Teaching Philosophy

For me, teaching is reaching out.

As a student, I had mentors who stimulated my curiosity, my determination and let my personality grow as an independent and responsible individual. Now, it is my turn to reach out so that to start an open dialogue, to realize big ideas and to accomplish great challenges.

When I taught compiler, I always rehearsed the material before every class. In practice, I like to prepare my own lecture materials so to be very familiar with the contents and clear in its delivery. I try to transmit my passion and keep a live class by asking question, leading discussion, using examples so to discover always new prospective. For example, take dead-code elimination --- i.e, to find the code parts of a program that will never be executed— I do not start saying "*so how we model such a problem and what could it be a solution ?…"*. I start with a motivation – why this is import; for example, because reducing the shear size of the code makes the compilation faster, it may discover a semantic error, *etc*. And once the motivation is clear, I go forward.

I present one idea per lecture/session, I reinforce the understanding by example and homeworks, and I reward the ability to think independently or in small groups. Most importantly, they (and I) should cultivate their curiosity and improve their ability every day. I use performance evaluations and feedback forms as the main means to gather feedback promptly (e.g., on a weekly basis) in order to understand the students' comprehension and participation, and to adjust the materials correspondently.

I have applied this approach especially in my compiler classes where I had the opportunity to present the theoretical foundation of computer science and put it in practice. I found this class extremely challenging for the instructor, for the TAs and extremely demanding to the students. However, I would not have it in any other way.