Research Article

An Overviuw Of The Teaching And Learning Process Basic Programming In Algorithm And Programming Courses

Wilda Susanti[1*], Jalius Jama[2], Krismadinata[3], Dochi Ramadhani[4], Torkis Nasution[5]

[1]Institut Bisnis dan Teknologi Pelita Indonesia, Indonesia
[2]Universitas Negeri Padang, Indonesia
[3]Universitas Negeri Padang, Indonesia
[4]IKIP PGRI Pontianak, Indonesia
[5]STMIK Amik Riau, Indonesia
*wilda@lecturer.pelitaindonesia.ac.id

Article History: Received: 11 January 2021; Accepted: 27 February 2021; Published online: 5 April 2021

Abstract: In this paper, we review the literature related to computer programming learning, where Algorithms and Programming are the topic domains of the Informatics and Computer science clusters. There are 4 competencies in learning outcomes, such as: 1) understand algorithmic concepts; 2) master algorithm concepts and principles; 3) master programming language concepts; and 4) master programming languages and algorithms. The main focus of this review is on beginner programming and topics related to student difficulties in learning programming. Various problems experienced by beginners were identified from the literature to some of the solutions offered by researchers.

Keywords: Programming, Student difficulty, Competence

1. Introduction

IT (Information Technology) graduates are still the prima donna. The field of information technology or other businesses that are supported by the application of information technology at present and in the future remains the government's concern. In 2020, the number of higher education graduates in Indonesia is around 6 million people per year, assuming 7 percent of students take the IT discipline(Tutang, 2009). Ironically, from the data of the Informatics and Computer Education Association (Aptikom) IT graduates in Indonesia only 10 percent is absorbed by the industry of the 25,000 IT graduates in Indonesia. Another case in India, 25 percent of IT workforce graduates are absorbed by the workforce of 3 million IT workers (Nair, 2020). The gap between universities and the industrial world is due to the unfulfilled competencies of graduates (Muchlis et al., 2020).

Education should be oriented to the world of work, the emphasis is not solely on cognitive aspects, but other personality aspects such as affective and psychomotor aspects are needed (Muhson et al., 2012). To face this challenge, higher education institutions must prepare an innovative learning system, and improve the competence of graduates who have 21st century skills (Learning and Innovation Skills), namely 4C skills (Critical thinking, Creativity, Collabarotion, and Communication (Muhson et al., 2012; Bakar et al., 2013;Cradler, J., McNabb, M., Freeman, M., and Burchett, 2002;Verawardina et al., 2020;Nouri et al., 2020; Trilling & Fadel, 2010).

KKNI APTIKOM (2015) establishes the standard of learning outcomes (Learning Outcomes) for subjects related to the subject area in the field of Computer Science / Informatics at the undergraduate level that Algorithm and Programming with reference to learning outcomes based on the KKNI APTIKOM which aims to meet the qualifications of Bachelor of Computer graduates in the S1 Information Engineering Program. Competence Algorithms and Programming are specifically defined and relevant to competencies based on the KKNI APTIKOM. Thus Programming Algorithms include aspects of the topic domain of the subjects in the S1 Informatics Engineering study program with related subjects, namely, Programming Basics, Data Structures and Algorithms, Algorithm Design and Analysis, Declarative Programming, Automata Language Theory, Intelligent Systems, Object Oriented Programming, and Web Programming. To achieve graduate competence, the core learning outcomes of the study program are formulated which refers to learning outcomes. The core learning outcomes in the field of Information Technology / Computer Science are grouped into six competency domains, namely (1) Mathematics (2) Basic Computer Science; (3) Algorithms and Programming; (4) Software Engineering; (5) Computer Systems; and (6) Life Skills (Success Skills).

Table 1. Learning outcomes of the Information Engineering / Computer Science Study Program to meet the qualifications of Bachelor graduates according to the KKNI level by referring to the learning outcomes recommended by APTIKOM

Table 1. Learning Outcomes of the Informatics Engineering / Computer Science Study Program

Topic Domain	Learning Outcomes
Algorithms and Programming	1. Understand the concepts of algorithms and complexity, covering the central concepts and skills needed to design, implement and analyze algorithms

	to solve problems. 2. Mastering the concepts and principles of algorithms and computer science theory that can be used in computer-based system modeling and design 3. Mastering programming language concepts, and being able to compare various solutions and various programming language models 4. Mastering programming languages and algorithms related to application programs to manipulate image, graphic and image models

Algorithms and Programming are the first programming courses given to students. This subject requires a set of cognitive processes that naturally develop through practice, writing solutions with algorithms(Francisco & Ambrosio, 2015). Programming is part of the curriculum in computer science education and is a basic skill that all computer science students should learn. Teaching programming languages aims to enable students to develop a set of skills needed to design computer programs and systems capable of solving real problems (Gomes et al., 2008). Learning programming requires students to increase creativity, teamwork, innovation and knowledge of data structures and algorithms(Nair, 2020). Computer programming skills require several types of thinking skills such as logical thinking and problem solving(Ozyurt, 2015). Developing a computer program requires the ability to translate and model one's thinking, problems and solutions in natural language into the selected programming language(Renumol et al., 2010). Computer programming requires problem-solving strategies and involves a lot of programming logic activities that pose challenges for students (Wang & Hwang, 2017).

Based on a survey between institutions around the world on research (Bennedsen & Caspersen, 2007) in 15 countries, in 63 institutions found 33% failure rate to learn computer programming or pass 67%. This research is strengthened by(Watson & Li, 2014) conducting a survey for quantitative evidence that has been done by (Bennedsen & Caspersen, 2007) whether the failure rate in computer programming increases or decreases over time. With data from 14 countries in 51 institutions, the failure rate was 32.3% or a graduation rate of 67.7%.

