CS 15-251 Great Theoretical Ideas In Computer Science

Spring 2004 **www.cs.cmu.edu/∼15251**

This course will take a philosophical and historical perspective on the development of theoretical computer science. The technical material will be self contained, assuming no specific knowledge beyond high school algebra and high school programming.

From a pile of stones to represent and manipulate numbers, humans have progressively developed an abstract vocabulary with which to mathematically represent their world. The ancients, especially the Greeks, realized that they could consistently reason about their representations in a step by step manner. In other words, by computing in abstract models, they could describe and predict patterns in the world around them.

Starting with ancient algorithms for arithmetic, we will revisit the development of mathematics from a computational point of view. Conversely, we will mathematically study the nature of computation itself. What is computation? What is computable, in principle? What is especially easy, or especially hard to compute? To what extent does the inherent nature of computation shape how we learn and think about the world?

Topics will include: representations of numbers, induction, ancient and modern arithmetic, basic counting principles, probability, number theory, the idea of a proof, formal proofs, logic, problem solving methods, polynomial representations, automata theory, cryptography, infinity, diagonalization, computability, time complexity, incompleteness and undecidability, random walks, and Kolmogorov/Chaitin randomness.

Weekly Schedule

LECTURE:	TR 3:00-4:20p	Doherty Hall 2315			
Section A:	M 10:30-11:20p	SH 220	Section D:	M 1:30-2:20p	OSC 200
Section B:	M 11:30-12:20p	SH 220	Section E:	M 2:30-3:20p	SH 224
Section C:	M 12:30-1:20p	SH 220	Section F:	M 3:30-4:20p	WeH 8427

Course Staff

Name	Office	Phone	Email	Office hours
Professors:				
Steven Rudich	Wean 7128	268-7885	rudich@cs.cmu.edu	TBA
Anupam Gupta	Wean 4109	268-7127	anupamg@cs.cmu.edu	TBA
Teaching Assistants:				
Andrew Gilpin	Wean 8th floor couches	268-1405	gilpin@cs.cmu.edu	M 8-10P
Brendan Juba	Wean 8th floor couches	862-2875	bjuba@andrew.cmu.edu	T 8-10P, R 7-9P
Susmit Sarkar	Wean 8th floor couches	268-2993	susmit@cs.cmu.edu	F 7-9P
Bella Voldman	Wean 8th floor couches		bvoldman@andrew.cmu.edu	T 4:30-5:30 & 7-10P, R 4:30-5:30P
Adam Wierman	Wean 8th floor couches	268-3056	acw@cs.cmu.edu	M 7-10P
Yinmeng Zhang	Wean 8th floor couches		ynz@andrew.cmu.edu	WR 7-9P
Secretary:				
Charlotte Yano	Wean 7120	268-7656	yano@cs.cmu.edu	

Spring 2004 **Lecture/Work Schedule**

Week	Date	Day	Lec.	Topic	Available	Due
1	Jan 13	T	1	Pancakes with a Problem!	Hwk 1	
	Jan 15	R	2	Choose Your Representation!		
2	Jan 19	M	**Martin Luther King Day**: no classes noon onwards			
	Jan 20	T	3	Unary, Binary, and Beyond	Hwk 2	Hwk1
	Jan 22	R	4	One Step at a Time: Induction		
3	Jan 27	T	5	On Raising a Number To a Power	Hwk 3	Hwk2
	Jan 29	R	6	Euclid's Great Recursive Algorithm For GCD		
4	Feb 3	T	7	Fibonacci Numbers: An Unexpected Formula	Hwk 4	Hwk3
	Feb 5	R	8	Modular Arithmetic and the RSA Cryptosystem		
5	Feb 9	M	**Quiz 1**			
	Feb 10	T	9	Counting I: One To One Correspondence and Choice Tree Representation	Hwk 5	Hwk4
	Feb 12	R	10	Counting II: Recurring Problems and Correspondences		
6	Feb 17	T	11	Counting III: Polynomials Count! Pascal's Triangle	Hwk 6	Hwk5
	Feb 19	R	12	The One Minute To Learn Programming Language: Finite Automata		
7	Feb 24	T	13	Playing Symbol Games: Logic, Language, and Meaning	Hwk 7	Hwk6
	Feb 26	R	14	Problem Solving: The Method Behind the AHA!		
8	Mar 2	T	15	On Time Versus Input Size		Hwk7
	Mar 4	R	16	Grade School Revisited: How to Add and Multiply		
	Spring Break					
9	Mar 16	T	17	Grade School Again: A Parallel Perspective	Hwk 8	
	Mar 18	R	18	Probability I: Counting in Terms of Proportions		
10	Mar 22	M	**Quiz 2**			
	Mar 23	T	19	Probability II: Probability Pitfalls and Paradoxes	Hwk 9	Hwk8
	Mar 25	R	20	Counting, Naming, and Worst-Case Compression		
11	Mar 30	T	21	Dating Theory: Who Wins the Battle of the Sexes?	Hwk 10	Hwk9
	Apr 1	R	22	Probability III: Random Variables and Great Expectations!		
12	Apr 6	T	23	Probability IV: Event Space In An Infinite Choice Tree	Hwk 11	Hwk10
	Apr 8	R	24	Probability V: Random Walks.		
13	Apr 13	T	25	Thales's Legacy: What is a Proof?		Hwk11
	Apr 15	R	**Spring Carnival**			
14	Apr 20	T	26	Cantor's Legacy: Infinity and Diagonalization.	Hwk 12	
	Apr 22	R	27	Turing's Legacy: The Limits of Computation		
15	Apr 26	M	**Quiz 3**			
	Apr 27	T	28	Gödel's Legacy: The Limits of the Symbol Game		Hwk12
	Apr 29	R	29	Ancient Paradoxes With An Incompressible Resolution		

