# Improving Students' Performances on Computational Thinking: A Systematic Literature Review

NRIAGU CHUKWUNONSO O[1], OKETAYO ABIMBOLA M[2], BAMIDELE OLUCHI JENNIE[3], ODUWOLE OLUWAKEMI O[4]

[1, 2, 3, 4]*Computer Science Department, National Mathematical Centre, Abuja, Nigeria*

***Abstract-*** ***The teaching of computer science in schools has been greatly influenced by the rapid development of technology, making Information Technology literacy a priority. Research literatures and experience in classrooms confirm that students have challenges on algorithms, function and programming. To overcome this, there is need to teach the concepts of mathematics in programming. Mathematics in programming constitutes an important role and should be included in the studies. The use of Mathematics in learning of programming offer a range of techniques to train students in computational thinking and programming skills. Foundational mathematical concepts such as logic, functions, and discrete mathematics, provide the cognitive framework for understanding core programming principles. The process of developing and debugging codes require precise and systematic approach similar to constructing a mathematical proof. Furthermore, students with a solid mathematical background tend to understand programming concepts and perform better in programming related tasks with the ability to design efficient solutions. The study reviewed the role of mathematics as the bedrock subject for computing; the integration of mathematical concepts and design of a systematic approach in training of students in problem solving. This comprehensive survey provides an in-depth examination of existing methods on computational thinking and mathematics approach in problem solving. The results showed that teaching programming with mathematics as an interdisciplinary approach increased students' programming and computational thinking skills. Additionally, the integration of mathematics improved the students' learning processes. This survey provide a valuable resource for researchers, teachers, and policymakers seeking to improve students' problem solving skills and to stimulate further research in this critical area.***

***Indexed Terms-*** ***Programming, Mathematical Concepts, Computational Thinking Skills, Interdisciplinary, Problem Solving***

## I. INTRODUCTION

Programming require a foundation in mathematics and computer science. The introduction of computer science at an early age is important, hence, the requirements for teaching programming to students has increased; with technology focused on programming in the development of students and problem-solving skills (Dagdilelis et al., 2004). Programming is seen as difficult due to the teaching methods. Moreover, programming require a level of knowledge and skills, methods of teaching are categorized in terms of the environment, knowledge of the domain, and the choice of programming tools. These categories make the teaching of the subject both improve the learning of programming (Kazimoglu et al, 2012).

Initiatives through stakeholders such as the Nigeria Computer Society (NCS), Computer Professionals of Nigeria (CPN), and Nigeria Universities Commission (NUC) have made computer science improved curricula, not only in universities, but also in secondary school level. Learning of programming enables users to conceptualize a problem better and allows them to select the best strategy and tools for problem-solving. Therefore, improving the programming skills of students would enable them to improve their computational thinking skills. A problem has been identified, appropriate analysis done, and potential solutions to the problem. Interdisciplinary exercises and examples are given to students during the learning of programming is important to enlighten them to create Innovative solutions and applications. (Nigeria Computer Society (NCS), Computer Professionals of Nigeria (CPN), Nigeria Universities Commission (NUC)).

Therefore, the role of interdisciplinary collaboration is important to the learning of programming. In this study, mathematics was chosen as the second subject since it has connection with programming and computational thinking.

Wing (2006) stated that computational thinking as the principle of thinking to solve computing problems. The learning of programming require a deep understanding of the problem in question, correct algorithmic process and solutions (Guzdial, 2004; Kelleher and Pausch, 2007; Resnick et al., 2009), It is essential for the integration of an interdisciplinary approach on the learning of programming to develop computational thinking and problem solving skills. The learning of programming skills would change the mindset of students in positive way and broadens their knowledge. Problem-solving using analytical skills to identify patterns, develop solutions, identifying bugs and optimize code (Grover and Pea, 2013; Kafai and Burke, 2014; Sengupta et al., 2013; Wing, 2006).

Draganoiu et al. (2017) examined the learning of programming require a significant level of thinking and problem-solving skill that programmers need to solve problems. The translation of abstract problems into programs require problem solving, algorithmic thinking, and computational thinking through interdisciplinary approach (Guzdial and DiSalvo, 2013; Kafai and Burke, 2014; Cassel, 2011; Mahadev and Connor, 2014). Since mathematics play a role in computer science, thus the combination of programming and mathematics creates the means to solve problems through the application of computer programming.

The study focused on three main areas; the use of mathematical concepts, computational thinking and interdisciplinary approach in the teaching and learning of programming. This research is on the application of mathematics in the teaching of programming. The study reviewed the collaboration of mathematics in the learning and the development of programming skills. In a broader context, this research is on the application of mathematics in the learning of programming. It is benefical to educators who teach programming by providing them with related works of training methods and the role of mathematics in learning programming.

