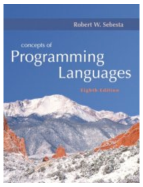
Syllabus for
Programming Languages
COMP 335

Instructor: Mark LeBlanc (`mleblanc`)

Office Hours: by appt. *or*
MW 11:30 – 12:20

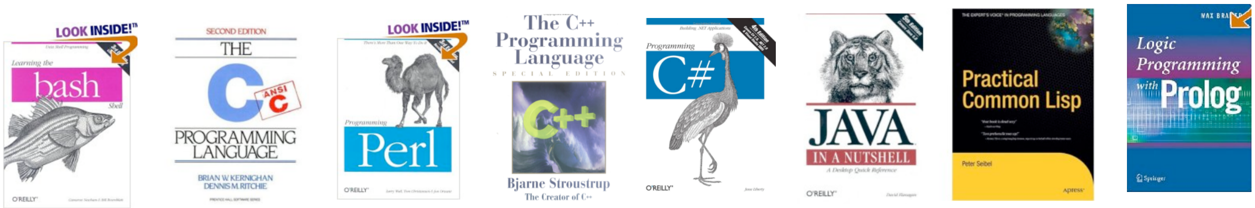
Office: SC-B103
Phone: 286-3970 (on campus: x3970)

Meeting: MWF 10:30-11:20
A118 or csLab



Required Text:
Concepts of Programming Languages (8th edition)
Robert Sebesta, Addison-Wesley, 2007.

You will need to augment this text with other language references. Our Wheaton library (as well as csLab and my own personal library) is well stocked with Perl, Visual Basic, Lisp, C, C++, Fortran, Cobol, Prolog, Modula, Java, Pascal, Scheme, Ada, assembler, and other titles.



Content

This course provides an introductory theoretical study of the principles that govern the design and implementation of contemporary programming languages. This includes language syntax (lexical properties, BNF notation, and parsing); language representations (data structures, control structures, and runtime environments); and language styles (procedural, functional, declarative/logical, scripting, and object-oriented). Particular focus will be on system and object-oriented languages. Labs, homeworks, and programming assignments will include the following languages listed below:

Language type	we will cover
system tools:	bash, C
data-interface/scripting:	Perl
object-oriented:	Java, C++, C#
functional:	Lisp
declarative (logic):	Prolog

Good web sites (see the accompanying **Bb** page for additional URLs)

Dr. Dobb's Journal
http://www.ddj.com

99 bottles of beer -- One program in 1214 languages
http://99-bottles-of-beer.ls-la.net/

Computer Languages History
http://www.levenez.com/lang/

The Language List
http://people.ku.edu/~nkinners/LangList/Extras/langlist.htm

An Open Directory Project index
http://dmoz.org/Computers/Programming/Languages/

Your grade:	**6 Programs**	**45%**	continual thru semester
	Program #1 (working with bash scripts) due Tuesday, Sept 09		
	Program #2 (head and tail in C) due Tuesday, Sept 16		
	Program #3 (web spiders in C) due Tuesday, Sept 23		
	Program #4 (recursive decent parser in C++) due Tuesday, Oct 07		
	Program #5 (yahooNews in Java): due Thur Oct 30		
	Program #6 (Lisp): due Thur Nov 13		
	2 Take Home Exams	**30%**	(15% each exam)
	#1 due Fri, Oct 17th in class		
	#2 due Fri, Nov 7th in class		
	Language Team	**25%**	TBA (last weeks of classes)
	Day #1 Intro to the language and lab		5%
	Day #2 Walk thru/demo/quality of final app		20%

<u>Honor Code Revisited</u>: **It goes without saying that all programming assignments and take-home exams will be the student's own, in keeping with the Wheaton Honor Code.** In labs, you may get help from fellow classmates, but remember that **all submitted work must be your own.** Again, **the programs and exams must be your own from beginning to end**. The exception of course is the final Team lecture/lab and team-developed application.

Week	Topic
1	**Aug 27/29** **History** of Programming Languages *Homework #1 due Wed, Sept 03 in class at the start of class*
2	Monday, Sept 01 Labor Day
2	**Sept 03/05** *So, your team language is .... ?* **bash** and UNIX *Program #1 (working with bash scripts) due Tuesday, Sept 09*
3	**Sept 08/10/12** **C** and UNIX *Program #2 (head and tail in C) due Tuesday, Sept 16* **Describing Syntax and Semantics** – intro to formal languages and grammars
4	**Monday Sept 15** Visit by alum Ken Aspeslagh ’00 and co-founder of ecamm network (http://ecamm.com) In lecture (10:30 – 11:20) Discover the **Objective-C** programming language and see how it has become the cornerstone of Mac and iPhone application development. Learn the basics, and then witness firsthand how Objective-C, combined with Apple's rich frameworks, enables rapid Application development on OS X. We'll compare and contrast Objective-C with C++, pointing out the advantages and pitfalls of each language. 3:30 - 5:00 (A102) “Apple’s Secret Sauce” Learn why visionary Steve Jobs, the man who brought you the iMac, iPod, and iPhone, chose Objective-C as the native tongue for application development on OS X. By combining the muscle of object-oriented programming with the ease of use you'd expect from Apple, Objective-C and Cocoa make programming fun again.
4	**Sept 17/19** *Program #3 (web spiders in C) due Tuesday, Sept 23* **Describing Syntax and Semantics** continued … **Compilers** BNF form, parse trees, syntax graphs

