COMPUTER SCIENCE

Emeriti: (Professors) Tom Binford, George B. Dantzig, Robert W. Floyd, John G. Herriot, Donald E. Knuth, William F. Miller, Nils J. Nilsson
Chair: Jean-Claude Latombe
Associate Chair for Education: Eric S. Roberts
Professors: David Cheriton, Bill Dally, David Dill, Qussama Khatib, Monica Lam, Hector Garcia-Molina, Gene H. Golub, Leonidas J. Guibas, Patrick Hanrahan, John Hennessy, Mark A. Horowitz, Oussama Khatib, Jean-Claude Latombe, Zohar Manna, John McCarthy, Edward J. McCluskey, John Mitchell, Joseph E. Oliger, Vaughan Pratt, Jeffrey D. Ullman, Terry Winograd
Associate Professors: Michael Genesereth, Marc Levoy, Rajeev Motwani, Serge A. Plotkin, Mendel Rosenblum, Yoav Shoham, Jennifer Widom
Assistant Professors: Mary G. Baker, Dan Boneh, Christoph Bregler, Dawson Engler, Ronald P. Fedkiw, Armando Fox, Daphne Koller, Christopher Manning, Nick McKeown, Balaji Prabhakar, Carlo Tomasi
Professors (Research): Richard Fikes, Gio Wiederhold
Professor (Teaching): Eric S. Roberts
Courtesy Professors: Giovanni De Micheli, Martin Kay, Michael Levitt, Teresa Meng, Grigori Mints, Fouad A. Tobagi
Courtesy Associate Professors: Russ Altman, Martin Fischer, John T. Gill, III, David Heeger, Teresa Meng, Mark A. Musen, Oyekunle Olukotun
Courtesy Assistant Professors: Joshua B. Tenenbaum, Benjamin Van Roy
Courtesy Assistant Professor (Research): Yuval Shahar
Senior Lecturer: Margaret Johnson
Lecturers: Gerald Cain, Nicholas J. Parlante, Robert Plummer, Patrick Young, Julie Zelenski
Acting Professor: John K. Salisbury
Acting Associate Professor: Colin Williams
Consulting Professors: Richard Gabriel, Kurt Konolige, Prabhakar Raghavan
Consulting Associate Professors: Craig Partridge, Feng Zhao
Consulting Assistant Professor: B. J. Fogg
Visiting Associate Professors: Ahmed Bahai, Hubert Comon, Greg Dudek, Moshe Tennenholz

The Department of Computer Science (CS) operates and supports computing facilities for departmental education, research, and administration needs. These CS systems are connected to SUNet, the campus-wide 100MB Ethernet backbone network, and SUNet is connected to the Internet through GTE.

All CS students have access to a departmental student machine, a Multi-CPU SUN Enterprise3000, as well as a cluster of public workstations in the Gates Building. In addition, most students have access to systems associated with their research areas.

Each research group in CS has systems specific to its research needs. These systems range from PC clones/Macs to high-end Multi-CPU SGIs and SUNs. Servers and workstations manufactured by DEC, SUN, HP, SGI, Intel, Apple, and IBM are also in place.

Support for course work and instruction is provided on systems available through Information Technology Systems and Services (ITSS).

UNDERGRADUATE PROGRAMS

The department offers both a major and a minor in Computer Science. The requirements for these programs are outlined in the "School of Engineering" section of this bulletin and described in more detail in the *Handbook for Undergraduate Engineering Programs* published by the School of Engineering. The department has an honors program, which is described in the following section.

In addition to Computer Science itself, Stanford offers several interdisciplinary degrees with a substantial computer science component. The Computer Systems Engineering major (also in Engineering) allows the study of issues of both computer hardware and software, bridging the gap between traditional CS and Electrical Engineering majors. The Symbolic Systems major (in the School of Humanities and Sciences) offers a chance to explore computer science and its relation to linguistics, philosophy, and psychology. Finally, the Mathematical and Computational Sciences major (also Humanities and Sciences) allows students to explore computer science along with more mathematics, statistics, and operations research.

