Lost in Syntax: Improving Student Implementation Skills in Programming

Akbar Wiguna[1](✉), Etistika Yuni Wijaya[2], and Cindy Cahyaning Astuti[1]

[1] Information Technology Education, Faculty of Psychology and Education, Universitas Muhammadiyah Sidoarjo, Sidoarjo, Indonesia
akbar.wiguna@umsida.ac.id

[2] Informatics Education, Faculty of Education, Universitas Trunojoyo Madura, Bangkalan, Indonesia

Abstract. This study aimed to analyze the problem-solving difficulties of Information Technology Education students in Basic Programming. A qualitative approach with case study research was conducted using data collection techniques such as interviews, document studies, participant observation, and data triangulation. The study revealed that students struggle with problem identification, formulating problem-solving plans, implementing the plans with syntax and semantic errors, and lack of alternative plans. The findings highlight the need to improve programming education and provide students with effective problem-solving strategies. This study has implications for programming educators to develop better teaching methods and tools to enhance student problem-solving skills.

Keywords: programming · problem-solving · difficulties · education · students

1 Introduction

Information Technology Education (PTI) is one of the Study Programs at the Faculty of Psychology and Educational Sciences (FPIP) Muhammadiyah University of Sidoarjo (UMSIDA). One of the competencies that must be mastered by PTI students is programming competence. [1] and [2] also stated that programming ability has become a core competency for engineering and computer science students. Programming languages are standard instructions for commanding computers. Programming languages are currently being studied more and more, this is related to the progress of the times which makes technology an important thing to support progress. One of the programming competencies that must be mastered by students is in the Basic Programming course.

Programming abilities is formed from the learning process. Learning programming is a complex task because programming requires new ideas in thinking and creative skills in problem solving [3]. [4] found that the majority of students felt that computer programming was a difficult and complex cognitive burden. [5] stated that novice programming students experience different types of difficulties which contribute to high dropout and failure rates in introductory programming courses. Observations that have

T. T. Y. Alabdullah et al. (Eds.): ICIGR 2022, ASSEHR 750, pp. 94–102, 2023.
https://doi.org/10.2991/978-2-38476-052-7_12

been made found that students experience difficulties when learning programming. Student had difficulty in understanding concepts and problem-solving. The difficulties that occur in the learning process have an impact on the quality of graduate results. The problem that often occurs in Basic Programming courses is that students find it difficult to solve problems in programming.

Problem solving is one of the skills needed by students in programming competence. Problem solving is the core of programming [6]. Problem solving skills are seen as one of the abilities that must be possessed to be able to master programming competencies. Problem solving steps according to Weir in [7] are: identifying problems, developing problem solving plans, implementing problem solving plans and checking problem solving results. Polya in [3] defines problem solving as a process of finding a solution to a problem to achieve a goal. In more detail, [8] define problem solving abilities as activities that use cognitive or cognitive and physical means to overcome obstacles (problems) and develop better ideas. Weir in [7] suggests 4 levels of problem solving: identifying problems, Developing Problem Solving Plans, Implementing Problem Solving Plans and checking problem solving results.

This research will examine students' problem-solving abilities in programming. With the study of student programming problem solving, it is hoped that it can be a solution in overcoming student problem solving problems in programming competence. This study aims to examine students' problem-solving abilities in Basic Programming courses.

2 Methodology

This research uses a qualitative approach with a case study method. Qualitative research is a scientific research, which aims to understand the phenomenon of what is experienced by research subjects such as behavior, perceptions, motivations, actions, etc. holistically and by means of descriptions in the form of words in a special context that naturally by utilizing various scientific methods. The case study method is a research strategy in which the researcher carefully investigates a program, event, activity, process, or group of individuals. The researcher describes the proportion of students' problem-solving abilities in coding programs. This study will describe the analysis of the problem-solving abilities of Information Technology Education students in coding programs.

The presence of the researcher in this study played a role as a key instrument during data collection. Researchers are also planners, executors of data collection, analysis, interpretation of data, and in the end become reporters of research results. In carrying out the research, researchers were supported by cameras, tape recorders, video recorders, interview guides and observation guides.

