A Programming Circus for Primary Schools

Katharina Geldreich, Alexandra Funke and Peter Hubwieser

TUM School of Education
Technical University of Munich, Germany
{katharina.geldreich, alexandra.funke, peter.hubwieser}@tum.de

Abstract. It is well known that students often hold many misconceptions and stereotypes towards computer science (CS). To prevent students from establishing these false ideas, it seems advisable to counteract at a very early stage in school. To face this, we designed an introductory programming course for children at primary schools. This poster describes the design of the extracurricular course as well as the pilot study we will conduct.

Keywords: Computer Science Education, Programming, Primary School

1 Introduction

Schools and universities have to deal with misconceptions and stereotypes towards CS [2] which are developed at a very young age [4]. One approach to prevent students from forming false and mostly negative attitudes is to introduce computer science concepts like programming at an early stage in school.

To create opportunities for making unadulterated experiences with technology and computer science, it is recommended to expose students to programming at primary or even kindergarten level. In order to provide such an experience, we planned a programming introduction course for grades 3 and 4. Two of our research questions are

- *Which programming concepts can the students learn during this course?*
- *How differ the interactions and programming results between the students?*

2 Course Development

2.1 Design

The course takes place over three days, on which we will spend four hours a day with the children. The students are supposed to learn basic principles of programming. To attract both girls and boys, we designed the course under the motto ”programming circus”, which should be appealing to both genders.

Day 1. The aim of the first day is to give the students a basic idea how computer programs work. To achieve this objective without the distraction of learning programming at first, we decided to use the unplugged approach [1]. To take up the circus theme, we will let the students program each other to find missing items and animals in a symbolic circus tent.

Day 2. On the second day, we want to enable the students to create simple multimedia products using the block based language Scratch [3].

We composed a learning circle in which the basic operations of Scratch get gradually introduced. For example, the children have to program the welcome greeting from the circus director, a joke telling clown and a dancing bear. The students first follow handed-out instructions and solve tasks afterwards.

Day 3. Our goal for the third day is getting an impression of what the students have learned so far and what they can apply in more open tasks.

They have to create their own circus story following several specifications like using a variety of characters or a repetition. At the end of the course, we will take a look at all projects and talk about the students' experiences.

2.2 Pilot Study

To test the course design and effectiveness we will run a pilot class. During the pilot study of this course we want to collect data with the following methods:

Videography. To analyze the interactions of the kids with each other and with the teacher, we will record the whole course on video.

Group interview. We will use a variety of interviewing and reflection methods to get an idea of the students' prior knowledge, what they think about programming and how they feel before and after the course.

Screen and audio capturing. Screens and audio of every student will be captured to get an image of their working methods.

Scratch programming products. We save all programming products the students produce during the three days to analyse them afterwards.

3 Conclusion and further research

After evaluating the results of the pilot study we will revise our course design. More courses with the revised curricula will be held.

We hope to get an insight of what concepts the students have of computer science and what prior knowledge they have. Furthermore we want to understand what they can learn during the programming course and how they learn.

References

1. CSUnplugged.org. Computer science unplugged. csunplugged.org, April 2016.

2. A. Funke, M. Berges, and P. Hubwieser. Different Perceptions of Computer Science. In *Proceedings of the 4th LaTiCE*, LaTiCE '16, New York, NY, USA, 2016. IEEE.
3. J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. The Scratch Programming Language and Environment. *ACM Transactions on Computing Education*, 10(4):16:1–16:15, 2010.
4. K. Prottsman. Computer science for the elementary classroom. *ACM Inroads*, 5(4):60–63, 2014.