Teaching Programming to Non-CS Major Students: Experiments with Storymaking Approach

HyungShin Kim, Hye Sun Jang, Hwan Cheol Lee, Dae Yong Kwon,
Yong Chul Yeum, Seung Wook Yoo, Hyun Chul Kim, WonGyu Lee
Computer Science Education, Korea University
Anam Dong, SungBuk Gu, Seoul 136-701, Seoul Korea
e-mail: {hyungshin, micro38, wink101, dykwon, yycok, swyoo, hkim, lee}@comedu.korea.ac.kr

Abstract

This paper reports on an experiment teaching educational programming language, Dolittle, to non-CS major students with storymaking approach. The Dolittle program offers an environment that enables students to code in Korean. This makes the students easily represent their ideas in the computer. In the experiment, students learned programming language, Dolittle, through creating their own stories. The preliminary results conclude that introducing the Dolittle program to non-CS major students with storymaking approaches has tremendous potential for encouraging students' motivation and furthermore developing students' learning achievement in the future.

1. Introduction

Many researches show that computer programming languages help to develop students' problem solving ability and analytical skills [7, 11, 15]. However, the opportunity for doing this seems limited to the students who are in engineering and science majors, because they will have several required computer programming courses in either first or second school year. We argue that social science and liberal arts major students also need programming competences because it would enhance and develop creative thinking, problem solving, and communication skills to handle changing situation for 21st century global knowledge economy [10]. A recent Korean Ministry of Education white paper, Adapting Education to the Information Age, has mentioned the importance of information and communication technologies in K-16 and life long learning [10]. Parallel to this call, Korea University offers several elective courses from science and engineering departments to non-major students as liberal arts courses. Since programming is not their primary interest, it is necessary to teach programming language to non-Computer Science (CS) major students in a new and untraditional way [6, 9, 12]. In this paper, we first present Dolittle supported by Korean language with 1:N multi-reserved words characteristics and then discuss the results of the experiment with non-CS major students with Dolittle. Lastly, our conclusion and future directions will be provided for further study.

2. Dolittle Program Supported by Korean Language

The object-oriented educational programming language, Dolittle, is originally developed by Dr. Susumu Kanemune for the purpose of

H.S. Kim, H.S. Jang, H.C. Lee, D.Y. Kwon, Y.C. Yeum, S.W. Yoo, H.C. Kim, W.G. Lee
Department of Computer Science Education, Korea University.

programming education in Japan [13, 14]. Dolittle program used for this experiment enables students to write coding in Korean language. This has developed at the Educational Programming Language (EPL) group at Korea University. The EPL group implemented not only simple Korean language localization (1:1 language exchange), but they also developed a set of 1:N multi-reserved words [5]. There are two reasons for developing 1:N multi-reserved words [5]. First, it can support a Korean language characteristic which has various endings of a word. Korean is a highly inflected language, especially the ending of a word based on the age group. Second, 1:N multi-reserved words are designed to consider learner's differences, preferences, and applicable fields based on their various backgrounds. Table 1 shows a sample set of 1:N reserved words.

Table 1. Sample Set of 1:N Multi-reserved Words

English Command	Current Command	Changes and Add → 1:N set
Turtle	거북	거북/거북이
Timer	타이머	타이머/시계
moveto	이동	이동/이동하고/이동하기/이동하다/이동한다 움직이고/움직이기/움직이다/움직인다 옮기기/옮기고/옮기며/옮긴다
position	위치한다	위치/위치한다/위치하다/위치하고/위치하기
hide	감추다	감추다/감춘다/감추고/투명거북
show	보이다	보이다/보인다/보이고
paint	색칠하다	색칠/색칠하다/색칠한다/색칠하고/색칠하기 칠/칠한다/칠하다/칠하고/칠하기
scale	확대한다	확대/확대한다/확대하다/확대하고/확대하기
closepath	닫기	닫기/닫다/닫는다/닫고 처음으로가기/처음으로간다/처음으로가고/ 처음으로가다/시작으로가다 시작으로가기/시작으로간다/시작으로가고
looks	변신	변신/변신한다/변신하다/변신하고 변형한다/변형하다/변형/변형하고
repeat	반복한다	반복/반복한다/반복하다/반복하고/반복하기
then	이라면	이라면/이면/만족하면
xpos?	x 위치는?	X 위치는?/세로위치는?/x 좌표는?
ypos?	y 위치는?	y 위치는?/가로위치는?/y 좌표는?