Important Course Information

Jan 13, 2004

Text

There is no required text for the course. The material is fairly diverse, and no standard text contains it. Copies of the slides used in the lectures will be handed out or made available on the web. If you must have a book, buy *Applied Combinatorics*, by Alan Tucker (available at Amazon.com) and/or *Discrete Mathematics And Its Applications*, by Kenneth H. Rosen. If you think that the absence of a real textbook in the course will conflict greatly with your learning style, you should consider taking Discrete Mathematics instead.

If you want to look at books which contain part of the course material, we recommend the following:

1. *Applied Combinatorics*, by A. Tucker, published by Wiley & Sons.
2. *Concrete Mathematics: A Foundation for Computer Science*, by R. Graham, D. Knuth, and O. Patashnik, published by Addison-Wesley.
3. *Introduction to Algorithms*, by T. Cormen, C. Leiserson, R. Rivest, and C. Stein, published by MIT Press.
4. *Discrete Mathematics and its Applications*, by K. H. Rosen, published by McGraw-Hill.
5. *Cryptography: Theory and Practice*, by D.R. Stinson, published by CRC Press.
6. *How To Solve It: A New Aspect of Mathematical Method*, by G. Polya, published by Princeton University Press.
7. *Programming Pearls* and *More Programming Pearls*, by J. Bentley, published by Addison-Wesley.
8. *Conceptual Blockbusting: A Guide to Better Ideas*, by J. L. Adams, published by W. W. Norton & Company.
9. *The Heritage Of Thales*, by W.S. Anglin and J. Lambek, published by Springer-Verlag.
10. *Proofs Without Words I and II (exercises in visual thinking)*, by Roger B. Nelson, published by The Mathematical Association Of America.
11. *The Book Of Numbers*, by John H. Conway and Richard K. Guy, published by Springer-Verlag.
12. *Aha! Gotcha (Paradoxes to puzzle and delight.)*, by Martin Gardner, published by Freeman Publishers.
13. *Proofs From The Book*, by Martin Aigner and Gunter Ziegler, published by Springer-Verlag.

Web Page, Bboards, and Mailing Lists

The class Web page will provide a wealth of up-to-date information about the course. The URL is

- `http://www.cs.cmu.edu/~15251` will contain the material for the course, and will reflect the course as it is given; this is the web site that you should read regularly.
- `http://www.discretemath.com` is currently inactive, but is expected to become active soon. It will be a repository of the latest material available, and may contain material over and above that in the above website.

`cyrus.academic.cs.15-251` is for staff announcements only;

`cyrus.academic.cs.15-251.discuss` is for general discussion among members of the class. Please keep discussion relevant and polite.

The Web page will include all assignments and on-line handouts, as well as facilities to check how you are doing in the course. You should visit it frequently.

The class will also have mailing lists: the list `cs-251@cs` mails to the entire class and teaching staff; `cs-251-A@cs`, `cs-251-B@cs`, `cs-251-C@cs`, `cs-251-D@cs`, and `cs-251-E@cs` mail only to the specified recitation sections (and their TAs). `cs-251-staff@cs` mails only to the teaching staff. Note that you usually stand a better chance of getting a response by mailing one staff member in particular, as opposed to `cs-251-staff@cs` (unless of course the one staff member you mail isn't around).

