Teaching programming outside computer science departments

By Cyril Pernet, University of Edinburgh, Krishna Kumar, University of Cambridge, Laurence Billingham, British Geological Survey.

Context

When: The Software Sustainability Institute Fellowship face-to-face selection day (November 2, 2016) involved several sessions including group discussions. This group discussion was about education and software.

Who: The group was composed of Dr Laurence Billingham, Dr Elena Vataga, Dr Krishna Kumar, Dr Cyril Pernet.

What: The discussion was about teaching programming and best practices at universities: whom we should teach, what should be taught and when.

Why

There are two primary reasons to teach programming and best practice to the widest possible community. Firstly, we now live in a digital society and knowing how computers and the web work is nowadays considered a form of literacy just like English and mathematics. Digital literacy is important because it allows individuals to be more responsible on how they use and interact on our digital age, and empowers them to better communicate and work.

Secondly, learning how to think like a software developer gives people powerful new ways of exploring ideas. Learning to code can change the way people think about the world and their place in it. Any programming language is a set of rules, but, through the act of learning the rules of the system and mastering the language, programmers acquire the ability to understand and modify the system. For more on deconstructing and improving broken systems, we recommend Jessica McKellar's PyBay2016 Keynote.

General idea

We unanimously agreed that programming and best practices have to be taught at every level in universities, catering to different needs of people, and have to start at undergraduate level. At undergraduate level, we recommend teaching algorithmic/programming thinking, and not dedicated software applications, emphasising the idea of learning to think, not learning to code. More emphasis should go towards best practices, such as writing good code documentation and manuals, version control systems and, at the same time, favour social aspects such as pair programming and code review. These are the kind of skills which are essential as a programmer, but can be transferred easily across disciplines. Such courses can be run across schools to promote the idea that, beyond coding, these are useful transferable skills.

We agreed that domain specific programming and specific software applications should be taught at masters level in respective schools and departments. More good practices like unit testing should be included in the curriculum.
Challenges

The challenges to implement on our curriculum example are twofold. The first one is trying to get schools to have common teaching. In the academic world, research collaborations have no boundaries at department, school, university, or country levels. When it comes to teaching, this is completely different. Each department oversees its students, which means that to run such a course, you need teachers from different departments and schools to organise it, keeping things simple at an administrative level. For students, it should be easy enough as undergraduates to have more common knowledge, and this will make them realise that common tools and practices exist across the range of professional activities. The second challenge is to convince graduate schools that coding skills are generic and translational skills. Once you have that, you teach it like teaching about writing a good article. Currently there is still too little recognition that coding is a generic skill and thus it remains inside labs, taught locally from PI to student or student to student. It doesn't allow to take full benefit from being in a university with computer science colleagues who can help. Only by pushing coding as a generic skill, one can improve our students' skills.

Conclusion

Hopefully, in 20 years from now, teaching programming thinking won't be necessary at universities, as the UK has made it a part of their official curriculum for all grade school kids. Until then, we should really push to get generic and specific skills taught to improve students' skills but also science, as better software (code) makes better research.