The previous experiment showed that programming with this 1:N multi-reserved words helped learners to concentrate on solving problems rather than sticking to the reserved words' errors [5]. For this study, we also used the Dolittle program which support 1:N multi-reserved words because this helps students to program natural language style programming beyond common programming languages' rigid characteristics [1, 16].

3. Experimental Design

3.1 Purpose

The way to learn computer programming language has dominantly been based on manual-based teaching [2, 3, 4, 8, 9]. Several studies show the way to teach computer programming should be changed with a different way [6, 9, 12]. One of the complaints about introductory computing courses is that programming is too abstract and not anchored in a relevant context [9]. Therefore, students easily loose their motivation to learn programming in the future. In this study, we designed to teach Dolittle Program with storymaking approach. We believe that this approach will provide more meaningful learning environments for non-CS major students. Because programming is not a primary interest of the students, it is essential to design the experiment with open-ended topics such as making their own stories through programming process [12].

3.2 Participants

58 students who have taken *the World Represented with Data* course at Korea University participated in this study. The students consisted of 39 male students and 19 female students. 85 % of the participants (49 students out of 58) were freshman and sophomore. Figure 1 shows the grade distribution of the participants. Participants consisted of non-CS major students such as social sciences, business, education, biology, sciences, humanities, and literatures. All of the participants were intermediately experienced computer users, but only 14 students responded they have an experience to learn programming before. The major distribution of the participants is shown in the Figure 2.

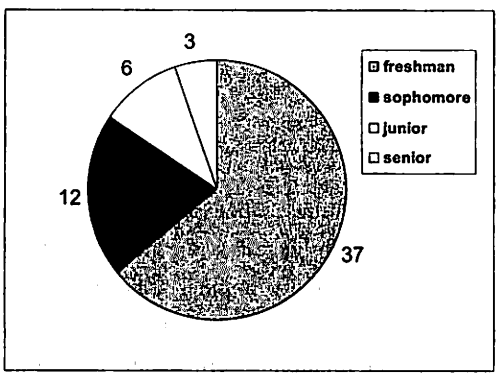

Figure 1. Participant's Grade Distribution

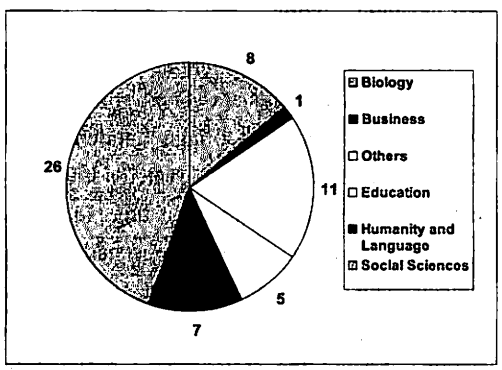

Figure 2. Participants' breakdown by major

3.3 Procedure

Our methodology included an initial survey on the participants' background of Computer Science and computer programming languages, three two-hour experimental lessons of Dolittle program, and the final survey which is consisted of the same questions with the initial survey in order to see the changes of participant's perception. Both initial and final survey questionnaires used in the experiment was adopted from *the media computation Course for non-majors* [8]. In the final survey, we also asked the participants what they think about *learning computer programming language through storymaking* approach in terms of motivation and achievement. To help non-CS major students to learn programming with meaningful contexts, we designed and developed an innovative instructional method called learning programming by making stories. Table 2 shows the four steps in terms of the goals of storymaking, learning objectives, and the tasks of the Dolittle program.

