Error Debugging Problem in Python Programming Learning Assistant System using Error Debugging Method extension Blank Element Selection Algorithms

hsu wai hnin*
Department of Computer Engineering and Information Technology, Yangon Technological University, Yangon, Myanmar
hsuwaipyplas01@gmail.com

Khin Khin Zaw
Department of Computer Engineering and Information Technology, Yangon Technological University, Yangon, Myanmar
thihakhinkhin85@gmail.com

ABSTRACT

Abstract: Python Programming Learning Assistant System (PYPLAS) has been developed to support python programming educations. Currently, PYPLAS provided the element fill-in-blank problem to improve the student programming skill. Students studied python error types and debugging technique in python code. In this paper, error debugging problem in PYPLAS is proposed so that students can study the types of python errors. Error debugging problem is generated by using error debugging method to replace incorrect element into correct ones. For evaluations, the 100 error debugging problems are generated to analysis the correctness of error debugging method according to python grammar. We generated 10 problems and asked 5 students in two universities to solve them. Eventually, the educational effects in python programming learning are verified by generating error debugging problems to assign the students in python programming course.

CCS CONCEPTS

• **Theory of computation** → Theory and algorithms for application domains; • **Information systems** → World Wide Web; • **Software and its engineering** → Software creation and management; • **Computing methodologies** → Artificial intelligence; • **Additional Keywords and Phrases: fill-in-blank problem, blank element selection algorithm, error debugging problem, error debugging method, python programming educations**;

ACM Reference Format:
hsu wai hnin and Khin Khin Zaw. 2022. Error Debugging Problem in Python Programming Learning Assistant System using Error Debugging Method extension Blank Element Selection Algorithms. In *2022 The 8th International Conference on Computing and Data Engineering (ICCDE 2022), January 11–13, 2022, Bangkok, Thailand.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3512850.3512858

ICCDE 2022, January 11–13, 2022, Bangkok, Thailand

ACM ISBN 978-1-4503-9571-7/22/01…$15.00
https://doi.org/10.1145/3512850.3512858

1 INTRODUCTION

Python programming has been educated in training centers, colleges and universities. Python programming is a general purpose programming language that is highly used in the current times [1]. Web-based (PYPLAS) Python Programming Learning Assistant System has been developed to improve Python Programming educations. PYPLAS has provided the element fill-in-blank problem to improve the student programming skill. Blank element selection algorithm is used to generate fill-in-blank problem. The element fill-in-blank problem asked the students about python grammar types. The blank element represents python basic grammatical elements such as a reserved word, an identifier, a control symbol, an operator in conditional expressions, and a library name [2].

In universities, teachers teach students to know the python error types. Three type of python error types are syntax errors, logical errors and run time errors. Debugging is the process of finding and fixing the errors in a program. Students learned debugging techniques to find and fix the errors in their program to improve their programming skill [2–4]. Error debugging problem for python code has generated to know the python error types and to improve the student's programming skill.

In this paper, the error debugging problem for python code is proposed to analysis the student's grammar skill. The error debugging problem is generated by using error debugging method to replace the selected elements into incorrect elements in python code. The error debugging method is generated error words in python code according to python error types such as syntax errors and logical errors. According to syntax errors, this method tested the elements such as reserved word, operator word, control word and variable word to replace the error words. According to logical error, this method tested the elements operator and variable word to replace the suitable error words. In PYPLAS, the error debugging problem is given to the students to test the student's debugging skill.

For analysis the correctness of error debugging method, the 100 error debugging problems have generated to replace the selected elements into incorrect elements in source codes according to syntax errors and logical errors. For evaluations, the five students in two universities solve the 10 problems. The average correct rate is 82% of all problems and the standard deviation is 6.099 of all problems. The rest of this paper arranged such as firstly we describe the review of blank element selection algorithm. Then, we present error debugging problem functions in PYPLAS. Then, we present the error debugging method for python code. Then, we present the

correctness and the evaluation results of each student. Finally, we conclude this paper with some future works.

