Science Journal of Education
2023; 11(6): 211-219
http://www.sciencepublishinggroup.com/j/sjedu
doi: 10.11648/j.sjedu.20231106.15
ISSN: 2329-0900 (Print); ISSN: 2329-0897 (Online)

Debugging Support Tool Using Knowledge of Failure

Xingyue Zhu

Graduate School of Information Sciences and Arts, Toyo University, Saitama, Japan

Email address:
zxy81774@gmail.com

To cite this article:
Xingyue Zhu. (2023). Debugging Support Tool Using Knowledge of Failure. *Science Journal of Education*, *11*(6), 211-219.
https://doi.org/10.11648/j.sjedu.20231106.15

Received: November 7, 2023; **Accepted**: December 9, 2023; **Published**: December 20, 2023

Abstract: The objective of this study is to realize a debugging support tool to help programming novices solve program errors efficiently. The proposed tool accumulates and stores failure knowledge, including examples of errors encountered by novice programmers and their solutions. By utilizing this database, the tool can search and find appropriate solutions to errors encountered by users in real time. Errors are categorized as compile errors, link errors, run-time errors, and logic errors, each of which has its own unique solution. In this study, particular emphasis is placed on assistance with compile errors. Failure knowledge consists of sample source code that includes the location and cause of errors encountered by the novice user, as well as solutions. When an error occurs, the user enters the relevant source code and error message into the tool, and the tool searches the database to provide the corresponding solution and explanation. The system also provides a space for users to enter their own analyses and explanations, which can be shared with other novices to promote mutual understanding. The proposed system utilizes a structured database for efficient error resolution, containing tables for storing and retrieving error messages, corresponding references, and user considerations. The effectiveness and applicability of the proposed tool should be verified.

Keywords: Programming Education, Web Application, Debugging Support, Machine Learning

1. Introduction

According to the 2020 Information and Communications White Paper, the number of companies in the information service industry has been on the rise since 2011, and sales have continued to rise [1], and the importance of software development has increased further. It can be said that.

According to the results of a survey by Fitzgerald et al. [2], more than half of the students do not have sufficient debugging skills even after completing an introductory programming course that includes 15 to 20 weeks of exercises in the Java language.

Many programming learners tend to learn programs according to their proficiency level in the early stages of learning a programming language. However, even if you understand the program at each level, if the scale of the program to be implemented increases even slightly, it is possible that you will no longer be able to understand the program.

On the other hand, beginners learning programming are likely to encounter errors when programming. However, due to the inaccuracy and imprecision of the diagnostic information, it is very difficult for beginners to utilize error messages to solve problems [9]. Therefore, learners make various attempts to deal with program errors during the debugging stage. In the process, you will acquire a wide range of knowledge not only about programming languages but also about creating programs.

Although programming languages often have official documentation, it is difficult to say that there is enough material available for beginners to learn. Nowadays, it is possible to look for hints that may lead to solutions on websites such as Stack Overflow [12], but there is a lot more information on these sites than just information about debugging. It may take a considerable amount of time to search through such various information until you arrive at the page that will help you solve your problem. Especially for beginners, it may be difficult to find a solution on your own, such as when you don't know the appropriate search keyword or search method.

The purpose of this research is to create a tool that supports debugging program errors for beginners in programming. Specifically, we propose a tool that has the ability to obtain information on error cases and how to deal with them, which can be used when debugging. The proposed tool is assumed to

be able to accumulate debugging examples using data on compilation errors and their solutions as failure knowledge, and apply machine learning to classify and match the examples. With the proposed tool, even in situations where only beginners are learning, it is possible to provide appropriate support while considering each person's programming ability when debugging.

2. Related Knowledge

2.1. Debug

Debugging is the process of searching for and identifying the cause and location of errors and defects discovered through testing, etc., in the program, and correcting them so that they operate as intended.

Using the debugging function of Eclipse, an integrated programming environment, first create a number of breakpoints near where the error is displayed and enter debugging mode. Perform Step Over and proceed to the next step. Estimate the approximate error and search for related words and phrases where the error occurs. Identify the solution to the part that causes the error, modify the source code, and finish debugging.

2.2. Machine Learning and Deep Learning

Machine learning involves providing a large amount of data, learning the regularities of the data, and making predictions and classifications on unknown data. Machine learning is particularly used in research in the field of pattern recognition and is widely used to solve complex problems.

Deep learning uses a structure with many hidden layers in a neural network, and has the characteristic of automatically calculating feature quantities. Application examples of deep learning include image recognition and natural language processing.

AI is being democratized in natural language processing with the release of pre-trained models. Data analysts as well as domain experts are using data-driven AI products to solve problems. Data augmentation by humans allows domain experts' knowledge to be fed back into models as data to improve accuracy [14].