This research was conducted at the Muhammadiyah University of Sidoarjo (UMSIDA) on students of the Information Technology Education Study Program (PTI) who were teaching Basic Programming courses.

In this study, data is information obtained from students through a research approach. What was said by the students is the main data source of qualitative data. More concretely, basically the qualitative data in this study consisted of student behaviors, descriptions of learning situations, events and interactions with students.

The basis for determining the subject/informant was carried out using a purposive sampling technique, namely determining sample members based on research objectives

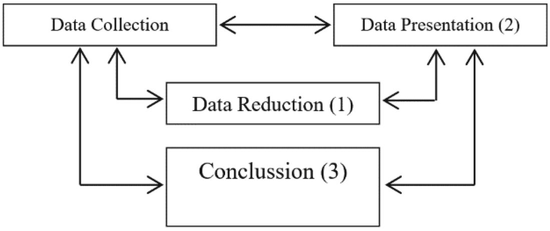

Fig. 1. Individual Case Data Analysis Flow

and researchers having strong professional reasons or considerations in selecting sample members [9]. Subjects in qualitative research are taken purposively with the intention that they do not have to represent the entire population, so that they have sufficient knowledge and are able to explain the actual situation of the research object. The informants to be used in this research are 19 students that learning Basic Programming Courses.

Data collection techniques used by researchers are interviews (interviews), observation (observation), documentation, and triangulation. The instruments used in this research were interviews, document studies, and observations. The flow of analysis of individual case data can be seen in Fig. 1.

The validity of the data (findings) is that each condition must fulfill: (1) demonstration of the correct value; (2) basic availability to apply; (3) the possibility of outside decisions that can be made about the consistency of procedures and the neutrality of findings and decisions [10]. Checking the validity of the findings is carried out with criteria for the degree of trust (credibility) which function: (1) carry out the inquiry in such a way that the confidence level of the findings can be achieved; (2) demonstrating the degree of trust in the findings by proving the multiple facts being studied [10].

This research was conducted with the following stages.

1. Pre-field stage

- Develop a research design, determine the problems to be discussed and determine the research design.
- Choosing a research campus through substantive theoretical considerations and studying and deepening the focus and formulation of research problems.
- Selecting and utilizing informants.
- Prepare research equipment.
- Maintaining good relations and adhering to their own background, norms, customs, habits and culture in dealing with research situations and contexts.

2. Field Work Stage

The stages of field work in this research are shown in Fig. 2.

Before the instrument is used, it will first be discussed with experts who are competent in the field of information technology education. To test the validity of the data in

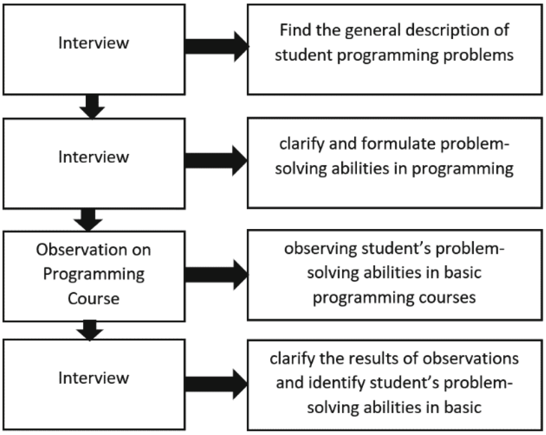
Interview
Find the general description of student programming problems
Interview
clarify and formulate problem-solving abilities in programming
Observation on Programming Course
observing student's problem-solving abilities in basic programming courses
Interview
clarify the results of observations and identify student's problem-solving abilities in basic

Fig. 2. Research Flow in Field Work

Table 1. List of Interview Questions on the aspect of problem identification ability

1	What the 1st things you do when you are doing programming assignment?
2	Do you have difficulty when identifying problems in programming?
3	How do you come up with a problem so that it can be solved programmatically?
4	Can you find out the conditions, data, and things that are not yet known in making the program?
5	Do you have difficulty describing the existing problems?

this study, it was planned to test the credibility of the data in the form of extended observations, triangulation with people outside the subject that have expertise in programming field.