2 REVIEW OF BLANK ELEMENT SELECTION ALGORITHM

In this section, the blank element selection algorithm in PYPLAS for python code is reviewed [2, 5, 6, 8].

2.1 Vertex generation for constraint graph

Each vertex describes a candidate element to be blank in python code by using the constraint graph. The vertices generated from python code by using LR-parser [5, 6].

2.2 Edge generation for constraint graph

Each edge is generated the connection between two vertices in python code that should not be blanked at the same time. Three categories of edge generation are group selection categories, pair selection categories and prohibition category [7, 8].

2.2.1 Group selection category. The vertices related to each other in python code are grouped together. Constraint graph is generated the edge among them. Firstly, the vertex as the largest number of incident edges in the constraint graph as is selected for each group. Then, edges are generated between selected vertices. The other remaining vertices cannot blank at the same time to satisfy the unique answer. Four conditions described in this category.

1. Identifier group: The same identifiers of variable, class name, method name grouped so that they cannot be blank at the same time for the unique answer.
2. Reserved word group: The reserved words in pairs grouped so that they cannot be blank at the same time.
3. Data types group: The data types for variables in one equation grouped together so that they cannot be blank at the same time for the unique answer.
4. Operator group: The three or more relational operators such as <, >, <=, >=, ==, = in the condition at expression are grouped together [6, 8].

2.2.2 Pair selection category. The pair is generated the two vertices connecting to each other in python code. For each pair, an edge extracted between the paired elements so that at least one element is not select for blank. Four conditions described in this category.

1. Continuous element pair: The continuous elements in each statement of python code paired. By pairing the continuous elements, at least one non-blanked element describes between two blanked elements. The parameter BG can increase or decrease the number of non-blanked elements in each statement of python code. The parameter CB can increase or decrease the number of non-blanked elements in each statement of python code.
2. Variable pair: In each equation of the python code, the two variable elements paired to select one blank element at the same time.
3. Reserved words pair: The elements representing any pair of reserved words in python code paired so that at least one element is not blank at the same time.
4. Control symbol pair: The elements representing two related control symbols paired such as “[”, “]” (square bracket), “(”, “)” (bracket), “{”, “}” (curly bracket) and then “:” (full column), “.” (Dot), “,” (comma) are also paired in a code [7, 9].

2.2.3 Prohibition category. An element prohibited to be blank so that it does not satisfy the uniqueness. There are three conditions in this category.

1. Constant: The constant element prohibited to be blank in the python code. If it is blanked, the student cannot be answered the correct one.
2. Single identifier: The element representing one identifier in python code prohibited to select the blank. If it is blanked, the student cannot be answered the original identifier.
3. Arithmetic operator: The element representing an arithmetic operator such as “=”, “+”, “*”, “%”, “/”, “-” and logical operator such as “&”, “|”, “^”, “!” is prohibited to select blank element in python code [2, 10].

2.3 Compatibility graph generation

The compatibility graph has generated by taking the complement of the constraint graph to represent the pairs of elements that can be blank simultaneously. According to python grammar, the inverse graph is generated to define the true of link vertex can be selected as blank element in python code [8, 11].

2.4 Maximal clique extraction of compatibility graph

By using the simple greedy algorithm, the maximal clique of compatibility graph extracted to find the maximal number of blank elements in python code. A clique of a graph represents its sub graph where any pair of two vertices connected by an edge [9, 10]. The steps of algorithm represented as follows:

1. It calculates the degree every vertex in the compatibility graph.
2. It selects one vertex among the vertices whose is a maximum degree.
3. If two or more vertices have the same maximum degree, select one randomly. If the selected vertex is a control symbol and the number of selected control symbols exceeds 1/3 of the total number of selected vertices, remove this vertex from the compatibility graph and go to (5).
4. It adds the selected vertex for blank, and removes it as well as its non-adjacent vertices from the compatibility graph.
5. If the compatibility graph becomes null, terminate the procedure.
6. The continuous blank number CB represented to control the number of continuous blank elements. The maximal number of CB elements is equal to the number of continuous blank elements in a statement of python code [12, 13].