3. Related Research

3.1. DESUS [3]

Tsuruko Eki proposed a support system that teaches tracing, which is one of the program construction techniques, in the early stages of supporting programming education [3]. Furthermore, we use it in actual education, evaluate the results, and discuss the influence of tracing behavior on programming learning.

DESUS's debugging support is mainly based on the results of analysis of the program generation process. This analysis views programming as an intelligent behavior and tracks, analyzes, and evaluates all dynamic data. These are also significant from the perspective of cognitive science and information education. In this proposed method, we showed that learners not only have the necessary knowledge of programming languages and algorithms, but also use a variety of learning plans for programming, and that this influences their programming learning. Based on the program generation process, the students are asked to introspect their own actions, and what they say is recorded and converted into data. Based on this data, a program generation process record is created and used to construct a program generation process model.

Tracing is the process of checking the status of a program that has errors or defects during operation in order to identify the error location. Tracing is useful from an early stage of learning when you are stuck trying to find a solution to a program. The authors note that tracking can help learners get out of an impasse.

However, the debugging method using DESUS requires an instructor, making it difficult to apply to individuals who are learning programming.

3.2. Software Debug Supporting Tool Based on Program Dependence Analysis [4]

This paper proposes a debugging support tool that uses program dependency analysis, static slicing, and partial evaluation techniques to extract parts related to errors and perform processing on the extracted parts [4].

By using slices, it is possible to extract from a program only the statements that are related to a certain output variable. Even if it is difficult to extract effectively by slicing, the target can be made smaller using the partial evaluation function. In this paper, the target language is Pascal, and debugging support is provided by reducing the program reference range by slicing and partial evaluation from the program dependency graph obtained as a result of program dependency analysis.

Partial evaluation replaces variables in a formula with constants and deletes parts that are not executed or unnecessary statements. By performing a partial evaluation, even if a bug exists in the part to be deleted, it is considered that there is no impact at this point. Additionally, once analyzed during debugging, it can be saved internally, making it possible to perform other tasks on the same program without having to analyze it again.

The authors suggested that one way to do this would be to extend the target language by applying a dependency analysis algorithm that can also handle pointer variables and structures. It is also a debugging support tool that was developed using the C language and uses Pascal as its target language. Nowadays, Java and Python are often used, so it may be difficult to apply the proposed method as is.

3.3. Open AI [6]

In these years, Open AI automatically website where you can program has been opened.

Log in directly to the site playground. If you copy the

source code or error message and ask a question, the corrected code will be displayed. However, depending on the way you ask questions, you may not be able to get the answer you want. For example, if you ask for only part of the error message, it might say, “To fix this problem, please re-examine the source code, identify the error, and fix it.” Detailed error messages are displayed, so you can refer to them to identify the problem and fix it." Furthermore, since the user's own learning status is unknown, explanations are often difficult for beginners to understand, and in these cases, conversational communication is difficult.

Playout In addition to ChatGPT to fix errors in the form of a conversation. However, depending on the error information you entered, ChatGPT may generate an error.

In addition, since artificial intelligence is used to analyze and compose the code, if the copied content contains confidential information, there is a risk of that information being leaked. I'm also concerned about safety.

3.4. Answer Selection [7]

There may be situations in which the optimal solution for the same error is different. Answer selection is the main method used to deal with this situation.

Answer selection is given a question and a choice of answer candidates for that question, and the goal is to find the best answer candidate among the choices that can accurately answer the question. The main problem with answer selection is that the correct answer may not share a lexical unit directly with the question. Rather, there are cases where there is only a semantic relationship. In addition, the answers may be noisy and contain a lot of irrelevant information.

The general idea for answer selection problems is to first learn the question-and-answer representations using LSTM or CNN neural encoders, and then perform text matching to transform the task into a classification or ranking problem. Neural ranking models rank answer candidates for questions in QA, and the core is the design of interaction functions between question expressions and answer expressions.

Yi Tay [7] et al. refer to deep sentences (sentences whose meaning and structure are complex to analyze and require deeper insight. For example, sentences that include complex structures and grammatical features are called ``deep sentences.'' We propose a holographic dual LSTM (HD-LSTM), which is a unified architecture that supports both modeling and semantic matching of (expressed). Essentially, the model is trained end-to-end, whereby the parameters of the LSTM are optimized to best explain the correlation between questions and answers. Yi Tay et al. first experimented with TREC QA, and the results clearly showed that HD-LSTM outperformed all other deep learning models.

Table 1 shows the results for all public models, including non-deep learning systems. It is clear that deep learning significantly outperforms traditional methods at this task. HD-LSTM also outperforms all models even on the smaller TRAIN dataset.