HONORS

The Department of Computer Science offers an honors program for selected undergraduates whose academic records and personal initiative indicate that they have the necessary skills to undertake high-quality research in computer science. Admission to the program is by application only. To apply for the honors program, students must be majoring in Computer Science, have a grade point average (GPA) of at least 3.5 in courses that count toward the major, and achieve senior standing (135 or more units) by the end of the academic year in which they apply. Coterminal master's students are eligible to apply as long as they have not already received their undergraduate degree. Beyond these requirements, students who apply for the honors program must also find a faculty member who agrees to serve as the thesis adviser for the project. Thesis advisers must be members of Stanford's Academic Council.

Students who meet the eligibility requirements and wish to be considered for the honors program must submit a written application to the undergraduate program office by May 1 of the year preceding the honors work. The application must include a letter describing the research project, a letter of endorsement from the faculty sponsor, and a transcript of courses taken at Stanford. Each year, a faculty review committee will select the successful candidates for honors from the pool of qualified applicants.

In order to receive departmental honors, students admitted to the honors program must, in addition to satisfying the standard requirements for the undergraduate degree, do the following:

1. Complete at least 9 units of CS 191 or 191W under the direction of their project sponsor.
2. Enroll in a research seminar, which allows students to share their experience with other students working on research projects.
3. Complete an honors thesis deemed acceptable by a committee consisting of the thesis adviser and at least one additional faculty member.
4. Present the thesis at a public colloquium sponsored by the department.
5. Maintain the 3.5 GPA required for admission to the honors program.

GRADUATE PROGRAMS

The University's basic requirements for the M.S. and Ph.D. degrees are discussed in the "Graduate Degrees" section of this bulletin.

MASTER OF SCIENCE

In general, the M.S. degree in Computer Science is intended as a terminal professional degree and does not lead to the Ph.D. degree. Most students planning to obtain the Ph.D. degree should apply directly for admission to the Ph.D. program. Some students, however, may wish to complete the master's program before deciding whether to pursue the Ph.D. To give such students a greater opportunity to become familiar with research, the department has instituted a new program leading to a master's degree with distinction in research. This degree is described in more detail in a subsequent section.

Applications for admission to the M.S. program, and all of the required supporting documents, must be received before December 15, 2000. Exceptions are made for applicants who are either Honors Co-op applicants or who are already students at Stanford (including coterminal applicants). Information on these deadlines is available from the department.

REQUIREMENTS

A candidate is required to complete a program of 45 units. At least 36 of these must be graded units, passed with an average 3.0 (B) grade point average (GPA) or better. The 45 units may include no more than 21 units of courses from those listed below in Requirements 1 and 2. Thus, students needing to take more than seven of the courses listed in Requirements 1 and 2 actually complete more than 45 units of course work in this program. Only extremely well-prepared students may expect to finish the program in one year; most complete the program in six quarters. Students hoping to complete the program with 45 units should already have a substantial background in computer science, including course work or experience equivalent to all of Requirement 1 and some of the courses in Requirement 2.

Requirement 1—The following courses may be needed as prerequisites for other courses in the program: CS 103X, 107, 108; 193L (for specialization 5 only); Electrical Engineering 182; Mathematics 109 or 120.

Requirement 2—Students must demonstrate breadth of knowledge in the field by completing the following courses:

1. Area A: Mathematical and Theoretical Foundations:
   a) Required:
      1) Statistics (Statistics 116 *or* Management Science and Engineering 120)
      2) Algorithms (CS 161)
      3) Automata (CS 154)
   b) Choose one of:
      1) Numerical Analysis (CS 137 or 237A)
      2) Logic (CS 157, 257, 258, *or* Philosophy 160A)
2. Area B: Computer Systems:
   a) Required: Architecture (Electrical Engineering 182 or 282)
   b) Choose two of:
      1) Operating Systems (CS 140)
      2) Compliers (CS 143)
      3) Introduction to Computer Networks (CS 244A *or* Electrical Engineering 284)
3. Area C: AI and Applications:
   a) Choose two of the following, with at least one 200-level course:
      1) AI (CS 121 or 221)
      2) Databases (CS 145 or 245)
      3) Graphics (CS 148 or 248)

Individual specializations are free to narrow the set of choices in specific areas of the breadth requirement; see the individual specialization sheets in the department office for details. Breadth courses are waived only if evidence is provided that similar or more advanced courses have been taken, either at Stanford or another institution. Courses that are waived rather than taken may not be counted toward the M.S. degree. Breadth courses may be taken on a Satisfactory/No Credit basis provided that a minimum of 36 graded units is presented within the 45-unit program.