### 1.1 Problem Statement

Programming is not considered to be easy because it requires the combined skills of critical thinking, creative thinking, algorithmic thinking and problem solving. In order to facilitate the learning of programming at a young age, educators seek new methods in teaching programming. The teaching of programming language using traditional approach focus more on paper and pen. However, through such traditional methods, students can only practice theories in programming. Studies have shown that learning components of programming language is about syntax and structure (Atmatzidou and Demetriadis, 2016; Jenkins, 2002; Lin et al., 2019). Hence, the challenge is not only in terms of teaching methods, but in the learning of programming.

### 1.2 Paper Organization

This study is organized into several sections. The initial section provides an introduction, problem statement, and outlines the paper's structure. This is followed by a review of related works in Section 2, which examines both current and previous studies. Section 3 elaborates on the methodology employed, detailing the systematic approach used to conduct the survey. The results, findings of the survey are presented in Section 4, with a discussion of the findings in Section 5. The study concludes with a summary of key points in Section 6.

## II. LITERATURE REVIEW

The following subsections consist of literature review of programming at a young age, interdisciplinary approach of Mathematics and Computer Science, the relationship between programming and mathematics.

### 2.1 Programming at a young age

Educators seek new ways to teach programming. Studies have indicated that the fundamentals should be introduction at a young age in stages to better understand programming (Duncan et al., 2014; Malan and Leitner, 2007; Bornat and Dehnadi, 2008). Moreover, learning programming at an early age enable children to learn how to follow algorithms, to be more creative in their thinking by viewing problems in different ways, encouraging them to improve

learning styles and solve problems (Yadav et al., 2017; Grover and Pea, 2013; De Raadt et al., 2002). In addition, Kelleher and Pausch (2007) argued that programming can be learned at any age if the programming taught are simple to learn and the feedback, support provided are adequate.

Holmboe et al. (2001) also stated that teaching should not just be about the transference of the theoretical knowledge to students, but teachers should have the required knowledge while teaching programming. Grover et al. (2014) stated that "CS unplugged" (Computer Science Unplugged); a teaching material based on games and puzzles should be used while introducing programming to students, as it provides several activities for students to practice and learn. On the other hand, storytelling-based projects made the learning of programming easier as it engages students through motivation and creative thinking (Kelleher and Pausch 2007). Thus, studies in these literatures showed how both CS unplugged and storytelling-based activities enhance students' skills towards the learning of programming (Burke and Kafai, 2010; Feaster et al., 2011; Taub et al., 2012; Thies and Vahrenhold, 2013).

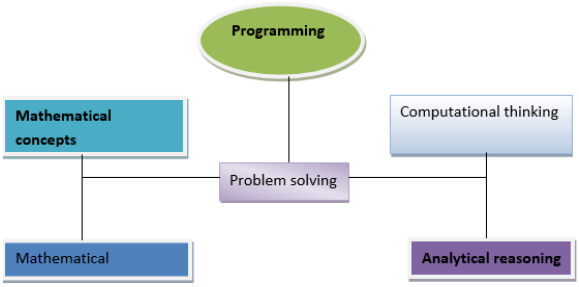


Figure 1: Building blocks in Programming

Learning of programming depends on the knowledge of computational skills and mathematics; these two aspects are required for proficiency in programming. Studies have shown the benefits of utilization of algorithmic thinking, critical thinking, and problem solving, which are components of computational thinking (Aho, 2012; Czerkawski and Lyman, 2015; Ioannidou et al., 2011; Israel et al., 2015; Lu and Fletcher, 2009; Wing, 2006). There are links between mathematical background and programming proficiency. Mathematics is posited as a gateway to programming due to algorithmic thinking, functions and manipulation of symbols of the two subjects. Several empirical studies indicated mathematical background is neccessary and there is correlation between mathematical proficiency and performance in introductory programming courses (Wilson and Shrock, 2001; Bennedsen and Caspersen, 2007).