2.5 Fill-in-blank problem generation in python code

In the maximal clique, procedure generated to sustain the total number of blank control symbols, because a python code usually has many control symbols. Then, the ratio of the average number of

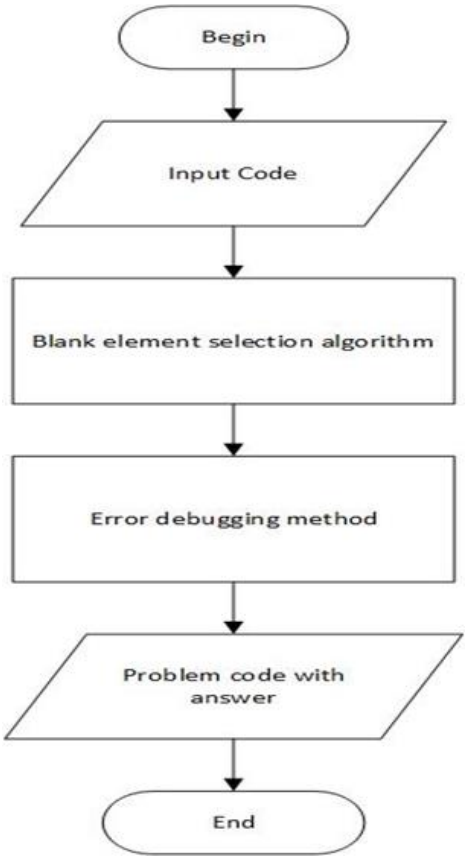

Figure 1: Flow diagram of error debugging problem generation

blanks for control symbols and other symbols defined as 1/3 by the algorithm. This appropriate ratio selected to generate the feasible fill-in-blank problems for novice students [2, 8].

3 ERROR DEBUGGING PROBLEM IN PYPLAS

In this section, Error debugging problem described in python programming learning assistant system (PYPLAS).

3.1 Error Debugging Problem Generation

In error debugging problem in PYPLAS, elements selected by the blank element selection algorithm in python code and replaced into incorrect elements by the error debugging method. The error debugging method changed the selected elements in python code into incorrect elements according to syntax errors and logical errors. In PYPLAS, the error debugging problems asked the students to find the error elements in python code and to answer the correct elements in the code. The correctness of each answer code has analyzed by comparing it with the correct answers according to string matching techniques. In PYPLAS, student can answer the error debugging problems and analyze their result of each problem by calculation average correct rate. In PYPLAS, the error debugging problems generated the following steps in figure 1.

4 ERROR DEBUGGING METHOD

In this section, the error debugging method extends the blank element selection algorithm of the python code is presented.

4.1 Features of Error debugging method

An error debugging method extends the blank element selection algorithm to replace the selected elements with incorrect elements in python code such as syntax and logical error grammatically. Firstly, the selected elements of each group or pair by using constraint graph are replaced with incorrect elements in python code according to python grammatically or logical errors. In error debugging method, at least one original element remained and others replaced with the incorrect elements from each group or each pair. Besides, the prohibited elements do not replace with incorrect elements in python code to answer the student difficulty. The error debugging method generated to obtain incorrect elements in python code according to syntax errors and logical errors in python grammatically.