Homework Assignments and Exams

There will be problem sets, programming assignments, three quizzes, and a final exam. The quizzes will be given in your recitation sections.

We will drop your lowest homework grade and your lowest quiz grade.

The grading formula will be:

40% Homework (problem sets, programming assignments)

30% Quizzes

30% Final

This formula is a lower bound on your grade. If the class average as computed by this formula is lower than a 75, we will use a curve to raise your final grades. *Under no circumstances will we use a curve to lower your grades.* From past experience, we expect the formula to determine your grade exactly. 90 and above is A range, 80-89 is B range, 70-79 is C range, and so forth.

Notice that if it is your habit to put in marginal effort on the homework and to come through on the exams, your grade will suffer a bit. This is deliberate; we think the homework is important.

Your mid-term grade will reflect your anticipated final grade.

The Homework Guru

Each homework assignment will have a specific person indicated as the guru for that assignment. The guru will be expert on all the details of that particular assignment. Though any staff member might be able to answer your questions, the guru knows all.

Typesetting

You must typeset your solutions to the problem sets. This makes the graders' job much easier. We recommend that you learn and use LaTeX, but anything typed is fine. Both Microsoft Word and FrameMaker can typeset equations. Many former students have told me how helpful it was to learn LaTeX.

Submitting Homeworks

Homeworks will be submitted electronically. We will accept files in postscript (ps) or pdf format. To submit your homework #*N*, copy your homework file to:

- `/afs/andrew/scs/cs/15251/student/assignment`*N*`/handin/`*userid*

Please see the web page for more information on submitting homeworks and converting to ps/pdf.

Late Work

The good news is that you can hand in any assignment up to one week late (seven days). The bad news is that you will lose seven points (out of 100) per day for the privilege. To be unambiguous, we define a "day" to start at midnight. You will lose seven points for each extra midnight that it takes you to do the assignment. *Late work makes a class much harder to administer. It also hurts you. Please try to avoid it.* If your assignment is more than seven days late, you will get a grade of zero (0) for that assignment.

Written assignments may be submitted multiple times. If you intend to submit any version more than one day late you must tell your TA to ensure we grade the most recent version.

Programming assignments may be resubmitted any number of times throughout the 7-day grading period. Each night at midnight, starting on the night the assignment is due, our scripts will automatically collect all the new handins and grade them, taking into account the 7-point-per-day lateness penalty. Your best current grade for the assignment will be posted to the Web page by the following morning.

If you have a good excuse (such as being very sick), you should contact Prof. Rudich. For compelling reasons (that extend beyond the fact that you have a lot of work lately and didn't plan ahead), we will excuse you from the lateness penalty.

Exercises

At the beginning of the problem sets may be a list of exercises. These will be clearly marked, and are **not to be handed-in**. The exercises will usually concern the lecture material from the previous week, and are meant to be relatively simple and to provide a first check of your understanding of recent material. If you are unable or unsure that you can answer these correctly, you should seek help from your fellow students and the course staff.

Extra Credit

A few of you will find the assignments too easy. For this reason, we will include more challenging extra credit problems. These problems will be substantially more sophisticated. Extra credit problems will not have a high point value and will not be a source of much partial credit. These are intended for the student who wants an increased challenge, despite the lack of point incentive.

Policy on Collaboration and Cheating

You *may* verbally collaborate on homework problems and the programming assignments. On each problem and program that you hand-in, you *must* include the names of the students with whom you have had discussions concerning your solution. Indicate whether you gave help, received help, or worked something out together. *You may* not *share written work or programs with anyone else.* You may *not* receive help from students who have taken the course in previous years. You may *not* review any course materials (or software) from previous years. You may *not* read the current solution (handed out) if you will be handing in the current assignment late. You may *not* look up the answer to a homework assignment which happens to appear in the published literature. You may *not* attempt to violate the security of the electronic grading system. You *may* get help from anyone concerning programming issues which are clearly more general than the specific assignment (e.g., what does a particular error message mean?).

Thus, clear examples of *cheating* include:

- Showing your code to another student.
- Copying a program from someone else.
- Getting help from someone whom you do not acknowledge on your solution.
- Showing a draft of a written solution to another student.
- Copying from another student during an exam.
- Receiving exam related information from a student who has already taken the exam.
- Submitting a program that attempts to alter or erase grading information.
- Looking at someone else's files containing draft solutions, even if the file permissions allow it.
- Lying to the course staff.

