Contact Information

Instructor: Dr. John C. Lusth  
Email: lusth@cs.ua.edu  
Web: http://beastie.cs.ua.edu/proglan/  
Office: 3427 SEC  
Hours: click here  
Phone: 205-348-1618

Catalog Statement

Formal study of programming language specification, analysis, implementation, and run-time support structures; organization of programming languages with emphasis on language constructs and mechanisms; and study of non-procedural programming paradigms.

Implementing the Catalog Statement

Students will embed features from various languages into Scam, a lexically-scoped variant of LISP. Students will also individually implement a programming language of their own design and write a significant program demonstrating the language.

Additional and orthogonal subjects to be covered in this class are: lexical analysis, syntactical analysis, parameter passing strategies, and axiomatic semantics. The last two topics will be covered near the end of the semester, as time permits.

At the end of this class, a student should be understand the concepts of: syntax vs. semantics, data vs. procedures, objects and state, functional programming, evaluation strategies, and interpretation vs. compilation.

Prerequisite

CS200 and CS201

Textbook and Supplies


A laptop computer is required for this course.

Larger Issues

This class is not only concerned with what you learn, but how fast you learn. The only way to learn quickly is read in advance, pay close attention in lecture, ask copious questions, take your study group seriously, and start all programming tasks early.

Topics

The following chapters and topics from the text will be covered:

<table>
<thead>
<tr>
<th>sections</th>
<th>topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1—1.3</td>
<td>data recursion; program recursion; scope</td>
</tr>
<tr>
<td>2.1—2.4</td>
<td>data interfaces, abstractions, and representations, user-defined data structures</td>
</tr>
<tr>
<td>3.1—3.3.5</td>
<td>environment passing interpreters</td>
</tr>
</tbody>
</table>

In addition, you will need to implement a programming language of your own design.

Attendance

Attendance is not required. If you miss a lecture, you are expected to get any missing information from other members of the class, especially your study group. I will not tell you what you’ve missed as I am too lazy to lecture over the same material more than once.
An absence will be excused if you provide a written, university authorized, excuse.

Grading
Grades will be determined by the following percentages:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>Prerequisite Exams (2)</td>
</tr>
<tr>
<td>30%</td>
<td>Periodic Concept Exams (3)</td>
</tr>
<tr>
<td>20%</td>
<td>Final Exam</td>
</tr>
<tr>
<td>20%</td>
<td>Large programming assignment (mandatory)</td>
</tr>
<tr>
<td>20%</td>
<td>Small assignments (3)</td>
</tr>
</tbody>
</table>

Grading Methodology
The Prerequisite Exam is a (mostly) multiple choice test consisting of questions that will be somewhat randomly selected from the set of questions found at:

http://beastie.cs.ua.edu/concepts/cs/ds/all.html
http://beastie.cs.ua.edu/concepts/cs/al/all.html

Note that selected questions may be randomly tweaked.

Small assignment submissions that fail to load into the Scam interpreter or do not compile will be given no credit. Points are deducted for failing to follow specifications, failure to provide sufficient testing, poor programming style, poor program organization, and failure on test cases. Each problem in an assignment is weighted equally.

For your designer programming language, you should implement the following support modules in the order given:

- Grammar
- Lexical Analyzer
- Parser
- Environment

The milestones for having these support modules completed are found in the schedule.

Your final project, at a minimum, must recognize programs written in your language. Otherwise, you will receive a failing grade for the class.

Other Details
Please be in your seat by the beginning of class. There is to be no food or drink in the classroom. Sleep is allowed only in the event of a stultifyingly boring lecture.

Signing up for the cs403 forum is mandatory. Schedule changes, assignment clarifications, and such will be announced via the mailing list.

Assignments are to be submitted electronically by 4:59:59 p.m. on the day they are due. There is a grace period: late assignments will be accepted up to 72 hours after the due date with no penalty. Each on-time small assignment submission (excluding pass/fail assignments) which grades 80% or better will be rewarded with 0.33 extra credit points added directly to your final percentage. No assignment will be accepted after the grace period. There will be no exceptions to this policy unless a student is incapacitated on the due date as well as during the 72 hour grace period. Please do not ask for an extension as I will most assuredly say no and then we’ll both end up feeling bad.

Grading is straight scale with the following cut-offs: 90% and above is an A, 80% and above is a B, 70% and above is a C, 60% and above is a D, and anything lower than 60% is an F. These cut-offs are strict. For example, 79.99999% is a C. The delta for plus-minus is two points on the low end and three points on the high end. For example, 90.92 will be an A-, while 97.100 will be an A+.

There are no make-up exams. If you miss an exam, a zero will be recorded for your score.

Limited discussion about the programming assignments is allowed, but only via the mailing list and only at a high level. Never post or share assignment code. If I can prove you even looked at someone else’s or some other team’s code or allowed someone to look at your code, you will be considered guilty of plagiarism. The minimum punishment requested for plagiarism,
or any other academic misconduct, is dismissal from the class with a failing grade. You are required to protect your work from plagiarism. If your work is plagiarized, it will be assumed that you were a willing participant in the plagiarism and you will receive the same punishment as the plagiarist. The Department of Computer Science uses sophisticated code “fingerprinting” tools to detect plagiarism.