Journal of ICT in Education (JICTIE)
ISSN 2289-7844 / 9 / 3 (Special Issue 1/2022: Student Engagement and EdTech) / 62-74

A Data Design for Integrating Problem Solving into Frame-Based Programming

Nor Farahwahida Mohd Noor, Aslina Saad*, Abu Bakar Ibrahim

Computing Department, Fakulti Seni, Komputeran & Industri Kreatif, Universiti Pendidikan Sultan Idris
norfarahwahida@gmail.com {aslina, abubakar.ibrahim} @fskik.upsi.edu.my

** correspondence author*

To cite this article (APA): Mohd Noor, N.F., Saad, A., & Ibrahin, A.B. (2022). A data design for integrating problem solving into frame-based programming. *Journal of ICT in Education*, *9*(3),62-74. https://doi.org/10.37134/jictie.vol9. sp.1.6.2022

To link to this article: https://doi.org/10.37134/jictie.vol9. sp.1.6.2022

Abstract

Data design is one of the main types of design in software development. This paper aims to design a database for a programming application as a part of the application development process. The purpose of the application is to help students of Introductory Programming in the higher learning institutions of Malaysia solve programming problems in a programming environment, especially in the C language. The objective of the study is to produce a data design to integrate data from a problem-solving process into a frame-based programming environment. To achieve this objective, the study is adopting the user design part of the Rapid Application Development (RAD) Model. Six validated problem-solving steps which are based on computational thinking (CT) concepts combined with the input-process-output (IPO) Model were utilized to gather data in the problem-solving process. Then, these data were integrated into ten validated sets of programming Code Patterns to be utilized in the frame-based programming environment. An entity-relationship diagram (ERD) which shows the relationship of seven entities was produced to illustrate the data design of the IPO database. This IPO database is the core of this programming application that integrates problem-solving into the frame-based programming environment. This integration contributes to the advancement of an integrated development environment (IDE) approach specifically for the needs of the novices. It helps to automate the instruction coding based on the guided problem solution, thus helping the novices to overcome the difficulties due to their incompetency in the programming language. Therefore, the design produced from this study is significant in developing an introductory IDE application for the students to improve their problem-solving and programming skills and prepare them for future demands of the industrial revolution challenges.

Keywords: problem-solving, frame-based programming, data design, C language programming.

Received: 1 February 2022; Accepted: 15 September 2022; Published: 1 December 2022

INTRODUCTION

Along with the fourth industrial revolution (4IR), the industry's need for the required skills also has changed. As computing and software technology is experiencing rapid development, the demand for expert programmers also has increased. Therefore, graduates must acquire problem-solving and programming skills in solving problems to meet this demand (Chaka, 2020). However, past studies have shown that programming is one of the most difficult courses to learn especially among beginners (Cheah, 2020).

Programming is not only about developing a computer program, but it also involves problem-solving tasks (Mohd Yusoff et al., 2020; Nelson et al., 2017). Therefore, programming needs proficiency in both problem-solving and program coding (Choi, 2019). However, due to their lack of programming experience, novice programmers often face difficulties in performing problem-solving (Hashim et al., 2017). Their unfamiliarity with the language syntax and patterns also has caused difficulties in writing instructions for the program in a programming environment (Ettles et al., 2018).

Although both problem-solving and program writing tasks are needed hand in hand in solving programming problems, most of the programming applications focus only on programming development but not on problem-solving processes. Moreover, the application usually used by the students which is the IDE is overwhelmed with many functions that could be intimidating for the novices (Warner & Guo, 2017). Various applications have been developed in facilitating programming but comparatively few are applicable for the C language (Egan & Mcdonald, 2020). Therefore, a programming application targeting the C language that serves as an introductory IDE with problem-solving guidelines for novices needs to be developed.

