Which programming language should students learn first?

A comparison of Java and Python

Chieh-An Lo
National Taiwan Normal University
No. 162, Sec. 1, Heping E. Rd.,
Taipei, Taiwan
60108012E@ntnu.edu.tw

Yu-Tzu Lin*
National Taiwan Normal University
No. 162, Sec. 1, Heping E. Rd.,
Taipei, Taiwan
linyt@ntnu.edu.tw

Cheng-Chih Wu
National Taiwan Normal University
No. 162, Sec. 1, Heping E. Rd.,
Taipei, Taiwan
chihwu@ntnu.edu.tw

***Abstract*—Programming is a fundamental ability for Computer Science majors. Most programming lectures use traditional languages, such as C, C++, or Java, as the first learning language for novices, which are effective for designing real applications and therefore are popular in industry. However, the complex syntax of these languages is challenging for novices, which becomes an obstacle to novices' learning.**

Python has simpler syntax and high-level data structures to enable succinct programs. The multiple paradigms of Python also provide learners opportunities to learn various features of programming languages. Therefore, Python gradually becomes a new option of the first language for learning.

This study investigates the features of the first learning programming language by comparing Python with the popular learning language Java. A questionnaire study will be conducted on both instructors and students to understand how the features of languages affect students' learning from the aspects of program translation and execution (interpretation vs. compilation), variable declaration (dynamic vs. static), the syntax of control structures, and the object-oriented syntaxes. For each aspect of questionnaire, questions are designed to understand whether and how the language features in this aspect influence program comprehension (including the occurrence of misconceptions or errors, and cognitive load) and program implementation. The research results will give instructors suggestions for selecting suitable learning languages in their lectures and designing adaptive instructional strategies/materials based on the features of the selected language.

Keywords—programming instruction; programming learning; first programming languag

I. SIGNIFICANCE

Programming is challenging for most novice learners, which involves many higher level thinking skills and complex concepts Novices usually cannot comprehend and apply programming knowledge properly. They learn programming from syntactic details and write pieces of codes based on their partial understanding of programming knowledge, but do not know how to combine them into a whole program [1]. Being stuck in syntactic details, novices tend to write programs in a line-by-line manner rather than considering the program as a meaningful structure [2]. Besides, they tend to make some common syntactic or semantic mistakes [3] and have misconceptions [4] which might due to their difficulties in learning the programming languages.

Although Java is a popular learning language, novice learners usually face challenges in learning the complex and abstract programming knowledge, e.g., classes, methods, types, and the assess level [5]. To reduce learning difficulties and failures, the criteria for choosing the first programming language to learn should include: (1) Simple input/output statements, (2) Readable and consistent syntax, (3) Clear syntax, and (4) Orthogonality [6]. Contrary to the complex syntax of Java, Python emphasizes code readability and simplicity. The succinct syntax of Python allows programmers to express concepts in fewer lines of code. Therefore, more and more researchers argued that Python should be taught as the first programming language [6]. To investigate which programming language should be taught first to improve novices' learning and its features, this research is aimed to compare Java with Python in terms of students' comprehension and perception, and also instructors' opinions.

II. RESEARCH MEHOD

A. Research process

The research process is illustrated in Figure 1. The literatures of programming instruction were at first survey to build the theoretical foundation of this study. Based on the survey results, the first version of the questionnaire was designed. Four experts were then reviewed the questionnaire and give comments to it. The questionnaire was then revised based on the experts' comments. The second version of questionnaire will be piloted in an undergraduate class of a university for the reliability analysis. We will revise the questionnaire again based on the reliability analysis results of the pilot test. Then, the final version of the questionnaire will be conducted on the undergraduate students and the instructors who teach programming in universities. Finally, a statistical analysis will be conducted to compare Java with Python and obtain conclusions about which one should be taught as the first language.

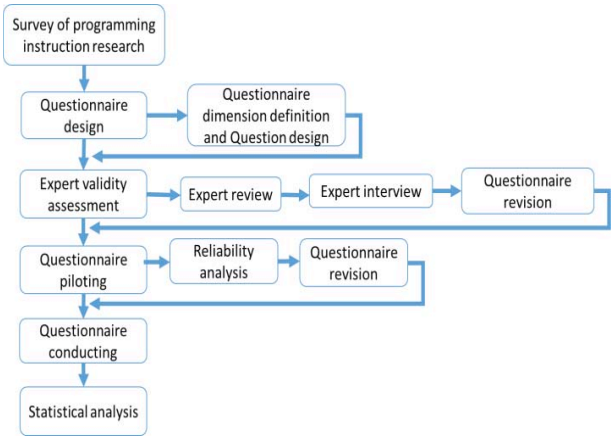

Figure 1. Research process.