Requirement 3—At least 1 but no more than 3 units of 500-level seminars must be taken.

Requirement 4—A program of 21 units in an area of specialization must be completed. All courses in this area must be taken for letter grades. Eight approved programs are listed below. Students may propose to the M.S. program committee other coherent programs that meet their goals and satisfy the basic requirements. Students who want to include a substantial research project as part of their degree program can arrange with their adviser to replace units in their specialization with a CS 393 (Computer Laboratory) project.

1. Numerical Analysis/Scientific Computation
   a) CS 237A, 237B, 237C
   b) At least two of: CS 260; Manage. Sci. & Engr. 121; Math. 131, 132, 220A, 220B, 220C; Stat. 200
   c) At least three of: CS 223A, 238, 326A, 327A, 328, 336, 337, 339; Aero. & Astro. 214A, 214B; Mech. Engr. 235A, 254; Stat. 227
2. Systems
   a) CS 240, 242
   b) At least three of: CS 243, 244A, 245, 248, 348B; Elect. Engr. 271, 275, 382
   c) At least 6 more units selected from '2b' and from the following: CS 194, 241, 244B, 244C, 248V, 249, 255, 315A, 315B, 341, 342, 343, 344, 345, 346, 347, 348A, 348C, 349, 448; Elect. Engr. 183, 272, 281, 318, 319, 374, 384A, 384B, 384C, 482A, 482B, 487, 488, 489; Psych. 267
3. Software Theory
   a) CS 242, 243, 256, 258
   b) At least one of: CS 244A, 245, 342, 343, 345
   c) At least one course from the following: CS 255, 261, 351, 355, 356, 361A, 361B, 365, 368
   d) At least one additional course selected from '3b,' '3c,' CS 346
4. Theoretical Computer Science
   a) CS 256, 258, 261 (361A, 361B, or 365 may be used as substitutes for 261)
   b) At least 12 more units from CS 228, 255, 345, 351, 352, 353, 355, 356, 357, 358, 359*, 361A, 361B, 365, 367A, 367B, 368, 369*; Manage. Sci. & Engr. 310
5. Artificial Intelligence
   a) At least four of: CS 222, 223A, 223B, 224M, 224N, 227, 228, 229, 326A
   b) A total of 21 units from the above and from the following: CS 205, 206, 225A, 225B, 226, 256, 257, 270A, 271, 274, 323, 327A, 328, 329, 354, 377, 379, 426
6. Database
   a) CS 245
   b) Two of: CS 345, 346, 347
   c) Four additional courses selected from '6b' and from the following: CS 222, 240, 242, 243, 244A, 244B, 244C, 249, 255, 270A, 270B, 271, 272, 315A, 315B, 341, 344, 395, 446; Elect. Engr. 489
7. Human-Computer Interaction
   a) CS 147, 247A, 247B
   b) At least 6 units from: CS 148 or 248, 377 (may be taken repeatedly), 378, 447
   c) A total of 21 units from the above and from the following: Comm. 269, 272; CS 249, 270A, 270B, 272, 320, 348A, 348B, 448; Engr. 145; Manage. Sci. & Engr. 234, 273, 280, 284; Linguistics 238; Mech. Engr. 101, 115, 215, 313; Psych. 203, 205, 221, 266, 267
8. Real-World Computing
   a) At least two of: CS 223A, 223B, 248
   b) At least three of: CS 205, 237A, 237B, 237C, 248V, 249, 270A, 270B, 271, 272, 326A, 348A, 348B, 368
   c) A total of 21 units from the above and from the following: CS 225A, 225B, 247A, 274, 327A, 328, 336, 399, 448; Psych. 267

* With consent of Specialization chair.

Requirement 5—Additional elective units must be technical courses (numbered 100 or above) related to the degree program and approved by the adviser. Elective courses may be taken on a Satisfactory/No Credit basis provided that a minimum of 36 graded units are presented within the 45-unit program.