Moreover, specific mathematical concepts, such as discrete mathematics (Rosen, 2019), Boolean algebra (Kandel A, 1998), logic (Hurley, 2022), and set theory (Halmos, 2017), are frequently used in programming. The concepts from complexity theory enable students analyze size of inputs, runtime and space resources. The mathematical analysis provide the most suitable algorithm for given problems. Students learn about algorithms for sort, search, and graph traversal, as well as various data structures like arrays, linked lists, trees, and hash tables. These concepts enable students to choose appropriate structures to organize and manipulate data efficiently. Mathematics in programming allows students understand the errors they may encounter in programming and how problems are solved.

| Mathematical Concepts | Programming concepts | Application |
|---|---|---|
| Discreet Mathematics | Graph theory, combinatorics | Algorithm design, data structures |
| Algebra | Variable manipulation, functions | functions, equations |
| Geometry | Vectors, transformations | Graphics, game design |
| Number system | Data representation, bitwise operations | binary, decimal, hexadecimal systems |
| Logic | Conditional statements, loops, Logical operators. | Algorithm design |
| Algorithms | Functions | Sorting, search algorithms |

| | | |
|---|---|---|
| Complexity, Big O notation | Comparative analysis | Resource allocation |

Table 1: Mathematical Applications in Programming

## III. RESEARCH METHOD

The research method used waa a survey on the role of mathematics, computational thinking in learning of programming. The literature search was conducted with five (5) databases (IEEE Xplore, Science Direct, Web of Science, ACM Digital Library, Research Gate) and using semantically the same search terms in the keywords, topics, titles, and abstracts of the articles. The researchers developed four criteria: The articles were based on computational thinking, mathematics, programming and research methods.

After the search, three selection stages were applied to reduce the initial set of 450 papers. The remaining final 110 papers were analyzed. In addition, articles written in peer-reviewed English language journals published between 2010 and 2024. However, articles that were not relevant with the technical aspects of computational thinking, mathematics or had no empirical data were removed. Articles that only focused on ICT without reference to mathematics, computational thinking or programming education were also removed. Table 2 shows the articles identified in the search, selected for the analysis with the focus on approach and methodolgy.

### 3.1 Design and Theme of the study

After thorough considerations, the researchers identified four dominant themes on the educational potential of mathematics, computational thinking in programming; the students' motivation and colloboration; and students' performance. Table 2 presents the themes and designs for some relevant articles identified for the literature review. However, it was difficult to determine the design for some studies.

| Article | Theme/ Approach | Focus | Methodology |
|---|---|---|---|
| Lambic (2011) | Mathematics background | The motivation of students to learn mathematics | Pre- and post-questionnaire |
| Moreno-León, Robles, and Román-González (2016) | Programming language, motivation | Impact of introducing programming in several areas like academic performance, student perception, and assessment of projects with Scratch | experimental and control groups, pre- and post-tests |
| Taylor, Harlow, and Forret (2010) | Computational thinking | To enhance mathematical and computational thinking | survey, interviews |
| Ardito, Mosley, and Scollins (2014) | Mathematics | Mathematical concepts, student experiences, problem solving and collaboration | interviews, classroom observations, examination |
| Leonard et al. (2016) | Computational thinking, motivation | Attitudes, computational thinking, self- | Survey |

| | | | |
|---|---|---|---|
| | | efficacy in technology | |
| Sinclair and Patterson (2018) | geometry environments, computational thinking | How computational thinking and mathematical thinking relate | interviews, survey |
| Husain, Kamal, Ibrahim, Huddin, and Alim (2017) | Mathematics | Mathematical thinking skills, problem solving | Pre-test and post-test |
| Bar and Stephenson (2021) | Computational thinking | Computational thinking in schools | Pre-test and post-test |

Table 2: The approach and methodolgies of some selected articles

Results

The four dominant themes from the studies are the educational potential of mathematics; computational thinking in programming; the students' motivation and collaboration; students' performance are presented. In this section, these themes are discussed.

Articles that mentioned student interest, attitudes, contribution, engagement, in learning are discussed under the motivation category. Student performance refers to students' academic achievements as measured quantitatively in test results and improvement in students' computational thinking and problem-solving skills based on similar quantitative and qualitative data, such as tests, questionaires, interviews, and surveys. The increased collaboration between students and teachers are discussed as students' learning processes.

3.2 The relationship between Computational Thinking and Computing

Computational thinking is the ability to organize and analyze data. Once a statement of problem is identified, it should be properly analyzed and possible steps are taken to solve the problem.

Studies have indicated that computer science enable students to utilize algorithmic thinking, critical thinking, and problem-solving skills (Aho, 2012; Czerkawski & Lyman, 2015; Ioannidou et al., 2011; Israel et al., 2015). Grover and Pea (2013) stated that in computational thinking, the decomposition of problem and pattern recognition are applied to solve problems. The use of mathematics improves computational thinking skills because they have the same objective of problem solving. In addition, learning programming is the ability to understand complex tasks through computational skills and translated into codes.

Interdisciplinary Approach

Interdisciplinary refers to the collaborative efforts involving multiple subjects in teaching and learning (Cassel, 2011; Mahadev & Conner, 2014). Establishing a link between Mathematics and Computational thinking would enhance the learning process as it provide students with diverse viewpoint in programming. Interdisciplinary programs require students to think broader, approach problems and find creative solutions. Therefore, the interdisciplinary teaching approach can be used in different levels of education as a key that develop students' knowledge (Cai and Sankaran, 2015).

