

CSE 755
Programming Languages
Autumn 2011

Syllabus

Instructor: Atanas (Nasko) Rountev

Course Summary

This is a course on the theory of programming languages. The goal is to study formal ways of defining the syntax and semantics of programming languages. The main topics are attribute grammars, axiomatic semantics, and operational semantics. In addition to these topics, we will cover some material on functional programming and type systems. This is a fairly theoretical course, and it requires understanding and applying various formalisms in the context of programming languages.

Prerequisites

CSE 625: Introduction to Automata and Formal Languages, and CSE 655: Introduction to the Principles of Programming Languages. This is a course from the graduate core (together with 725, 760, 775, and 780), and the level of difficulty is relatively high. If you are an undergraduate student with good math abilities and willingness to do graduate-level work, you should do fine; if not, you should probably not take this course. If you are a graduate student in another department, you *must* have in-depth experience with imperative and object-oriented programming, and some background in formal languages and automata.

General Information

- Credits: 3
- Instructor: Atanas (Nasko) Rountev, routev@cse.ohio-state.edu, 292-7203
- Instructor's office hours: DL 685, Tues 2:00 – 3:00, Thurs 1:00 – 2:00, or by appointment
- Grader: Pawas Ranjan, ranjan@cse.ohio-state.edu
- Grader's office hours: DL 474, Tues and Wed 2:00 – 3:00, or by appointment
- Course web page: <http://www.cse.ohio-state.edu/~routev/755>

Topics

- Attribute Grammars
- Operational Semantics and Interpreters for Lisp
- Type Systems

- Axiomatic Semantics
- Operational Semantics
- Other Topics

Objectives

The principal course objective is to discuss ways to define formally the syntax and semantics of programming languages. We will also talk about programming methodologies (e.g., functional programming). Upon successful completion of the course, students will be able to:

- Understand the role of certain theoretical formalisms, and apply them in the context of programming languages.
- Use attribute grammars to specify context-sensitive conditions, compile-time analyses, and translational semantics.
- Define the axiomatic semantics of simple imperative constructs, and use it to prove program properties.
- Define the operational semantics of simple imperative languages and functional languages.
- Use type systems to specify compile-time analyses.
- Understand some of the differences between programming methodologies.
- Implement parts of simple interpreters and compilers.

Reading

This course does not have a required textbook. We will use materials from several different books.

- Frank Pagan, *Formal Specification of Programming Languages: A Panoramic Primer*, Prentice-Hall, 1981.
- Kenneth Slonneger and Barry Kurtz, *Formal Syntax and Semantics of Programming Languages*, Addison-Wesley, 1995.
- Glynn Winskel, *The Formal Semantics of Programming Languages: An Introduction*, MIT Press, 1993.
- Ryan Stansifer, *The Study of Programming Languages*, Prentice-Hall, 1995.
- Benjamin C. Pierce, *Types and Programming Languages*, MIT Press, 2002.
- Lisp 1.5 Programmer's Manual, McCarthy and others.

Your most important reading will be the lecture notes and your own notes. Copies of all notes will be handed out in class, and will also be available on the course web page. For each topic, I will give you pointers to relevant parts of the books.

Course Web Page

<http://www.cse.ohio-state.edu/~rountev/755>: the course web page will contain all notes, handouts, assignments, a detailed schedule, pointers to reading materials, etc. Copies of assignments etc. handed out in class become official, independent of whether they are on the web page or whether you are able to access the page.

Assignments

- There will be several assignments, typically due in 7 to 10 days.
- Assignments should be done independently. General high-level discussion of assignments with others in the class is allowed, but *all of the actual work* should be your own. Assignments that show excessive similarities will be taken as evidence of cheating and dealt with accordingly.
- Assignments should be turned in **by the beginning of class** on the due day. Late assignments turned in by the beginning of the next class will be graded with 30% reduction. Assignments turned in later than that will not be accepted.
- Make the assignments readable and understandable. They should be handed in on regular paper, legibly written or typed. If you have more than one sheet, **staple the sheets together**. If the grader has trouble reading or understanding what you have done, points will be deducted even if it can finally be determined that you have the correct answer.
- *Important:* A common theme of this course is the application of theoretical principles to small problems in the domain of programming languages. As with all theoretical foundations, your solutions have to be *precise and detailed*: you have to work out *all* details that are necessary to solve the problem using the approaches discussed in class. You also have to write your solutions in a way that convinces the grader that you understand all these details. Be careful, precise, and thorough.

Project

- There will be one programming project, which has to be submitted electronically on **stdsun** by midnight on the due date. The project *must* compile and run on **stdsun**. Some people prefer to implement the project on a different machine, and then port it to **stdsun**. This usually creates ugly problems in the last minute, and I would advise against it. Still, if you decide to use a different machine, it is **entirely** your responsibility to make the code compile and run correctly on **stdsun** before the deadline.
- The project should be done independently from other students in the class. General discussion of the project with others in the class is allowed, but you have to do all the design, programming, testing, and debugging independently. Projects that show excessive similarities will be taken as evidence of cheating and dealt with accordingly.
- The projects are due by 11:59 pm on the due day. **Absolutely no exceptions** will be made to this deadline: if you submit at 12:00 am, your submission will be considered to be late. The time stamp on the electronic submission will be used to determine the submission time. A reduction of **10% per day** will be applied to late submissions. Submissions more than three days late will **not** be accepted.

Exams

- There will be a midterm exam and a final exam.
 1. Midterm: some time during the middle of the quarter
 2. Final: at the standard time determined by the university

Both will be comprehensive, in-class, closed book. You will be allowed to use a *cheat sheet* — one standard-sized piece of paper, with notes on both sides.

- The exam questions will typically require creative application of the general approaches discussed in class. Memorizing things will not be enough; you need to have conceptual understanding of the techniques we have covered, and how these techniques could be applied to small problems. Exam questions will be very similar to the questions from the homeworks; thus, you should make sure that you have very solid understanding of all details in the homework solutions.
- Missing the midterm or the final without prior written (e-mail) approval from me will result in a score of zero for that exam. To get my approval to reschedule an exam, e-mail me **at least one week** before the exam is scheduled. I will not give such approval unless the reasons are justifiable.

Grading

Assignments	20%
Project	25%
Midterm	20%
Final	35%

Grading Policy

The entire course will be graded on a curve. I expect the average grade to be around B+. For this reason, I will deduct points rather liberally and I will encourage the grader to do the same. Keep this in mind if you get a score that you consider to be relatively low. Of course, when grading on a curve, the absolute score is not important. For homeworks, exams, and the project, I will give you statistics that will help you understand your standing in the class.

It is a course policy that whoever graded something will be responsible for handling grading disputes. I will grade the midterm exam and the final exam. The grader will grade the assignments and the project. Grades become final one week after an assignment or an exam is handed back. This should leave plenty of time to resolve grading disputes.

Honesty

I will treat you as professionals, and you should conduct yourselves as such. You are free to discuss the assignments and the project with others. However, the solutions you submit should be developed by yourself. **Cheating is a very serious offense and will not be tolerated.** Supplying others with materials is also against this rule. Additional details on academic integrity are available at <http://oaa.osu.edu/coamresources.html>. **Please read this information carefully.**

Students with Disabilities

Any student who feels he or she may need an accommodation based on the impact of a disability should contact me privately to discuss his or her specific needs. Please contact the Office of Disability Services at (614) 292-3307, or visit 150 Pomerene Hall, to coordinate reasonable accommodations for students with documented disabilities.

Religious Obligations

I will do my best to accommodate any religious obligations you may have. Please contact me privately, at least one week in advance, to work out any relevant details.