As part of the software development process, the design phase is important in a software process model. It is one of the main processes in most software process models. In the RAD model, the design phase involves several design processes. One of the main design types in software development is data design. For any software utilizing a database, a data design needs to be carefully planned to produce a well-designed database as it is the pillar in supporting system operations (Kurnianda, 2018). Therefore, it is crucial to have a well-designed database to ensure good database performance.

Therefore, this paper aims to design the integration of problem-solving to a frame-based programming application through its data design. For that purpose, this application is designed as an integrated system that guides problem-solving towards facilitating program development. The data design is done to develop the application database that will be the core service of this introductory IDE as a tool for learning C programming. Thus, this introductory IDE could help students enhance their problem-solving and programming skills to get them ready for using the professional IDE in the future and develop the qualities of a competent programmer.

LITERATURE REVIEWS

To meet the demand of 4IR, graduates should acquire one of the most in-demand skills which are programming skills. Programming skills should come together with problem-solving skills as a generic soft skill to play in the industry competently (Chaka, 2020). However, the challenges in learning programming have made this course one of the most difficult courses with the highest dropout rates (Edwards et al., 2019; Hosanee & Rana, 2018).

Three significant challenges of programming often faced by the students are problem-solving deficiency (Islam et al., 2019; Mohd Yusoff et al., 2020), language difficulties (Hashim et al., 2017; Hosanee & Rana, 2018; Qian & Lehman, 2017), and the use of intimidating IDE for writing programs (Hashim et al., 2017; Sim & Lau, 2018). In most of the cases, the low achievement of the programming course in the higher learning institutions was due to students' difficulties in programming which is related to their weaknesses in performing problem-solving (Mat Isa & Md Derus, 2017).

To overcome the difficulties in problem-solving, CT techniques were suggested which require the students to do task decomposition, abstraction, pattern recognition, and algorithm design (Chen, 2017; Mohd Yusoff et al., 2020). To implement the CT, Tawfik et al. (2020) recommended using scientific instructions and inquiries to help in problem-solving. This is supported by Saad (2020), who stated that strong CT skills for problem-solving could be developed through investigative inquiries. Nevertheless, a specific guideline or model to develop instructions and inquiries for programming problem-solving should be used.

Alshaye et al. (2019) and Hasan et al. (2020) suggested that the instructions and inquiries for problem-solving in programming could be guided by the IPO model to identify the important elements of a problem solution. The IPO information which are the inputs, the processes and the expected outputs from a problem is important in designing the program algorithm (Veerasamy et al., 2019). By using specific scientific instructions and inquiries to guide in problem-solving, this IPO information could be identified thus could help in planning for the program algorithm. Moreover, it could reduce the student's cognitive load in solving problems and thus prepare them to start coding for their program (Hasan et al., 2020; Margulieux et al., 2020).

Coding using text-based programming has been conventional especially in higher education. Practically, students use the IDE to develop a C program and to demonstrate programming (Hundhausen et al., 2017). The existing IDE used in higher education is usually employing text-based programming which is quite challenging for the novices (Hashim et al., 2017). Therefore, the students being novice programmers need to cope with syntax learning, at the same time getting used to the programming environment. This has raised a concern that it might be daunting for novices to start coding and score in a programming course (Ishizue et al., 2018; Xinogalos, 2016).

During program coding, the most frequent programming errors among novices came from syntax errors (Qian & Lehman, 2017). Based on the studies by Ettles et al. (2018) and Qian & Lehman (2017), the common syntax errors often observed among novices are due to missing semicolons, mismatched parentheses, brackets, or quotation marks. These errors are usually the result of typographical errors due to unfamiliarity with the language syntax and patterns. These are the common challenges in text-based programming due to the language difficulties, for every single symbol and character is visible to the compiler.

Since text-based programming has been challenging to novices, visual programming was suggested to overcome the programming incompetency among the novices where programming is done using instruction blocks instead of typing the tedious program code. This block-based programming simplifies the coding activity as the users only need to drag and configure the parameters within a particular block and combine it with other blocks to make a complete application program.