This research focused on how to improve computational thinking skills through the integration of interdisciplinary method.

3.3 Interdisciplinary Approach of Mathematics and Computer Science

Interdisciplinary approach is considered to be an appropriate means of reducing the complexity of learning of programming (Cassel, 2011; Mahadev and Conner, 2014). In programming, the steps to solve complex problems are the problem definition, designs, solve the problem and create computer programs.

Mathematics, on the other hand, is a broad discipline in which individuals learn to use critical thinking and spatial reasoning in order to solve mathematical problems (Park and Mills, 2014).

In Nigerian educational system, mathematics and computer science are addressed as two distinct subjects; although in actuality, the two disciplines have more in common than many would assume, with modern computer science areas like cyber security that uses techniques that are related or derived from mathematics. An attribute of programming is the definition of logic statements to reduce the complexity to true or false statements. The mathematical representation of this programming aspect is the use of numbers "1" and "0," where "1 implies true" and "0 implies false". Mathematics and computer science has links with other disciplines; however, the relationship between these two science subjects is undeniable and much more than systematic. Firstly, the basic level where mathematics manifests in computer science is in the binary system of communication. Thus, in order to acquire a basic understanding of the computing; an understanding of mathematics is needed. At the secondary school level is to encourage students to embrace both subjects. Herbsleb (2005) stated that in order to cope with the complexity of programming, disciplines related with programming, such as mathematics, should be included in learning of programming. Research has found that people with an understanding of mathematics achieve a deeper understanding of programming. Moreover, there is connection between mathematics and computer science as both subjects aim to understand a problem and logically build their solutions (Burns et al., 2012; Graham and Fennell, 2001; Lu and Fletcher, 2009; Pruski and Friedman, 2014; Bruce et al., 2003).

The accessibility of Information Communication Technology (ICT) resources that are learner-centered improved students' learning experience increase their interest in mathematics (Aghware, et al., 2010; Umar and Musa 2022). Integration of computer science and mathematics is important at an early stage as mathematics is a subject that entails comprehensive problem-solving interactions, this attribute creates an environment where students can learn more about programming. The mathematics in programming require interpretation, design and implementation.

Researchers agree that the mathematics play a role on learning of programming and computer science, because both subjects use reasoning to find solutions (Bruce et al., 2003; Havill and Ludwig, 2007; LeBlanc and Leibowitz, 2006). Mathematics entails understanding various problems and using reasoning to develop mathematical solutions. The combination of mathematics and computer science increases the skills of students in both mathematical reasoning and programming to comprehend different approaches for tasks hence enhance cognitive skills (Stozhko et al., 2015). Hence students examine tasks, its structures, functions, to better understand its concepts, its designs through computational thinking and problem-solving skills.

3.4. The integration of mathematics in programming

The integration of mathematics into learning of programming has emerged as a theme for fostering students' motivation, transforming the perception of mathematical concepts in programming from an abstract and challenging subject to an engaging and innovative one. Programming languages provide an environment where students practice, solve problems, experience a sense of accomplishment, thereby boosting motivation. Several research underscore the positive impact on students' attitudes towards mathematics in programming (Draganoiu et al., 2017; Husain et al, 2017). There is a shift from passive learning to hands on approach. This is supported by studies on block-based programming environments, which showed to be effective in enhancing logical thinking and motivation of students of programming (Malan and Leitner, 2007; Moreno-León et al., 2016)

Empirical studies have consistently demonstrated the motivational benefits of integrating programming into maths classrooms. Research highlighted that using programming as a tool lead to statistically significant improvements in students understanding of mathematics (Lambic, 2011; Ardito, 2014)

The study's findings, derived from collected and analyzed data, are presented below.

Mapping of Published articles

This review synthesizes existing literature, highlighting key themes and objectives across

published articles. The analysis also underscores the importance of mathematics in learning programming. A summary of the reviewed papers is provided in Table 3, offering an overview of the research landscape.

The categories of published articles are presented in Table 3.

| S/N | Article Category | Frequency of Articles | Percentage of Articles |
|---|---|---|---|
| 1 | Computational thinking approach | 33 | 30% |
| 2 | Mathematical based learning | 22 | 20% |
| 3 | Interdisciplinary approach | 12 | 10% |
| 4 | Introductory aspect and Programming | 24 | 21% |
| 5 | Motivation and collobration | 11 | 10% |
| 6 | Other relevant articles | 10 | 9% |
| | Total | 110 | 100% |

Table 3: Published studies on Mathematical reasoning and Computational thinking

## IV. DISCUSSIONS

Mathematics equip students with tools on problem decomposition, and algorithmic thinking and logical reasoning There is a connection between mathematics, logical reasoning, and programming. The training of students in programming requires these mathematical and computer science concepts.

