CS 15-251 Great Theoretical Ideas In Computer Science

Fall 2005 Course Document **http://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	Baker Hall 136A
Recitation Section A:	M 9:30-10:20p	SH 220
Recitation Section B:	M 10:30-11:20p	SH 220

Course Staff

Name	Office and Phone	Email	Office hours
Professors:			
Anupam Gupta	Wean 4109, 268-7127	`anupamg@cs.cmu.edu`	Thu 5-6:30p
John Lafferty	Wean 4111, 268-6791	`lafferty@cs.cmu.edu`	Tu 5-6:30p
Teaching Assistants:			
Kanat Tangwongsan	Wean 8th floor couches	`ktangwon+15251@andrew.cmu.edu`	M 5-6:30p
Yinmeng Zhang	Wean 8th floor couches	`ynz@andrew.cmu.edu`	Su 3-4:30p
Secretary:			
Dorothy Zaborowski	Wean 4116, 268-3779	`daz@cs.cmu.edu`	

Important Course Information

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. *Discrete Mathematics: Elementary and Beyond*, by L. Lovasz, J. Pelikan, and K. Vesztergombi, published by Springer Verlag.
2. *Applied Combinatorics*, by A. Tucker, published by Wiley & Sons.
3. *Concrete Mathematics: A Foundation for Computer Science*, by R. Graham, D. Knuth, and O. Patashnik, published by Addison-Wesley.
4. *Introduction to Algorithms*, by T. Cormen, C. Leiserson, R. Rivest, and C. Stein, published by MIT Press.
5. *Discrete Mathematics and its Applications*, by K. H. Rosen, published by McGraw-Hill.
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`

The web page will have 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 bulletin boards are:

- `cyrus.academic.cs.15-251`, for staff announcements only;
- `cyrus.academic.cs.15-251.discuss`, for general discussion among members of the class. Please keep discussion relevant and polite.

Mailing lists: the list `cs-251@cs` mails to the entire class and teaching staff; `cs-251-A@cs` and `cs-251-B@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 and Exams

Over the course of the semester, there will be

- 12 (twelve) **problem sets**, which will include some programming assignments.
- three **tests**, which will be 50 minutes long, and given in recitation.
- five short **quizzes**, which will be 10 minutes long, and given at the beginning of classes.
- one **final exam**, which will be 3 hours long.

The dates of all of these (except the date for the final exam) appear on the calendar on the course web-page.

Your Grade

We will drop your lowest homework grade, your lowest in-class quiz grade, and your lowest test grade. The grading formula will be:

- **40% Homework (problem sets, programming assignments)**
- **25% In-Recitation Tests**
- **5% In-class 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. Your mid-term grade will reflect your anticipated final grade.

If you put in marginal effort on the homework and come through on the exams, your grade will suffer a bit. This is deliberate; we think the homework is important.

The Homework Guru

Each homework assignment will have a specific *guru*: (s)he 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 Homeworks

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

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 the directory:

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

Please see the web page for more information on converting files to postscript or 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 resubmit 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 the professors. 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.

Extra Credit

A few of you will find the assignments too easy. Hence, we will include more challenging extra credit problems, which 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, or on the web.
- You *may not* attempt to violate the security of the electronic grading system.
- However, 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.
- Looking up answers to homework problems on the web

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 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.

Some course material 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 read, sign and return the back page of this document.

Commitment to Honor the Course 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, or on the web.
- You *may not* attempt to violate the security of the electronic grading system.
- However, 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.
- Looking up answers to homework problems on the web

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 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.

Some course material 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 above statement
(PRINT YOUR NAME)
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)