EECS 101:

An Introduction to Computer Science for Everyone

Syllabus

Web Page

www.nucs101.org

What roles this class can play for you

- For CS and CIS majors and minors: EECS 101 is a *required core course* in the new curriculum (all new majors starting September, 2007).
- For Weinberg students: EECS 101 satisfies the *area III (social and behavioral sciences) distribution requirement.*
- For everyone: This course explains the field of computer science

Instructor

Peter A. Dinda
Technological Institute, Room L463
pdinda@northwestern.edu
Office hours: Thursdays, 1:30-3:30pm or by appointment

Teaching assistant

J. Scott Miller
Ford Motor Engineering Design Center ("FoMo") 2-221
jeffrey-miller@northwsestern.edu
Office hours: Wednesdays, 1-4pm or by appointment

Undergraduate assistants

Eugenia Gabrielov, EugeniaGabrielov2010@u.northwestern.edu
Patrick Shankland, p-shankland@northwestern.edu
Office hours and location will be announced later

Location and Time

Lecture: Tuesdays and Thursdays, 3:30-4:50pm, Tech L251

Undergraduate Assistant-led Recitation/Discussion
Location and time to be announced later

Prerequisites

None
All students at Northwestern are welcome
No prior knowledge of Computer Science is needed
There is no programming in this course

Who Should Take This Class

- All students at Northwestern who are interested in learning about what computer science is, what computer scientists do, and how both affect the world from the practical to the esoteric.
- Freshmen and those who are thinking about the pursuing the computer science majors/minors in either McCormick or Weinberg.
- Computer Science and Computer Engineering students, who should become familiar with the breadth of the intellectual endeavor of the field.
- Computer Science students on the new curriculum (this course is required)
- Weinberg students to satisfy the area III distribution requirement

What This Class Is About

The primary goal of this course is to answer these simple questions:

- "What is Computer Science?"
- "What do Computer Scientists do?"
- "How does Computer Science interact with the rest of the world?"

The very ubiquity of the *products* of Computer Science has sadly not been coupled with a corresponding growth in the understanding of the *intellectual content, structure, history, and aspirations* of the field, or of what the people in the field actually think and do. Many people might answer the questions by saying that Computer Science is "programming" or "coding", that Computer Scientists are programmers or manage programming projects, and that the field interacts mostly by creating products. While the art of computer programming is important and can be joyful, Computer Science encompasses much more. Here are just a short selection of the kinds of questions and issues that Computer Science grapples with that we will touch on in this class:

- How can we make it possible for human beings to design and build ever more complex artifacts? Computer software is among the most complex creations of human beings and is at the bleeding edge of what we understand how to design and build? How do we manage its complexity?
- How do we sustain the exponential growth of computing power, storage capacity, and communications capacity? How do we exploit it?
- What are the mathematical and physical limits to computation and communication? What *can't* computers do? What *is* computation? What *is* communication?
- What kinds of computational problems are there? How does the universe limit how fast they can be solved?
- What kinds of computers are there? Are some fundamentally more powerful than others?
- What kinds of languages are there? Can we find one that's equally adept at getting things done on a computer and explaining how it's done to a person?
- How can we translate from one language to another? How is this problem different for human languages and computer languages? Do answers for one have influence on the other?
- How can computers and human beings best interact?
- Is software creation an art, a science, or an engineering discipline?
- Can we make a computer learn by itself instead of programming it?
- Can we, and how can we, make a mind? How can we tell that we're successful?
- Is intellectual property theft? What are the implications to society of the fact that a computer can process any information and making copies of information is free? What are the implications of the Free (as in Freedom) software movement?
- What are the limits to security and privacy that can be achieved computationally? What implications do these fundamental limits have for politics and law?
- Is computation based on physics or is physics based on computation?
- Is the universe a simulation?

Reading

In addition to introducing questions and concepts, the readings are also intended to present the personalities and history that are involved, and to provoke discussion and debate. Computer Science is a dynamic intellectual endeavor and many core intellectual questions are still open.

A separate reading list shows all the sources the class draws on.