As the review showed, sparse research exists on the educational potential of mathematics in programming education. Most of the articles had results showing better performance in computational thinking, mathematics and motivation to learn programming. The review focused on mathematics, computational thinking and programming. The aim was to map existing research examining the use of mathematical concepts in programming, computational thinking in programming to determine whether there was sufficient evidence to justify the need for integration of mathematics in learning of programming. In 100 selected articles were analyzed to determine the educational potential of mathematics, computational thinking in learning of programming. This study concentrated only on studies discussing mathematics, computational thinking and programming.

Furthermore, teachers should also be retrained prior to implementation in order to ensure its success because the knowledge may only be theoretical, therefore are ill-equipped to teach their students (Gadanidis et al., 2017). Teachers should therefore provide mathematical concepts as an integrated component rather than teaching it separately. There is need for good foundation as the core concepts behind computer's algorithms are about mathematics. Educators with vast knowledge in mathematics provide programming students with problem-solving techniques that incorporate mathematical options in finding appropriate computing solutions.

## CONCLUSION

Strategies are continuously being explored to integrate these mathematical concepts into learning of programming to make abstract concepts understandable, methodologies in problem-solving, teaching relevant mathematical and computational thinking in the area of programming. There is mathematics in programming, the relationship between computer science and mathematics is undeniable, and finding of solutions for given problems (Burns et al., 2012; Dagdilelis et al., 2004). Therefore, review work on the mathematical concepts in learning of programming as mathematics and programming share similarities in creative thinking and solving problems. The role of mathematics and computer science in learning programming with its abstraction, logical reasoning, problem-solving, empower students to become proficient and versatile. Students with the knowledge of mathematical concepts and computer science analyze problems, design, implement, test and refine solutions.

The study examined the teaching of programming through an interdisciplinary approach by integration of mathematics to enhance computational thinking and programming skills. The study showed that the integration of mathematics in computer science played an important role in the improvement of students' programming skills.

REFERENCES


[1] Aghware, F., Egbuna, E.; Aghware, A. & Ugboh, E. (2010). The implications of the internet and associated information and communication technologies (ICT) on science, technology and mathematics (STM) education in Nigeria. Agbor Journal of Science Education, 163-170.

[2] Aho, A. V. (2012). Computation and Computational Thinking. The Computer Journal, 55(7), 832-835. https://doi.org/10.1093/comjnl/bxs074

[3] Ardito, G., Mosley, P., & Scollins, L. (2014). We, robot: Using robotics to promote Collaborative and mathematics learning in a middle school classroom. (Report). Middle Grades Research Journal, 9(3), 73.

[4] Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. Robotics and Autonomous Systems, 75(B), 661-670. https://doi.org/10.1016/j.robot.2015.10.008

[5] Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? Inroads, 2(1), 48-54. https://doi.org/10.1145/1929887.1929905

[6] Belanger, C., Christenson, H., & Lopac, K. (2018). Confidence and common challenges: the effects of teaching computational thinking to students ages 10-16. St. Catherine University, MN. https://sophia.stkate.edu/maed/267

[7] Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. ACM SIGCSE Bulletin, 39(2), 32-36. https://doi.org/10.1145/1272848.1272879

[9] Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. Digital Experiences in Mathematics Education, 3(2), 115-138. https://doi.org/10.1007/s40751-017-0028-x

[10] Bornat, R., & Dehnadi, S. (2008). Mental models, consistency and programming aptitude. S. Hamilton & M. Hamilton (Eds.), Proceedings of the Tenth Conference on Australasian Computing Education, 78, 53-61. Australian Computer Society. https://dl.acm.org/doi/epdf/10.5555/1379249.1379253

[11] Borrego, M., & Newswander, L. K. (2010). Definitions of interdisciplinary research: Toward graduate-level interdisciplinary learning outcomes. The Review of Higher Education, 34(1), 61-84. https://doi.org/10.1353/rhe.2010.0006

[12] Bruce, K. B., Drysdale, R. L. S., Kelemen, C., & Tucker, A. (2003). Why math? Communications of the ACM, 46(9), 40-44. https://doi.org/10.1145/903893.903918

[13] Bort, H., & Brylow, D. (2013). CS4Impact: Measuring computational thinking concepts present inCS4HS participant lesson plans. Proceedings of the 44th ACM Technical Symposium on Computer Science Education, 427–432.