However, text-based programming is still important because it is an industry practice and should be preserved for syntax learning by implementing frame-based programming (Perera et al., 2021; Sim & Lau, 2018). Frame-based programming employs the block-based programming approach at the same time incorporates text-based programming (Sim & Lau, 2018). In frame-based programming, certain common programming coding or instructions are grouped into blocks where each block performs a specific task. These blocks of visible pre-written programming codes known as Code Patterns are used to build a C program, at the same time allowing parameters modification, thus supporting the transition to text programming.

With some modification, problem-solving data can be integrated into these Code Patterns. As CT and the IPO model managed to extract out the important variables from a problem, these variables can then be integrated into the Code Patterns which are utilized in the frame-based environment. Thus, the relationships between problem-solving data and program development parameters could be established in the IPO database and retrieved in the Code Patterns. This approach could facilitate program coding and accelerate text-based programming in the programming IDE. Therefore, the frame-based approach could reduce the extraneous cognitive load in programming, hence will reduce anxiety (Bakar et al., 2019).

This study establishes a data design in which problem solutions are directly translated from the problem-solving process into programming coding. With this data design, an IPO database of the application could be developed to serve for this introductory IDE to encounter the main issues of programming frequently faced by the students. This introductory IDE will be a better solution to introduce students to C language programming as it could simplify the problem-solving process and automate coding activities at the same time maintaining syntax learning to empower the programming skills of the students.

RESEARCH METHODOLOGY

This research is adopting a part of the RAD Model, that is the User Design phase as shown in Figure 1 below.

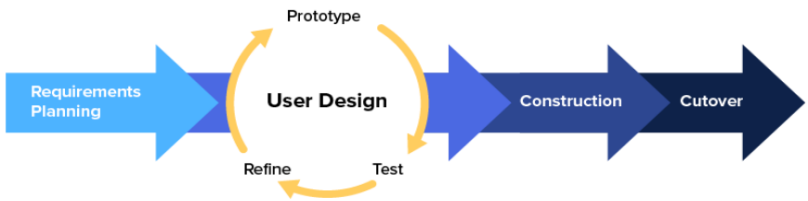

Figure 1: User design phase in the RAD model. Source: (Rahmawati & Rosyida, 2020)

For this user design phase, this paper focuses on the database construction by analyzing the functional requirements of the application to form the ERD (Lubis & Zamzami, 2019). The design of this application is based on the application requirements that have been produced in the Requirement Planning phase. From the requirement planning, six validated Problem-Solving Steps and ten validated sets of programming Code Patterns were found to be analysed to get the data design of the introductory IDE. The data design process is illustrated in Figure 2.

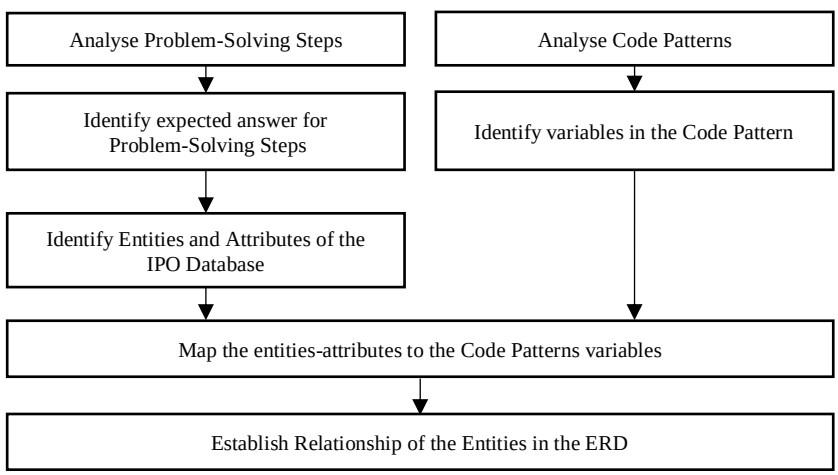

Figure 2: Data design process

Based on the illustration in Figure 2, the data design process starts with Analysing the Problem Solving Steps and the Code Patterns. The Problem-Solving Steps are used to gather data in the problem-solving process. From the analysis of the Problem-Solving Steps, the expected answers for each step were identified. Then, these expected answers for each step need to be stored in a database that is identified as the IPO database. The IPO database is the main data structure that stores answer from users in the

guided Problem-Solving process. Based on these Problem-Solving answers, the entities and attributes are then identified including the key attributes to build the IPO database. By identifying the key attributes of each entity, the relationship between one entity and another entity can be analyzed (Lubis & Zamzami, 2019).