Table 2. Four Steps for Dolittle Learning

	Storymaking Goal	Dolittle Learning Goal and Task
STEP 1	Write a story students want to create or choose a topic from several stories instructor offered	How to use basic syntax of the Dolittle program O Run Dolittle
STEP 2	Compose a scene of the story to be animated (e.g. define characters, create background)	How to (1) create object, (2) use method of a object (function), (3) use the Turtle object - forward, turn, move, looks, penup, pendown, color, and (4) create new object by using turtle object - makefigure O Create characters to be hero or heroine O Create backgrounds such as tree, river, sun etc.
STEP 3	Define each character's action and write code for the characters'	How to (1) use a simple repeat statement, (2) create a method, (3) use Timer

	movement	Object, (4) use GUI Object - textfield, button ○ Create backgrounds using repeat statement ○ Create methods of character's action using Timer Object ○ Create a scene title using GUI object
STEP 4	Complete the story by making the order of each character's action	How to use (1) repeat statement, (2) conditional statement, (3) communication among objects ○ Arrange the characters' action based on the defined functions ○ Use repeat and condition statement ○ Create a Play button using GUI Object

4. Results and Discussion

The results and analysis have been done and some discussions have been made throughout the participant's written responses from the surveys.

4.1 Comparison of Initial and Final Survey Results

This section describes the comparison of the survey results at the beginning and the end of the experiment. First, Table 3 shows the responses from the question: *What is Computer Science to you?* As shown Table 3, the number of students who think CS is programming is increased. The majority of the students think CS is use computer for some purposes in both initial and final surveys.

Table 3. Definition of CS at the beginning and the end of the experiment (raw count)

	Initial Survey	Final Survey
Don't know	7	3
How computer works	18	12
Programming	4	13
Use computer for some	25	28
Scary and tough	2	1
Required elective course	1	
Web and the Internet	1	1

As shown Table 4, 31 students think Computer Science is about problem solving. After learning the Dolittle program, there is a slight increase of positive thinking and decrease of negative thinking on CS.

Table 4. More definitions of CS at the beginning and the end of the experiment

What is CS to you?	Problem Solving	Logic	Positive	Negative
Initial survey responses	31	19	2	6
Final survey responses	31	20	3	4

Six-hour Dolittle lessons would not make significant influence to the students' conception of Computer Science and programming. However, we believe that this will be an important cornerstone for further longitudinal studies.

4.2 Dolittle Lessons

This section shows a sample work from the classroom assignments at the end of the lessons. A freshman female college student from communication department created a love story between a bear and a raccoon dog. First, she created a bear, a raccoon dog, and several objects which she wanted to be in her story. Then she programmed the objects to have actions. Lastly, the story was completed by defined functions. Figure 3, Figure 4, Figure 5, and Figure 6 shows the stories and its coding pages in turn.