- Syntax errors : syntax errors are created to test variable, identifier, reserved words and control words by the followings:
- The same variables selected to obtain grouped together, where at least one variable name remained and others selected. The multiple dictionary (enchant) is used to obtain the list of similar variable names by matching with selected variable names. The suitable incorrect variable names chose from the list of similar variable names by using the fuzzy method.
- The same identifier (class name or method name) selected to obtain grouped together, where at least one variable name is remained and others selected. The selected variable names are matched with multiple dictionary (enchant) to obtain the list of similar variable names. The suitable incorrect variable names chose from the list of similar variable names by using random and fuzzy method.
- The pairs of reserved words such as “if-elif-else”, “try-except-finally”, “from-import” and “while-else” selected to obtain grouped or paired together, where at least one reserved word is remained and others replaced with suitable reserved words.
- The reserved words such as “class”, “def”, “break”, “return” and “pass” replaced with suitable reserved words to obtain error elements. For example, “return” replaced with “pass”.
- The pairs of control symbols such as “{,}” (curly bracket), “(,)” (bracket) and “[,]” (square bracket) are paired, where at least one control symbol is remained and others are placed with suitable control symbols.
- The same variables such as list name, array name, dictionary name and tuple name are group together and others replaced with suitable variables by using fuzzy method.
- The same library names selected to obtain grouped together, where at least one element is remained and others replaced with suitable library names.
- The name of a library function replaced with the same name where only the upper/lower case reversed.
- Logical errors: logical errors are created to test the operators in python code such as arithmetic operators, assignment

operators, comparison operators and logical operators by the followings:

- Arithmetic operators such as "*, +, -, /, **, //, %" are replaced with suitable operators to obtain error words. For example, "*" operator is replaced with "/" operator.
- Assignment operators such as -=, +=, *=, /= are replaced with different such operators to obtain error words. For example, "-=" operator is replaced with "+=" operator.
- Comparison operators such as >=, <=, >, <, "! =" and == are replaced with suitable such operators to obtain error words. For example, ">=" operator is replaced with "<=" operator.
- The logical operators such as and, or, not, are replaced with suitable such operators to obtain error words. For example, 'or' operator replaced with 'and' operator.
- In the conditional expression, the relational operators selected to obtain grouped together, where at least one operator is remained and others replaced with different operators.

Method 1: Error Debugging Method

```
INPUT: read word (word)
OUTPUT: error_word
Variable: String reserved_word, operator_word, control_word,
variable_word
Set debuggingmethod(word)
reserved_word=changereservedword(word)
if reserved_word==word then
operator_word=changeoperatorword(word)
    if operator_word==word then
      control_word=changecontrolword(word)
      if control_word==word then
                variable_word=changevariableword(word)
                error_word=variable_word
                   else
                error_word=control_word
   else
      error_word=operator_word
else
   error_word=reserved_word
endif
return error_word
```

4.2 Example of Constraint Graph

Figure 2 illustrates the constraint graph for python code of "RandomNum" class. Firstly, the elements paired together by pair selection category, where at least one element is remained and others are replaced with suitable incorrect elements. Then, the elements grouped together by grouped selection category, where at least one element is remained and others replaced with suitable incorrect elements.

In figure 2, the dotted line shows the elements grouped together and the straight line shows the elements paired together by the group selection category or pair selection category. In figure 2, there are three identifier groups such as "method", "random" and "RandomNum", two variable groups such as "n", "sum" and two control groups such as "(,)" and ":". In figure 2, there are eight pairs are continuous pair in graph. For example, "import" and "random" is continuous pair in graph.

4.2.1 Selecting python code for RandomNum. In this paper, we select the python code of "RandomNum" class.

```
1: import random
2: class RandomNum:
3:  def method(self,n):
4:  sum$=$o
5:  for i in range(n):
6:   sum+$=$random.randrange(1,100)
7:  print(sum)
8: obj$=$RandomNum()
9: obj.method(10)
```

5 EVALUATION

In this section, the evaluation of error debugging problems in python code described.

5.1 Generated error elements in problem code

Error debugging problem code of "RandomNum" class shows the number of lines (LOC) is nine lines in the code. There are 20 elements in the code and 7 incorrect elements in the code according to CB=1 and BG=1. In python code for "RandomNum" class, "import" of line 1 in the "import random" pair replaced with "Import" incorrect reserved element. In the "class Random" pair, the "class" of line 2 replaced with "Class" incorrect reserved element. In the "def method" pair of line 3, the "def" replaced with "Def" incorrect reserved element. In "for i in range" pair, the "range" in line 5 replaced with "rance" incorrect identifier element. The group of ":" in line 2, 3 and 5, the ":' in line 5 is replaced with ";" incorrect control element.

