Object Oriented Programming and Program Correctness: The Students' Perspective

Ioanna Stamouli and Meriel Huggard

University of
Dublin
Trinity College

embarkinitiative
Investing in People and Ideas

Research Objectives

- How do students experience object oriented programming concepts and formal definitions?
- How do students reason about specific programming problems, and programming as a whole?
- How do students perceive their learning environment and which aspects of it do they consider either beneficial or redundant?

The study

- 16 students first year Computer Science.
- *Introduction to Object Oriented Programming*.
- Java.
- 4 sets of interviews.
- Transcribed verbatim.

Themes of the study

Themes	***Theoretical***	***Object Oriented***	***General programming***
Programming	✓		
Learning to program	✓		
Software engineering	✓		
Object Orientation	✓		
Algorithms	✓		
Objects		✓	
Classes		✓	
Methods		✓	
Attributes		✓	
Iterations			✓
Decision making			✓
Arrays			✓
Constructors		✓	
Approach to writing programs	✓		
Understanding of correctness	✓		

Phenomenography

- The first goal of a phenomenographic project is to describe the experience of learning something.
- The focus of such a study is to describe the variations in the ways a phenomenon is experienced.
- Results:
  - Descriptive categories.
  - Hierarchical manner.
  - The variations between the categories present the critical points in understanding and are highlighted in the analysis.
- Semi-structured interviews.

Current status of the research

- Understanding of learning to program.
- Understanding of correctness.

Learning to program

Category Label	***Category Description***
Learning the syntax of the language	Learning to program is experienced as learning the syntax of the language.
Learning and understanding the programming constructs	Apart from learning the syntax of the language, the focus here also includes learning and understanding the constructs involved in programming in general.
Learning to write programs	As above, but also utilising all these to write programs.
Learning a way of thinking	Learning to program is also experienced as learning how to think logically in general.
Learning to "Problem Solve"	Utilising this way of thinking to solve programming problems.
Acquiring a new skill	The whole process is experience as learning a new skill that affects the way one thinks in real life.

Learning to program

- Category 1: Learning the syntax of the language

  Alan: Oh yeah, the syntax, you know, learning to fix the small errors that you would... you know, you have the basic knowledge but then you still need to refine it... you know, you may put the wrong brackets at some point and you may not know the equals method to compare the two strings, you know, stuff like that.

- Category 2: Learning and understanding the programming constructs

  Neil: [...] you need to know the syntax of the language that you are programming in and the concepts like iterations and conditions. But all these… you kind of need to know how to use them, like, and when to use them... The syntax of these things is also important but you need to understand the concepts first, not in a single language, like, because they kind of exist for most of the programming languages.

- Category 3: Learning to write programs

  Anthony: It involves setting out your ideas after thinking how to, like, put these ideas into the proper syntax and you need to know the commands in whatever language you are working in and then writing out the program. It's all about the end result.

Learning to program

- Category 4: Learning a way of thinking

  I: What do you believe is required, or what does it take to learn how to program?

  Patrick: You definitely need the sense of logic, that is the basic thing because everything else then is based in your ability to logically think of a solution, even with some of the other programs that we are doing you need the logic to be able to see it. Basically, you need decent mathematics and a basic sense of logic and the practice, of course. Pay attention and then it clicks, I guess.

- Category 5: Learning to "Problem Solve"

  Brian: I think it's more trying not to look at it as one, as more like… not just as the problem, but more as to how you are going to deal with the problem. There is a lot of ways probably... an infinite number of ways that you can do any one problem. It's just trying to do it the smartest and quickest way really… not the quickest way, the best, the more efficient way.

- Category 6: Acquiring a new skill

  Ken: [...] sort of you work out how to analyse problems and you realise that maybe there are ways of doing something, that you haven't thought of before and you can apply that to other things like in assembly.

Understanding of correctness

- *When do you believe a program is correct?, what is your idea of a correct program?.*

Category label	***Category description***
Syntactical correctness	A program is perceived to be correct when it is syntactically right, that is when it compiles without any errors.
Functional correctness	Apart from being syntactically correct the program needs to fulfill the requirements of the problem specification.
Design correctness	In addition to the above, the program should be correctly structured in order to enable extendibility.
I/O validation and performance correctness	The program should also cater for invalid input and it is should also be optimised in terms of code length and how fast it executes.

Understanding of correctness

- Category 1: Syntactical correctness

  I: When do you think a program is correct?

  Liam4: When it had no bugs.

  I: Anything else?

  Liam: No, not really, as long as it runs, it's right.

- Category 2: Functional correctness

  Patrick: I suppose when it satisfies all the things in the question and when it compiles successfully or when the red lines disappear from Eclipse [laughs]. I suppose it is right when it does what you want it to do.

- Category 3:Design correctness

  Eamonn: I felt it was correct when it runs, did what it was supposed to do and it is structured properly. [...] I feel design is part of correctness. It is easier and better when it is correctly structured, because people can understand it... and because then you can go back and extend and reuse what you had there. So appropriate design is really important and it adds to the solution.

- Category 4: I/O validation and performance correctness

  Colin: A program is correct when it does what you want it to do first of all, so you give it the values you want it to use and then it just works. It works also when it is user proof so if you use the wrong values then you cannot crash it, you cannot pass values that would make it not work. You have to be able to respond properly when you don't give exactly what it wants. It should be able to distinguish among what is valid and what is not.

Observations & implications

- *Our findings suggest that the way students experience learning to program is related to their perception of what constitutes a correct program*

C1. Learning the syntax of the language
C2. Learning and understanding the programming constructs
C3: Learning to write programs
C4: Learning a way of thinking

→

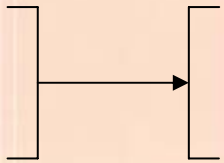
C1. Syntactical correctness
C2. Functional correctness

- *Although a one-to-one relationship was not established between the two themes: from the 9 students that fell into these categories, all, except one, viewed program correctness as syntactical or functional.*

Observations & implications

- *Our findings suggest that the way students experience learning to program is related to their perception of what constitutes a correct program*

C5. Learning to 'Problem Solve'
C6. Acquiring a new skill



C3. Design correctness
C4. I/O Validation correctness

- A one-to-one relationship is observed
- From the 7 students, 3 experienced program correctness as *design correctness* while 4 experienced programming as *I/O validation and performance*.

Observations & implications

- The study's population is not sufficiently large to draw final conclusions on the relations observed.
- It is sufficiently representative to indicate the existence of this trend.
- The results suggest that students develop a general view this then influences their experience throughout the course.
- The findings also show that more than half of the population of the study does not develop a complete and mature understanding of learning to program.
- A similar relationship holds for the understanding of program correctness.

Future work

- Does this hold true for the students understanding of other object oriented constructs such as objects, classes and other programming constructs like loops and if statements?
- Could it be that by positively influencing one theme (such as the students' perceived criteria for program correctness), does this have a concomitant impact on the students' experience, and attitude, towards programming?
- Observe the relationships between the themes.

Thank you for your attention!