At the same time, after the Code Patterns are analysed, all variables in the Code Patterns are identified. After that, the entities and attributes of the database gathered from the Problem-Solving answers are then mapped to the Code Patterns variables. These entities and attributes should be able to be retrieved in the Code Pattern to be used in the frame-based environment. This is where the IPO database is integrated into the Code Patterns by identifying related fields of the database that link to any part of the instruction codes. Finally, the relationships of the entities are established to develop a conceptual data modelling which is often used to illustrate a logical representation of data, which is the ERD (Kurnianda, 2018).

FINDINGS AND DISCUSSION

The Problem-Solving Steps comprises six sets of scientific instruction and inquiries which were developed to extract data from a specific programming problem. These scientific instructions and inquiries were developed based on CT concepts combined with IPO Model. For each instruction/ inquiry, there is an expected answer from the user. Every expected answer should be mapped according to the IPO model. Each answer could be an input, an output or a process to solve a problem as shown in Table 1.

Table 1: Problem-solving steps and IPO data entries

Instructions / Inquiries		**Expected Answer/ Input**	**IPO type**	**IPO Database**	
				Entities	**Attributes**
Step 1	What data will you get from the user?	Input name	Input	Input	Input_name
	State the data type.	Input data type			Input_datatype
Step 2	What output should you calculate?	Output name	Output	Output	Output_name
	State the data type.	Output data type			Output_datatype
Step 3	What other data is given/ needed?	Input2 name	Input	Input2	Input2_name
	Give the value	Input2 data value			Input2_value
	State the data type.	Data type			Input2_type
Step 4	Give the formula to calculate the output.	Formula	Process	Formula	Arithmaticformula

Instructions / Inquiries		Expected Answer/ Input	IPO type	IPO Database	
				Entities	Attributes
Step 5	Conditional action: State a selection condition.	Condition	Process	Selection	Selection_cond
	State an action if the condition is met.	Action 1			If_true_action
	State an action if the condition is met.	Action 2			If_false_action
Step 6	Repeating action: Name a counter.	Counter name	Process	Counter	Counter_name
	Counter data type	Data type			Counter_datatype
	State a repetition condition.	Condition		Repetition	Repetition_cond

Based on the Problem-Solving steps in Table 1, specific answers have been identified to each of the instructions and inquiries. These answers need to be registered to an IPO database. From the analysis, seven types of problem-solving data were identified which are Input, Output, Input2, Formula, Selection, Counter, and Repetition. These types of data are identified as the entities of the database. Each entity contains several attributes that describe each entity. Note that, in each of the Problem-Solving Steps, details of data were prompted in separated instructions/ inquiries. These details are identified as the attributes of the entities. Therefore, based on the expected answers in the Problem-Solving Steps, the entities and their corresponding attributes could be identified to build the IPO database as shown in Table 2.