3 Results and Discussion

The purpose of the interviews in this study was to determine problem solving abilities in Basic Programming courses. The interview was conducted on March 24, 2022. A list of interview questions related to aspects of problem identification ability is shown in Table 1.

Interviews that were conducted on March 24 to 8 students who had been selected obtained the following results:

Student 1 still has difficulty identifying problems, and relies a lot on existing references (modules or references from the internet). Student 2 is still not able to identify the problem correctly, when he gets an assignment he does it right away without looking for existing problems. Student 3 is still not able to identify the problem correctly, when he gets the assignment directly done without looking for existing problems, and relies on references to identify problems. Student 4 can still identify problems with the help of

Table 2. List of Interview Questions on the aspect of problem identification ability

1	Do you in working on programming assignments always develop a problem solving plan?
2	Did you develop a problem-solving plan in these forms: algorithm, pseudocode, flowchart?
3	Why don't you draw a problem-solving plan?
4	What were your difficulties when developing a problem-solving plan?
5	Do you like the activity of putting together a problem-solving plan?

flowcharts and references. Students 5 can identify problems by making initial designs and assisted with references (modules). Student 6 can identify problems by looking for examples of programs that are similar in solving problems. Student 7 can identify problems by looking for examples of programs that are similar in solving problems, but often have difficulty identifying problems. Student 8 has not been able to identify problems in doing assignments, but students have been able to guess what the program will look like.

A list of interview questions related to the Ability to Develop a Problem Solving Plan can be seen in Table 2.

Interviews that were conducted on March 24 to 8 students who had been selected obtained the following results:

Student 1 has difficulty making a problem-solving plan when the problem is too complicated, and still has difficulty planning program flow. Student 2 compiles a problem solving plan only when assigned to make a problem solving plan, and has difficulty determining the program flow. Student 3 compiles a problem solving plan only when assigned to make a problem solving plan, and has difficulty determining the program flow. Student 4 in doing programming assignments always plans to solve problems with flowcharts, and has difficulty determining program flow. Student 5 in doing programming assignments always compiles a problem solving plan with a flowchart, and has difficulty determining the program flow. Student 6 in doing programming assignments does not always develop a problem solving plan. If the description of the program flow is clear, then student 6 will not make a problem solving plan. Student 7 in doing programming assignments did not make a problem solving plan, and experienced difficulties in complexity and took a long time to make a problem solving plan. Student 8 in carrying out programming assignments did not develop a problem solving plan, due to the complexity of the process of preparing a problem solving plan.

A list of interview questions related to aspects of the ability to implement a problem-solving plan can be seen in Table 3.

Interviews that were conducted on March 24 to 8 students who had been selected obtained the following results:

Student 1 experiences difficulties in the ability to implement a problem-solving plan, such as errors still occur when completing programming assignments, even though they have re-checked and documented. Student 2 has difficulty implementing a problem-solving plan, such as still often forgetting instructions when completing programming assignments, even though they have re-checked and documented. Student 3 experienced

Table 3. List of Interview Questions on the Ability aspect of Implementing a Problem Solving Plan

1	Do you always check every step of the plan in carrying out your programming assignments in completing the task?
2	What are your difficulties in solving problems in programming? (syntax, semantics)
3	Do you make reports/documents regarding the correctness of each existing problem solving step?

Table 4. List of Interview Questions

1	Do you always check your programming task after you finished?
2	If there is an error or mistake what do you do?
3	Do you have an alternative plan in case of errors or mistakes?

difficulties in implementing a problem-solving plan, as there were still errors in semantics when completing programming assignments, even though they had re-checked and documented. Student 4 has difficulty implementing the problem-solving plan, such as errors still occur when completing programming assignments, even though they have re-checked and documented. Student 5 has difficulty implementing the problem-solving plan, such as errors still occur when completing programming assignments, even though they have re-checked and documented. Student 6 experienced difficulties in implementing a problem-solving plan, such as errors still occurring when completing programming assignments, even though they had re-checked and documented. Student 7 experienced difficulties in implementing a problem-solving plan, such as errors still occurring when completing programming assignments, even though they had re-checked and documented. Student 8 experienced difficulties in implementing a problem-solving plan, such as errors still occurring when completing programming assignments, even though they had re-checked and documented.