5	**Sept 22/24/26** Special Event: **Wed, Sept 24** Paul Karasik lecture: The Language of Comics” 4:00 - 5:00 May Room, Mary Lyon **EBNF** ambiguous grammars, recursive descent parsing **Designing a grammar:** Together, we design a language **Intro to lex**
6	**Sept 29 Oct 01/03** **Recursive Decent Parser** *Program #4 (recursive decent parser in C++) due Tuesday, Oct 07* **Intro to yacc** **Special Event: Norman W. Johnson Lecture Series** Thur, Oct 2nd 5:30pm Norman W. Johnson, Wheaton Professor Emeritus of Mathematics and the honoree of this lecture series, will deliver this year's address entitled “Symmetry.” You can read more about Norman at his wikipedia site.
7	**Oct 06/08/10** **Object-Oriented Design** *Take Home Exam I (due Fri, Oct 17 in class)*
8	Mon Oct 13 – Tue Oct 14 – **Fall Break**
8	**Oct 15/17** **Object Syntax** classes, methods, messages, case studies (C++ *vs*. C# *vs*. Java) **Inheritance/ Software Reuse**
9	**Oct 20/22/24** **Polymorphism** *Program #5 (yahooNews in Java): due Thur Oct 30*

10	**Oct 27/29/31** *Take Home Exam II due Fri, Nov 07* *<buffer week in the schedule>* Wrap up OOP? Perl?
11	**Nov 03/05/07** **Functional Programming in Lisp** *Program #6 (Lisp): due Thur Nov 13*
12	**Nov 10/12/14** **Logic/Declarative Programming in Prolog**
13	**Nov 17/19** **Team I**
13-14	**Nov 21/24** **Team II**
15	**Dec 01/03** **Team III lab** **Dec 05** Evaluations

PHILOSOPHY of the COURSE

This course will run very much like a graduate school course. That is, the pace will be quick, the workload heavy, a strong dose of introductory theory applied throughout, and an **air of professionalism** maintained by both the instructor and the student. Said differently, the instructor places high expectations on your participation in class and equally high expectations on the amount of *independent* effort you apply. NOTE: *The exact pages to be assigned for reading will be given in lecture.*

PROGRAMMING HOMEWORK

It is expected that you spend *at least* **2 hours** on reading and practice problems for every one hour of lecture. This computes to *at least* **6 hours of work in the text(s) per week**. This course is *not* just about learning new languages. Rather the focus is on the theoretical basis for programming languages and on systems and object-oriented languages in particular. Let there be no misunderstandings: you *must* spend time in the texts and associated articles -- reading, re-reading. Programs must be electronically submitted by 5am of the day after the due date (yes, you get 5 extra hours after midnight of the due date if you need and are willing to stay up). A program beyond 5am and one day late will be docked 10%. No programs can be submitted after more than one day late. Of course, one day late extends for 24 hours after the 5am deadline. Assume this deadline is fixed.

TAKE HOME EXAMS

There are two take home exams during the semester. Again, you may *not* work together on take home exams.

Language TEAMs

Lecture/Lab

You will join two or three other students (I believe there will be 2 groups of two, 1 group of three) to lead two 50-minute meetings on a unique language that has not already been covered in class (e.g., R, PHP, Ruby, Visual Basic, Scheme, etc). Milestones for selecting your language and preparing for these labs will be requested throughout the semester. In the first meeting, you will lead the other students and the instructor at the whiteboard and with handouts, followed by a *series* of "hello world" hands-on programming examples. This implies that you must have at least enough computers installed with the necessary language and programming environment, that is, you are responsible for setting up the lab before your initial session. In the second meeting you will demonstrate a semi-large project that you jointly developed over the semester. Some of the time in this second meeting will be spent in a walkthrough of some of the software written. For each of the two meetings, I will allot your group a certain number of "presentation points" and then the group members will decide on how to share the points. For example, a very strong lab between a pair of students might be assigned 180 points. Between them, they might decide one person took the lead so they agree to split the points 94 and 86. You are expected to deliver a professional lab. It will be clear to everyone in the audience if (i) you have rehearsed the order of materials to present; (ii) if the examples introduced show an adequate exposure to your language; and (iii) if you have set up the Desktops of participants with code examples and environments that are ready to run/modify.

Team App

Your small group (the same group that does the lab together) will produce one significant application you have developed over the semester. This should dovetail very well with your preparation of your hands-on lab since the material you learn while building your app can be shared with the other students. Again, this is a *significant* software effort to take place over the entire semester, not something that will be done in one week. Like the presentation, points will be assigned to the group as a whole and the group will decide how those points should be distributed to individual members.