

Teaching Statement

Benedikt Schmidt

Teaching classes is an integral part of an academic position which I highly value as an opportunity to interact with students and get them excited about different fields of computer science. To achieve this, I believe it is important to first enable students to see the core of a problem and then teach them how to transfer the skill of solving such well-defined problems to the messy, ill-structured problems that show up in industry or research. As a teacher, the process of simplifying a research problem and solution to present it in class is also often very useful to give a new perspective, which in turn benefits research.

Teaching experience. Teaching was one of my main responsibilities as a research assistant at ETH Zurich. I was a teaching assistant for *Linear algebra*, *Discrete maths*, and *Computing tools* for students from other departments. As a teaching assistant, I was responsible for giving exercise sessions, developing exercises for the exercise sheets, supervising lab sessions and grading and supervising exams. For the undergrad course *Formal methods and functional programming* with 100-120 students, I was the lead teaching assistant and had the additional responsibilities of supervising the student teaching assistants, devising and organizing the written exams, and substituting for the main lecture occasionally. Since I did not have the opportunity to perform local teaching duties at the IMDEA Software Institute, I have actively sought opportunities to lecture on my research topics at summer schools. For example, I taught lectures on *Computer-aided cryptography* in Algiers, Paris, and an IACR school at University of Maryland, which I co-organized. I am looking forward to perform the same task of designing and teaching a class over the course of a whole semester.

Teaching philosophy. I believe that for a successful class, it is of great importance to clearly communicate what is expected of the students and what services I provide in return. For example, in classes where handing in solutions to the programming exercises was optional, I stressed that in previous years, very few students did well in the exams without handing in exercise solution. The problem was that these students never made use of our service to provide feedback on their work. I also believe that to prepare a class, it is necessary to reflect not only on the topic itself, but also on which teaching strategies are the best fit for the topic and the audience. This requires assessing the students' knowledge before the course and providing students with the means to give feedback early on. If the background knowledge of the students is very diverse, it can be useful to go away from the traditional lecture format and make courses more interactive and non-linear. I have experienced this in courses on computer-aided cryptography where one half of the audience consists of cryptographers and the other half consists of researchers in the area of formal methods. Since both groups often have limited knowledge in the other field, I believe the best approach is to cover both perspectives whenever possible, but to provide additional material for self-study instead of spending too much time on material that is well-known to one of the groups. Of course, the opportunity to discuss the self-study material should be provided. Finally, I think an important skill that should be taught to students is the ability to evaluate their own solutions. Here, automated tests and tools like compilers or proof assistants that give immediate feedback to students play an important role. Since one of my main research interests are tools for analyzing the security of cryptographic constructions, I believe that such tools can be adapted to be highly useful in teaching.

Conclusion. I look forward to having the opportunity to design and teach my own classes. I am willing and able to teach a broad range of courses in computer science. In particular, I would be interested in teaching classes on security, programming languages and formal methods on any level. I would also like to design a class that simultaneously focuses on attacks on real-world implementations of cryptography and how techniques and tools from programming languages and formal methods can be used to make implementations resilient against such attacks.