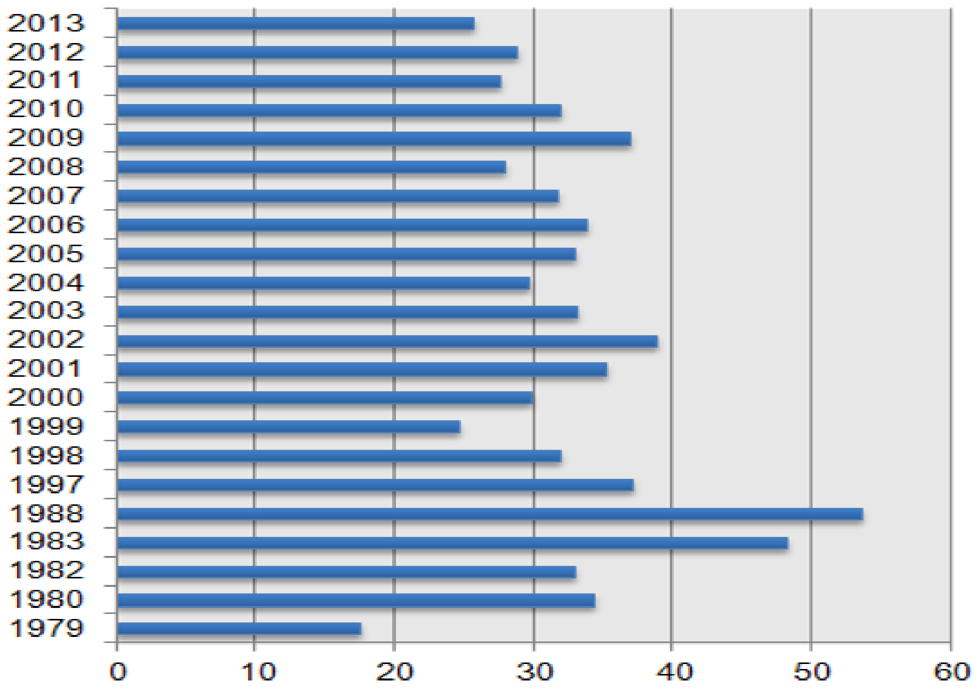

Figure 1. Students who do not graduate from year to year(Watson & Li, 2014)

Figure 1. Shows the mean percentage of students who did not graduate by year ranging from 53.5% to 17.4% that there was no significant increase in the pass rate of computer programming over time.

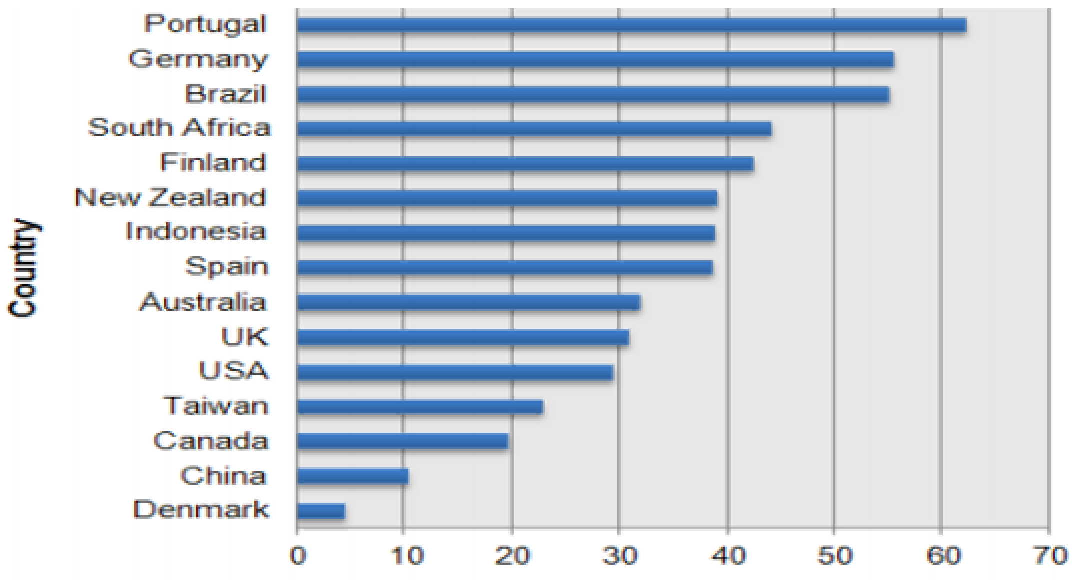

Figure 2. Students who do not pass by country(Watson & Li, 2014)

The educational practice of each country will be different, further (Watson & Li, 2014) giving the average percentage of students who do not pass computer programming by country, as seen in Figure 2. Portugal has the highest failure rate followed by Germany and Brazil. Meanwhile, Indonesia is ranked 7th out of 15 countries.

