Variables and variable naming in introductory programming education

Werf, V. van der

Summary

Learning a programming language is challenging. Even one of the first –but crucial– concepts to learn, variables, is hard to grasp. Variables can represent different roles and functions within a code and need to be named appropriately to support code comprehension and debugging. Yet, students can hold various misconceptions about variables that can be (unintentionally) encouraged by the explanations and analogies used to introduce and explain the concept. Moreover, also names can be (unintentionally) misleading, hindering the understanding of the concept and the code.

This dissertation approaches the teaching of a programming language from the reasoning that (familiar) natural language, like the language used to name variables, serves as a bridge between complex programming problems and the programming language itself. The dissertation first investigates students' code explanations in plain English, after which it delves deeper into the teaching of variables and (variable) naming practices in introductory programming education across educational levels. These include secondary education, vocational education, university education, adult education, massive open online courses (MOOCs), and programming textbooks for children and novices.

Throughout the chapters, this dissertation aims (1) to open a scholarly discussion on how naming practices can or should be implemented in programming education, and (2) to inform and support educators and developers of educational materials in the fields of *Computer Science* and *Software Engineering* who want to know how to address the topic in their courses and materials. The performed research is mostly qualitative and exploratory: it involves an analysis of student artifacts (chapter 2), open coding of in-depth interviews with teachers on their perceptions and practices (chapter 3), and observations of educational courses and materials (chapters 4 and 5). However, the research ends with the design and implementation of interactive learning activities to support the adoption of good naming practices among learners (chapter 6).

In short, the dissertation presents the following results. Novice programmers rely on natural language elements that are present in code, to understand and explain it (chapter 2). Yet, students are rarely taught how to name their variables and functions nor how names can interfere with code comprehension (chapters 3, 4, 5). While educators believe that using good naming practices is an important skill for professional programmers, they assume learning this skill is not difficult and is generally done 'naturally' and 'by example' (chapter 3). As such, students are seldom required to use appropriate names and receive little to no feedback on their naming practices (chapter 3), which limits their opportunities to learn from mistakes. Moreover, code examples used in courses and textbooks do not always reflect what is described as good naming practices by teachers *and* those same (course) materials (chapters 3, 4, 5). Finally, the introduction of the concept of variables appears programming-language-dependent, and the used explanations and analogies show a dominant focus on storing information; other functions or benefits of using variables are rarely mentioned and potential misconceptions appear not addressed (chapters 4, 5).

These results show that many students are expected to learn from conflicting information without much support or guidance. Hence, the expectation that students learn by example could be compromised and naming practices deserve more careful and explicit attention in programming courses. The designed interactive activities (chapter 6) include whole-class dialogue based on various naming examples which can raise awareness for the topic's importance, allow students to experience the effect of (unintentionally) misleading names, and provide opportunities for feedback needed to develop one's understanding of good naming practices. Moreover, such activities revealed issues among students that may prevent the adoption of good naming practices, such as rejecting the topic's importance and cost-benefit-related issues (chapter 6).

For educators, the main takeaways are: (1) pay more explicit attention to the importance of developing good naming practices; (2) use consistent naming examples in regular example codes; (3) integrate interactive activities to discuss *how* and *why* names are appropriate or inappropriate; (4) stimulate students' critical thinking on the effect of naming choices; and (5) provide regular feedback on students' naming choices. Furthermore, educators are encouraged to (6) expand the definitions and analogies they use for introducing the concept of a variable in their teaching to better reflect variables' purpose and use in programming and address potential misconceptions.

Lastly, this dissertation challenges the academic community to further investigate the effect of naming practices on students' *learning process* and the adoption of (future) programming skills. Ideally, such research should provide guidelines and contribute to a structured learning trajectory with an appropriate focus on aspects of programming that go beyond problem-solving and code-writing abilities and include more reading and (interactive) comprehension activities.