Table 1. *Performance comparison of all public Models on TREC QA datasets [7].*

Model	P@1	MRR
Random Guess	0.2000	0.4570
Okapi BM-25	0.2250	0.4927
CNN	0.4135	0.6323
LSTM	0.4875	0.6829
NTN-LSTM	0.5548	0.7309
HD-LSTM	0.5569	0.7347

In addition, as shown in Table 2, Yi Tay et al. used LSTM and conducted the same experiment on Yahoo QA and found that it had state-of-the-art performance.

Table 2. *Experimental results on Yahoo QA dataset [7].*

Model	MAP	MRR
Wang et al. (2007)	0.6029	0.6852
Heilman and Smith (2010)	0.6091	0.6917
Wang and Manning (2010)	0.5951	0.6951
Yao (2013)	0.6307	0.7477
Severyn & Moschitti (2013)	0.6781	0.7358
Yih et al. (2013)	0.7092	0.7700
Yu er al. (2015)	0.7113	0.7846
Severyn et al. (2015)	0.7459	0.8078
HD-LSTM TRAIN	0.7520	0.8146
HD-LSTM TRAIN-ALL	0.7499	0.8153

The HD-LSTM designed by Yi Tay et al. has been confirmed to be practical for TREC QA and Yahoo QA, but in this study, it is questionable whether it is effective in finding the optimal solution for the same error.

3.5. Similar Class/Method Detection Method Using Program Execution History [8]

Recently, plagiarism of source code has increased. Source code plagiarism can be either plagiarism of the whole source code or plagiarism of a part of the source code such as classes or methods. When a part of the source code is plagiarized, a code clone detection method can be used to identify the plagiarized part, which can identify the duplicated parts among software. However, there are obfuscation techniques that rewrite software to make it difficult to understand, and if the plagiarist uses obfuscation techniques on the source code, it will be difficult to identify the plagiarism. To solve this problem, we proposed a method to detect similar class and method pairs by analyzing the execution history of programs that are less affected by obfuscation. Obfuscation can protect a program from unauthorized analysis. When obfuscation is applied, it is difficult for existing code clone detection techniques to identify equivalent processes as code clones.

The purpose of this paper is to identify cases where a plagiarist has used obfuscation techniques to transform a static structure into a program, such as a method being moved to another class. Possible.

This paper proposed a method to detect class clone pairs and method clone pairs from the two program execution histories given two program execution histories. Since this method targets the program execution history, it can be detected even if it is obfuscated.

This paper proposed a method to detect class clone pairs and method clone pairs by dividing the program execution history into phases and comparing the method call sequence of each phase. We then applied the proposed method to an actual application and confirmed that it was possible to identify the same components before and after obfuscation. However, it is necessary to confirm the resistance to dynamic name resolution using reflection, which is a typical obfuscation technique other than name conversion and method distribution. Also, ProGuard not support dynamic name *resolution using reflection. Therefore, the proposed method* that performs execution history analysis is expected to be highly resistant to this obfuscation.

4. Debugging Support Tool Using Failure Knowledge

Although there is a proposal by Murata Takumi [10] to realize debug-supported chatbot by tracing method, this method is based on DevBot [11]. However, this method also requires teachers to trace students.

In addition, by using bugs that are actually created by novice programmers for learning debugging, it is thought that novice programmers can learn how to deal with bugs that they are likely to face. However, there is no support for debugging that captures the tendency of the bugs of the novice students under study, such as forgetting semicolons [13].

The proposal is to realize a tool that allows novice programmers to accumulate knowledge of errors that occur in programming and how to deal with them as failure knowledge, and to search the accumulated data when an error occurs to efficiently find a solution to the error.

4.1. Types of Bugs and Errors

Types of bugs and errors include compilation errors, link errors, runtime errors, and logic errors [5].

Compilation errors are failures or interruptions due to some fatal problem occurring when converting source code written in a programming language to machine language code, and code errors are the main cause. Link errors occur when generating executable files from object code. A runtime error is an error that occurs when a program is executed. A logic error is an error in which the execution result of a program does not turn out as intended.

Error messages generated by syntax errors in compiled languages usually include the number of lines or characters in the line where the error occurs and a string indicating the type of error [15].