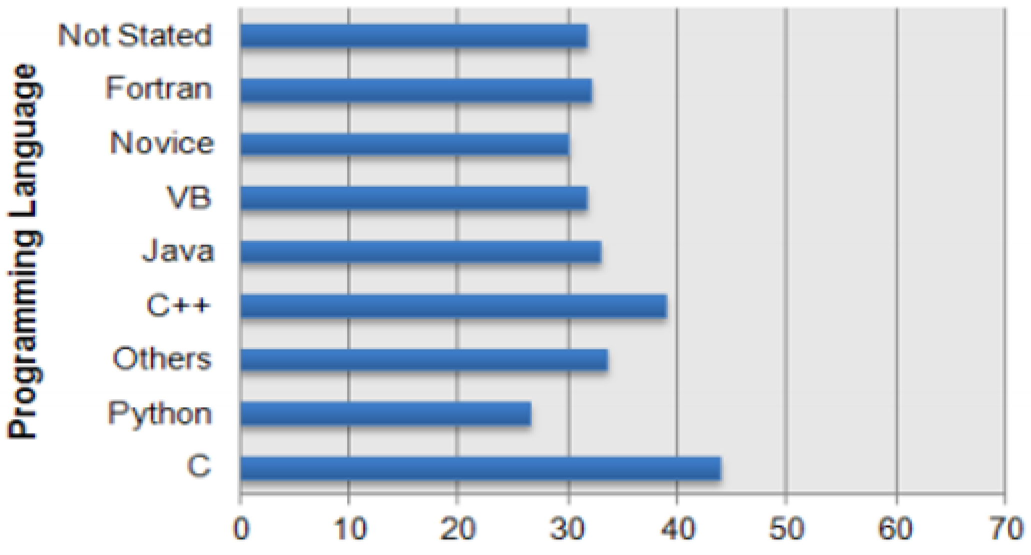

Figure 3. Students who do not pass are grouped according to programming language(Watson & Li, 2014)

In Figure 3. the selection of programming languages at the student graduation level according to (Watson & Li, 2014)the first rank is seen in C language and followed by C ++. Furthermore (Bennedsen & Caspersen, 2019) conducted another study on the failure rate of learning programming where previous research found that the student failure rate was an average of 33% in 2007 and in 2018 the average failure rate was 28%. Not surprisingly, learning to program can be such a difficult task, to the point where the failure rate phase in programming classes is almost the same(Bennedsen & Caspersen, 2007).

From the literature, collected on the problem of programming failure is a programming language commonly used in programming classes such as C, C ++, C # and Java which has a broad and complex syntax, making learning difficult for beginners(Gomes et al., 2008;O'Kelly & Gibson, 2006; Robins et al., 2003; Milne & Rowe, 2002; Watson & Li, 2014;Bravo et al., 2005). The number of students who fail when starting computer programming learning because of difficulty understanding: basic concepts, problem solving skills, by identifying problems, developing algorithms and coding algorithms with programming languages (Milne & Rowe, 2002; Jenkins, 2002; Esteves et al., 2009; Lahtinen et al., 2005);Susanti et al., 2020). Ineffective problem solving strategies (Febrian & Lawanto, 2018; Whittington, 2004). Learners also experience learning barriers because they have the wrong perspective about computers (Ben-Ari, 1998; Brennan, K., & Resnick, 2012; Lischner, 2001) or face an unwanted learning environment (for example, lack of human interaction)(Ben-Ari, 2001). Some of the misconceptions include misconceptions related to initialization of variables, loops, conditions, pointers and recursion(Milne & Rowe, 2002).

The results of the study(Khaleel et al., 2017) show that the low value of learning programming is because students experience ineffective learning, lack of interest and lack of motivation. Another most important problem faced by students in learning programming is in the practicum section, which involves their need to practice extensively to achieve higher programming skills(Wei, 2010).

Researchers from the literature review have done a lot of research on strategies to help students learn computer programming which is a challenging problem of how to improve beginners' understanding and programming skills. Cognitive theory is offered to answer the question: why do so many students fail to learn programming. These include difficulty understanding program objectives and their relationship to computers, difficulty understanding the syntax and semantics of a particular programming language(Robins et al., 2003). Misunderstanding of programming constructs (Lane & VanLehn, 2003), inability to solve problems (McCracken et al., 2001) and inability to read and understand program code (Lister et al., 2004).

Two cognitive factors that make learning to program difficult are learning styles and motivation(Milne & Glenn, 2002; Tandon & Ravikumar, 2013). Traditional teaching methods, usually based on lectures and specific programming language syntax, fail to motivate students to engage in programming (Berlin & Bennedsen, 2006; Lahtinen et al., 2005; Esteves et al., 2007). Therefore, it is important to incorporate concept knowledge and strategies for its use in the learning process(Lahtinen et al., 2005).

2.

3. Programming Learning Solutions

Teaching programming languages aims to make students develop a set of skills needed to design computer programs and systems capable of solving real problems. The 36 literature reviewed showed eight failure rates in computer programming. The first is not applying the correct algorithm(4 articles).. Second, programming is difficult (14 articles). Third, ineffective problem-solving strategy skills (18 articles). Four, the inability to solve the problem (6 articles). Five, wrong perspective about computers (3 articles). Six, lack of human interaction (2 articles). Seven, learning styles and motivation (4 articles). Eight, lecturer-centered learning (3 articles). Figure 1 shows the percentage of literature review of students' failure rates in computer programming.