List of interview questions related to aspects of the ability to examine the results of problem solving can be seen in Table 4.

Interviews that were conducted on March 24 to 8 students who had been selected obtained the following results:

Student 1 always double-checks the work and is able to handle errors, but does not have an alternative plan if something goes wrong or goes wrong. Student 2 re-checks the work and is able to handle errors, but does not have an alternative plan if an error or mistake occurs. Student 3 re-checked the work and was able to handle errors, but did not have an alternative plan if an error or mistake occurred. Student 4 re-checked the work and was able to handle errors, but did not have an alternative plan if an error or mistake occurred. Student 5 re-checked the work and was able to handle errors, but did not have an alternative plan if an error or mistake occurred. Student 6 re-checked the work and was able to handle errors, but did not have an alternative plan if an error or mistake occurred. Student 7 re-checks the work and is able to handle errors, and has an alternative plan if an error or mistake occurs. Student 8 re-checked the work and was able to handle errors, but did not have an alternative plan if an error or mistake occurred.

The research findings on the analysis of students' problem-solving abilities in programming were obtained through interviews, document study, and observation. The research findings are presented following the order of the research focus.

1. Research Findings Focus 1: A. Problem Identification Ability

   In the problem identification step, the first thing to understand is what problems arise. Then, what are the conditions, data, and things that are not yet known. After knowing the existing conditions and data, the next step is to find ways to overcome the conditions of these problems, then describe the existing problems [7]. Based on the results of interviews with students, students still have difficulty identifying problems. and depends a lot on existing references (modules, examples of similar programs, and references from the internet)
2. Research Findings Focus 2: Ability to Develop a Problem Solving Plan

   In the step of preparing a problem solving plan/method, the first thing that must be understood is the analysis of the problems that arise, looking for almost the same problems that have been encountered. Then, find important things that are useful from the data, look for other appropriate data to find solutions to problems. If you have found a solution, write a solution with operational steps. Based on the results of interviews with students, students still have difficulty in preparing problem-solving plans, especially for planning appropriate program flow. The majority of students do not develop problem-solving plans if they are not assigned. Students are still not able to develop a student solution plan. This is in line with [11] research which states that the ability to solve and plan problem solving is still far from expectations.
3. Research Findings Focus 3: Ability to Implement a Problem Solving Plan

   In implementing the problem-solving plan, it is important to check each step of the plan in its implementation. After that, make a report regarding the correctness of each existing problem-solving step. Based on the results of interviews with students, all students experienced difficulties in their ability to implement problem-solving plans, such as syntax and semantic errors still occurring when completing programming assignments, even though they had re-checked and documented. This findings is in line with the findings in the research that conducted by [12] that students having issues in understanding filing, pointers, understanding syntax or debugging. [13] also stated that student having difficulty in understanding the basic concepts of programming structure, a program designed to complete a particular task, and learn the syntax of programming languages.
4. Research Findings Focus 4: Ability to Check Problem Solving Results

   In examining the results of problem solving, it is important to check the correctness of the solutions that have been implemented. After that, check the results and check the possible alternative solutions that appear. If there is, compare alternative solutions with the solutions you have found. Based on the results of interviews with students, all students re-checked their work and were able to handle errors, but did not have an alternative plan if an error or mistake occurred. [11] and [14] also find that consequently, students only write down what is known and what is being asked.

4 Conclusion

Students still have difficulty identifying problems. and much depends on existing references (modules, examples of similar programs, and references from the internet). Students still find it difficult to plan problem solving, especially to plan the appropriate program flow. The majority of students do not develop problem-solving plans if they are not assigned. All students experienced difficulties in their ability to implement problem-solving plans, such as syntax and semantic errors still occurring when completing programming assignments, even though they had re-checked and documented. All students re-checked their work and were able to handle errors, but did not have an alternative plan if an error or mistake occurred.