4.2. Knowledge of Failure

Failure knowledge in this research is a summary of source code containing errors that occur when compiling a program, as well as the causes and solutions for those errors. First, record the source code in which the error occurred in the tool, and then search for a solution that is suitable for dealing with this error from the solutions stored in the tool.

```
6⊖    public static void main(String[] args) {
7         int c;
8         int sum = 0;
9
10        System.out.print("Please enter a value: ");
11
12        c = new Scanner(System.in).nextInt();
13        for (int i = 1; i <= c; i++) {
14            sum = i + sum;
15        }
16
17        System.out.println(sum);
18    }
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Syntax error on tokens, delete these tokens

        at text00002.Text00002.main(Text00002.java:9)
```

Figure 1. *Compilation error example.*

Figure 1 shows an example of a compilation error, and the error message says: Unresolved compilation problem: Syntax error on tokens, delete these tokens. This error is caused by the source code containing double-byte spaces. The proposed tool first determines the type of error, removes solutions that do not correspond to the error, selects the optimal solution, and outputs it to the screen.

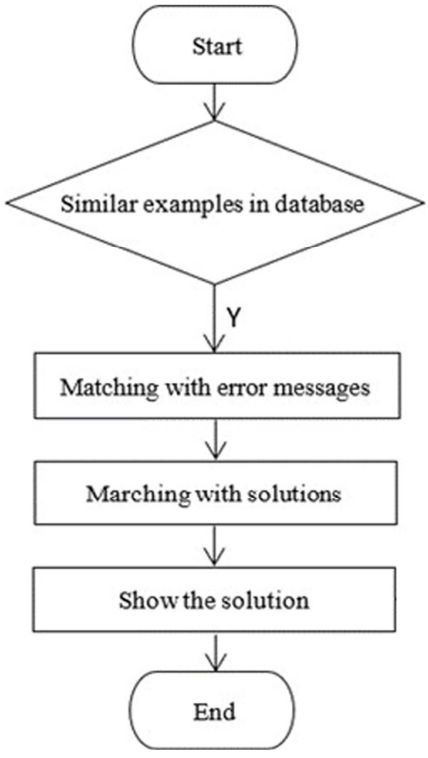

Figure 2. *Processing flow.*

Figure 2 shows the processing flow of the proposed tool. When an error occurs, if a solution is recorded in the tool, the solution will be displayed, and if the corresponding data is not found in the tool's database, the solution will be searched

for on the Internet and written into the database. When writing to the database, the proposed tool accumulates and collects records of errors and debugging history. Many existing development support tools often present common debugging errors and solutions. Search accuracy is important for ease of use by programming beginners.

4.3. Design of Debugging Support Tools Using Failure Knowledge

4.3.1. Identifying Error Messages

It is quite common for beginners to encounter the same error over and over again while programming. The same error message may correspond to different solutions, so it is necessary to specify which source code corresponds to which solution.

In this research, we propose a method for identifying error messages and their countermeasures.

At the beginning of the system, the instructor enters the solution and explanation first, making it ready for beginners to search.

As shown in Figure 3, the image diagram is divided into three columns.

From the left, the source code and error message columns are columns that the user actually inputs, input the source code and error message, and search for the input error message by keyword. The solution and explanation column contains the solution and explanation for the error that was previously saved in the database by the instructor.

When the "Search" button is clicked, this is where the previously saved solutions and explanations will be output. The reference example and reference example discussion columns are the source code and its considerations for beginners using the tool. When the "Search" button is clicked, not only the solution and explanation columns, but also the reference example and reference example discussion columns are already saved. Outputs.

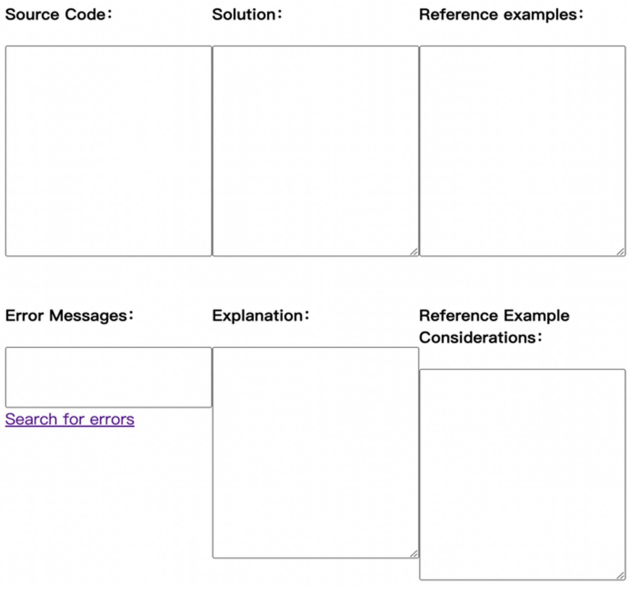

Figure 3. *Operational image of the proposed system.*

If there is only one solution, the specific flow is shown below.

First, for beginners, users paste their own source code into the source code section. Paste the error message output by eclipse in the error message section. On the first screen, the solution and explanation fields are blank. After copying, press the error search button to display the error, its solution, and a detailed explanation. In addition, the reference example and discussion of the reference example output the already saved source code of another user who encountered the same error and its discussion.