MASTER OF SCIENCE WITH DISTINCTION IN RESEARCH

A student who wishes to pursue the M.S./CS with distinction in research must first identify a faculty adviser who agrees to supervise and support the research work. The research adviser must be a member of the Academic Council and must hold an appointment in Computer Science. The student and principal adviser must also identify another faculty member, who need not be in the Department of Computer Science, to serve as a secondary adviser and reader for the research report. In addition, the student must complete the following requirements beyond those for the regular M.S./CS degree:

1. *Research Experience:* the program must include significant research experience, at the level of a half-time commitment over the course of three academic quarters. In any given quarter, the half-time research commitment may be satisfied by a 50 percent appointment to a departmentally supported research assistantship, 6 units of independent study (CS 393, 395, or 399), or a prorated combination of the two (such as a 25 percent research assistantship supplemented by 3 units of independent study). This research must be carried out under the direction of the primary or secondary adviser.
2. *Supervised Writing and Research:* in addition to the research experience outlined in the previous requirement, students must enroll in at least 3 units of independent research (CS 393, 395, or 399) under the direction of their primary or secondary adviser. These units should be closely related to the research described in the first requirement, but focused more directly on the preparation of the research report described in the next section. Note that these units must be taken in addition to the 21 units required for the specialization, although they do not count toward the 45 units required for the degree.
3. *Research Report:* students must complete a significant report describing their research and its conclusions. The research report represents work that is publishable in a journal or at a high-quality conference, although it is presumably longer and more expansive in scope than a typical conference paper. Three copies of the research report must be submitted to the Student Services office in the department three weeks before the beginning of the examination period in the student's final quarter. Both the primary and secondary adviser must approve the research report before the "distinction in research" designation can be conferred.

DOCTOR OF PHILOSOPHY

Applications to the Ph.D. program and all supporting documents must be received before December 15, 2000. The following are department requirements (see the Computer Science graduate programs administrator for further details, or visit http://cs.stanford.edu/Admissions):

1. A student should plan and successfully complete a coherent program of study covering the basic areas of computer science and related disciplines. The student's adviser has primary responsibility for the adequacy of the program, which is subject to review by the Ph.D. program committee.
2. Each student, to remain in the Ph.D. program, must satisfy the breadth requirement covering introductory level graduate material in major areas of computer science. Once a student fulfills six of eight whole areas of the breadth requirement, he or she may apply for admission to candidacy for the Ph.D. This is typically done by the end of the second year in the program. The student must completely satisfy the breadth requirement by the end of nine quarters (excluding summers), and must pass a qualifying exam in the general area of the expected dissertation.
3. As part of the training for the Ph.D., the student is required to complete at least 4 units (a unit is 10 hours per week for one quarter) as a teaching assistant or instructor for courses in Computer Science numbered 100 or above.
4. The most important requirement is the dissertation. After passing the qualifying examination, each student must secure the agreement of a member of the department faculty to act as the dissertation adviser. (In some cases, the dissertation adviser may be in another department.)
5. The student must pass a University oral examination in the form of a defense of the dissertation. It is usually held after all or a substantial portion of the dissertation research has been completed.
6. The student is expected to demonstrate the ability to present scholarly material orally, both in the dissertation defense and by a lecture in a department seminar.
7. The dissertation must be accepted by a reading committee composed of the principal dissertation adviser, a second member from within the department, and a third member chosen from within the University. The principal adviser and at least one of the other committee members must be Academic Council members.

Ph.D. MINOR

For a minor in Computer Science, a candidate must complete 20 units of computer science course work, including at least three of the master's core courses to provide breadth, and one course numbered 300 to provide depth. The remaining courses must be numbered 200 or above. One of the courses taken must include a significant programming project to demonstrate programming proficiency. A grade point average (GPA) of 3.0 or better must be maintained.

TEACHING AND RESEARCH ASSISTANTSHIPS

Graduate student assistantships are available. Half-time assistants receive a tuition scholarship for 9 units per quarter during the academic year, and in addition receive a monthly stipend.

Duties for half-time assistants during the academic year involve approximately 20 hours of work per week. Teaching assistants (TAs) help an instructor teach a course by conducting discussion sections, consulting with students, grading examinations, and so on. Research assistants (RAs) help faculty and senior staff members with research in computer science. Most teaching and research assistantships are held by Ph.D. students in the Department of Computer Science. If there is an insufficient number of Ph.D. students to staff teaching and research assistantships, then these positions are open to a limited number of master's students in the department. However, master's students should not plan on being appointed to an assistantship.