You should purchase the following books. It's cheapest if you buy the paperback versions. Additional materials will be handed out in class or provided via the

web. Note that **we will read only selected portions of these books**, but you should feel free to dive deeper into these books and the other materials listed on the reading list.

- David Harel, *Computers Ltd: What They Really Can't Do*, Oxford University Press, 2003.
- Martin Davis, *The Universal Computer: The Road from Leibniz to Turing*, W.W. Norton and Company, 2000.
- Paul Graham, *Hackers and Painters: Big Ideas from the Computer Age*, O'Reilly Media, 2004.
- Lawrence Lessig, *Code: Version 2.0*, Basic Books, 2006. This book is also available for free online at http://codev2.cc/ (Creative Commons license)
- Richard Stallman, *Free Software, Free Society: Selected Essays of Richard M. Stallman*, Free Software Foundation, 2002. This book is also available for free online at http://www.gnu.org/ (GNU Public License)
- Steven Levy, *Hackers: Heros of the Computer Revolution*, Penguin, 2001.
- Stephen Wolfram, *A New Kind of Science*, Wolfram Media, 2002. This book is also available for free online at http://www.wolframscience.com/
- Simon Singh, *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor, 2000
- Ray Kurzweil, *Are We Spiritual Machines? Ray Kurzweil vs. the Critics of Strong AI*, Discovery Institute, 2002. This book is also available for free online at http://www.kurzweilai.net
- Charles Petzold, *Code: The Hidden Language of Computer Hardware and Software*, Microsoft Press, 2000.
- Eric Raymond, *The Cathedral and the Bazaar: Musing on Linux and Open Source by an Accidental Revolutionary*, O'Reilly, 2001. Also mostly available for free from http://www.catb.org/~esr/writings/cathedral-bazaar/
- Scott Rosenberg, *Dreaming in Code: Two Dozen Programmers, Three Years, 4732 Bugs, and One Quest for Transcendent Software*, Crown, 2007.

It's a lot more fun if students discuss and debate the readings! There is a discussion group that is accessible from the course web page, and students are also strongly encouraged to participate in the recitation section.

Papers

During the course of the quarter students will pair up (i.e., work in teams of two people) to write essays. The first essay will be on a personality in the history (or present) of computer science. The second essay will be on a computer science topic, such as one of the questions given above.

The intent of these essays is threefold. First, the student will get a view of computer science from two distinct cross-cutting points of view. Second, she will have the opportunity to dive more deeply into particular aspects of the field. Third, students will have the opportunity to do research and write as a team.

Exams

There will be a midterm and a final. The final will not be cumulative.

Grading

40 % Term papers
20 % Class participation informed by reading.
This includes participation in the recitation section and the online discussion group
20 % Midterm
20 % Final

Schedule

Week 1 Introduction

The long arc from the Antikythera mechanism to today
Historical roots of modern computer science in mathematics, electrical engineering, and psychology
A look at the big questions
Structure of the field (the core areas)
What computer scientists do (how people are employed)

Reading

- Jeannette Wing, "Computational Thinking", *Communications of the ACM*, Volume 49, Number 1, January 2006, pp. 33-35. http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.pdf
- Bernard Chazelle, "Could Your iPod be Holding the Greatest Mystery in Modern Science?", *Math Horizons*, April, 2006. http://www.cs.princeton.edu/~chazelle/iPod
- Vannevar Bush, As We May Think, *The Atlantic Monthly*, July, 1945. http://www.theatlantic.com/doc/194507/bush
- Wikipedia, "The Antikythera Mechanism", http://en.wikipedia.org/wiki/Antikythera_mechanism
- "Hackers and Painters", Chapter 2 of Paul Graham, *Hackers and Painters*, O'Reilly, 2004.
- http://money.cnn.com/magazines/moneymag/bestjobs/2006/
- Preamble from David Harel, *Computers Limited: What They Really Can't Do*, Oxford University Press, 2000.