Table 2: IPO database

Table No	Tables	Fields
1	Input	Input_ID
		Input_name
		Input_datatype
2	Output	Output_ID
		Output_name
		Output_datatype
3	Input2	Input2_ID
		Input2_name
		Input2_value
		Input2_datatype

Table No	Tables	Fields
4	Formula	Formula_ID
		Arithmaticformula
5	Selection	Selection_ID
		Selection_cond
		If_true_action
		If_false_action
6	Counter	Counter_ID
		Counter_name
		Counter_datatype
7	Repetition	Repetition_ID
		Repetition_cond

The IPO database presented in Table 2 shows that each entity consists of two or three attributes each. The entities of the database are represented as tables, while the attributes are represented as fields. The primary keys for each entity were identified and shown in underlined text. Note that each field is corresponding to the Problem-Solving answers. These data can be manipulated as variables for the program development. The Input, Input 2, Counter, Output data from the user will hold unique variable names. The problem-solving instructions also asked for a data type for these variables to allow them to be automatically declared through the Code Patterns. These fields are linked to the Code Patterns which are implemented in the frame-based programming environment to integrate problem-solving data into the program development code editor as shown in Table 3.

Table 3: Integrated code patterns

	Code Patterns	Integrated C Coding Blocks
CP1	Include stdio.h	#include <stdio.h>

CP2	main( )	int main( )
		{
		return 0;
		}

CP3	Declare Variables	<*Input_datatype*> <*Input_name*>; <*Output_datatype*> <*Output_name*>; <*Counter_datatype*> <*Counter_name*>; <*Input2_datatype*> < *Input2_name*> = <*Input2_value*>;
CP4	Insert Formula	<*Output_name*> = <*Arithmeticformula*> ;
CP5	printf	printf(" *insert your text here* ");
CP6	scanf	scanf("% *v* ", *&variable*);

	Code Patterns	**Integrated C Coding Blocks**

CP7	if - else	if(<*Selection_cond*>)
		{
		<*If_true_action*>
		}
		else
		{
		< *If_false_action* >
		}

CP8	for loop	for(<*Countername*>=0;<*Repetition_cond*>;<*Counter_name*>++)
		{
		}

CP9	while loop	while(< *Repetition_cond* >)
		{
		}

CP10	do-while loop	do { } while(< *Repetition_cond* >);

Based on the integrated C coding blocks of these Code Patterns in Table 3, the underlined code are the variables from Problem-Solving Steps that could be retrieved from the IPO database. It can be seen that one programming problem could have many Input, Input2, Output and Counter variables. It also can have many Formula, Repetition and Selection functions. Therefore, all entities in the IPO database is having a one-to-many relationship. The entity-relationship diagram is illustrated in Figure 3.