Students with fellowships may have the opportunity to supplement their stipends by serving as graduate student assistants.

COURSES

(WIM) indicates that the course meets the Writing in the Major requirement.

(AU) indicates that the course is subject to the University Activity Unit limitations (8 units maximum).

GUIDE TO SELECTING INTRODUCTORY COURSES

Students arriving at Stanford have widely differing backgrounds and goals, but most find that the ability to use computers effectively is beneficial to their education. The department offers many introductory courses to meet the needs of these students.

For students whose principal interest is an exposure to the fundamental ideas behind computer science and programming, CS 105 is the most appropriate course. It is intended for students in nontechnical disciplines who expect to make some use of computers, but who do not expect to go on to more advanced courses. CS 105 meets the Area 2b General Education Requirement and includes an introduction to programming, and the use of modern Internet-based technologies. Students interested in learning to use the computer should consider CS 1C (Introduction to Computing at Stanford) or 1U (Introduction to Unix).

Students who intend to pursue a serious course of study in computer science may enter the program at a variety of levels, depending on their background. Students with little prior experience or those who wish to take more time to study the fundamentals of programming should take 106A followed by 106B. Students in 106A need not have prior programming experience. Students with significant prior exposure to programming or those who want an intensive introduction to the field should take 106X, which covers most of the material in 106A and B in a single quarter. All instruction in CS 106 uses ANSI C, although the prior programming experience required for 106X may be in any language. In all cases, students are encouraged to discuss their background with the instructors responsible for these courses.

After the introductory sequence, Computer Science majors and those who need a significant background in computer science for related majors in engineering should take 103, 107 and 108. CS 103 offers an introduction to the mathematical and theoretical foundations of computer science. CS 107 exposes students to a variety of programming paradigms that illustrate critical strategies used in systems development; 108 builds on this material, focusing on the development of large interactive programs based on the object-oriented programming paradigm.

In summary:

For exposure—1C or 1U
For nontechnical use—105
For scientific use—106A
For a technical introduction—106A
For significant use—106A,B or 106X, along with 103, 107, and 108

NUMBERING SYSTEM

The first digit of a CS course number indicates its general level of sophistication:

0-99	service courses for nontechnical majors
100-199	other service courses, basic undergraduate
200-299	advanced undergraduate/beginning graduate
300-399	advanced graduate
400-499	experimental
500-599	graduate seminars

The tens digit indicates the area of Computer Science it addresses:

00-09	Introductory, miscellaneous
10-19	Hardware Systems
20-29	Artificial Intelligence
30-39	Numerical Analysis
40-49	Software Systems
50-59	Mathematical Foundations of Computing
60-69	Analysis of Algorithms
70-79	Typography and Computational Models of Language
90-99	Independent Study and Practicum

NONMAJOR

1C. Introduction to Computing at Stanford—For those with limited experience on computers. Introduction to the basics of computing, and a variety of programs, encouraging individual exploration of the programs covered. Topics: word processing, spread sheets, using the WWW and the Internet, and computing resources at Stanford. Macintosh and PC systems. One-hour lecture/demonstration in dormitory clusters. Weekly short assignments and final project. Not a programming course.
1 unit, Aut (Staff)

1I. The Internet—For a computer-literate but not technical audience. What is the Internet and what is it good for? The foundations, resources, and uses of the Internet, emphasizing practical skills for finding, reading, and authoring Internet material. Topics: HTML, FTP, HTTP, web publishing and searching; evolution and future directions; security and privacy issues. Programming-oriented course is 193I. Prerequisites: basic computer skills at level of 1C, e.g., file editing, and access to a computer on the Internet.
1 unit, Win (Staff)

1U. Introduction to Unix—Tutorial on using the Unix operating system. Topics: text editors, the file system, the C shell, standard Unix utilities, PERL. Includes simple shell programming, but is not a programming course and assumes no prior exposure to programming.
1 unit, Spr (Staff)

50. Problem Solving with Mathematica—For engineers, physicists, mathematicians, and others who need to solve mathematical or quantitative problems. Comprehensive introduction to Mathematica, an interactive mathematical software package that includes a high-level programming language. Symbolic, numerical, graphical, animation, and programming capabilities, including use of Mathematica to manipulate expressions, find roots, solve differential equations, visualize functions and data, import and export data in arbitrary formats, work with expressions in standard mathematical notation, and perform statistical analyses.
2 units (Williams) alternate years, not given 2001-02