Penalty for Cheating

Our reaction to your cheating will vary according to the situation.

Unsolicited Confession If you seek us out and admit that you have cheated, we will probably let it go.

Solicited Confession If we come to you and ask if you have cheated and you freely admit it, we will take that into consideration. We will either give you a zero on the assignment, ask you to drop the class, or fail you in the course.

Denial If you do not admit that you have cheated, we will provide our evidence that you have done so. We will at the very least fail you in the class; furthermore, we will take our evidence to the dean and seek more substantial penalties.

Pedagogical Rationale and Advice

Collaboration not only helps you get the job done, it teaches you how to explain your (inchoate) ideas to others. This is why we permit discussion of the problems between students. Be careful not to let other people do all the work. If you misuse the opportunity for collaboration in this manner, you will fail the exams and do poorly in the course.

The best is usually to find a single partner with whom you have a relatively balanced collaboration. A group of four is usually too big for everyone to be following the joint problem solving process.

Many of the course materials will be the same as in previous years. This is not because we are lazy. It takes years to develop good problems. The only reason to change them is to make cheating more difficult. It is far better for you to work on the most excellent problems that we have been able to find in over a decade of teaching. We appeal to your sense of honor because this is what is optimal from a pedagogical point of view.

Signing the Statement on the Back Page

We understand that most of you would never consider cheating in any form. There is, however, a small minority of students for whom this is not the case. In the past, when we have caught students cheating they have often insisted that they did not understand the rules and penalties. For this reason we require that each student sign and return the back page of this document.

Commitment to Honor the Cheating Policy

Policy on Collaboration and Cheating

You *may* verbally collaborate on homework problems and the programming assignments. On each problem and program that you hand-in, you *must* include the names of the students with whom you have had discussions concerning your solution. Indicate whether you gave help, received help, or worked something out together. *You may* not *share written work or programs with anyone else.* You may *not* receive help from students who have taken the course in previous years. You may *not* review any course materials (or software) from previous years. You may *not* read the current solution (handed out) if you will be handing in the current assignment late. You may *not* look up the answer to a homework assignment which happens to appear in the published literature. You may *not* attempt to violate the security of the electronic grading system. You *may* get help from anyone concerning programming issues which are clearly more general than the specific assignment (e.g., what does a particular error message mean?).

Thus, clear examples of *cheating* include:

- Showing your code to another student.
- Copying a program from someone else.
- Getting help from someone whom you do not acknowledge on your solution.
- Showing a draft of a written solution to another student.
- Copying from another student during an exam.
- Receiving exam related information from a student who has already taken the exam.
- Submitting a program that attempts to alter or erase grading information.
- Looking at someone else's files containing draft solutions, even if the file permissions allow it.
- Lying to the course staff.

Penalty for Cheating

Our reaction to your cheating will vary according to the situation.

Unsolicited Confession If you seek us out and admit that you have cheated, we will probably let it go.

Solicited Confession If we come to you and ask if you have cheated and you freely admit it, we will take that into consideration. We will either give you a zero on the assignment, ask you to drop the class, or fail you in the course.

Denial If you do not admit that you have cheated, we will provide our evidence that you have done so. We will at the very least fail you in the class; furthermore, we will take our evidence to the dean and seek more substantial penalties.

Pedagogical Rationale and Advice

Collaboration not only helps you get the job done, it teaches you how to explain your (inchoate) ideas to others. This is why we permit discussion of the problems between students. Be careful not to let other people do all the work. If you misuse the opportunity for collaboration in this manner, you will fail the exams and do poorly in the course. The best is usually to find a single partner with whom you have a relatively balanced collaboration. A group of four is usually too big for everyone to be following the joint problem solving process.

Many of the course materials will be the same as in previous years. This is not because we are lazy. It takes years to develop good problems. The only reason to change them is to make cheating more difficult. It is far better for you to work on the most excellent problems that we have been able to find in over a decade of teaching. We appeal to your sense of honor because this is what is optimal from a pedagogical point of view.

I, _____ , have read and understood the
(PRINT YOUR NAME)
above statement of the CS 15-251 policy on collaboration and cheating which was distributed on the first day of class. I agree to honor the rules which the policy statement describes.

_____
(SIGN AND DATE)