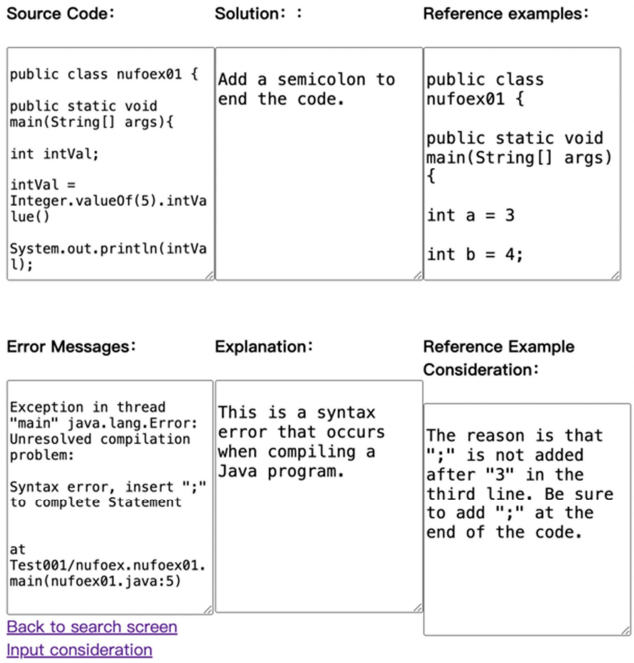

Figure 4. *Error message search result diagram,*

Regarding the search method, Java programs are divided into ``Errors" and ``Exceptions" depending on the content of the exception that occurs, and the difference lies in whether the exception can be handled by the program or not.

Specifically, "Exception in thread ‘main’ java.lang. ArrayIndexOutOfBoundsException: 10” Take for example. First, change "." and ":"split separate with (), “Exception in thread ‘main’ java”, “lang”, “ArrayIndexOutOfBoundsException”, “10”. Of the four, "lang" and "ArrayIndexOutOfBoundsException" are used as keywords. First, "lang" is caught and the direction of the error is identified. Then, catch "ArrayIndexOutOfBoundsException," which is more detailed, to identify the error message and corresponding solution.

As shown in Figure 4, after inputting the source code and error message, the corresponding solution and explanation from the database will be displayed. In addition, the reference example location can also present errors that beginners have encountered and their considerations.

In addition, the proposed system allows users to search for their own search history and input their thoughts.

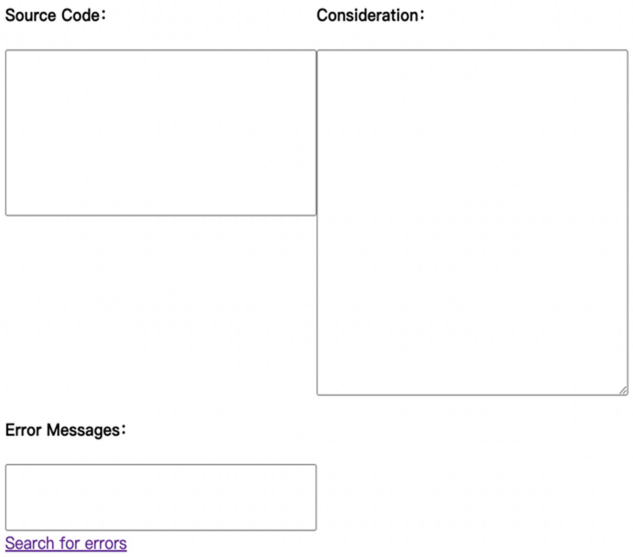

Figure 5. *Consideration input screen diagram.*

As shown in Figure 5, you can enter your own thoughts on the source code and error messages entered on the right side. Also, what you enter can be viewed by others as a reference example.

For consideration, since this is where the same novice would enter, name the Consideration and save the user failure knowledge with save (). When a beginner encounters an error and searches based on the error message, it is sometimes difficult to understand if the instructor only thinks in terms of saved solutions and explanations. In this situation, the same novice's insights would be of considerable help in deepening the user's understanding. When the user wants to retrieve a study, he or she can use get() to output his or her study and knowledge of failure from the database on the same screen. In addition, the source code and the discussion are output as the reference example and the discussion column of Figure 3, the source code as the reference example and the discussion as the discussion of the reference example.