978-1-4799-9967-5/15 $31.00 © 2015 IEEE
DOI 10.1109/LaTiCE.2015.15

IEEE computer society

B. Participants

There are four types of subjects: (1) undergraduate students in the Department of Computer Science in the university who have learnt Java, (2) Python, or (3) both or at least half year, and (4) instructors who are familiar with both Java and Python.

C. Questionnaire

The 5-point Likert scale questions in the questionnaire involve several major programming concepts, as listed in Table I.

TABLE I. Programming concepts involved in the questionnaire.

Topic	Subtopic
Data type	Various data types
	Ranges of data types
Variable	Dynamic and static variables
	Ranges of variables
Expressions and Assignment	Relational and Boolean Expressions
	Assignment statement
Control structure	If statement
Loop structure	For statement
	While statement
	Nested for loop
Function	Definition of functions
	Function parameters
Object and class	Definition of objects and classes
	Constructor
	Object creation

There are three types of questionnaires: (1) the questionnaire for students only have learnt Java or Python, (2) the questionnaire for students have learnt both Java and Python, and (3) the questionnaire for the instructors, in which (1) is to understand students' perceptions about learning and using Java or Python, and (2) and (3) are to ask the students who have learnt both Java and Python, and instructors who are familiar with these two languages to compare the characteristics of Java and Python.

The questionnaire for (1) includes three dimensions to understand the students' comprehension levels, programming skills, and perception of learning the programming topics by using Java or Python:

1. Programming concept comprehension

In this dimension, students' comprehension and perceptions about data types, expressions and assignment, the control structure, the loop structure, function, and objects is examined.

2. Program implementation

This dimension is to understand whether students can successfully write programs using suitable data types, structures, functions, or objects.

3. Cognitive load

Except students' comprehension levels and implementation skills, we also intend to examine students' cognitive load in learning the programming concepts and skills.

The questionnaires for (2) and (3) have four dimensions to investigate students' perceptions and instructors' opinions about the differences between Java and Python:

1. Programming concept comprehension

Students have to answer about their comprehension levels when learning different programming languages. Instructors have to compare the students' comprehension situations for different languages.

2. Program implementation

Students have to answer about their perceptions and difficulties while writing the programs. Instructors have to compare the students' difficulties in writing programs with different languages.

3. Misconception

Students have to explain their understanding about some concepts to understand whether they have misconceptions. Instructors have to compare students' misconceptions in learning and writing programs with different languages. The questionnaire questions of this dimension are based on students' common misconceptions. Some examples of the misconceptions [4] are presented in Table II.

TABLE II. Programming misconceptions.

Topic	Misconception
Loop	· The students cannot measure the execution time of a loop. · The students do not know the initial value and the increment statement should be assigned in the conditional part of the for loop.
Function	· The student tries to start the function call before they have evaluated a variable. · The student creates parameter variables in the caller's frame, not in the callee's. · The student thinks that the functions are executed immediately when a program starts.
Object	· The student creates a new object rather than copying a reference. · A new object should be created, but the student makes a copy of a reference to an existing object. · Instead of creating a new instance variable for the object, the student creates a new local variable.

REFERENCE

[1] L.E. Winslow, "Programming pedagogy – A psychological overview", SIGCSE Bulletin, vol. 28, pp. 17–22, 1996.

[2] J.C. Spohrer, and, E. Soloway, "Novice mistakes: Are the folk wisdoms correct?", Communications of the ACM, vol. 29, no. 7, pp. 624-632, 1989.

[3] L. Mannila, M. Peltomäki, and T. Salakoski, "What about a simple language? Analyzing the difficulties in learning to program", Computer Science Education, vol. 16, no. 3, pp. 211–227, 2006.

[4] T. Sirkiä, & J. Sorva,"Exploring programming misconceptions: an analysis of student mistakes in visual program simulation exercises", Proceedings of the 12th Koli Calling International Conference on Computing Education Research, pp. 19–28, 2012.

[5] A. Radenski, "Python first: A lab-based digital introduction to computer science", Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, pp. 197-201, 2006.

[6] M. Vujoševic-Janicic, and D. Tošić, "The role of programming paradigms in the first programming courses", The Teaching of Mathematics, vol. XI, no. 2, pp. 63–83, 2008.

[7] G. Van Rossum, "Computer programming for everyone", Retrieved from http://www.python.org/doc/essays/cp4e.html, 1999