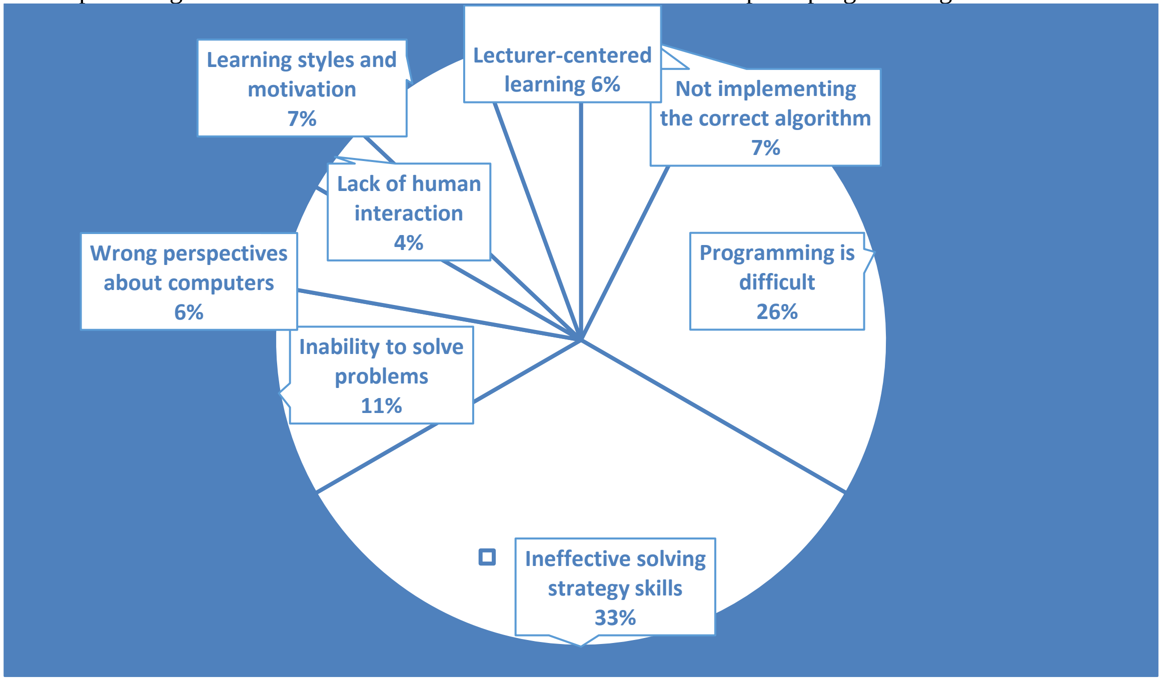

Figure 1. Percentage of failure in programming

Literature on computer programming failure, from studies there are states that the same problem is obtained so that the percentage is as above.

To be able to compete, students must have cognitive skills, ability to solve complex problems, have attitudes and motivation(Tsai & Tsai, 2018). The solution to dealing with problems that will become a trend in the 21st century is to combine mastery of knowledge with technology (Voskoglou & Buckley, 2012).

Many researchers have attempted to overcome this problem by developing various computer-based instructional tools and improving existing learning methods (O'Kelly & Gibson, 2006; Olapiriyakul & Scher, 2006; Adams, 2007; Febrian & Lawanto, 2018). Designing metacognitive related activities can be done through the use of technology that is integrated into educational activities. Designing metacognitive-related activities that focus on social and cognitive development is a theoretical and practical challenge, especially in supporting teaching and learning computer programming (Rum & Ismail, 2017).

4. METODOLOGY

The results of the search for scientific sources such as international journals found solutions that have been done by previous researchers to answer student failures in computer programming. Appropriate samples were collected and analyzed. The following are the systematic steps taken in conducting a literature study as shown in Figure 2.

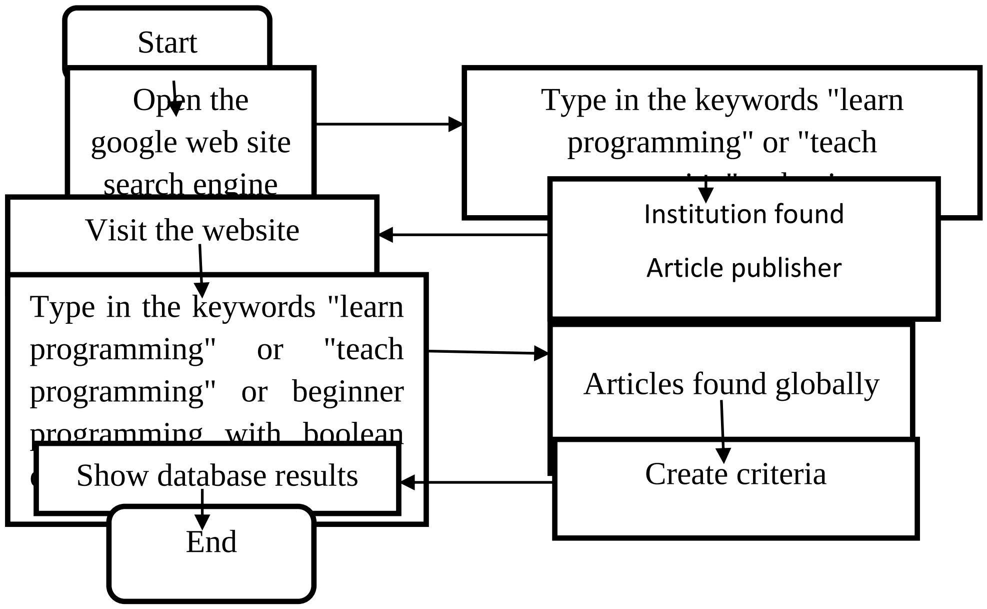
Start
Open the google web site search engine
Type in the keywords "learn programming" or "teach
Institution found
Article publisher
Visit the website
Type in the keywords "learn programming" or "teach programming" or beginner programming with boolean
Articles found globally
Create criteria
Show database results
End

Figure 2. Flowchart of systematic literature review

After selecting and screening, based on the criteria for articles that meet the requirements for literature review, the website database page. A total of 20 articles were used as a source of review reading as a strategy to overcome the failure of basic programming learning.