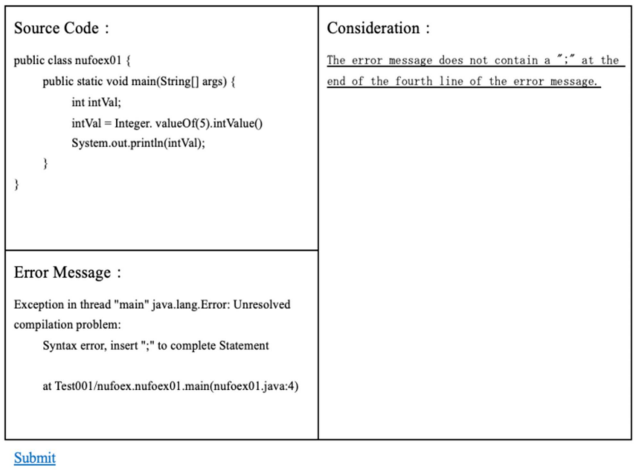

Figure 6. *Search result diagram for failure knowledge.*

Figure 6 shows the search results for failure knowledge by beginners. When you want to output your considerations, use get () to retrieve them from the database and output them to the screen based on your error message.

4.3.2. Concrete Explanation

The structure of the data mainly used in this research is as follows.

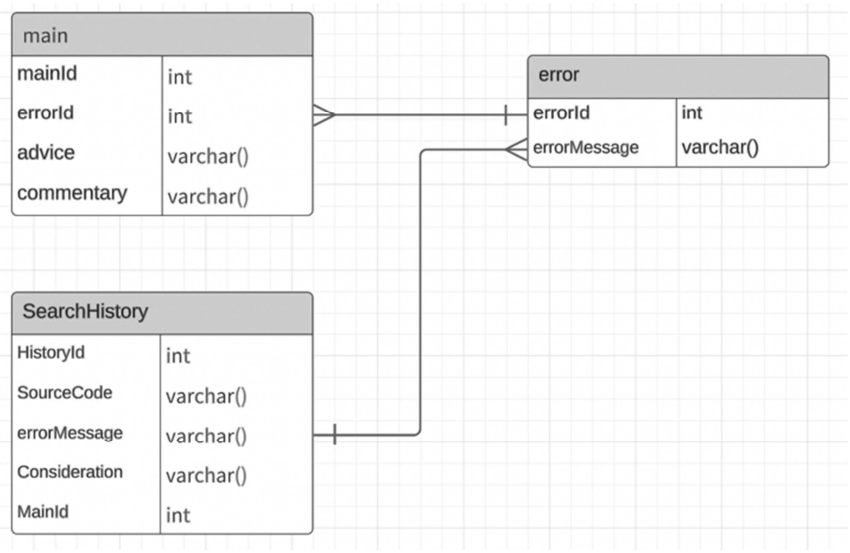

Figure 7. *ER model diagram.*

Figure 7 is a diagram modeling the database. As shown in Figure 7, important databases can be divided into three types.

The specific contents are explained below.

First, as shown in Table 3, the main table stores solutions and explanations for certain types of errors. In errorId, it is concatenated with errorMessage corresponding to Table 4. When you input an error message, you can specify the errorId, and use errorId to search for the corresponding solution and explanation in the main table and output it to the screen.

Table 3. *Main.*

MainId	errorId	advice	commentary
1	1	Add a semicolon when you want to end the code.	A syntax error occurred when compiling a Java program. This indicates that a syntax error has occurred because the sentence is not complete. It seems that part of the code does not end with a semicolon.
…	…	…	…

Table 4 shows the database that stores error messages. errorId is connected to errorId in Table 4.

Table 4. *Error Messages and Corresponding Reference Examples.*

Id	errorId	errorMessage
1	1	exception in thread "main" java.lang.Error: Unresolved compilation problem: Syntax error, insert ";" to complete Statement at Test001/nufoex.nufoex01.main (nufoex01.java:4)
…	…	…

Table 5 MainId and Table 3 MainIdinputted by the same beginner in Table 5. SourceCodeand its corresponding Consideration outputs a reference example and a discussion of the reference example on the screen.

Table 5. *Search History.*

Id	HistoryId	SourceCode	errorMessage	Consideration	MainId
1	1				1

5. Consideration

Tsuruko Eki [3] uses the tracing method. If a beginner makes an error in tracing, the instructor must advise the beginner on the spot through DESUS. During support, the instructor asks a number of questions about the learner's program. Learners can answer these questions in order, create a conversation according to the progression of questions, and arrive at a teaching method.

The question progression in Figure 7 is constructed assuming the teacher's questions at this time.