51. Introduction to Quantum Computing and Quantum Information Theory—For computer scientists, physicists, mathematicians, engineers, and others who want to learn the capabilities of quantum computers and the necessary quantum mechanics and complexity theory. Topics: quantum algorithms (including Shor's polynomial time algorithm for integer factorization, Grover's database search algorithm, quantum tree search, quantum wavelets), quantum information theory, quantum cryptography, breaking the RSA cryptosystem, quantum teleportation, circuit design, quantum error correction, and examples of prototype quantum computers. Prerequisites: familiarity with elementary matrix algebra and complex numbers.
2 units, Win (Williams)

99A. Stanford Introductory Seminar: The Downside of Computing Systems—Preference to freshmen. Computers are critical components of our world in such tasks as surgery, air traffic control, and international banking. How computing systems fail, how such failures may affect our society in the future, and how to build and maintain systems to avoid failures. Case studies of computer-related disasters, including the Therac-025 accidents, the Internet worm, and the Ariane 5 crash. Topics: computer security, robust distributed systems, fault-tolerant architectures, organizational behavior.
3 units (Baker) not given 2000-01

99C. Stanford Introductory Seminar: Computers—Fact and Fiction—Preference to freshmen. Can a computer be world chess champion? Can it learn to do something it wasn't told to do? Can it create? Even computer scientists have disagreed. Some interesting and/or controversial predictions are compared to the state of the art technology. Things computers have already done, things that they might be able to do, and things that computers cannot do. Possible topics: virtual reality, the world wide web, machine learning, computer game playing, security and crptography, etc. Prerequisite: basic understanding of how computers work (i.e., how we write programs that tell a computer what to do).
3 units (Koller) not given 2000-01

99D. Stanford Introductory Seminar: The Science of Art—Preference to freshmen. The interwoven histories of science and Western art from the Renaissance to the 19th century. Emphasis is on the revolutions in science and mathematics that have inspired parallel revolutions in the visual arts (e.g., Brunelleschi's invention of linear perspective, Newton's discoveries in geometric optics, and the theories of color vision proposed by Goethe, Young, Helmholtz, etc.). The scientific principles behind image making, including a brief survey of digital image synthesis (i.e., computer graphics). GER:2b (DR:6)
3 units, Win (Levoy)

99E. Stanford Introductory Seminar: Great Ideas in Computer Science—Preference to freshmen. The power and limitations of computers; concrete strategies for solving problems using computers. What can a computer do efficiently? Why are programs hard to test? How can we make computers appear clairvoyant? How do you keep secrets in computers? Should tables be sorted? When is it a good idea to be greedy? These questions involve ideas whose impact ranges from the philosophical foundations of computation to concrete applications in everyday life. Prerequisite: mathematical maturity (e.g., AP Math) and exposure to computer programming.
3 units (Motwani, Raghavan) not given 2000-01

99F. Stanford Introductory Seminar: **Paradox—Bug or Feature?**—Preference to freshmen. Conflict in thought is as inevitable as in territory and relationships. The discovery, influence, and use of paradoxes in mathematics, logic, nature, cognition, and computation. The paradoxical continuum: Zeno, Democritus, Newton, Leibniz, Cohen, Itano. Logical paradoxes: Eubulides' liar, the set of all sets, incompleteness of arithmetic, the halting problem. Paradoxes in nature: quantum mechanics, chaos. Cognition: mind-body interaction, free will, and determinism. Computer hardware and software: cyclic circuits and the fixpoint operator. Prerequisite: AP Math.
3 units, Win (Pratt)

99G. Stanford Introductory Seminar: The Two Cultures—Bridging the Gap—Preference to freshmen. In 1959, the British physicist and novelist C. P. Snow delivered a lecture at Cambridge University in which he argued that "the intellectual life of the whole of western society is increasingly being split into two polar groups." In Snow's view, these groups, which can be characterized roughly as humanists and scientists, exist as separate cultures that have "almost ceased to communicate at all." Professors in Computer Science and English collaborate to examine the nature of this split, reflected at Stanford by the tendency to divide the campus community into "techies" and "fuzzies," and explore ways to bridge this cultural gap.
3 units, Aut (Roberts, Saldivar)

