

I Object!

A polemic

Michael Kölling
University of Kent

A change occurred recently in the discussion of introductory programming teaching: the procedural backlash. Suddenly, respected and experienced teachers start claiming that teaching object orientation early is not only difficult, but impossible. Procedural programming once again is supposed to be the solution, with objects having to take a back seat.

This development is not only questionable and unfortunate, but wrong and dangerous.

The practice of teaching programming at universities, colleges and high schools went through a change roughly in the mid 1990s: The teaching of object orientation in introductory courses slowly became mainstream. Fairly soon, the *Object First* or *Objects Early* school of thought was formulated, stating that teaching object orientation by discussing objects from the very start is a good thing.

More recently, opposition to this development has surfaced strongly. One highly visible event that fuelled the debate was a discussion in March 2004 on the SIGCSE mailing list. In one of the posts, Elliot Koffman described the difficulties instructors have with teaching objects early, and Stuart Reges, in one of the responses, started to consider the "objects first experiment" as "a failure".

The following discussion showed clearly – both by number and content of the posts – that the argument had struck a nerve in the community. The arguments were subsequently summarised by Kim Bruce¹, and surfaced again in a variety of papers. Importantly, this discussion brought a feeling held by many teachers individually out into the open, and established this view as an officially stated school of thought, which I shall call *Proceduralism*.

Now, gathering a number of supporters for any chosen viewpoint does not make that view right. The argument presented by the Proceduralists generally goes like this: I have taught in an objects-first style; It has failed; Therefore it does not work.

This, obviously, is not a valid argument. It is, however, exactly what Reges states in his post to the SIGCSE mailing list, and it is what gets reaffirmed by other

¹ Bruce, K., Controversy on How to Teach CS 1: A discussion on the SIGCSE-members mailing list, The SIGCSE Bulletin 29, Vol 36, Number 4, December 2004

supporters of this view. I have two words in reply: existence proof. But more on that later.

So is teaching objects first difficult? I think it is fair to say that many experienced teachers find the teaching of object orientation more difficult than they found the teaching of procedural programming. But we have to ask: Why is this the case?

I believe at the heart of the issue is *inherent* versus *accidental* complexity.

Object orientation – or the teaching thereof – is made complex by two factors: intrinsic complexity – complexity caused by the nature of the problem – and accidental complexity – complexity caused by external factors, such as the use of inadequate languages, tools, teaching strategies, lack of experience by teachers, and so on.

Whether actual problems with teaching OO stem from intrinsic or accidental complexity is an important question: If it were intrinsic complexity, then it cannot be fixed, and an objects first approach would indeed be too hard for beginning students. If it is accidental complexity we can fix it.

I claim (and very strongly believe) that the problems many teachers currently observe originate almost exclusively in accidental complexity. We use non-optimal programming languages, inadequate software tools, many textbooks are appallingly bad, many teachers have no experience or training in object orientation themselves, pedagogical strategies used were developed for entirely different paradigms, and so the list goes on. (Note that I do not claim that good books, tools, etc. do not exist. They do. And they get used in some courses. But they do not get used anywhere close to routinely or widely enough.)

The thing about accidental complexity is that it can be removed. Problems can be fixed, and the situation can be improved. It may not be easy, but it can be done.

I fear that the Proceduralists' arguments are not only weak, but dangerous. They give credibility to a simplistic cop-out of a challenging situation. If we view the problems as intrinsic, then we stop working on improving the aspects that need improvement. The result would be that we give up development of the tools and strategies that could benefit a whole generation of students.

Let us look at some of the arguments made by the Proceduralists in their published statements. Typical is a recent paper which investigates the effect of prior procedural experience before taking an objects-first introductory course. The paper reports that students who had prior programming experience in a procedural language did better in the course than those students that did not have prior programming experience. It concludes that "some initial teaching of Java imperatives and syntax would have benefited the objects-first approach to OO learning". What the paper actually has shown is that having programmed for a

longer time helps. To draw any valid conclusion from the experiment, both groups would at least have to have a similar length of time of programming experience.

Other papers that try to argue the benefits of the procedural approach are similarly full of holes. Claims are made attributing a wide range of observations, such as rising student enrolment, higher percentages of female students and student satisfaction, to the procedural approach, without any data showing any causality between the phenomena. In fact, it seems that in one course, the student rise occurred in the same year as the procedural teaching was started – students therefore obviously anticipating the paradigm change and flocking into the course as a result of it.

Another claim is that student marks were improving when courses are changed back from an objects early to a procedural approach, and this is taken as an indicator that procedural first is better.

In the same vein, I might argue that I was first teaching mathematics, and when I switched to teaching woodwork instead, the marks went up. Therefore, teaching woodwork is better than teaching mathematics.

To turn this into a valid comparison, one would need defined intended outcomes and compare the methods against how close they come – in similar time – to reaching those outcomes. The attempt of comparing two approaches with different outcomes is nonsensical.

So can an objects early strategy work? Of course it can.

As we all know, an existence proof needs only a single case to conclude. There are numerous publications in the literature that describe various strategies, tools, programming examples and practical experiences with using them and report success. Some of them are using tools such as Karel J Robot, Alice, BlueJ, ObjectDraw, DrJava, JPT, and others. Some just use good pedagogic approaches.²

Is it easy to do? That depends. Teaching objects first requires a radical change in our approach to teaching – a change in language, tools, pedagogy, teaching material, strategy and – most of all – mind set. Making this change is not easy at all. It's a radical change. But once that change has taken place, teaching OO is no harder than teaching procedural programming.

Let's not give up on objects just yet. I don't believe that object orientation and objects-first teaching is the solution to all our problems. But I do believe that whatever will eventually replace it should be a step forward, not a step back.

² I had originally intended to reference the relevant papers here, but they quickly became too numerous for a reasonable citation, so I am leaving that out – they are really easy to find if you only care to look.