma dan pemrograman. Jurnal Pendidikan Teknologi Dan Kejuruan, 27(2), 123–134. https://doi.org/10.21831/jptk.v27i2.41327

Sari, D. P., & Efendi, A. (2025). Effectiveness of coding in enhancing computational thinking skills among elementary students. Edunesia: Jurnal Ilmiah Pendidikan, 6(2), 150–162.

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License

https://www.edunesia.org/index.php/edu/article/download/1299/590/9544

Sorva, J., Karavirta, V., & Seppälä, L. (2011). A review of recent research on the teaching of programming. In Lecture Notes in Computer Science (Vol. 6882, pp. 32–40). https://doi.org/10.1007/978-3-642-22023-2_4

Sugiyono. (2021). Metode Penelitian Kuantitatif, Kualitatif, dan R&D. Alfabeta.

Watson, C., Li, F. W. B., & Godwin, J. L. (2020). Learning programming at scale: Challenges and opportunities in MOOCs. Journal of Computer Assisted Learning, 36(1), 121–133. https://doi.org/10.1111/jcal.12381

Whalley, J., Lister, R., Thompson, E., Clear, T., Robins, P., Kumar, P. K. A., & Prasad, C. (2006). A multi-national study of reading and tracing skills in novice programmers. ACM SIGCSE Bulletin, 38(3), 119–123. https://doi.org/10.1145/1151869.1151890

Yahaya, N., Abd Kadir, R., Miskon, M. F., & Jusoh, A. (2020). Integrating programming skills in mechatronics education: A case study. International Journal of Emerging Technologies in Learning (IJET), 15(10), 65–79. https://doi.org/10.3991/ijet.v15i10.12793

Zhong, B., & Xia, L. (2020). Threading and anchoring: Scaffolding computational thinking in block-based programming. Journal of Educational Computing Research, 58(6), 1176–1200. https://doi.org/10.1177/0735633120919950

Copyright at authors some right reserved this work is licensed under a Creative Commons Attribution-Share Alike 4.0 International License