Week 2 Fundamental theories – The core ideas and how they are evolving

Discrete mathematics and logic – set theory and logic, not calculus, is the basis of computer science
Theory of computation and its surprising facts
Theory of communication and its ubiquity
Classical computers
The "laws": Moore, Gilder, Murphy

Reading

- Chapters 1-2, from David Harel, *Computers Limited: What They Really Can't Do*, Oxford University Press, 2000.
- Chapters 1, 6, 7 from Martin Davis, *The Universal Computer: The Road from Leibniz to Turing*, W.W. Norton and Company, 2000.

Week 3 Algorithms and data structures – Solving problems, Manipulating Data

Note: Instructor and TA will probably be out of town on Tuesday. There will probably be a guest lecture.

Tractability of problems
Algorithms
Data structure concept
Simulation and its impact

Reading

- Chapters 3-5, from David Harel, *Computers Limited: What They Really Can't Do*, Oxford University Press, 2000.
- Chapter 8 from Martin Davis, *The Universal Computer: The Road from Leibniz to Turing*, W.W. Norton and Company, 2000.

Week 4 Computer Systems – Raising the abstraction

Architecture
Operating systems
Databases
Networking
Middleware

Videos

- ARPANET Video (origins of the Internet) (http://www.newmediamedicine.com/blog/2006/08/16/arpanew-video/) (Watch on your own)
- Possible demos of assembler, Linux Trace Toolkit and Ethereal. (In Class)

Reading

- Chapters 2, 3, 10, 11, 18, 22 from Charles Petzold, *Code: The Hidden Language of Computer Hardware and Software*, Microsoft Press, 2000
- http://en.wikipedia.org/wiki/Operating_system
- http://en.wikipedia.org/wiki/Middleware
- http://en.wikipedia.org/wiki/Database_management_system (skim)
- http://en.wikipedia.org/wiki/Internet (skim)

Week 5 More systems, and languages and software engineering

Compilers
Software Engineering
Mathematical thinking about grammar and semantics
Computation as understanding languages
Formal language design
Programs and proofs
Language design and implementation in practice
and its convergence with theory

Reading

- Chapter 24 from Charles Petzold, *Code: The Hidden Language of Computer Hardware and Software*, Microsoft Press, 2000
- History of Programming Languages (chart and content) http://www.levenez.com/lang/ (skim)
- Programming Languages (Wikipedia article) : http://en.wikipedia.org/wiki/Programming_language and http://en.wikipedia.org/wiki/History_of_programming_languages (skim)
- Chapter 1, 2, 3, and 10 from Scott Rosenberg, *Dreaming in Code: Two Dozen Programmers, Three Years, 4732 Bugs, and One Quest for Transcendent Software*, Crown, 2007.
- "Programming Languages Explained", Chapter 10 of Paul Graham, *Hackers and Painters*, O'Reilly, 2004.
- http://en.wikipedia.org/wiki/Compiler

Week 6 Human Computer Interaction and Graphics

Applied geometry on a computer
Issues in interface design
Psychology of using computers
Where your PC came from, and where it might go…

Videos

- SIGGRAPH Video Review, Issue 137, *The Story of Computer Graphics*, Video, 1999. (We may watch in class)

- Doug Engelbart Demo. http://sloan.stanford.edu/MouseSite/1968Demo.html (Watch on your own – OK to skim)
- Alan Kay Video "Doing With Images Makes Symbols": http://video.google.com/videoplay?docid=-533537336174204822 (Watch on your own – great history)

Reading

- Chapter 6 from Scott Rosenberg, *Dreaming in Code: Two Dozen Programmers, Three Years, 4732 Bugs, and One Quest for Transcendent Software*, Crown, 2007.
- "Taste For Makers", Chapter 9 of Paul Graham, *Hackers and Painters*, O'Reilly, 2004.