5. RESULT AND DISCUSSION

The results of the literature review explain several programming learning strategies that have been carried out by several researchers along with the results of these studies. The solution offered is learning programming with collaobarotive, problem-based programming with PBL or PjBL and an approach with computational thinking. More details can be seen in Table 2.

Table 2.Researcher's solutions in programming learning

No	Author	Research Solutions	Result	Publisher/Source
1.	(Voskoglou & Buckley, 2012)	Problem solving-based learning with a computational thinking approach to synthesizing critical thinking and existing knowledge.	Identifying students' critical thinking	Egyptian Computer Science Journal ,ECS ,Vol.36 No.4, September 2012
2.	(Esteves et al., 2011)	Computer learning and programming is developed in cyberspace. Observations are focused on 1) how students and teachers interact 2) virtual classroom activities 3) use of interfaces 4) challenges and constraints	Identify problems that hinder teacher intervention in cyberspace and detect solutions to	British Journal of Educational Technology doi:10.1111/j.1467-8535.2010.01056.x

			those problems.	
3.	(Esteves et al., 2007)	Use a collaborative virtual environment to increase student effectiveness and motivation	Enhances the learning experience for computer programming students	International conference on multimedia and information and communication technologies in education
4	(Wang & Hwang, 2017)	Problem posing based collaborative learning strategy using the C # language. Experimental group trial data of 25 students were taught using collaborative practice problem solving strategies and 28 students as a conventional control group.	Improve learning achievement and programming skills	Education Tech Research Dev DOI 10.1007/s11423-017-9551-0 Springer
5	(Nouri et al., 2020)	Skills development related to computational thinking concepts	Cognitive skills, language skills, creative problem-solving skills and attitude and collaborative attitude skills	Education Inquiry Volume 11, 2020 - Issue 1
6	(Febrian & Lawanto, 2018)	Identifying and investigating programming task understanding skills	The thought process of the participants	International Education Studies; Vol. 11, No. 12; 2018
7	(Saltan, 2016)	Perform online algorithmic visualization as an instructional approach	Knowing the effect of online algorithmic visualization on student achievement	Journal of Education and Learning; Vol. 6, No. 1; 2017 Published by Canadian Center of Science and Education The
8	(Renumol et al., 2010)	Identify students' cognitive processes. Eight cognitive processes identified 1) confusion 2) hypothesis 3) interrogation 4) repetition 5) monitoring 6) memory 7) relapse 8) translation	Make programming teachers aware of the cognitive difficulties of programming and the importance of the programming teaching process	ACM Transactions on Computing Education
9	(Esteves et al., 2009)	Learning approach with Second life is a problem-based 3-dimensional online virtual world	Observe and reflect on the problems that arise	Journal of Virtual Worlds Research
10	(Ideris et al., 2019)	Students work in groups to solve programming problems using the Scratch software as a teaching aid	Improve student scores on tests and higher order thinking skills	Konferensi ASEAN ke-4 tentang Psikologi, Konseling, dan Humaniora (ACPCH 2018)
11	(Serrano-Cámara et al., 2016)	Uses a mobile collaboration tool called MoCAS	For motivation	International Journal of Engineering Education
12	(Bravo et al., 2005)	Using animation and simulation programs in computer-supported collaborative learning	Provides educational tools to support teaching and learning of computer programming	Jurnal Ilmu Komputer Universal
13	(Chen,	Problem-based learning with a	Students' ability	Advances in Social

	2018)	computational thinking approach to programming languages.	to think computationally in solving problems	Science, Education and Humanities Research (ASSEHR), volume 156. Atlantis Press
14	(Othman et al., 2013)	PBL requires students to work collaboratively	Investigate student interests, learning styles and student learning preferences.	International Conference on University Learning and Teaching. Science Direct
15	(Da Rodrigues et al., 2017)	Designing a prototype model for online collaborative learning systems in a virtual environment. The technique used is the collaborative learning technique "Think-Pair-Share"	The results indicate the need to develop online small group discussions	Proceding
16.	(Serrano-Cámara et al., 2014)	The methodology used is in 3 stages 1. Data collection 2. Analysis and design 3. Implementation phase. Correspondent of 50 students of Diploma 3 in Computer Science	Overcoming student difficulties by improving practice in programming	Computers in Human Behavior. Elseiver
17	(Peng, Yuan, et al., 2019)	Doing collaborative learning practice computer programming	Students are more motivated by sharing	Jurnal Teknologi Pendidikan Australasia
18	(Peng, Wang, et al., 2019)	Collaborative learning supported by MoCAS. Evaluation of motivation on 139 students	CIF's collaborative instructional approach and MoCAS tools	Int. J. Smart Technology and Learning
19	(O'Kelly & Gibson, 2006)	Propose a visualization-based and progressive learning environment as cognitive tools to support collaborative learning to support PjBL programming.	Incorporating a simple approach to complex visualization-based cognitive tools is more effective in improving student programming performance	ITiCSE'06, June 26–28, 2006
20	(Lovos, 2015)	Learning programming by integrating technology	Critical review of how emerging technologies have been integrated in programming learning	Revista Internacional de Ciencia, Matemáticas y Tecnología

A review of 20 literature studies with problem-based, collaborative and computational thinking approaches

Applying technology in programming learning has become hot during these 40 years (Douce et al., 2005). The programming problem-solving process forces a person to think about what problems and what needs to be done to solve the problem and then various program functions and procedures are identified in the form of inputs and outputs, so solving problems is a key component for programming activities so that learning programming by embracing PBL / PjBL as one of the instructional approaches. Problem-based programming learning involves various aspects of programming knowledge, strategies, and problem-solving processes. Incorporating collaborative work in programming courses has been identified as a potential strategy for maximizing student participation and having a positive impact on learning. However, social interaction in collaborative learning does not happen automatically. Appropriate guiding strategies and supporting tools for collaborative learning are indispensable. Practice-based problem solving strategies to support collaborative learning activities in computer programming practice courses.

