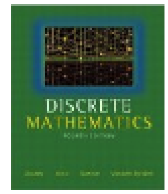
syllabus for

foundations of computing theory

computer science **COMP 121**

Instructor: Mark LeBlanc (**mleblanc**)	Office Hours: by appt. *or* MON, TUE 10-11, WED 11-12	
Office: SC-1322		
Phone: 286-3970 (on campus: x3970)	Meeting: MW 12:30 – 1:50, Rm. **SC B234**	

Texts (*selected, xeroxed pages from multiple texts in a binder and online sources*)



Discrete Mathematics (4th Ed.) by Dossey, Otto, Spence, and Vanden Eynden. Addison-Wesley, Boston, MA, 2002.

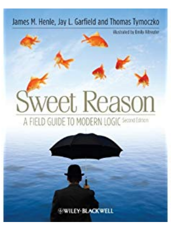
Sweet Reason – A Field Guide to Modern Logic by Henle, Garfield, and Tymoczko. Wiley-Blackwell, 2012.

Introduction to the Theory of Computation (2nd Ed.) by Sipser. Course Technology, 2005.

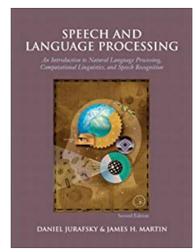
Speech and Language Processing (2nd Ed.) by Jurafsky and Martin. Pearson Prentice Hall, 2008.

Content:

Discrete mathematics represents the language, symbolic notation, and problem solving principles that lead to a rich appreciation of computing. This course is an initial semester of exposure to the tools for precise vocabulary, powerful notation, useful abstractions, and rigorous thinking that is needed as



someone works in computing. And just who does not work with computing these days in one way or another? A working premise of the course is that it is not possible to make excellent and effective use of computers without involving oneself in mathematical considerations. It seems everyone these days wants to apply computers to the problem at hand, but very few have experience with the fundamental mathematical principles to ensure that things are done correctly and efficiently. Simply put, someone in your group has to know with certainty that an answer is wrong or that a task could be performed more efficiently! This course provides practice with an introduction to symbolic expressions, abstractions, and mathematics that enables *you* to be that person.



> *"As the field of computer science matures, more and more sophisticated analysis techniques are being brought to bear on practical problems. To understand the computational techniques of the future, today's students will need a strong background in discrete structures."* (Computing Curricula).

Curriculum: Many areas of computing require an ability to work with discrete mathematical structures. Most of the material covered in this course serves as an initial exposure to and practice with the discrete mathematical topics that appear in later computer science courses. In addition to satisfying the Quantitative Analysis (QA) general education requirement, this course counts toward a computer science minor and major. Computer science majors who enjoy the theory and/or plan to go to graduate school in computing should also plan to later enroll in MATH 211 for a deeper exposure to writing proofs, counting, and graph theory.

Your grade:	In class participation	6%	attendance and participation required
	10 Homeworks	50%	continual throughout the semester
	Visual Trumpery Talk	(points given)	Friday, Feb. 15, 2pm, in Hindle Auditorium
	Exam1	12%	Wednesday, Feb 27th, in class
	Exam2	12%	Monday, Apr 8th, in class
	Final Exam	20%	**Thursday, May 9th, 2pm** (room TBD)

"In computer science, if you are almost correct you are a liability."
Fred Kollett (1941-1997), Math/CS, Wheaton College

Week	Open Questions	Reading Exams	Topics
1 Jan. 23	So you can recite the powers of 2 by memory, right? So red is `0xFF0000`, correct? What is the largest possible value I can store in a memory location on this chip?	websites	**Number systems** Binary/Octal/Hexadecimal HTML colors **Compiling source code to …**
2 Jan. 28 Jan. 30	How can we describe this situation with propositional statements? How can we use boolean algebra to find design flaws in our software? How should we document our functions so that others can understand the logic in our software?	Dossey *et al*. Appendices A.1 A.2 Henle *et al*.	**Logic** Statements Equivalence Negation with quantifiers **Tarsky's World** **Formal Methods** Algorithm documentation PRE/POST conditions Loop invariants
3 Feb. 04 Feb. 06	How do tiny embedded microprocessors control larger machines based on a set of inputs?	Dossey *et al*. 9.1 9.2	**Circuits** Logic gates Boolean algebra
4 Feb. 11 Feb. 13	How much do we spend on coffee and candy a day? How can we store the data points for a cube? How can we rotate the cube?	Dossey *et al*. Appendix B Handouts, websites, and notes	**Matrices** Matrix operations Representing and moving objects in 2-space
5 Feb. 18 Feb. 20	Hey, the relational database model is based on set theory and first order predicate logic, right?	Dossey *et al*. 2.1, 2.2, 2.3 Appendix B	**Sets, Relations, and Databases** Set Operations Equivalence Relations Matrices of Relations Digraphs of Relations
6 Feb. 25 Feb 27	How can we leverage this math to help us design efficient queries to databases?	websites	**Relational Databases** Intro to SQL **Exam I**
7 Mar 04 Mar 05	How can we store our huge graph in the computer?	Dossey *et al*. Appendix B 3.1, 3.2, 3.3	**Graphs** Notation Data Structures for Graphs Adjacency Matrix and Adjacency List
8 March 11-15	**Spring BREAK**		

