

Course proposal: CSC 312, “Programming language implementation”

Department of Computer Science · November 22, 2013

The Department of Computer Science proposes to introduce a new course, “Programming language implementation,” in which students learn the essential concepts in the theory of programming languages (such as evaluation, binding, procedural abstraction, state, control flow, data abstraction, types, and inheritance) by constructing operational models (interpreters) that implement those concepts.

Here is a proposed outline and schedule of topics. The selection reflects the learning outcomes suggested under the headings “Functional programming,” “Basic type systems,” “Program representation,” “Language translation and execution,” “Syntax analysis,” “Static analysis,” “Type systems,” “Formal semantics,” and “Language pragmatics” in *Computing Curricula 2013* (<http://ai.stanford.edu/users/sahami/CS2013/>).

1. Introduction and overview
2. Recursively specified data; contracts; higher-order procedures
3. Data abstractions; `define-datatype` and `cases`
4. Concrete and abstract syntax; context-free grammars in Backus–Naur form; syntax trees
5. Scanning and parsing
6. Evaluating arithmetic expressions
7. Environments and binding
8. Procedures and closures; recursion
9. State and assignment; lvalues and rvalues
10. Storage structures
11. Parameter passing; eager and lazy evaluation
12. Control flow and continuations
13. Imperative style
14. Exceptions
15. Threads and synchronization
16. Continuation-passing style; tail-call optimization
17. Type rules and type checking
18. Type inference
19. Objects, classes, and inheritance
20. Type checking in object-oriented languages
21. Final examination

As a textbook for the course, we propose *Essentials of programming languages*, by Daniel P. Friedman and Mitchell Wand (Cambridge, Massachusetts: MIT Press; third edition, 2008; ISBN 978–0–262–96279–4)

We propose to offer the course for two credits. The class will meet three times a week through the first seven weeks of the semester, each session lasting fifty minutes. Most class sessions will be labs consisting of short programming exercises, which the students will work on in pairs. We’ll assess students on the basis of outside programming assignments, lab work, and a fifty-minute final examination.

This course will be an elective, designed primarily for majors, and is not intended to serve as a prerequisite for any other course.