5.1.1 Generating error debugging problem code for RandomNum. In this paper, we describe the error debugging problem code for "RandomNum" class.

```
1:  Import random
2:  Class RandomNum:
3:     Def method {self,n} :
4:         sum$=$0
5:         for i in rance (n) ;
6:             sum+$=$random.randrange(1,100)
7:             print(sum)
8:  obj$=$RandomNum()
9:  obj.method(10)
```

5.2 Evaluation of error elements in problems code

In table 1, the 100 error debugging problems generate to calculate the evaluation of error elements in each code. The blank element selection algorithm used BG=1 (blank gap number) and CB=1 (Continuous blank number) is default and error debugging method is used to generate the incorrect elements in code. In error debugging method, the error elements automatically generated in each code depending on reserved words, operator words, control words and variable words.

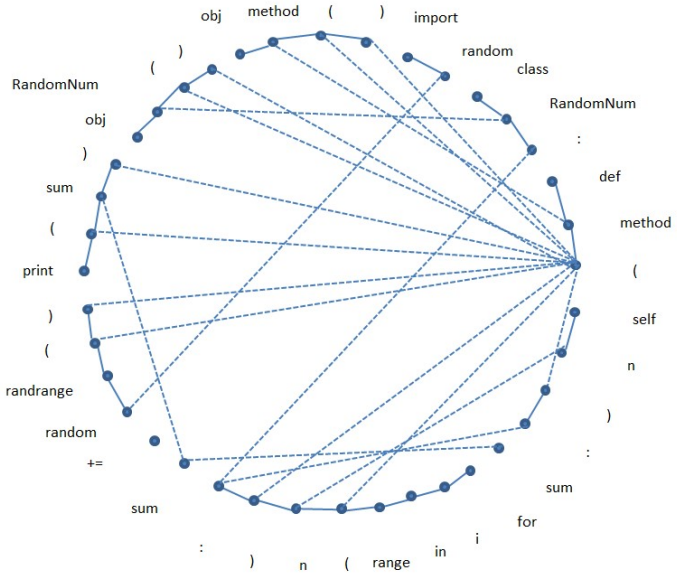

Figure 2: Constraint graph for RandomNum

Table 1: Generated error elements in 100 code for evaluation

No	Category	#of generated Problems	Total # of elements	Total # of replaced elements	#of reserved words	#of operator words	#of Control words	#of variable words
1	Operator	10	236	40	6	3	2	29
2	String	10	583	83	24	1	25	33
3	Decision	10	827	150	39	2	57	52
4	Dictionary	10	746	124	37	0	62	25
5	list	10	802	123	26	0	70	30
6	Math, date time	10	676	102	30	0	40	32
7	Class and object	10	597	88	21	1	34	32
8	File	10	610	90	21	0	36	33
9	Try, catch and exception	10	675	107	29	0	43	33
10	Tuple, function	10	677	110	22	0	56	28
Total # of 100 problems		100	6429	1017	255	7	425	327
Average # of replaced elements				15.82%	4%	0.1%	6.6%	5.1%

The average number of replaced elements in 100 problems code is 15.82%. The average number of reserved words is 4% to generate incorrect words. The average number of operator words is 0.1%, the average number of control words is 6.6% and the average number of variable words is 5.1%. The rate of replaced elements in operator words is less than other words because the problem code used operator words are less than other words. The rate of replaced elements in control words is more than other words.

5.3 Evaluation of solution performance of students

In COVID 19, the students are currently studying the python programming from online training. We asked 5 students to answer the 10 problems. In table 2, the average correct rate of 5 students is 82%. The standard deviation of 5 student's correct answer elements is 6.099.

