Tutorials Embedded in an IDE: A Feasible Way for CS Students to Learn Debugging? – A Study Design

Olli Kiljunen
Aalto University
olli.kiljunen@aalto.fi

1. Introduction

Although the importance of teaching novice programmers to debug computer programs is well-noted (*e.g.* McCauley et al., 2008; Li, Chan, Denny, Luxton-Reilly, & Tempero, 2019), finding optimal ways to help students to learn debugging is still an ongoing quest. Nevertheless, a common understanding among both the computing education researchers and practitioners seems to be that debugging, possibly due to its complex and often complicated nature, is a difficult subject for students in their early phases of programming education.

The goal of my PhD thesis, is to gain new insight into how debugging can be learnt and develop novel ways to teach CS students to debug. As a part of that, in this doctoral consortium abstract, I present and outline my plans for an evaluative study that tries to answer the question whether a certain learning tool and method – designed and developed by me in collaboration with our research group – would carry potential for becoming a both efficient and practical way to teach debugging.

The learning method I have been developing is based on interactive software tutorials that students take using an educational tool embedded in a programming environment, IDE. In my planned study, I test this learning method with novice CS students to see how they interact with the tool, whether they feel satisfied when learning with it, which features of it they found either useful or useless, and what kind of debugging skills or knowledge can be taught with it. The results are hoped to provide better understanding on how teaching of debugging can be improved and help me in the further development of this method of learning.

2. Background and wider context

Previous computing education research literature has suggested various approaches to teach debugging. For example, Carter (2014) used interactive software tutorials in her study with promising results. Those tutorials, however, take place in an exercise environment that is specifically designed for them and not otherwise used by the students.

Social constructivism and sociocultural learning theories emphasise the situated nature of learning and the learner's interaction with their environment and tools. In order to better align the tutorial-based approach with the understanding of socioculturally mediated learning, the learning method I am suggesting places the tutorials within an authentic programming environment – that is, a fully-featured, professional-grade IDE.

The study I am planning to carry out this autumn is a part of the larger continuum of my PhD thesis. The series of studies – of which this one is the first – takes the form of design-based research where the suggested teaching method and tool are developed, put into use, and evaluated iteratively. Drawing conclusions of my observations on each iteration, I aim at not only coming up with new learning methods but also contributing to the theory of how debugging can be taught and learnt.

3. Description of the tool

The tool I have developed is a software system that integrates into an authentic programming environment and allows students to work with interactive tutorials and/or exercises. In the context of this study, the system guides a student through various debugging tutorials step-by-step. At each step, the tool visually highlights those parts of the programming environment that the student is expected to pay attention to and interact with. It also provides students with hints and suggestions about what they should try next to be able to proceed in their debugging process.

The tool is implemented as a plug-in for IntelliJ IDEA development environment. The concept and underlying design could, however, be ported to mostly any modern programming environment.

4. Planned observation and analysis

In my study design, I plan to recruit a few (4–6) student participants currently taking their first Computer Science course with no or very little preliminary experience in programming. The study will be carried out at the stage of the course when the students have been taught some basic programming language constructs and they have written a couple of simple programs as exercises. At that point, they have not, however, received explicit training on how to debug programs.

The student subjects are taken one by one into an observation room where there is a computer on a desk and a researcher acting as a supervisor of the study. The subjects' interaction with the computer as well as their discussion with the supervisor are recorded. Before starting the study, each subject is asked for their previous knowledge of programming and debugging as well as familiarity of the programming and debugging tools the study involves. In the beginning, the computer has IntelliJ IDEA open with some buggy program's source code. The subject is asked to debug the program – that is, to locate the error in the code and fix it or at least verbally suggest a fix. For this task, subjects are given a few minutes, after which – whether successful or not – they are interviewed about their performance and how they felt about it.

After that, the study proceeds to its second phase, where our debugging tutorial tool is introduced to the subject. A new buggy program is opened for them, and they are asked to debug it following the tutorial. The procedure is repeated so that each subject goes through three tutorials in total. After the tutorials, subjects are asked about how they felt using the tutorials, what they think they learnt, and what they liked and what they did not like in the tutorials and why.

In the last phase, subjects are again given a buggy program to debug but this time without a tutorial. The time limit is similar as in the first phase. Finally, subjects are interviewed of their overall experience. They are asked to reflect on what they learnt when using the tutorials and how they used that knowledge in the last phase.

The target of this study is not to answer the question whether or not our tutorial system is a superior way to learn debugging or – by any means – compare it to other learning methods. Instead, the observed data is scrutinised to understand whether students find this way of learning intuitive and meaningful and whether it can teach them at least some aspects of debugging. Moreover, I analyse the observed data to recognise such interactions with the tutorials where students failed to follow the intended course of actions or felt frustrated, confused, or irritated about the tutorials. From these results, I seek to draw conclusions on how the tutorial system should be developed further to best serve its purpose. Comparing this method of learning debugging to other learning methods is a potential question of future research, based on the results I get.

5. Expectations for the doctoral consortium

By taking part in the PPIG 2022 Workshop and, in particular, its doctoral consortium, I expect to be able to review and refine my study plan based on the discussions with the other participants and the feedback I get. Furthermore, I hope those discussions and working sessions will provide me with creative ideas and viewpoints that will open new interesting directions for my research in the areas of debugging and learning. Also, I would be more than delighted if participating in the doctoral consortium sparked some future collaboration with other researchers who have overlapping areas of academic interest.

6. References

Carter, E. (2014). *An intelligent debugging tutor for novice computer science students* (Doctoral dissertation, Lehigh University, Betlehem, PA, USA). Retrieved from `https://preserve.lib.lehigh.edu/islandora/object/preserve\%3Abp-7256300`

Li, C., Chan, E., Denny, P., Luxton-Reilly, A., & Tempero, E. (2019). Towards a framework for teaching debugging. In *Proceedings of the twenty-first australasian computing education conference* (pp. 79–86).

McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: a review of the literature from an educational perspective. *Computer Science Education*, *18*(2), 67–92.

About me

My name is Olli Kiljunen, and I am a second-year doctoral student in the field of *Computing Education Research* (CER) at Aalto University, Finland. I am writing my thesis as a member of *Learning+Technology Research Group* and under the supervision of Professor Lauri Malmi. In this study, I collaborate with DSc Otto Seppälä and DSc Juha Sorva.

I live in a suburban area in Helsinki, Finland. In addition to programming and education research, my interests include various forms of drama and improvisational theatre, where I fluctuate between the roles of actor, director, playwright, composer, and spectator.