A literature review also shows that collaborative programming learning can increase student motivation (Esteves et al., 2007). The research supports that collaboration is an effective educational feature for programming (McDowell et al. 2002; Esteves et al. 2009). According to Roe and Queensland (2008), in their research, there was a change in student perceptions about collaborative learning, sucha as: 1) a more pleasant environment; 2) more confident with a peer environment; and 3) increased skills. W. W. M (2009) also stated that collaborative learning supported by computers can improve thinking skills, social interaction, criticality and creativity. Computational thinking involves solving problems, designing systems, and understanding human behavior based on computer science concepts (Grover & Pea, 2013). Understand the cognitive processes underlying collaborative (A. Soller et al., 2004).

Computational thinking is one of the programming learning approaches to train students to solve problems to be able to think in a structured, logical, and algorithmic manner (Yeni Anistyasari, Ekohariadi, 2020).Students' skills in programming computers are in line with computational thinking which is a skill to learn and to think in a structured, abstract, algorithmic and logical manner, and to be ready to solve complex and open problems (Yasin, 2020). Computational thinking is in accordance with 21st century skills in solving problems, designing systems, and understanding human behavior by drawing on the basic concepts of computer science (Nouri et al., 2020).

6. Conclusion

In this literature review, the failure rates in computer programming are presented and the factors that cause the failure rates of students to learn programming and the solutions that have been offered by several researchers. Based on the study: 1) good knowledge of problem solving skills; 2) knowing the syntax and semantics of programming languages; 3) being able to understand existing code; 4) the ability to analyze problems; 5) being able to compile solutions; and 6) being able to express the program into a programming language and being able to test it,

References

1. Adams, J. C. (2007). Alice, middle schoolers & the imaginary worlds camps. *SIGCSE 2007: 38th SIGCSE Technical Symposium on Computer Science Education*, 307–311. https://doi.org/10.1145/1227310.1227418
2. Bakar, M. A., Jilani, J., Jailani, N., Razali, R., Shukur, Z., & Aziz, M. J. A. (2013). Student Centered Learning Environment for Project Monitoring. *Procedia Technology*, *11*(Iceei), 940–949. https://doi.org/10.1016/j.protcy.2013.12.279
3. Ben-Ari, M. (1998). Constructivism in computer science education. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, *30*(1), 257–261. https://doi.org/10.1145/274790.274308
4. Ben-Ari, M. (2001). Constructivism in computer science education. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, *20*(1), 257–261. https://doi.org/10.1145/274790.274308
5. 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
6. Bennedsen, J., & Caspersen, M. E. (2019). Failure rates in introductory programming - 12 years later. *ACM Inroads*, *10*(2), 30–35. https://doi.org/10.1145/3324888
7. Berlin, D.-, & Bennedsen, J. (2006). *What do Teachers Teach in Introductory Programming ?* 17–28.
8. Bravo, C., Marcelino, M. J., Gomes, A., Esteves, M., & Mendes, A. J. (2005). Integrating educational tools for collaborative computer programming learning. *Journal of Universal Computer Science*, *11*(9), 1505–1517.
9. Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Educational Research Association Meeting*. http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf
10. Chen, G. (2018). *Programming Language Teaching Model Based on Computational Thinking and Problem-based Learning*. *156*(Seiem), 128–131. https://doi.org/10.2991/seiem-17.2018.31
11. Cradler, J., McNabb, M., Freeman, M., and Burchett, R. (2002). Subject: Research on academic performance and technology. *Learning & Leading with Technology*, *29*(8), 46–56. http://educ116eff11.pbworks.com/w/file/fetch/44935610/Article.StudentLearning.pdf
12. Da Rodrigues, P. L. R., Franz, L. P., Cheiran, J. F. P., Da Silva, J. P. S., & Bordin, A. S. (2017). Coding Dojo as a transforming practice in collaborative learning of programming: An experience report. *ACM International Conference Proceeding Series*, 348–357. https://doi.org/10.1145/3131151.3131180
13. Douce, C., Livingstone, D., & Orwell, J. (2005). Automatic Test-Based Assessment of Programming: A Review. *ACM Journal on Educational Resources in Computing*, *5*(3), 4. https://doi.org/10.1145/1163405.1163409
14. Esteves, M., Fonseca, B., Morgado, L., & Martins, P. (2009). Using Second Life for Problem Based

Learning in computer science programming. *Journal For Virtual Worlds Research*, *2*(1). https://doi.org/10.4101/jvwr.v2i1.419