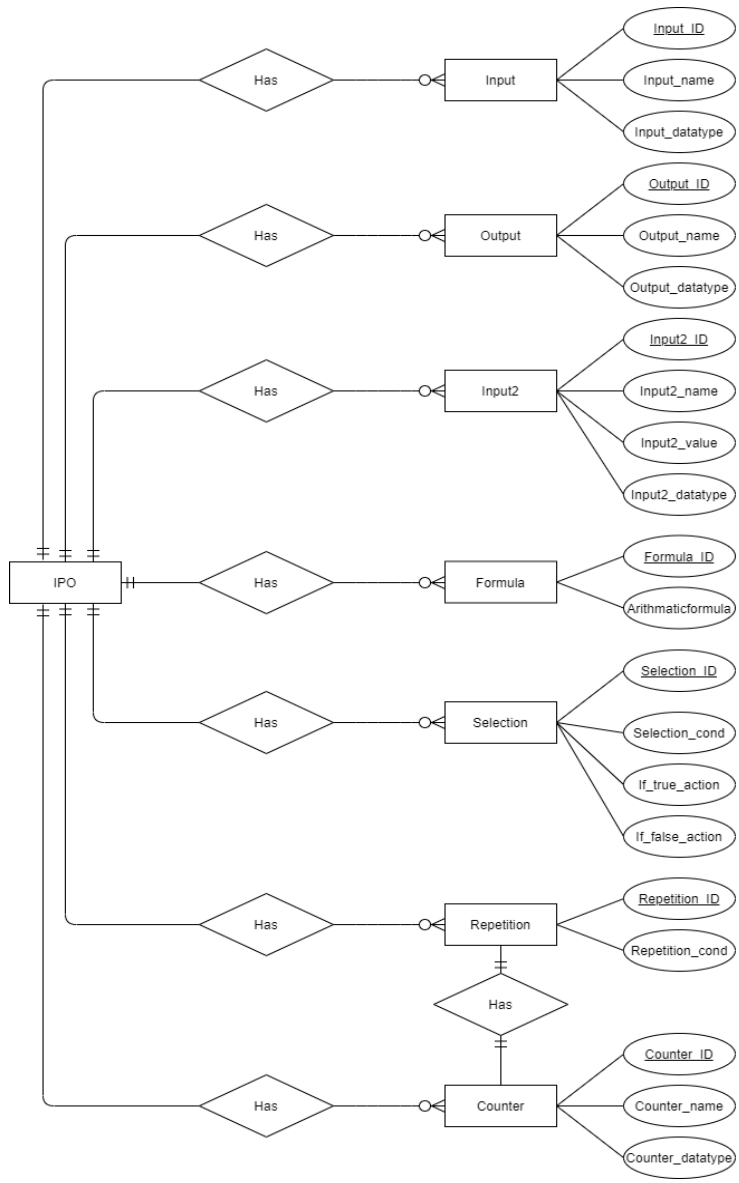

Figure 3: ERD for the IPO database

Figure 3 illustrates the ERD of the IPO database which consists of seven entities. Each entity is detailed by several attributes. For each entity, there is one primary key which is an index that uniquely defines the entity (Lubis & Zamzami, 2019). The IPO table has one-to-many relationships with all entities. However, the Repetition entity also has a one-to-one relationship to Counter.

The data design found from this research can be used as guidance and reference to software developers and other researchers in the field of software engineering in developing an integrated programming problem-solving IDE for the use of teaching and learning especially for basic C programming. This data design integrates the problem-solving guidelines into the IDE that making this IDE different as compared to the existing IDE for programming which only focuses only on the programming part. Therefore, it contributes to further diversifying the teaching tools for programming courses that balance the focus on both problem-solving and programming parts. This design is significant to be the basis of programming instructional tools as an effort to produce graduates with a strong foundation of problem-solving and programming skills to become expert programmers. Thus, it significantly contributes to programming learning to fulfil the demand for skilled programmers for IR4.0.

RECOMMENDATION FOR FUTURE RESEARCH

Since the data design was developed to cater for the introductory programming courses, thus it provides Problem-Solving Steps for simple and basic problems of C programming. It also provides Code Patterns for only basic instructions for structured programming. Therefore, for future research, it is recommended that these Problem-Solving Steps and Code Patterns can be expanded further to cater for modular and advanced programming. Modular and advanced programming would need advanced Code Patterns and additional problem-solving guidelines to suit the complexity of the problem. With some modification, it also could be implemented for other programming languages as well.

CONCLUSION

This study has produced a data design for an introductory IDE that implements frame-based programming which is integrated with problem-solving. The integration of problem-solving in the frame-based programming environment could be done with a well-designed database that has been illustrated in a form of an entity-relational diagram. This design is the basis of developing the core service of the application, which is the IPO database. The scientific instructions and inquiries in the Problem-Solving Steps were used to gather data for solving the problem and kept in the IPO database, while the pre-written Code-Patterns retrieve these data from the database to be used in the programming environment. This approach is applied for the introductory IDE to help novices perform problem-solving and writing programming syntax.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude to the Ministry of Higher Education and the Department of Polytechnic and Community College Education for their full support of this study. The authors are also grateful to Universiti Pendidikan Sultan Idris who also contributed to guiding this study. The authors would also like to thank the Malaysian Polytechnic and all the lecturers who have participated and contributed to this study.

