

CSC 4101

Programming Languages

Fall 2006

Syllabus

Gerald Baumgartner

Course Summary

Principles of programming language design; specification of syntax and semantics; underlying implementation of block structured languages; dynamic memory allocation for strings, lists, and arrays; imperative versus applicative programming; logic programming; modern programming languages.

Prerequisites

- CSC 3102: Advanced Data Structures and Algorithm Analysis.
- Credit will not be given for both this course and CSC 7001.

Office Hours

Who	Where	Phone	E-Mail	When
Gerald Baumgartner	Coates 290	578-2191	gb@csc.lsu.edu	TTh 2:30-3:30pm
Hua Cao Looney	Coates 168	N/A	hcao@csc.lsu.edu	Th 3:10-4:50pm

Other office hours by appointment (recommended).

Reading

- Michael Scott, *Programming Language Pragmatics*, 2nd ed., Morgan Kaufmann Publishers, 2005.
- Robert W. Sebesta, *Concepts of Programming Languages*, 5th or 6th ed., Addison-Wesley, 2003 (optional).
- Guy L. Steele Jr., *Growing a Language*, OOPSLA '98, ACM, 1998.
- Henry G. Baker, *I Have a Feeling We're Not in Emerald City Anymore*, ACM SIGPLAN Notices, Vol. 32, No. 4, April 1997.
- Richard P. Gabriel and Ron Goldman, *Mob Software: The Erotic Life of Code*, OOPSLA '00, ACM, 2000.
- Mailing List cs4101@byte.csc.lsu.edu.
- Web page <http://www.csc.lsu.edu/~gb/csc4101/>.

Important Dates

- Fall Holiday: Thu, Oct 5
- Midterm Exam: Tue, Oct 17, 12:10–1:30pm
- OOPSLA conference: Oct 24–26
- LCPC '06 conference: Nov 2–4
- Turkey Day Break: Thu, Nov 23
- Final Exam: Sat, Dec 16, 3:00–5:00pm

Both exams are comprehensive.

Depending on the number of graduating students, the final exam might have to be moved to another date.

Homeworks

There will be five homework assignments, which will be due at the beginning of class on the due date. A penalty of 15% will be assessed for each day a homework is late up to a maximum of 30%.

Programming Assignments

There will be two programming assignments, which will be due at midnight (11:59pm) of the due date. The first two programming assignments will be in Java (or maybe in C++). The third one will be in Scheme. There will also be small programming assignments, such as 1–10 liners in Scheme, ML, or Prolog as part of homeworks.

A penalty of 10% will be assessed for each day a project is late up to a maximum of 30% after which it will not be accepted.

Grading

Homework	15%
Projects	36%
Midterm	24%
Final	25%

The course will be graded partly on a curve. For this reason, I will deduct points rather liberally and I will encourage the grader to do the same. Don't be too upset if you don't get what you consider to be a high score. When grading on a curve the absolute score is not that important. To give you a feeling about where you are standing in class, statistics about the scores will be provided periodically.

Topics

The following list of topics is the ordering of subjects covered in this course. The time spent on each subject will vary. I don't know yet when the homework and project due dates will be. So the following list of topics is only a rough outline.

Topic	Due
Introduction, Functional Programming	
Scheme, OO Programming	HW 1
Syntax and Parsing, OO Programming	HW 2
OO Programming, Data Types	Project 1
OO Programming, Types, Scoping	HW 3
Subprograms and Parameter Passing	Midterm
Parameter Passing, Subprogram Impl.	HW 4
Subprogram Impl., Exception Handling	Project 2
Functional Programming, ML	HW 5
Logic Programming, Prolog	Project 3

Due Dates and Grading

The time allotted for each homework assignment will be made quite generous, as such, the penalty for turning in late is high: 15% per day up to a maximum of 30%. If you turn in a homework on a weekend, write on the submission date and time and slide it under my door. I usually stop by my office on weekends. Predating a late submission is considered a cheating offense. On weekends, submission by Saturday 5pm will incur a penalty of 15%; submission by the class period on Monday will incur a penalty of 30%.

Since the time needed for finishing a programming assignment is harder to estimate and to allow fixing severe bugs that show up close to the deadline, programming assignments can be submitted up to three days after the official deadline. For each day past the deadline, a penalty of 10 percent will be incurred. Programming assignments will be submitted electronically. Projects will be due at midnight. The submission facility will reject your submissions three days after the due date.

Without prior arrangements in case of extenuating circumstances, submission of homeworks and projects past the late deadline is *not* allowed and such work will *not* be graded and you will receive *no* credit. It is your responsibility to make sure that you have completed your work with enough time to submit your materials.

Grading disputes can be submitted in writing with accompanying documentation, or in person during regular office hours. It is course policy that whoever graded the work will be responsible for handling disputes. In general, I will grade the midterm exam and the final exam. The grader will grade the homeworks and the programming assignments. Grades become final one week after a homework, project, or exam is handed back. This should leave ample time for resolving grading disputes.

Homework Standards

All written work submitted must carry the student's name and must be reasonably neat and well organized. Any work that cannot be read easily will score zero points. A reasonable standard of English expression and grammar is also required. The same requirements apply to exams.

Programming Standards

The algorithm used must be essentially correct. Obviously, the program should (compile and) run. Because of the complexity of some of the programs, very little or no credit can be given for a program that doesn't run. If a program dumps core or throws a runtime exception, only partial credit will be given.

Since programming assignments might build on top of a previous project, it is very important to get each submission to run without core dumps and to structure the program so it can be easily extended.

I expect your work to exhibit high standards of programming style and layout, reflecting your expertise as a computer professional. Poor style and documentation may result in points being deducted.

Honesty

I will treat you as professionals, and you should plan on conducting yourself as such. This course presents many important concepts you will need throughout your career as a computing professional, so it is important that *each student* do *all* the assignments and projects and learn the material.

There will be several homework assignments and programming projects. You are free to discuss these assignments with others. However, the programs and homework solutions you submit are to be developed by yourself. *Cheating is a very serious offense and will not be tolerated.* The grader or I may use tools for detecting cheating on programming assignments. Supplying others with material is also against this rule. The policy is that the supplier and receiver of information will both be punished.

Save all handwritten notes and printouts you generate as you work on a project and keep them until the end of the quarter so as to protect yourself in the event that someone “borrows” your program, or the version you submit is lost. For your protection, cases of missing output should be *immediately* reported.

Computer Account Security and Use

To help others resist the temptation of using your work, you should maintain proper security on your computer account. Especially, keep your password from others and do not alter the protection on any of your files. To give others access to your account or files or printouts of your programs is the same as giving them the information directly and will be dealt with accordingly. Any trouble with computer accounts should be referred to an instructor as soon as possible.

When a program has been submitted electronically, you should maintain an *unedited* version of what you submitted (with the correct date stamp) until after that program has been graded. It has also been found beneficial to make a copy of a file before editing that file, in case a system crash occurs during the edit session. When typing in a program for the first time, or when making major changes, you should save your work regularly.