15. Esteves, M., Fonseca, B., Morgado, L., & Martins, P. (2011). Improving teaching and learning of computer programming through the use of the Second Life virtual world. *British Journal of Educational Technology*, *42*(4), 624–637. https://doi.org/10.1111/j.1467-8535.2010.01056.x
16. Esteves, M., Morgado, L., Martins, P., Fonseca, B., Esteves, M., Morgado, L., Martins, P., & Fonseca, B. (2007). *The use of Collaborative Virtual Environments to provide student ' s contextualisation in programming To cite this version : INTERNATIONAL CONFERENCE ON MULTIMEDIA AND INFORMATION AND COM- student ' s contextualisation in programming*.
17. Febrian, A., & Lawanto, O. (2018). Do Computer Science Students Understand Their Programming Task?—A Case Study of Solving the Josephus Variant Problem. *International Education Studies*, *11*(12), 26. https://doi.org/10.5539/ies.v11n12p26
18. Francisco, R. E., & Ambrosio, A. P. (2015). Mining an online judge system to support introductory computer programming teaching. *CEUR Workshop Proceedings*, *1446*.
19. Gomes, A., Areias, C., Henriques, J., & Mendes, A. J. (2008). Aprendizagem de programação de computadores: dificuldades e ferramentas de suporte. *Revista Portuguesa de Pedagogia*, 161–179. https://doi.org/10.14195/1647-8614_42-2_9
20. Husein, Ismail H Mawengkang, S Suwilo "Modeling the Transmission of Infectious Disease in a Dynamic Network" Journal of Physics: Conference Series 1255 (1), 012052, 2019.
21. Husein, Ismail, Herman Mawengkang, Saib Suwilo, and Mardiningsih. "Modelling Infectious Disease in Dynamic Networks Considering Vaccine." *Systematic Reviews in Pharmacy* 11.2, pp. 261-266, 2020.
22. Husein, Ismail, Dwi Noerjoedianto, Muhammad Sakti, Abeer Hamoodi Jabbar. "Modeling of Epidemic Transmission and Predicting the Spread of Infectious Disease." *Systematic Reviews in Pharmacy* 11.6 (2020), 188-195. Print. doi:10.31838/srp.2020.6.30
23. Husein, Ismail, YD Prasetyo, S Suwilo "Upper generalized exponents of two-colored primitive extremal ministrong digraphs"AIP Conference Proceedings 1635 (1), 430-439, 2014
24. Husein Ismail, Rahmad Syah, "Model of Increasing Experiences Mathematics Learning with Group Method Project", *International Journal of Advanced Science and Technology*, pp. 1133-1138, 2020.
25. Ideris, N., Baharudin, S. M., & Hamzah, N. (2019). *The Effectiveness of Scratch in Collaborative Learning on Higher-Order Thinking Skills in Programming Subject Among Year-Six Students*. *304*(Acpch 2018), 421–425. https://doi.org/10.2991/acpch-18.2019.99
26. Jenkins, T. (2002). ON THE DIFFICULTY OF LEARNING TO PROGRAM. *Centre for Information and Computer Sciences*, 53–58.
27. Khaleel, F. L., Ashaari, N. S., Wook, T. S. M. T., & Ismail, A. (2017). Programming learning requirements based on multi perspectives. *International Journal of Electrical and Computer Engineering*, *7*(3), 1299–1307. https://doi.org/10.11591/ijece.v7i3.pp1299-1307
28. Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 14–18. https://doi.org/10.1145/1067445.1067453
29. Lane, H. C., & VanLehn, K. (2003). Coached program planning: Dialogue-based support for novice program design. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, 148–152. https://doi.org/10.1145/792548.611955
30. Lischner, R. (2001). Explorations: Structured labs for first-time programmers. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, 154–158. https://doi.org/10.1145/366413.364571
31. Lister, R., Fone, W., McCartney, R., Seppälä, O., Adams, E. S., Hamer, J., Moström, J. E., Simon, B., Fitzgerald, S., Lindholm, M., Sanders, K., & Thomas, L. (2004). A multi-national study of reading and tracing skills in novice programmers. In *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)* (Vol. 36, Issue 4). https://doi.org/10.1145/1041624.1041673
32. Lovos, E. (2015). Ambiente de desarrollo virtual para el aprendizaje de la programación: un estudio de caso en la Lic. de Sistemas de la Universidad Nacional de Río Negro, Patagonia Argentina. *Revista Internaciona de Ciencia, Matemáticas y Tecnología.*, *2*(1), 11.
33. McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., Laxer, C., Thomas, L., Utting, I., & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ACM SIGCSE Bulletin*, *33*(4), 125–180. https://doi.org/10.1145/572139.572181
34. Milne, I., & Glenn, D. A. N. (2002). Kesulitan dalam Pemrograman Pembelajaran dan Pengajaran — Pandangan Siswa dan Tutor. *Teknologi Pendidikan Dan Informas*, 55–66.

