THE ROLE OF DOMAIN FAMILIARITY TO LEARN INTRODUCTORY PROGRAMMING

Polat Sendurur and Emine Sendurur
Ondokuz Mayis University
Samsun, Turkey

ABSTRACT

Learning introductory programming concepts can be challenging for novice learners. Providing tasks related to the main field of study might serve as a facilitator during the learning process. This study aims to observe how a group of university students, who enrolled in a programming class for the first time, perform programming tasks by domain familiarity—110 university students involved in this mixed study. One group performed the domain familiar task while the other one performed the unfamiliar one. In the second activity, the groups were reversed. The quantitative analysis indicated that domain familiarity does not affect programming performance.

KEYWORDS

Programming Education, Domain Familiarity, Learning How to Code

1. INTRODUCTION

Changing paradigms of education have brought about new roles for learners. Today's learners are expected to manage their learning (Reigeluth, 2016), so such higher-order thinking skills as problem-solving might be required. 21st-century skill set combines many skills combining problem-solving and information science. Computational thinking, new media literacy, and virtual collaboration can be listed among these skills (Curzon et al., 2009; Mohaghegh & McCauley, 2016). To improve these skills, interdisciplinary approaches might be needed. Based on this motivation, the current study aims to explore the effects of domain familiarity on programming performance, referring to a group of computational thinking skills, including formulating problems, organizing and analyzing data, and abstraction (ISTE, 2011).

The practice is generally based on domain-specific approaches in traditional instructional settings. On the other hand, such disciplines as programming education might borrow various practices of different disciplines. For example, the creation and testing of an algorithm might apply in many domains ranging from biology to language classes. Programming pedagogy is evolving, so integrating known instructional approaches might contribute to its development. Moving beyond the boundaries of the taught domain is an approach triggering creativity, motivation, and learning (Baer, 1998; Bong, 2004; Ding et al., 2013; Shen et al., 2008). It is known that creativity is essential in the development of algorithmic thinking skills (Curzon et al., 2009; Grover, 2013).

Moreover, creativity is also related to domain familiarity. According to Mishra et al. (2012), breaking the boundaries of domains can contribute to creativity. On the other hand, the lack of domain knowledge might hinder creativity (Sendurur et al., 2018).

In summary, the literature about domain familiarity still needs to be more conclusive, and no studies investigate domain familiarity in the programming education context. This study assigned domain-familiar or domain-unfamiliar tasks to students without computer programming experience. Unlike previous studies, this study has a different perspective on utilizing domain familiarity. In other words, domain specificity was used as a content delivery method instead of the content itself, which might shed light on programming education pedagogy. This study aims to observe how a group of university students perform programming tasks by domain familiarity with the help of the following research question:

1. Do domain-familiar tasks affect novice undergraduate students' programming performances?

2. METHOD

A mix of qualitative and quantitative techniques was planned to utilize in the research. However, only quantitative results are presented as the work is in progress. 110 sophomore university students from the department of Math Education participated in the study. They attended the programming course, which is a must-course aiming to teach introductory programming topics. The participants do not have any programming experience. They were assigned randomly to one of the two groups. Each group completed two different activities; one was domain-specific, and the other was non-domain-specific. The following table summarizes the activities, their covered programming topics, and the group-activity matching. Participants completed four different activities, as shown in Table 1.

Table 1. Programming Activity-Group Matches

Group	Domain Familiarity	Activity Description	Covered Programming Topic
Group1	Familiar	Odd/even numbers	Variables, conditionals
Group2	Unfamiliar	Tools with/without electricity	Variables, conditionals
Group1	Unfamiliar	Vowels/consonants	Variables, conditionals, loops, random numbers
Group2	Familiar	Symmetric/non-symmetric geometric objects	Variables, conditionals, loops, random numbers

Multiple data sources were utilized multiple to observe the performance change across domains. Students were assigned to complete tasks within the Scratch environment, and their final products were evaluated according to task-specific rubrics. The students were also required to explain the code block's function and contribution to the program. Independent samples t-test conducted on groups' rubric scores and content analysis was applied to their explanations about code block's function and contribution. However, this short paper reports only the quantitative results as the content analysis is still in progress. The quantitative data were analyzed in SPSS 21.0. To understand the differences between groups, two independent samples t-tests were run on the scores gained through activities 1 and 2.

3. FINDINGS

The independent samples t-test conducted on the first activity scores showed that domain familiarity has no significant effect on programming performance, t (127) =1.12, p=.26. The group structure was reversed in the second activity. Participants who completed the first activity in the domain familiar format participated in the second activity in the non-domain familiar form. The other group similarly took the second activity in domain familiar format. The independent samples t-test based on the scores of the second activity also did not indicate a significant difference between the two groups, t (123) =-1.09, p=.28. In other words, domain familiarity has not been found significant in participants' performances in the second activity.

4. CONCLUSION

Quantitative findings indicated that domain familiarity did not significantly affect programming performance. There are several possible reasons for such a conclusion. The fact that the programming course was at the introductory level may have caused this situation. The programming concepts within the scope were simple, which may have prevented observing the domain familiarity effect. Another point is that quantitative findings may need to be triangulated with more detailed investigations, such as qualitative findings. A thorough examination of the participants' codes may provide more information about domain familiarity. For this reason, in the following study processes, the code structure of the participant products will be subjected to content analysis, and the process will be examined in more detail. This study reports the preliminary analysis of the gathered data; thus, the findings cannot be generalized.

REFERENCES

Baer, J. (1998). The case for domain specificity of creativity. *Creativity research journal*, *11*(2), 173-177.

Bong, M. (2004). Academic motivation in self-efficacy, task value, achievement goal orientations, and attributional beliefs. The Journal of Educational Research, 97, 287–297.

Curzon, P., Black, J., Meagher, L. R., & McOwan, P. (2009). cs4fn. org: Enthusing students about Computer Science. *Proceedings of Informatics Education Europe IV*, 73-80.

Ding, H., Sun, H., & Chen, A. (2013). Impact of expectancy-value and situational interest motivation specificity on physical education outcomes. *Journal of Teaching in Physical Education*, *32*(3), 253-269.

Grover, S. (2013). OPINION: Learning to Code Isn't Enough, from https://www.edsurge.com/n/2013-05-28-opinion-learning- to-code-isn-t-enough

ISTE (2011). Opeartional Definition for Computational Thinking. Retrieved from https://www.iste.org/explore/computational-thinking-all.

Mishra, P., Henriksen, D., & The Deep-Play Research Group. (2012). Rethinking technology and creativity in the 21st century: on being in-disciplined. TechTrends, 56(6), 18–21.

Mohaghegh, D. M., & McCauley, M. (2016). Computational thinking: The skill set of the 21st century.

Reigeluth, C. M. (2016). Instructional Theory and Technology for the New Paradigm of Education. Revista De Educación a Distancia, (50). Retrieved July 20, 2019 from https://revistas.um.es/red/article/view/270781

Sendurur, E., Ersoy, E., & Çetin, I. (2018). The Design and Development of Creative Instructional Materials: The Role of Domain Familiarity for Creative Solutions. *International Journal of Technology and Design Education*, *28*(2), pp. 507-522.

Shen, B., McCaughtry, N., & Martin, J. (2008). The influence of domain specificity on motivation in physical education. *Research Quarterly for Exercise and Sport*, *79*(3), 333-343.