

Position Statement FYI 2000

Dwight House
Fayettevill State University

There are several issues associated with teaching the first year of undergraduate computer science. Issues such as math prerequisites, computer prerequisites, the language to be used (assuming there is to be a language used for implementing algorithms), closed labs versus open labs, the nature of the programming assignments, topics to be covered are a few that spring to mind immediately.

The issue that has recently (in the past few years) become prominent is the paradigm for the language in the first year. More institutions are moving toward OOP as the beginning paradigm - most using C++ (as opposed to Smalltalk or Eiffel). But, C++ allows the use of two paradigms - so, do we stick with just OOP (much like Decker and Hirshfield), or should we teach both (like Deitel and Deitel)? I used to think we should follow the approach of Decker and Hirshfield, and I held to this position rather strongly. I am no longer so sure.

What I still feel strongly is that the primary task of those teaching the first year of undergraduate programming is address (teach ?) problem solving - no matter what paradigm is being used. This is much easier said than done. Students naturally want to spend more time on language syntax. (or even the specifics of an OS or text editor - depending upon the course prerequisites) One way to keep problem solving prominent in the course is to demonstrate it repeatedly during the course. The problems should be chosen carefully, and the steps involved in solving explained clearly, slowly, and (often) more than one time. I am not sure I do this as well as I should. It is particularly difficult at the beginning of the course.