[14] Burke, Q., & Kafai, Y. B. (2010). Programming & storytelling: opportunities for learning about coding & composition. In Proceedings of the 9th International Conference on Interaction Design and Children, 348-351. https://doi.org/10.1145/1810543.1810611

[15] Burns, R., Pollock, L., & Harvey, T. (2012). Integrating hard and soft skills: software engineers serving middle school teachers. In Proceedings of the 43rd ACM technical symposium on Computer Science Education, 209-214. ACM.

[16] Cai, W., & Sankaran, G. (2015). Promoting critical thinking through an interdisciplinary

study abroad program. Journal of International Students, 5(1), 38-49. https://doi.org/10.32674/jis.v5i1

[17] Cassel, L. N. (2011). Interdisciplinary computing is the answer: now, what was the question?

[18] ACM Inroads, 2(1), 4-6. https://doi.org/10.1145/1929887.1929888

[19] Coenraad, M., Mills, K., Byrne, V. L., & Ketelhut, D. J. (2020). Supporting teachers to integrate computational thinking equitably. Proceedings of 2020 Research on Equity and Sustained Participation in Engineering, Computing, and Technology. https://doi.org/https://doi.org/10.1109/RESPECT49803.2020.9272488

[20] Czerkawski, B. C., & Lyman, E. W., III. (2015). Exploring issues about computational thinking in higher education. TechTrends, 59(2), 57-65. http://dx.doi.org/10.1007/s11528-015-0840-3

[21] Dagdilelis, V., Satratzemi, M., & Evangelidis, G. (2004). Introducing secondary education students to algorithms and programming. Education and Information Technologies, 9(2), 159-173. https://doi.org/10.1023/B:EAIT.0000027928.94039.7b

[22] Davenport, J. H., Wilson, D., Graham, I., Sankaran, G., Spence, A., Blake, J., & Kynaston, S. (2014).

[23] Interdisciplinary teaching of computing to mathematics students: Programming and discrete mathematics. MSOR Connections, 14(1), 1-8. https://doi.org/10.11120/msor.2014.00021

[24] De Raadt, M., Watson, R., & Toleman, M. (2002). Language trends in introductory programming courses. In Proceedings of the 2002 Informing Science+ Information Technology Education Joint Conference, 229-337. Informing Science Institute.

[25] Draganoiu, R., Moldoveanu, A., & Braescu, A. (2017). The Math of Programming-Interdisciplinary Approach. The International Scientific Conference eLearning and Software for Education, Carol National Defense University, 2, 473-481. https://doi.org/10.12753/2066-026X-17-152

[26] Duncan, C., Bell, T., & Tanimoto, S. (2014). Should your 8-year-old learn coding? Schulte (Ed.), Proceedings of the 9th Workshop in Primary and Secondary Computing Education, 60-69. ACM

[27] Feaster, Y., Segars, L., Wahba, S. K., & Hallstrom, J. O. (2011). Teaching CS unplugged in the high school (with limited success). In Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, 248-252. ACM.

[28] Fofang, J. S., Weintrop, D., Walton, M., Elby, A., & Walkoe, J. (2020). Mutually supportive mathematics and computational thinking in a fourth-grade classroom. In Gresalfi, M. and Horn, I. S. (Eds.), The Interdisciplinarity of the Learning Sciences, 14th International Conference of the Learning Sciences (ICLS), Nashville, Tennessee: International Society of the Learning Sciences, 3, 1389-1396.

[29] Fojtik, R. (2014). Design patterns in the teaching of programming. Procedia - Social and Behavioral

[30] Sciences, 143, 352-357. https://doi.org/10.1016/j.sbspro.2014.07.493

[31] Gadanidis, G. (2017). Artificial intelligence, computational thinking, and mathematics education.

[32] The International Journal of Information and Learning Technology, 34(2), 133–139. doi:10.1108/ijilt-09-2016-0048

[33] Gadanidis, G., Cendros, R., Floyd, L., & Namukasa, I. (2017). Computational thinking in mathematics teacher education. Contemporary Issues in Technology and Teacher Education, 17(4), 458-477.

[34] Graham, K. J., & Fennell, F. (2001). Principles and standards for school mathematics and teacher education: Preparing and empowering teachers. School Science and Mathematics, 101(6), 319-327. https://doi.org/10.1111/j.1949-8594.2001.tb17963.x

[35] Grover, S., Cooper, S., & Pea, R. (2014). Assessing computational learning in K-12. In Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education, 57-62. ACM.

[36] Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field.

[37] Educational Researcher, 42(1), 38-43. https://doi.org/10.3102%2F0013189X12463051

[38] Guzdial, M., & DiSalvo, B. (2013). Computing education: Beyond the classroom. Computer, 46(9), 30-31. https://doi.org/10.1109/MC.2013.306

[39] Guzdial, M. (2004). Programming environments for novices. Computer Science Education Research, 127-154. Routledge Falmer.