REFERENCES

Alshaye, I., Tasir, Z., & Jumaat, N. F. (2019). The Conceptual Framework of Online Problem-Based Learning Towards Problem-Solving Ability and Programming Skills. *2019 IEEE Conference on E-Learning, e-Management and e-Services, IC3e 2019*, *January*, 12–15. https://doi.org/10.1109/IC3e47558.2019.8971780

Bakar, M. A., Mukhtar, M., & Khalid, F. (2019). The development of a visual output approach for programming via the application of cognitive load theory and constructivism. *International Journal of Advanced Computer Science and Applications*, *10* (11), 305–312. https://doi.org/10.14569/IJACSA.2019.0101142

Chaka, C. (2020). Skills, Competencies and Literacies Attributed to 4IR/Industry 4.0: Scoping Review. *International Federation of Library Associations and Institutions*, *46*(4), 369–399. https://doi.org/10.1177/0340035219896376

Chen, G. (2017). Programming Language Teaching Model Based on Computational Thinking and Problem-based Learning. *Advances in Social Science, Education and Humanities Research (ASSEHR), 2nd International Seminar on Education Innovation and Economic Management (SEIEM 2017)*, *156*(Seiem), 128–131. https://doi.org/10.2991/seiem-17.2018.31

Choi, S. Y. (2019). Development of an instructional model based on constructivism for fostering computational thinking. *International Journal of Innovative Technology and Exploring Engineering*, *8*(3C), 381–385.

Edwards, J. M., Fulton, E. K., Holmes, J. D., Valentin, J. L., Beard, D. V., & Parker, K. R. (2019). Separation of syntax and problem solving in Introductory Computer Programming. *Proceedings - Frontiers in Education Conference, FIE*, *2018-Octob*(May 2019), 1–5. https://doi.org/10.1109/FIE.2018.8658852

Egan, M. H., & Mcdonald, C. (2020). An Evaluation of SeeC : A Tool Designed to Assist Novice C Programmers with Program Understanding and Debugging. *Computer Science Education*, *00*(00), 1–34. https://doi.org/10.1080/08993408.2020.1777034

Ettles, A., Luxton-Reilly, A., & Denny, P. (2018). Common Logic Errors Made By Novice Programmers. *ACM International Conference Proceeding Series*, 83–89. https://doi.org/10.1145/3160489.3160493

Hasan, A. H., Hilmi, M. F., Ibrahim, F., & Haron, H. (2020). Input Process Output (IPO) AI Chatbot As Personal Learning Assistant For Programming Coursework. *Proceedings of International Conference on The Future of Education IConFEd) 2020*, *November 2020*, 17–18.

Hashim, A. S., Ahmad, R., & Shahrul Amar, M. S. (2017). Difficulties in Learning Structured Programming: A Case Study in UTP. *Proceedings - 2017 7th World Engineering Education Forum, WEEF 2017- In Conjunction with: 7th Regional Conference on Engineering Education and Research in Higher Education 2017, RCEE and RHEd 2017, 1st International STEAM Education Conference, STEAMEC 201*, 210–215. https://doi.org/10.1109/WEEF.2017.8467151

Hosanee, M., & Rana, M. E. (2018). A Refined Approach for Understanding Role of Variables in Elementary Programming. *Journal of Advanced Research in Dynamical and Control Systems*, *10*(11), 238–248.

Hundhausen, C. D., Olivares, D. M., & Carter, A. S. (2017). IDE-Based Learning Analytics for Computing Education: A Process Model, Critical Review, and Research Agenda. *ACM Transactions on Computing Education*, *17*(3), 1–26. https://doi.org/10.1145/3105759

Ishizue, R., Washizaki, H., Sakamoto, K., & Fukazawa, Y. (2018). PVC: Visualizing C programs on web browsers for novices. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, *2018-Janua*, 245–250. https://doi.org/10.1145/3159450.3159566

Islam, N., Shafi Sheikh, G., Fatima, R., & Alvi, F. (2019). A Study of Difficulties of Students in Learning Programming. *Journal of Education & Social Sciences*, *7*(2), 38–46. https://doi.org/10.20547/jess0721907203

Kurnianda, N. R. (2018). Database Design for Customer Retention and Loyalty Administration Information System. *IJCSMC:: International Journal of Computer Science and Mobile Computing*, *7*(10), 1-8.