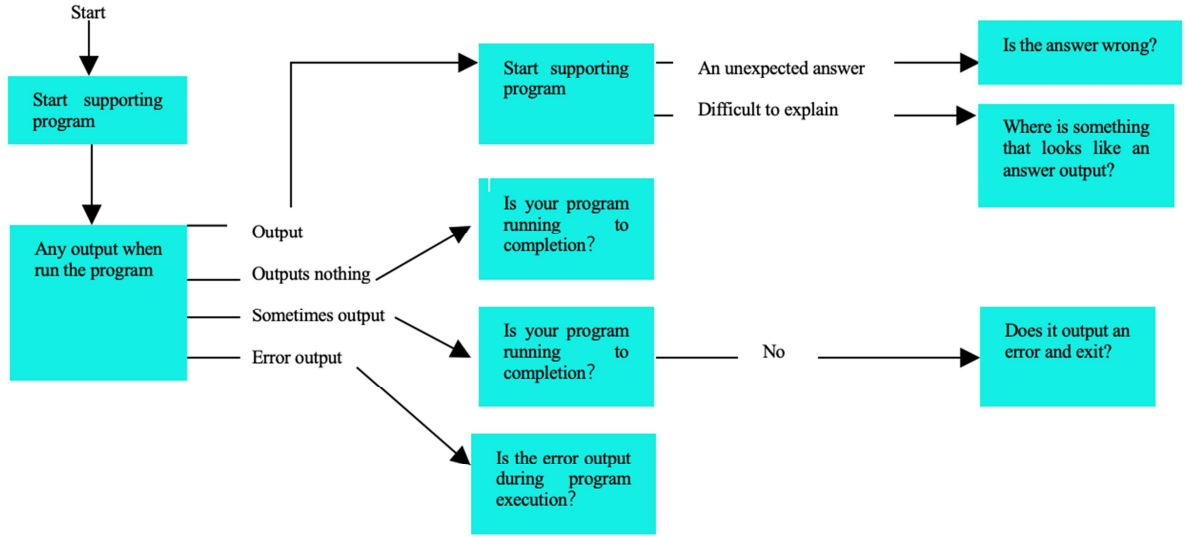

Figure 8. *A series of questions that determine the tracing instruction method [3].*

As shown in Figure 8, in this method, an instructor is essential at every step, and if there is no instructor at a certain step, it will be difficult for beginners to learn on their own.

We proposed a method that uses failure knowledge to solve problems where errors cannot be resolved without a supervisor. With the proposed method, beginners can search for solutions through previously accumulated data without the need for an instructor to be present. In addition, not only the instructor but also the source code and thoughts of other learners can be used as explanations. It is thought that the same beginner's explanation would be easier to understand.

In Shinichi Sato's research [4], we prototyped a debugging support system that uses slices and partial interpretation to extract only the parts related to the statement or variable of interest from the entire program and performs processing only on those parts. A slice is a set of statements that affect the value of a variable in a statement s in a program, or a set of statements that are affected by the definition of a variable in s. By using slicing, only the statements related to a certain output variable can be extracted from the program.