[40] Halmos, P.R. (2017) Naive Set Theory. Courier Dover Publications, New York. ISBN0387900926, 9780387900926.

[41] Harvie, D. P., Estes, T., & Major, K. (2018). Use of Commercial Online Training to Augment Programming Language Education. In Proceedings of the 19th Annual SIG Conference on Information Technology Education (SIGITE '18) (p. 89). ACM. https://doi.org/10.1145/3241815.3241832

[42] Havill, J. T., & Ludwig, L. D. (2007). Technically speaking: fostering the communication skills of computer science and mathematics students. ACM SIGCSE Bulletin, 39(1), 185-189. https://doi.org/10.1145/1227504.1227375

[43] Herbsleb, J. D. (2005). Beyond computer science. In Proceedings of the 27th International Conference on Software Engineering, 23-27. https://doi.org/10.1145/1062455.1062466

[44] Holmboe, C., McIver, L., & George, C. E. (2001). Research Agenda for Computer Science Education. In Annual Workshop of the Psychology of Programming Interest Group, 13, 207-223. Sheffield Hallam University.

[45] Hurley, P. J. (2012). A Concise Introduction to Logic (11th ed.). Boston, MA: Wadsworth.

[46] Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers and Education. https://doi.org/10.1016/j.compedu.2018.07.004

[47] Husain, H., Kamal, N., Ibrahim, M. F., Huddin, A. B., & Alim, A. A. (2017). Engendering problem solving skills and mathematical knowledge via programming. Journal of Engineering Science and Technology, 12(12), 1–11.

[48] Ioannidou, A., Bennett, V., Repenning, A., Koh, K. H., & Basawapatna, A. (2011). Computational Thinking Patterns [Paper presentation]. Annual Meeting of the American Educational Research Association, New Orleans, LA.

[49] Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. Computers & Education, 82, 263-279. https://doi.org/10.1016/j.compedu.2014.11.022

[50] Jehlička, V. (2010). Interdisciplinary relations in teaching of programming. In Proceedings of the 2010 International Conference on Applied Computing. World Scientific and Engineering Academy and Society (WSEAS), 33-38.

[51] Jenkins, T. (2002). On the difficulty of learning to program. In Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, 4, 53-58.

[52] Jones, C. (2010). Interdisciplinary approach-advantages, disadvantages, and the future benefits of interdisciplinary studies. Essai, 7(1), Article 26. http://dc.cod.edu/essai/vol7/iss1/26

[53] Kafai, Y., & Burke, Q. (2014). Connected code: why children need to learn programming. MIT.

[54] Kale, U., Akcaoglu, M., Cullen, T., Goh, D., Devine, L., Calvert, N., & Grise, K. (2018). Computational what? Relating computational

thinking to teaching. TechTrends, 62(6), 574-584. https://doi.org/10.1007/s11528-018-0290-9

[55] Kandel A, Langholz G, Mott J (1998). Foundations Of Digital Logic Design. World Scientific Publishing Company, ISBN 9813105100, 9789813105102.

[56] Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play.

[57] Kelleher, C., & Pausch, R. (2007). Using storytelling to motivate programming. Communications of the ACM, 50(7), 58-64. https://doi.org/10.1145/1272516.1272540

[58] Lambic, D. (2011). Presenting practical application of mathematics by the use of programming software with easily available visual components. Teaching Mathematics and Its Applications: An International Journal of the IMA, 30(1), 10–18. doi: 10.1093/teamat/hrq014

[59] LeBlanc, M. D., & Leibowitz, R. (2006). Discrete partnership: a case for a full year of discrete math. ACM SIGCSE Bulletin, 38(1), 313-317. https://doi.org/10.1145/1121341.1121438

[60] Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2019). Computational Thinking from a Disciplinary Perspective: Integrating Computational Thinking in K-12 Science, Technology, Engineering, and Mathematics Education. Journal of Science Education and Technology, 29(1), 1–8. doi:10.1007/s10956-019-09803-w Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. Journal of Science Education and Technology, 25(6), 860–876. doi: 10.1007/s10956-016-9628-2

[61] Lin, Y.-T., Wang, M.-T., & Wu, C.-C. (2019). Design and Implementation of Interdisciplinary STEM Instruction: Teaching Programming by Computational Physics. Asia-Pacific Education Researcher, 28(1), 77-91. https://doi.org/10.1007/s40299-018-0415-0