Student S1 has 96% highest correct rates. Student S3 has 74% correct rate and has 126 submissions. Students S2, S3 and S4 have

Table 2: Solution performance of each student

ID	Total # of replaced elements	Total # of correct answer elements	# of submissions	Correct rate (%)
S1	72	69	13	96%
S2	72	57	147	79%
S3	72	53	126	74%
S4	72	56	167	78%
S5	72	59	98	82%
Average correct rate				82%
Standard Deviation (SD)				6.099

less than 80% correct rates. Then, the S3 student has lower correct rate than other students and the smaller number of submission than S2 and S4. The S3 student must study the python programming. Generally, the larger number of submissions contributes the higher correct rate for S3 student. If the student has lowest correct rate, this student must study the python programming and python grammar.

6 CONCLUSION

In this paper, we proposed the error debugging problems of python programming learning assistant system. We generated the error debugging problem by using error debugging method extended the blank element selection algorithm. For evaluation, 100 problems code generated to analysis the number of replaced elements in the problem code. The replaced elements are definitely elements by using python dictionary and fuzzy logic. Then, 10 problems assign 5 students to analysis the problem correctness and calculate the student programming skill. In the future, we must ask the many students and to analysis the answer of these students. Finally, we must upload the 100 error debugging problems in PYPLAS system.

REFERENCES

[1] Akshansh Sharma, Firoj Khan, Deepak Sharma, Dr. Sunil Gupta, 2020. “Program: The Programming Language of Future”, International Journal of Innovative Research in Technology (IJIRT-149340), volume 6 issue 12, ISSN: 2349-6002, may 2020.

[2] H. W. Hnin and K. K. Zaw, 2020. "Element Fill-in-Blank Problems in Python Programming Learning Assistant System," 2020 International Conference on Advanced Information Technologies (ICAIT), 2020, pp. 88-93, doi: 10.1109/ICAIT51105.2020.9261778.

[3] H.P.Langtangen, 2014. “Debugging in Python”, Center for Biomedical Computing, Simula Research Laboratory, Department of Informatics, University of Oslo, pp.1-3, may 8 2014.

[4] Qingkai kong, Timmy Siauw, Alexandre Bayen, 2020. “Python Programming and Numerical Methods-A Guide for Engineers and Scientists”, 2020.

[5] LR parser, http://www.javapoint.com/lr-parser.

[6] Quartz25, D. Rass, Jesdisciple, H. Rost, L. D, 2021. “Python Programming”, “https://en.wikibooks.org/wiki/Python Programming”, May,23,2021.

[7] G. V. Rossum and python development team, 2021. “Python Tutorial Release 3.9.6”, python software foundation, pp.57-63, July, 20, 2021.

[8] Nobuo Funabiki, Khin Khin Zaw, Nobuya Ishihara, and Wen-Chung Kao, 2017. “A Graph-based Blank Element Selection Algorithm for Fill-in-Blank Problems in Java Programming Learning Assistant System”, IAENG International Journal of Computer Science, 44:2, IJCS_44_2_14, 24.May.2017.

[9] Tana and Nobuo Funabiki, 2015. “A proposal of graph-based blank element selection algorithm for java programming learning with fill-in-blank problems”, Preceedings of the international multiconference of engineers and computer scientists 2015 Vol I, IMECS 2015, March 18-20,2015, Hong Kong.

[10] Nobuo Funabiki, Member, IEEE, Shin Sasaki, Tana, and Wen-Chung Kao, Senior Member, IEEE, 2015. “An operator fill-in-blank problem for algorithm understanding in java programming learning assistant system”, IEEE 4th global conference on consumer electronic (GCCE), 2015.

[11] N. Funabiki, Tana, N. Ishihara, and W.C.Kao, 2016. “Analysis of fill-in-blank problem solution results in java programming course”, to appear in Proc.GCCE 2016, Oct.2016.

[12] Hsu Wai Hnin, Khin Khin Zaw, Nobuo funabiki, 2018. “Error debugging problem in java programming learning assistant system”, ICSE, 9th international conference on science and engineering, December 8, 2018.

[13] Khin Khin Zaw, Nobuo Funabiki, and Minoru kuribayashi, 2016. “Extensions of blank element selection algorithm for java programming learning assistant system”, IEICE Technical Report, ET2016-17.