COMP 3351 Programming Languages

Christian Grothoff
christian@grothoff.org

http://grothoff.org/christian/

README

http://grothoff.org/christian/teaching/2008/3351/

Overview

- Programming languages $\equiv$ theory and practice
- **Theory:** written assignments and exams
  Practice: use and implementation of language features
- You will write code in C, Java, ML and Prolog.
- You must already be able to write non-trivial code in either C, C++ or Java.

Academic dishonesty

- Webpage says what is allowed.
- If in doubt, ask first.
- Cheating can be detected with automated tools.
- Any violation will be reported.

Expectations

- Read the indicated chapters of the textbook – not every detail is covered in class, but it may still be helpful in exams!
- Deliver tested, working versions of assignments on time using subversion.
- Present the assigned programming language in class.
- Answer theoretical questions in midterm and final exams.

Theoretical content

- Syntax and semantics
- Functional programming and logic programming
- Types
- Scope
- Calls, references, pointers, classes
- Interpreters and cost models

Practical content

- Tools: subversion, JavaCC, JTB
- Functional programming in Java
- Implementation of an interpreter
- Many other programming languages

Presentations

- Ada ("defense"), Cobol ("business"), Fortran ("science")
- AMPL ("operations research"), TeX ("type setting")
- Haskell ("functional"), PL/1 ("high-level language")
- Perl ("text processing"), Emacs Lisp ("extension programming")

Presentations

- Ada, Cobol, Fortran
- Haskell, PL/1, Modula-3, Scheme
- AMPL, M4, Make, AWK, sed, TeX
- Perl, Python, Ruby, Emacs Lisp, JavaScript

Questions

?

Why study programming languages?

Why study programming languages?

- Helps or enables programming, debugging and profiling
- Selecting programming language for project is the most important software engineering decision
- Sometimes writing a new language is the best solution
- Understanding language constructs abstractly helps learn new languages
- Understanding language implementations enables emulation of constructs in other languages

Language Systems (1/2)

Language systems include features like:

- Editor
- Compiler
- Linker
- Loader
- Runtime system

Language Systems (2/2)

Language systems include features like:

- Runtime library
- Debugger
- Profiler
- Build tools
- Packaging tools

Kinds of programming languages

- Machine language and assembly
- Imperative
- Functional
- Object-oriented
- Logical
- Untyped, weakly-typed, strongly-typed

Kinds of language systems

- Ahead-of-time compiled
- Just-in-time compiled
- Interpreted
- Interactive
- Garbage collected

Other considerations

- Availability
- Documentation
- Library
- Maturity
- Standardization

Questions

?

Homework hints

- $ svn add filename ; svn commit -m “logmessage”
- $ gcc -o binary sourcename.c ; ./binary
- $ latex filename.tex ; xdvi filename.dvi
- $ javac pack/Type.java ; java pack.Type
- $ sml < filename.sml

Homework summary

Before the next lecture:

- Generate password with `htpasswd` and register account.
- Read textbook chapter 1 and skim chapter 2.
- Install software (or use department machines).
- Implement "Hello World" a few times.
- Test with provided script and submit!

Questions

?