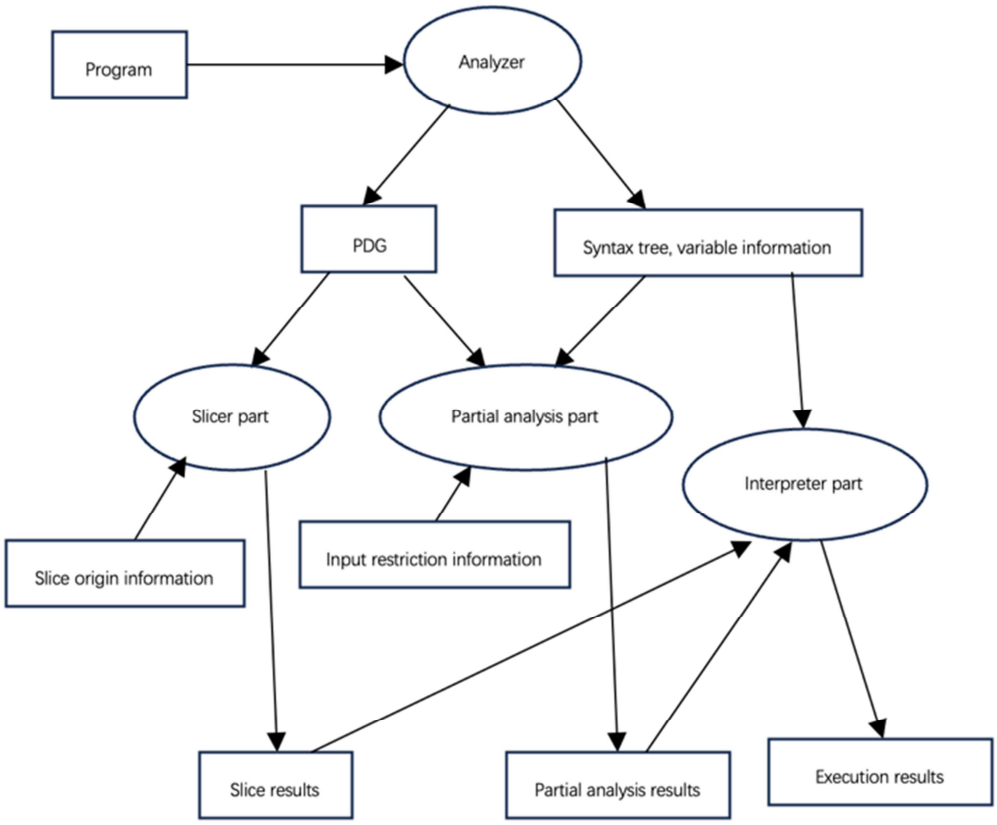

Figure 9. *System configuration diagram [4].*

However, it is unclear whether this tool is applicable to other languages. Also, although this tool talked about the analysis method, it is unclear how specifically it should be modified. For beginners, even if you understand it, you may not be able to modify the source code.

6. Conclusion

In this study, we proposed a debugging support tool that uses failure knowledge for beginners. So far, we have collected error messages encountered during software development and their solutions, examined functions to record and store them in a database, and recorded considerations for error messages encountered during software development and source code, and have provided reference examples. I thought it might be helpful for others.

A future challenge is to confirm how effective the tool proposed in this research is.

ORCID

0009-0001-8020-4542

References

[1] Ministry of Internal Affairs and Communications, Information and Communications White Paper 2020 Edition, https://www.soumu.go.jp/johotsusintokei/whitepaper/r02.html.

[2] Sue Fitzgerald, Gary Lewandowski, Renée McCauley, Laurie Murphy, Beth Simon, Lynda Thomas, Carol Zander "Debugging: finding, fixing and flailing, a multi-institutional study of novice debuggers", Computer Science Education, vol. 18, No. 2, PP. 93-116 (2008).

[3] Tsuruko Eki, "Research on learning support for debugging tactics for programming beginners", Kyushu Institute of Technology doctoral dissertation (2009).

[4] Shinichi Sato, Hajime Iida, Katsuro Inoue. "Prototype of debugging support tool based on program dependency analysis". Journal of the Information Processing Society of Japan, 37 (4), PP. 536-545 (1996).

[5] Keita Matsumura, Masayuki Arai. "Development of a debugging support tool for C language beginners to collect bug reports" ISSN 2186-5647 (2018) Summary of the 51st Academic Conference of the College of Industrial Engineering, Nihon University.

[6] OpenAI homepagehttps://openai.com/api/

[7] TAY Yi, Luu Anh Tuan, Minh C. Phan, Siu Cheung Hui. "Learning to rank question answer pairs with holographic dual lstm architecture", Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval, PP. 695- 704. (2017).

[8] Masakazu Ioka, Norihiro Yoshida, Katsuro Inoue. "Similar class/method detection method using program execution history" ISSN 0289-6540 (2014) Volume 31, Issue 1, PP. 1_110-1_115.

[9] Davin McCall, Michael Kolling. "A New Look at Novice Programmer Errors", ACM Transactions on computer Education, Vol. 19, No. 38, PP. 1-30 (2019).

[10] Takumi Murata, Toya Nakayama, Hiroaki Hashiura. "A Method for Debugging Java Programs using Chatbots", Information Processing Society of Japan, 84 (2022-1) PP. 349-350 (2022).

[11] Linda Erlenhov, Francisco Gomes de Oliveira Neto, Philipp Leitner. "An Empirical Study of Bots in Software Development: Characteristics and Challenges from a Practitioner's Perspective", Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software, US, PP. 445–455 (2020).

[12] Moriuchi kai, Taku Okuno, "A Study on a Debugging Support System Using Crowd Knowledge ", ISSN 2188-8825, SE-210, vol. 4, PP. 1-5 (2022).

[13] Gakuto Akiyama, Tsukasa Nakamura, Masanari Kondo, Yasutaka Kamei, Naoyasu Ubayashi, "A case study of automatic debugging problem generation using novice programmers' bug fix histories" Computer Software, vol. 39, No. 4, PP. 10-16 (2022).

[14] Masato Ota, Faisal Hadiputra. "Uncertainty and Explanation-based Human Debugging of Text Classification Model", The 37th Annual Conference of the Japanese Society for Artificial Intelligence, PP. 3L5GS1103-3L5GS1103 (2023).

[15] Ryota Kondo, Masataka Nagura. "A Learning Support Method for Novice Programmers based on Categorization of Compilation Error Messages", Japan Society for Software Science and Technology, PP. 63-74 (2023).