Problem Instances, Problem Classes, and the Problem of Transitioning

Dirk Schmerenbeck

Year of Ph.D.: 0
Supervisor: JProf Dr Jacqueline Staub

Trier University, Germany
schmerenbeck@uni-trier.de
staub@uni-trier.de

1 Introduction

Python Bebras is a programming environment designed for beginners to learn basic programming through grid navigation tasks. Users guide a beaver through a lake using a set of simple commands (e.g. `move()`, `turnLeft()`, `turnRight()`, etc.). Initially, students solve specific problems by counting fields, then gradually learn to generalize these solutions to broader task classes. They develop skills in comparing problem instances, detecting similarities, and using loops and conditionals to gradually handle all instances of a class with one program. This process of generalization is a key aspect of computational thinking [1].

Beginner programmers are quickly overwhelmed and oftentimes already challenged enough with the syntax and the local context of their solution [2]. This is why educators often start by introducing well-defined problem instances. Writing code that works for one specific example is more manageable for novices than grappling with the complexities of an entire problem class from the outset. However, as novice programmers advance, they must learn to generalize their solutions to address entire problem classes. This means developing programs that can handle any valid input and produce the correct output, not just solving one particular case.

A simple introductory task might involve guiding the beaver three fields forward to reach a log.

Fig. 1: Simple problem instance where the beaver needs to move three fields ahead before picking up the log.

This task can be solved with the following code:

```
move()
move()
move()
removeLog()
```

As learners progress, they move from solving specific problem instances like the one above to tackling more generalized problems (see Figure 2).

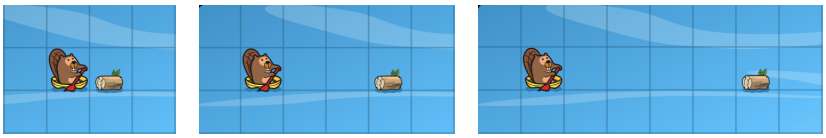

Fig. 2: Three similar problem instances, but each instance requires a different solution.

In the example above the beaver is required to move forward an unknown number of fields until it reaches the log. To solve all three instances with a single program the students have to use a `while` loop to continuously check if the beaver has reached the log:

```
while not onLog():
    move()
removeLog()
```

This is the moment when novices are introduced to the concept of a problem class. The solution described above is applicable to any situation where the beaver needs to move forward until it reaches a log, regardless of how far away the log is.

2 Research Question

The initial phase of my Ph.D. focuses on the further development of the programming environment. Having our own environment at our disposal presents numerous opportunities to customize and tailor it to our specific needs. Our programming environment is particularly well-suited for beginners, enabling them to practice core concepts such as loops and conditionals through a series of carefully designed worlds. Each task represents a set of problem instances that students must solve with a single program. Consequently, a key focus of my research will be to explore how students transition from solving individual problem instances to addressing entire problem classes.

3 Methodology

The Python Bebras online environment offers numerous opportunities for collecting user data, which can be valuable for further analysis and research. We have

successfully partnered with the Swiss Beaver Competition Committee, allowing our environment to be utilized for a bonus task in the upcoming national Beaver Competition. This collaboration will provide us with extensive user data, greatly benefiting our research efforts.

4 Related Work, Open Questions

As I am at the beginning of my Ph.D. journey, I am eager to receive feedback to further refine and clarify my research questions. Additionally, during the consortium, it would be valuable to explore how other researchers might utilize the environment.

References

1. Shuchi Grover and Roy Pea. *Computational Thinking: A Competency Whose Time Has Come*. 12 2017.
2. Donna Teague, Malcolm Corney, Alireza Ahadi, and Raymond Lister. A qualitative think aloud study of the early neo-piagetian stages of reasoning in novice programmers. In *Proceedings of the 15th Australasian Computing Education Conference [Conferences in Research and Practice in Information Technology, Volume 136]*, pages 87–95. Australian Computer Society, 2013.