[62] Lu, J. J., & Fletcher, G. H. (2009). Thinking about computational thinking. ACM SIGCSE Bulletin, 41(1), 260-264. https://doi.org/10.1145/1508865.1508959

[63] Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? Computers in Human Behavior, 41, 51-61. https://doi.org/10.1016/j.chb.2014.09.012

[64] Mahadev, A., & Conner, S. (2014). An interdisciplinary approach to teaching a computer ethics course. Journal of Computing Sciences in Colleges, 29(5), 202-207. https://dl.acm.org/doi/10.5555/2600623.2600663

[65] Malan, D. J., & Leitner, H. H. (2007). Scratch for Budding Computer Scientists. In SIGCSE '07: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education, ACM, 223-227. https://doi.org/10.1145/1227504.1227388

[66] Moreno-León, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it
Journal of Information Technology Education: Research, 15, 283–303. doi: 10.28945/3521

[67] Oliveira Aureliano, V. C. (2013). A methodology for teaching programming for beginners. In Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research, 169-170. ACM.

[68] Olteanu C. (2022). Programming, mathematical reasoning and sense-making. International Journal of Mathematical Education in Science and Technology 53(8), 2046-2064.

[69] Park, J. Y., & Mills, K. A. (2014). Enhancing interdisciplinary learning with a learning management system. Journal of Online Learning and Teaching, 10(2), 299-313. https://jolt.merlot.org/vol10no2/park_0614.pdf

[70] Porter, L., Guzdial, M., McDowell, C., & Simon, B. (2013). Success in introductory

programming: What works? Communications of the ACM, 56(8), 34-36. https://doi.org/10.1145/2492007.2492020

[71] Pruski, L., & Friedman, J. (2014). An integrative approach to teaching mathematics, computer science, and Physics with Matlab. Mathematics and Computer Education, 48(1), 6-18.

[72] Rosen K. (2019). Discrete Mathematics and Its Applications. 8th Edition, illustrated Publisher McGraw-Hill, ISBN1260091996, 9781260091991.

[73] Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2011). Teaching programming in secondary school: a pedagogical content knowledge perspective. Informatics in Education, 10(1), 73-88. https://www.ceeol.com/search/article-detail?id=69618

[74] Seng, W. Y., & Yatim, M. H. M. (2014). Computer game as learning and teaching tool for object-oriented programming in higher education institution. Procedia - Social and Behavioral Sciences, 123, 215-224. https://doi.org/10.1016/j.sbspro.2014.01.1417

[75] Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: a theoretical framework. Education and Information Technologies, 18, 351-380. https://doi.org/10.1007/s10639-012-9240-x

[76] Sinclair, N., & Patterson, M. (2018). The Dynamic Geometrisation of Computer Programming.

[77] Mathematical Thinking and Learning, 20(1), 54–74. doi: 10.1080/10986065.2018.1403541 Stozhko, N., Bortnik, B., Mironova, L., Tchernysheva, A., & Podshivalova, E. (2015).

[78] Interdisciplinary project-based learning: technology for improving student cognition.

[79] Research in Learning Technology, 23, Article 27577. https://doi.org/10.3402/rlt.v23.27577

[80] Tan, J., Guo, X., & Zheng, W. (2014). Case-based teaching using the Laboratory Animal System for learning C/C++ programming. Computers & Education, 77, 39-49. https://doi.org/10.1016/j.compedu.2014.04.003

[81] Taub, R., Armoni, M., & Ben-Ari, M. (2012). CS unplugged and middle-school students' views, attitudes, and intentions regarding CS. ACM Transactions on Computing Education (TOCE), 12(2), Article 8. https://doi.org/10.1145/2160547.2160551.

[82] Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. Procedia - Social and Behavioral Sciences, 8, 561–570. doi: 10.1016/j.sbspro.2010.12.078

[83] Thies, R., & Vahrenhold, J. (2013). On plugging unplugged into CS classes. In Proceedings of the 44th ACM Technical Symposium on Computer Science Education, 365-370.

[84] Umar, A., & Musa, S. (2022). ICT and Learning of Mathematics in Nigeria. Journal of Mathematics Instruction, Social Research and Opinion, 1(3), 143-152. ISSN 2962-7842.

[85] Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016).

[86] Defining computational thinking for mathematics and science classrooms. Journal of Science Education and Technology, 25, 127-147

[87] Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: A study of twelve factors. ACM SIGCSE Bulletin, 33(1), 184-188. https://doi.org/10.1145/366413.364581

[88] Wing, J. M. (2008). Computational thinking Communications of the ACM, 49(3), 33-35. https://doi.org/10.1145/1118178.1118215

[89] Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. P. J. Rich & C. B. Hodges (Eds.), Emerging research, practice, and policy on computational thinking, 205-220. Springer.