35. Milne, I., & Rowe, G. (2002). Difficulties in learning and teaching programming - Views of students and tutors. *Education and Information Technologies*, *7*(1), 55–66. https://doi.org/10.1023/A:1015362608943
36. Muchlis, L. S., Rukun, K., & Krismadinata, K. (2020). Efektifitas Pengembangan Model Diva Learning Manajemen System Pada Matakuliah Algoritma Dan Pemrograman. *Jurnal Pendidikan Teknologi Kejuruan*, *3*(2), 104–108. https://doi.org/10.24036/jptk.v3i2.7123
37. Muhson, A., Wahyuni, D., & Mulyani, E. (2012). Analisis Relevansi Lulusan Perguruan Tinggi Dengan Dunia Kerja. *Jurnal Economia (Yogyakarta)*, *8*(1), 42–52. https://doi.org/10.21831/economia.v8i1.800
38. Nair, P. R. (2020). Increasing Employability of Indian Engineering Graduates through Experiential Learning Programs and Competitive Programming: Case Study. *Procedia Computer Science*, *172*, 831–837. https://doi.org/10.1016/j.procs.2020.05.119
39. Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, *11*(1), 1–17. https://doi.org/10.1080/20004508.2019.1627844
40. O'Kelly, J., & Gibson, J. P. (2006). RoboCode & problem-based learning. *ACM SIGCSE Bulletin*, *38*(3), 217. https://doi.org/10.1145/1140123.1140182
41. Olapiriyakul, K., & Scher, J. M. (2006). A guide to establishing hybrid learning courses: Employing information technology to create a new learning experience, and a case study. *Internet and Higher Education*, *9*(4), 287–301. https://doi.org/10.1016/j.iheduc.2006.08.001
42. Othman, M., Othman, M., & Hussain, F. M. (2013). Designing Prototype Model of an Online Collaborative Learning System for Introductory Computer Programming Course. *Procedia - Social and Behavioral Sciences*. https://doi.org/10.1016/j.sbspro.2013.07.094
43. Ozyurt, O. (2015). An analysis on distance education computer programming students' attitudes regarding programming and their self-efficacy for programming. *Turkish Online Journal of Distance Education*, *16*(2), 111–121. https://doi.org/10.17718/tojde.58767
44. Peng, J., Wang, M., Sampson, D., & van Merriënboer, J. J. G. (2019). Using a visualisation-based and progressive learning environment as a cognitive tool for learning computer programming. *Australasian Journal of Educational Technology*, *35*(2), 52–68. https://doi.org/10.14742/ajet.4676
45. Peng, J., Yuan, B., Spector, J. M., & Wang, M. (2019). Integrating technology in programming learning and instruction: a critical review. *International Journal of Smart Technology and Learning*, *1*(4), 323. https://doi.org/10.1504/ijsmarttl.2019.106538
46. Renumol, V. G., Janakiram, D., & Jayaprakash, S. (2010). Identification of cognitive processes of effective and ineffective students during computer programming. *ACM Transactions on Computing Education*, *10*(3). https://doi.org/10.1145/1821996.1821998
47. Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *International Journal of Phytoremediation*, *21*(1), 137–172. https://doi.org/10.1076/csed.13.2.137.14200
48. Rum, S. N. M., & Ismail, M. A. (2017). Metocognitive support accelerates computer assisted learning for novice programmers. *Educational Technology and Society*, *20*(3), 170–181.
49. Saltan, F. (2016). The Impact of Online Algorithm Visualization on ICT Students' Achievements in Introduction to Programming Course. *Journal of Education and Learning*, *6*(1), 184. https://doi.org/10.5539/jel.v6n1p184
50. Serrano-Cámara, L. M., Paredes-Velasco, M., Alcover, C. M., & Velazquez-Iturbide, J. Á. (2014). An evaluation of students' motivation in computer-supported collaborative learning of programming concepts. *Computers in Human Behavior*, *31*(1), 499–508. https://doi.org/10.1016/j.chb.2013.04.030
51. Serrano-Cámara, L. M., Paredes-Velasco, M., Velázquez-Iturbide, J. Á., Alcover, C. M., & Castellanos, M. E. (2016). MoCAS: A Mobile Collaborative Tool for Learning Scope of Identifiers in Programming Courses. *International Journal of Engineering Education*, *32*(2), 969–981.
52. Susanti, W., Sukrianto, D., & Ramadhani, D. (2020). *Pengaruh Model Discovery Learning dalam Kemampuan Berpikir Kritis dan Cognitif Mahasiswa Program Studi Sistem Informasi*. *20*(3).
53. Tandon, R., & Ravikumar, P. (2013). On the difficulty of learning power law graphical models. *IEEE International Symposium on Information Theory - Proceedings*, 2493–2497. https://doi.org/10.1109/ISIT.2013.6620675
54. Trilling, B., & Fadel, C. (2010). 21St Century Skills: Learning for Life in Our Times. *Choice Reviews Online*, *47*(10), 47-5788-47–5788. https://doi.org/10.5860/choice.47-5788
55. Tsai, M. C., & Tsai, C. W. (2018). Applying online externally-facilitated regulated learning and computational thinking to improve students' learning. *Universal Access in the Information Society*, *17*(4), 811–820. https://doi.org/10.1007/s10209-017-0542-z
56. Tutang. (2009). *Peluang Dan Tantangan Lulusan Bidang Teknologi Informasi Di Indonesia*. 1–7.

57. Verawardina, U., Ramadhani, D., Susanti, W., Lubis, A. L., Simeru, A., & Ambiyar. (2020). Studying technology-based XXI century learning using Mooc in education. *International Journal of Psychosocial Rehabilitation*, *24*(9), 2644–2649. https://doi.org/10.37200/IJPR/V24I9/PR290297
58. Voskoglou, M. G., & Buckley, S. (2012). *Problem Solving and Computational Thinking in a Learning Environment*. *36*(4), 28–46. http://arxiv.org/abs/1212.0750
59. Wang, X. M., & Hwang, G. J. (2017). A problem posing-based practicing strategy for facilitating students' computer programming skills in the team-based learning mode. *Educational Technology Research and Development*, *65*(6), 1655–1671. https://doi.org/10.1007/s11423-017-9551-0
60. Watson, C., & Li, F. W. B. (2014). Failure rates in introductory programming revisited. *ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference*, 39–44. https://doi.org/10.1145/2591708.2591749
61. Wei, X. (2010). *Research of Practical Course Teaching of JAVA Language Abstract-At present , many institutions of higher learning set up JAVA course , how to better integrate the characteristics of the course with students ' hands-on practical ability , and to accumulate*. 10–12.
62. Whittington, K. J. (2004). Infusing Active Learning Into Introductory Programming Courses. *Journal of Computing Sciences*, *19*(5), 249–259. http://dl.acm.org/citation.cfm?id=1060081.1060111
63. Yeni Anistyasari, Ekohariadi, M. (2020). Strategi pembelajaran untuk meningkatkan keterampilan pemrograman dan berpikir komputasi: sebuah studi literatur. *Journal of Vocational and Technical Education*, *02*, 37–44.