Week	Open Questions	Reading Exams	Topics
9 Mar 18 Mar 20	What is the shortest path between cell towers to transmit a wireless message across the country? How can we visit all nodes on the graph?	Dossey *et al.* Appendix B 3.1, 3.2, 3.3	**Paths** and Circuits in Graphs Shortest-path, Breadth-First Search
10 Mar 25 Mar 27	So we know that graphs that are connected and have no cycles are Trees... How can we help seven farms in Iowa build a communications network to relay storm information with the minimum number of expensive fiber optic lines?	Dossey *et al.* 4.1, 4.2, 4.3 4.4, 4.5, 4.6 Notes	**Trees** Notation Spanning Trees **Algorithms** (on Trees) Depth-First-Search Binary Trees
11 Apr 01 Apr 03	How might I store all the words in a dictionary to facilitate fast lookup? How many moves should my computer game “look ahead” when playing in expert-mode?	Websites Notes	**More Tree algorithms** **Game Trees**
12 Apr 08 Apr 10	How can we find all the six or seven letter palindromes in a dictionary?	Jurafsky and Martin; LeBlanc and Dyer	**Regular Expressions** **Exam II**
13 Apr 15 Apr 17	How do those vending machines work? What is the syntax for a legal variable name in our programming language? What do we use XML for?	Dossey *et al.* 9.4 Sipser and notes	**Finite State Machines** **Automata** **Languages and Grammars** Chomsky hierarchy Context-free grammars, BNF Lexical analysis XML
14 Apr 22 Apr 24	If we use our recursive algorithm, how many arithmetic operations will be required? Can we find a closed form for that recurrence relation? Can we prove it is correct?	Dossey *et al.* 8 Notes Dossey *et al.* 2.6	**An Introduction to Recursion** Counting instructions **Recurrence relations** **Proof by Induction**
15 Apr 29 May 01	So we know our program must deal with *really* large number of data items, how can we compare the rates of growth of two algorithms? How can we fit multiple cubic and quadratic polynomials together to approximate a data set?	Handouts and notes	now for something continuously different … **Differential Calculus** Functions Rates of growth Algorithm efficiency Algorithm analysis, “Big Oh” Spline curves
Final		**Final Exam**	**Thursday, May 9th, 2pm**

Exact pages to read and homework exercises to be submitted will be assigned in lecture.

Honor Code Revisited: It goes without saying that all submitted work will be the student's own, in keeping with the Wheaton Honor Code. For homework, all work must be your own from beginning to end.

Get a 3-ring Binder (I have some free binders near my office; come by and snag one)
I will pass out many(!) handouts to facilitate note taking in class. Please keep an organized binder.

Homework solutions *must* show all your work. Let me say that more directly: do not just submit a homework exercise that shows only your answer. You will *not* get credit for homework problems that do not show *all* your work.

Homework solutions *must* be neat! I know you do not give your English professors "hen-scratch" when you write a paper. No, you write drafts, edit, print, correct, print, and submit a neat final draft. I expect the same in your homework submissions. As you initially work on the homework, do not concern yourself with how things look, in fact, you should have multiple sheets of scrap paper about as you work on a solution. **BUT**, once you are finished, **you must professional format all your work** using a word processor of your choice (e.g., use MS Word, LaTeX, etc**.** Your homework should be formatted as professionally as you would expect to see in a textbook. So, I reserve the right to deduct points for sloppy submissions or submissions that are not stapled together, even if the answers are correct. Again: all homeworks require typed, formatted solutions.

Accommodations for Disabilities *Wheaton is committed to ensuring equitable access to programs and services and to prohibit discrimination in the recruitment, admission, and education of students with disabilities. Individuals with disabilities requiring accommodations or information on accessibility should contact Autumn Grant, Associate Director Accessibility Services at the Filene Center for Academic Advising and Career Services. ~ grant_autumn@wheatoncollege.edu or (508) 286-8215 ~*

Learning Goals

(0) To apply abstraction and decomposition when attacking a large complex task
(1) To model relevant aspects of a problem to make it tractable
(2) To read and explain algorithms for solving computational problems
(3) To identify an appropriate symbolic notation to represent a problem at hand
(4) To simplify and solve problems by manipulating symbolic notation
(5) To recognize how discrete math topics permeate the discipline of computing

You "earn" your grade in this course by showing mastery of these learning goals as they appear within and are applied to current topics in computing. In addition to numerous opportunities to "play with the math" in class, your level of mastery is directly dependent on your time on task outside of class. Of course, you will be graded on homeworks and exams, but you should view each submission of homework as the culmination of hours of *practice*.

The analogies of an athlete or musician are appropriate. You know when you watch a game or listen to a concert, much practice has occurred prior to the game or performance. In fact, most performers would *never* attempt to perform without hours and hours of practice. Is practice glorious? Does anyone cheer? Nope. Yet, returning to homework, **it is sometimes the case that a student thinks that one attempt at a problem is sufficient for submission; why is that?** *Think about it*. The spectators in the audience *know* when they see a poorly prepared performer. Your professor (*a.k.a.* your future boss) is in the audience. Be disciplined. Avoid ambiguity. Impress me.

Visual Trumpery – a talk by Alberto Cairo [www.trumperytour.com]

Friday, February 15, 2 – 4pm, Science Center, Hindle Auditorium

The English word "trumpery" means worthless nonsense, something that is showy and deceitful at the same time. Trumpery can occur in text, verbally, or visually. This non-partisan talk focuses on the visual, examining misleading charts, graphs, and data maps designed by individuals and organizations from all over the political spectrum. Cairo will use these examples to equip you with a solid understanding of "graphicacy," the word he uses to refer to visual literacy. He believes a literate, numerate, and graphicate citizenry is the best antidote for a world where trumpery runs rampant.

Attendance is required; points will be awarded to your overall homework grades. Please see me if you have a conflict and we will arrange for you to attend an additional "data-related" event.