Week 7 Artificial Intelligence – Making minds and solving problems we don't know how to solve

Turing test
Are people fancy computers?
Logic-based approaches
Heuristic search
Vision and image understanding
Robotics
Machine learning – statistics as if computers existed

Videos

- Koza Genetic Programming Example (In Class)

Reading

- Chapter 7 from David Harel, *Computer Limited: What They Really Can't Do*, Oxford University Press, 2000.
- Chapter 9 from Martin Davis, *The Universal Computer: The Road from Leibniz to Turing*, W.W. Norton and Company, 2000.
- John McCarthy, "*What is Artificial Intelligence?*", http://www-formal.stanford.edu/jmc/whatisai/
- Chapters 2+5 (John Searle + Rebuttal) and Chapters 5+9 (Thomas Ray + Rebuttal) : Selections from Ray Kurzweil, *Are We Spiritual Spiritual Machines? Ray Kurzweil vs. the Critics of Strong AI, Discovery Institute*, 2002. Also available for free on Kurzweil's site: http://www.kurzweilai.net/ (http://www.kurzweilai.net/meme/frame.html?main=/meme/memelist.html?m%3D19)

Week 8 Security and a bit of Law and Politics

Cryptography

Cracking the uncrackable
Secure protocols and their magic
Intrusion detection

Reading

- Chapters 4,6,7 from Simon Singh, *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor, 2000
- Chapter 6 from David Harel, *Computer Limited: What They Really Can't Do*, Oxford University Press, 2000.
- Tadayoshi Kohno, Adam Stubblefield, Aviel Rubin, and Dan Wallach, *Analysis of an Electronic Voting Machine*, IEEE Symposium on Security and Privacy, 2004. (available online. It is sufficient to skim this)
- Feldman, Halderman, and Felten, Security Analysis of the Dielbold AccuVote-TR Voting Machine. (available at http://itpolicy.princeton.edu/voting/ - it is sufficient to skim this)

Week 9 Politics, Law, Culture, Entrepreneurship

Hacker culture
Startup culture
The Free Software and Open Source Movements
Code as Law (ex: DRM)

Reading

- Essays 1 ("The GNU Project"), 2 ("The GNU Manifesto"), 4 ("Why Software Should Not Have Owners"), and 16 ("The Danger of Software Patents") from Richard Stallman, *Free Software, Free Society: Selected Essays of Richard M. Stallman*, Free Software Foundation, 2002. This book is also available for free online at http://www.gnu.org/ (GNU Public License)
- Chapter 1 ("Code is Law"), Chapter 3 ("Is-Ism: Is the Way it is the way it must be?"), and Chapter 10 ("Intellectual Property") in Lawrence Lessig, *Code: Version 2.0*, Basic Books, 2006. This book is also available for free online at http://codev2.cc/ (Creative Commons license)
- The Sony Root Kit Scandal: http://en.wikipedia.org/wiki/2005_Sony_BMG_CD_copy_protection_scandal (Skim)
- Chapter 2 ("The Hacker Ethic") from Steven Levy, *Hackers: Heros of the Computer Revolution*, Penguin, 2001. [The Epilogue ("The Last Hacker") on Richard Stallman is also a good read]
- Title essay from Eric Raymond, *The Cathedral and the Bazaar: Musing on Linux and Open Source by an Accidental Revolutionary*, O'Reilly, 2001. Also mostly available for free from http://www.catb.org/~esr/writings/cathedral-bazaar/

- "The Other Road Ahead" and "How To Make Wealth", Chapters 5 and 6 of Paul Graham, *Hackers and Painters*, O'Reilly, 2004.

Week 10 The bleeding edge and crazy ideas / slack time

Quantum computing
Digital Physics
Biological computing with DNA
Smart Dust
More

Videos

- Play Conway's Game of Life: http://www.bitstorm.org/gameoflife/

Reading

- Chapters 1 and 2 from Stephen Wolfram, *A New Kind of Science*, Wolfram Media, 2002. This book is also available for free online at http://www.wolframscience.com/
- Rule 110 (Wikipedia Article): http://en.wikipedia.org/wiki/Rule_110_cellular_automaton
- Conway's Game of Life http://en.wikipedia.org/wiki/Conway's_Game_of_Life
- Nick Bostrom, *Are You Living in a Computer Simulation?*, Philosophical Quarterly, Volume 53, Number 211, 2003. (Available online at http://www.simulation-argument.com)
- Chapter 8 from Simon Singh, *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor, 2000
- … More TBD depending on available time / slack