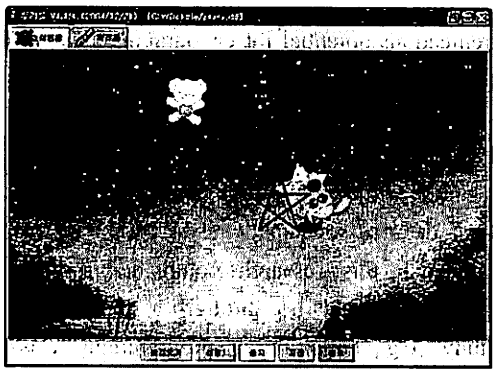

Figure 3. Student's sample work in the execution window

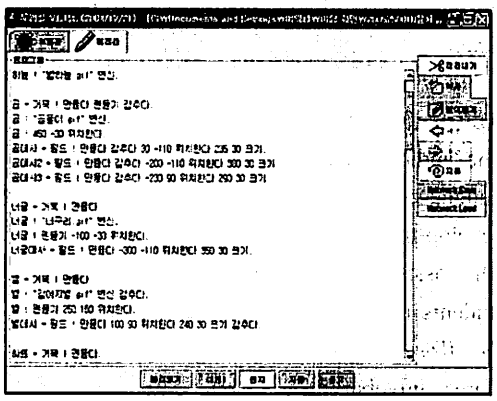

Figure 4. Student's sample work in code window

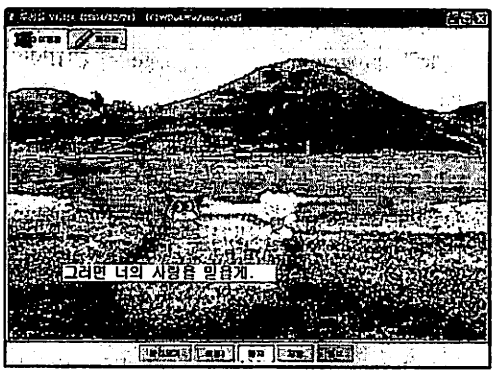

Figure 5. Student's sample work in execution window

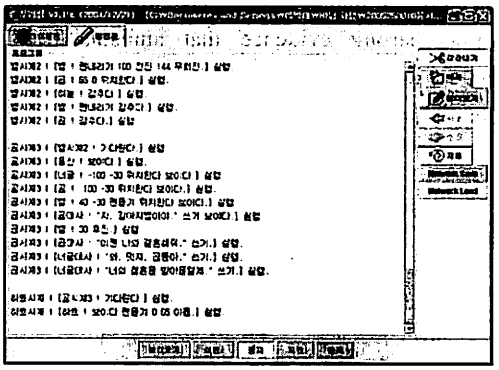

Figure 6. Student's sample work in code window

We also asked them whether the Dolittle program is easy to learn comparing to other computer programming languages. As shown Table 5, 69% of the students responded that Dolittle is easy to learn.

Table 5. Final Survey Responses

	Easy	Difficult	No response
Raw count	40	15	3
Percentage	69%	25%	5%

Most students who answered positively explained that the reason they felt easy for learning Dolittle was because they could write the program in Korean. Unlike learning other programming languages, Korean coding helps students to remember the reserved words and furthermore to feel much familiarity to use. The set of 1:N multi reserved words also offers students the programming coding environments to express the students' idea with flexibility.

4.3 Impact of Storymaking approach

This section includes how students think about a new way of learning the Dolittle program. We designed and developed the instructional method called *learning programming making stories*. Students learned the Dolittle program by making their own stories. Figure 7 shows the 90% (47 students) out of the students who answered (52 students) on this question thinks learning programming by storymaking was

meaningful and helped them to code. This shows strong evidence that students learn better when they are in a learner-centered contextualized environment.