Acknowledgement. Thank you to DRPM Umsida for funding this research, and Universitas Muhammadiyah Umsida Information Technology Education Student, and Information Technology Education research team at Universitas Muhammadiyah Sidoarjo.

References

1. T. C. Yang, G. J. Hwang, S. J. H. Yang, and G. H. Hwang, "A two-tier test-based approach to improving students' computer-programming skills in a web-based learning environment," *Educ. Technol. Soc.*, vol. 18, no. 1, pp. 198–210, 2015.
2. H. C. Ling, K. L. Hsiao, and W. C. Hsu, "Can Students' Computer Programming Learning Motivation and Effectiveness Be Enhanced by Learning Python Language? A Multi-Group Analysis," *Front. Psychol.*, vol. 11, no. January, 2021, doi: https://doi.org/10.3389/fpsyg.2020.600814.
3. A. K. Emmanuel, O. G. Oluwadamilare, and O. A. Yomi, "Strategies for Improving Computer Science Students' Academic Performance in Programming Skill," *Curr. Top. Math. Comput. Sci. Vol. 4*, no. January 2015, pp. 41–48, 2021, doi: https://doi.org/10.9734/bpi/ctmcs/v4/2615f.
4. L. P. Baldwin and J. Kuljis, "Visualisation techniques for learning and teaching programming," *J. Comput. Inf. Technol.*, vol. 8, no. 4, pp. 285–291, 2000, doi: https://doi.org/10.2498/cit.2000.04.03.
5. A. Alammary, "Blended learning models for introductory programming courses: A systematic review," *PLoS One*, vol. 14, no. 9, pp. 1–26, 2019, doi: https://doi.org/10.1371/journal.pone.0221765.
6. I. Govender *et al.*, "Increasing self-efficacy in learning to program: exploring the benefits of explicit instruction for problem solving," *J. Transdiscipl. Res. South. Africa*, vol. 10, no. 1, pp. 187–200, 2014, doi: https://doi.org/10.4102/td.v10i1.19.
7. S. Purnakanishtha, P. Suwannatthachote, and P. Nilsook, "Development and Validation of a Problem Solving Skill Test in Robot Programming Using Scaffolding Tools," *Open J. Soc. Sci.*, vol. 02, no. 02, pp. 47–53, 2014, doi: https://doi.org/10.4236/jss.2014.22007.
8. M. G. Voskoglou and S. Buckley, "Problem Solving and Computational Thinking in a Learning Environment," vol. 36, no. 4, pp. 28–46, 2012, [Online]. Available: http://arxiv.org/abs/1212.0750.
9. Sukardi, *Metodologi Penelitian Pendidikan.* 2004.
10. F. Fiantika, *Metodologi Penelitian Kualitatif. In Metodologi Penelitian Kualitatif*, no. March. 2022.

11. Y. Darma, D. Suratman, A. Yani, and U. D. Susiaty, "Improving Problem-Solving Ability and Character in Subject-Specific Pedagogic with Heuristic Strategy," *JETL (Journal Educ. Teach. Learn.*, vol. 4, no. 2, pp. 333–338, 2019.
12. N. Islam, G. Shafi Sheikh, R. Fatima, and F. Alvi, "A Study of Difficulties of Students in Learning Programming," *J. Educ. Soc. Sci.*, vol. 7, no. 2, pp. 38–46, 2019, doi: https://doi.org/10.20547/jess0721907203.
13. M. D. Siti Rosminah and M. A. Ahmad Zamzuri, "Difficulties in learning Programming: Views of students," *1st Int. Conf. Curr. Issues Educ.*, pp. 74–78, 2012, [Online]. Available: http://rozmiens.com/myarticle/ICCIE2012_sitirosminah.pdf.
14. S. I. Malik, R. Mathew, R. Al-Nuaimi, A. Al-Sideiri, and J. Coldwell-Neilson, "Learning problem solving skills: Comparison of E-learning and M-learning in an introductory programming course," *Educ. Inf. Technol.*, vol. 24, no. 5, pp. 2779–2796, 2019, doi: https://doi.org/10.1007/s10639-019-09896-1.