Lubis, J. H., & Zamzami, E. M. (2019). Relational database reconstruction from SQL to Entity Relational Diagrams Relational database reconstruction from SQL to Entity Relational Diagrams. *Journal of Physics: Conference Series*, 10–15. https://doi.org/10.1088/1742-6596/1566/1/012072

Margulieux, L. E., Morrison, B. B., & Decker, A. (2020). Reducing Withdrawal and Failure Rates in Introductory Programming with Subgoal Labeled Worked Examples. *International Journal of STEM Education*, *7*(1). https://doi.org/10.1186/s40594-020-00222-7

Masura, R., Shahrina, S., Rodziah, L., Noor Faezah, M. Y., Noor Faridatul Ainun, Z., & Rohizah, A. R. (2012). Major Problems in Basic Programming that Influence Student Performance. *Procedia - Social and Behavioral Sciences*, *59*, 287–296. https://doi.org/10.1016/j.sbspro.2012.09.277

Mat Isa, N. A., & Md Derus, S. R. (2017). Students experience in learning Fundamental Programming : An analysis by gender perception. *Advanced Journal of Technical and Vocational Education*, *1*(1), 240–248.

Mohd Yusoff, K., Ashaari, N. S., Tengku Wook, T. S. M., & Mohd Ali, N. (2020). Analysis on the Requirements of Computational Thinking Skills to Overcome the Difficulties in Learning Programming. *International Journal of Advanced Computer Science and Applications*, *11*(3), 244–253. https://doi.org/10.14569/ijacsa.2020.0110329

Nelson, N., Sarma, A., & Hoek, A. van der. (2017). Towards an IDE to Support Programming as Problem-Solving. *Proceedings of the 2017 Psychology of Programming Interest Group (PPIG)*, 15.

Perera, P., Tennakoon, G., Ahangama, S., Panditharathna, R., & Chathuranga, B. (2021). A Systematic Mapping of Introductory Programming Languages for Novice Learners. *IEEE Access*, *9*, 88121–88136. https://doi.org/10.1109/ACCESS.2021.3089560

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

Rahmawati, V., & Rosyida, S. (2020). Analisa Model Rapid Application Development Dalam Membangun Sistem Informasi Sekolah Mengemudi. *Paradigma – Jurnal Informatika Dan Komputer*, *22*(1).

Rubio, M. A., Romero-Zaliz, R., Manoso, C., & De Madrid, A. P. (2015). Enhancing an introductory programming course with physical computing modules. *Proceedings - Frontiers in Education Conference, FIE*, *2015-Febru*(February). https://doi.org/10.1109/FIE.2014.7044153

Saad, A. (2020). Students' computational thinking skill through cooperative learning based on hands-on, inquiry-based, and student-centric learning approaches. *Universal Journal of Educational Research*, *8*(1), 290–296. https://doi.org/10.13189/ujer.2020.080135

Sim, T. Y., & Lau, S. L. (2018). Online Tools to Support Novice Programming: A Systematic Review. *2018 IEEE Conference on E-Learning, e-Management and e-Services, IC3e 2018*, 91–96. https://doi.org/10.1109/IC3e.2018.8632649

Tawfik, A. A., Graesser, A., Gatewood, J., & Gishbaugher, J. (2020). Role of questions in inquiry-based instruction: towards a design taxonomy for question-asking and implications for design. *Educational Technology Research and Development*, *68*(2), 653–678. https://doi.org/10.1007/s11423-020-09738-9

Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M. J. (2019). Relationship between perceived problem-solving skills and academic performance of novice learners in introductory programming courses. *Journal of Computer Assisted Learning*, *35*(2), 246–255. https://doi.org/10.1111/jcal.12326

Warner, J., & Guo, P. J. (2017). *CodePilot : Scaffolding End-to-End Collaborative Software Development for Novice Programmers*. *9*, 1136–1141.

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies*, *21*(3), 559–588. https://doi.org/10.1007/s10639-014-9341-9