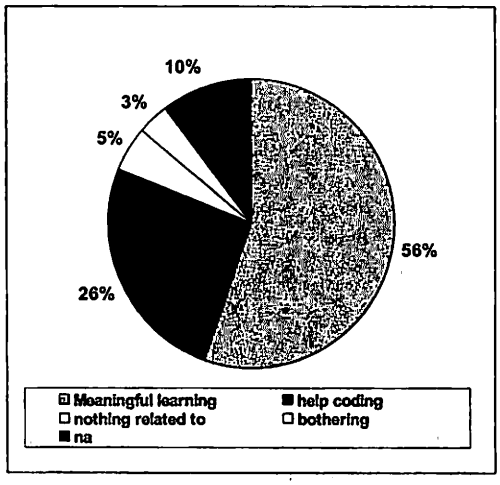

Figure 7. . Student's responses on storymaking instruction

5. Conclusion

This paper reports on an experiment teaching educational programming language, Dolittle, to non-CS students by making story. The results show that this new way of learning programming is very effective to encourage students who are already de-motivated at the outset. Furthermore, the Dolittle program environment that enables students to code in Korean makes the students easily represent their ideas in the computer. Therefore, the rigidity, one of the common popular difficulties of leaning to program [3, 15, 16] is solved when students learn the Dolittle program. However, the limitation of this study, such as its small subject pool, prevented us from getting any clear distinctions between the initial and final surveys. For a more general standpoint, we believe that our very preliminary results support the conclusion that introducing educational computer programming language to non-CS major students with storymaking approaches has tremendous potential for encouraging students' motivation and furthermore developing students' learning achievement in the future. Lastly, for further study, a larger sample size should be utilized to provide a good foundation for a detailed statistical analysis. Another way in which this research could be fruitfully extended would be through a longitudinal study (e.g., following a programming class through an entire school year), in order to measure how educational programming language embedded into an entire year's curriculum contributes to student learning over time.

Acknowledgements

The authors thank Educational Programming Language Group members who helped to develop the Dolittle Storymaking course, those who assisted with the evaluatizon and the students in the experimental course who took *the World Represented with Data* class at Korea University.

Reference

1) A. Bruckman and E. Edwards. Should we leverage natural-language knowledge? An analysis of user error in a natural-language style programming language. CHI'99 Pittsburgh PA USA, 1999.
2) A. Quinn. An interrogative approach to novice programming. Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02), 2002.
3) D.B. Palumbo. Programming language/problem-solving research: A review of relevant issues. Review of Educational Research, Vol. 60, no. 1, pp 65-89, 1990.
4) E. Soloway and J. Spohrer. Studying the novice programmer. Hillsdale, NJ: Lawrence Eribaum Associates, 1989.

5) H. Choe, D. Kwon, H. Kim, Y. Yeum, S. Yoo, and W. Lee. Multi-reserved words supporting system for object-oriented educational programming language "Dolittle" The Journal of Korean Association of Computer Education Vol 8, no 2, 2005.
6) J. Bennedsen. Teaching Java programming to media students with a liberal arts background. Proceedings of the seventh Java & the Internet in the Computing Curriculum Conference, 2003.
7) J.D. Bransford, A. L. Brown, and R.R. Cocking, editors. How People Learn: Brain, Mind, Experience, and School. National Academy Press, Washington, D.D., 2000.
8) M. Guzdial. A media computation course for non-majors. In Proceedings of the Innovation and Technology in Computer Science Education Conference. ACM. Canterbury, UK, 2001.
9) M. Guzdial and E. Soloway. Teaching the Nintendo generation to program. Communications of the ACM, Volume 45, Issue 4, April 2002.
10) M. J. Jacobson, Y., Kim, J. Lee, H. Kim, and S. Kwon. Learning sciences principles for advanced e-learning systems: Implications for computer-assisted language learning. Multimedia-Assisted Language Learning, *8*(1)., 2005.
11) M. Resnick,. (1995). New paradigms for computing, new paradigms for thinking. In A. diSessa, Hoyles, C., & Noss, R. (Eds.), Computers and Exploratory Learning (pp. 31-43). New York: Springer-Verlag.
12) P. B. Andersen, J. Bennedsen, S. Brandorf, M. Caspersen, and J. Mosegaard. Teaching programming to liberal arts students – a narrative media approach. Proceedings of the eighth Annual Conference on Innovation and Technology in Computer Science Education. Thessaloniki, Greece, 2003.
13) S. Kanemune. Dolittle Programming language. http://www.logob.com/dolittle.
14) S. Kanemune, T. NAkatani, and R. Mitarai. Dolittle – Experiences in Teaching Programming at K-12 Schools.
15) S. Papert. Mindstorms: children, computers, and powerful ideas. Basic Books. 1980.
16) T. Wright and A. Cockburn, Writing, Reading, Watching: A task-based analysis and review of learners' programming environments. Proceeding of International workshop on Advanced Learning Technologies, 2000.