

Comp 311  
Principles of Programming Languages  
Lecture 1  
Course Overview and Culture

Corky Cartwright  
August 24, 2009



# Course Facts

See web page

[www.cs.rice.edu/~javaplt/311](http://www.cs.rice.edu/~javaplt/311)

Participate in the course newsgroup



# What is Comp 311?

- Anatomy (Syntax) and Physiology (Semantics) of Programming Languages
  - What is the anatomy of a programming language
    - Parsing and abstract syntax
    - Lexical nesting and the scope of variables
  - What are the conceptual building blocks of programming languages? (common anatomical structures and their functions)
  - Use high-level interpreters to define meaning of languages (expression evaluators)

# What is Comp 311, cont.

- Using anatomy to prevent bugs
  - Type systems
  - Type checking
  - Type inference (reconstruction)
- Mechanisms for language extension
  - Syntax extension (macros)
  - Reflection
  - Custom class loaders
- Sketch how the interpretation process can actually be efficiently implemented by machine instructions
  - CPS transformation
  - Garbage Collection



# Subtext of Comp 311

- Teach good software engineering practice.
- You have to write lots of lines of conceptually challenging lines of code in this course. With good software engineering practices, the workload is reasonable.
- With poor software engineering practices, the workload is unreasonable.

# Good Software Engineering Practice

- Test-driven design
  - Unit tests for each non-trivial method written before any method code is written
  - Unit tests are a permanent part of the code base
- Pair programming
- Continual integration
- Continual refactoring to avoid code duplication
- Conscientious documentation (contracts)
- Avoiding mutation unless there is a compelling reason



# Why Study Programming Languages?

- Programmers must master the programming languages of importance within the domains in which they are working.
- New languages are continually being developed. Who knows what languages may be involved in computing 25 years from now?
- Many software applications involve defining and implementing a programming language.
- A deep knowledge of programming languages expands the range of possible solutions available to a software developer. A program design may involve extending the designated implementation language either explicitly (macros, new/revised translators & custom class loaders) or implicitly (new libraries, hand-translation)
- Some implementation languages are extensible through macros, reflection, or customizable class loaders.



# Course Culture

- Approximately 8 programming assignments  
7 required  
1-2 extra credit
- Assignments must be done in either Generic Java (Java 6.0). We encourage you to use DrJava. Both JUnit and javadoc are built-in to DrJava and it fully compatible with command line compilation, execution, and testing (using for example ant scripts).
- Late assignments not accepted, but ...
  - Every student has 7 slip days to use as he/she sees fit.
  - Saving as many as possible until late in the term is advantageous.



# Course Culture, cont.

- Assignments are cumulative.
- Class solutions are provided for the first three assignments within three days after they are due.
- After the third assignment, you are on your own. Extensive unit testing is important because you can reuse previous unit tests on subsequent assignments with no change in most cases.



# Course Culture, cont.

- My teaching style
  - Encourage you to develop a passion for the subject and personally digest and master the material.
  - Make the course accessible to students who don't aspire to become language researchers (avoid repeating my experience in complex analysis with Andy Gleason)
  - Weaknesses:
    - Tendency to digress
    - Explain concepts at too abstract a level without sufficient examples

