CSCI 365 SYLLABUS

COURSE NAME, NUMBER	Programming Language Concepts, CS 365
SEMESTER, YEAR	Spring 2008
INSTRUCTOR	T. VanDrunen
OFFICE / TELEPHONE / EMAIL	Armerding 112 752-5692 Thomas.VanDrunen@wheaton.edu
OFFICE HOURS	MTuWF 3:10-4:10 pm; Th 8:30-11:30 am.
COURSE WEBSITE	`http://cslab.wheaton.edu/~tvandrun/cs365`

RESOURCES Tucker and Noonan, *Programming Languages: Principles and Paradigms,* (First Edition). McGraw Hill, 2002.

COURSE DESCRIPTION

Formal definition of programming languages, including syntax and semantics; recursive descent parsing, data structures, control constructs, recursion, binding times, expression evaluation, compiler implementation; symbol tables, stacks, dynamic allocation, compiler compilers.

GOALS AND OBJECTIVES

1. Students will be able to articulate and analyze the structure and formal aspects of programming languages
   - Using formal models of a language's lexical structure, syntax, and semantics
   - Comparing the strengths and weaknesses, similarities and differences among various programming paradigms.
2. Students will be able to prove propositions about programming languages
   - Using structurally inductive proof techniques.
   - Using notions of type-safety.
   - Using the progress and type-preservations lemmas
3. Students will be able to write fragments of compilers and other programming language systems
   - Articulating the compilation process.
   - Writing program analyzers (for example, type-checkers) to discover information about programs
   - Writing interpreters to execute programs in a programming language
   - Writing source-to-source compilers to illustrate the equivalence among programming languages.

ASSESSMENT PROCEDURES

1. Problem sets will exercise students' abilities to use language models and write proofs of language attributes.
2. Projects will teach students to apply their understanding of programming languages in building programming language systems.
3. The midterm and final exam will evaluate students' mastery of the comprehensive material.

Grading:

	weight
Daily work	5
Projects	50
Midterm exam	22.5
Final exam	22.5

SPECIAL EXPECTATIONS

Academic Integrity

A course of this class size and difficulty may present situations where the line between beneficial collaboration and cheating is blurry. To clarify, we make a distinction between *daily work* and *projects*.

At the end of most class periods I will assign a few small problems for you to work on for the next class period. The purpose is to reinforce the material, to prepare you for the next period's discussion, and to show you what to expect on the exams. Daily work has a minimal affect on your grade for the course. Thus collaborating on daily work usually will be acceptable—judge for yourself whether you are benefiting from it.

Most of your work in this class will be in the programming projects, and they contribute to 50% of your course grade. These are to be your own work. You may discuss the problem with your classmates and you are encouraged to share testcases. Your code, however, must be solely your own work.

Attendance

While I was an undergraduate, I missed a grand total of two classes, and one of them was to take the GRE. I expect the same from my students. Since being a student is your current vocation, since your learning now will affect your ability to support a family and church later in life, and since you, your family, and/or a scholarship fund are paying a large sum of money to educate you, being negligent in your schoolwork is a sin. I do not take attendance, but I do notice.

Special needs

Whenever possible, classroom activities and testing procedures will be adjusted to respond to requests for accommodation by students with disabilities who have documented their situation with the registrar and who have arranged to have the documentation forwarded to the course instructor. Computer Science students who need special adjustments made to computer hardware or software in order to facilitate their participation must also document their needs with the registrar in advance before any accommodation will be attempted.

Lectures. To prepare for class, you should complete the assigned reading (when assigned) and a problem based on the previous class period's discussion. Class will begin with a discussion of the assigned problem, followed by an exploration of the topics from the reading.

Projects. Most of the work outside of class will be on programming projects, of various lengths. All of these will involve writing part of a programming system (that is, an interpreter, analyzer, or compiler) for a language based on the textbook's example language, Jay, or a similar language. As mentioned above, these should be worked on independently.

General outline.

- Syntax (2 weeks)
- Semantics (1 week)
- Imperative languages (3 weeks)
- Object-oriented languages (2 weeks)
- Functional languages (3 weeks)
- Exceptions (1 week)

See the course website for the current plan for specific topics and their order. This schedule is subject to change, and I am open to hearing suggestions from you about what topics you are interested in that are not currently planned. See the table of contents of your textbook for a list of potential topics.

Examinations. The midterm is Wed, March 5, during class period. That is the week before spring break. (There may also be a take-home portion.) The final exam is on Tuesday, May 6, at 1:30 pm. I do not allow students to reschedule examinations because of travel, so make your plans accordingly. The final will be "not explicitly" cumulative.