99H. Stanford Introductory Seminar: Programming and Problem Solving Seminar—Preference to freshmen. Students are given five problems to solve. Each involves programming, but programming is not sufficient to solve the problem (i.e., the approach to the problem is at least as important). Students experiment individually and as a group with techniques, and write a working program as a solution. Group discussions include general problem-solving approaches and concepts relevant to the problem at hand. Prerequisite: 106B or 106X.
3 units (Ullman) not given 2000-01

99I. Stanford Introductory Seminar: Business on the Information Highways—Preference to freshmen. Understanding the capabilities of the Internet and its services. The effect on commerce, education, and healthcare. Technical and business alternatives. Who will be hurt and who will benefit from the changes occurring? The central project develops sections of a Web publication.
3 units, Win (Wiederhold)

99J. Stanford Introductory Seminar: Computer Security in the Electronic Age—Preference to sophomores. Based on readings and discussions of current issues in computer security. Topics: the history of codes and ciphers and a summary of basic mathematics used in current cryptography; causes of computer vulnerabilities, including program errors, design flaws, and inherent network and browser limitations; policies and practices that restrict or monitor access to information.
3 units (Mitchell) not given 2000-01

99K. Stanford Introductory Seminar: Digital Actors—Preference to sophomores. Digital actors are an emerging field, with applications to video games, movies, simulation and training, manufacturing, and animated web pages. Introduces the computational techniques used to create and animate robotics, geometric computing, computer vision, and graphics. The problem of creating/animating digital actors, technical subproblems. Prerequisite: knowledge of elementary geometry. Recommended: some programming experience.
3 units (Latombe) not given 2000-01

99M. Stanford Introductory Seminar: Computer and Information Security—Preference to freshmen. Topics: aspects of computer security, including the damage caused by break-ins, common holes in computer systems, technological solutions for preventing attacks, cryptography, and legal issues in computer security.
3 units (Boneh) not given 2000-01

99N. Stanford Introductory Seminar: Ruler, Compass, and Computer-Computational Representations of Geometry—Preference to sophomores. Representations of geometry play an important role in computer science in the physical world. Models of physical objects and processes as used in computer graphics, computer vision, and robotics use geometry as an essential component of representing shape, motion, and other physical modalities. The mathematical ideas behind commonly used representations and algorithms for geometric objects, focusing on intuitive understanding as opposed to formal development. Prerequisite: introductory CS class, e.g., 106A, B or X. Recommended: general background knowledge in mathematics and physics.
3 units, Spr (Guibas)

99P. Stanford Introductory Seminar: Smart Computers and other Technological Opportunities—Preference to freshmen. How smart are computers now? How smart can we make them, and how soon, and what good will they be? Of the current "smart" things, how smart are they and what genuine benefits are offered? Outside of computers, what are the technological opportunities for humanity? To what extent is humanity in difficulty with natural resources? What significant threats are there to humanity?
3 units (McCarthy) not given 2000-01

99Q. Stanford Introductory Seminar: Universal Ideas in Computation—Preference to freshmen. Techniques and ideas that cut across computer science, emphasizing concepts that also show up extensively in non-computer systems. Examples: indirection (especially naming and translation); using the past to predict the future; using randomness to make robust decisions; the pervasiveness of "skew" in data (20% of the code consumes 80% of CPU cycles, 4% of fighter pilots account for 40% of all kills); and the universality of scheduling and concurrency control, which are used to coordinate access to shared resources such as shared memory, intersections, classrooms, bathroom stalls, etc.
3 units, Spr (Engler)

99R. Stanford Introductory Seminar: Digital Dilemmas—Preference to freshmen. The history and evolution of computing and communication technology, and how it affects the fabric of society. Topics: the military-academic-industrial research complex, the Cold War, and the public good; intellectual property meets the Internet, using MP3 and Napster as a case study; the balance between individual privacy and freedom and the security and stability of the state, and the effect of strong cryptography on this balance. Readings, discussion, debates, guest speakers, field trips.
3 units, Aut (Fox)

UNDERGRADUATE

103A. Discrete Mathematics for Computer Science—The fundamental mathematical foundations required for computer science. Topics: logic, relations, functions, basic set theory, proof techniques, combinatories, recursion, and recurrence relations. GER:2c (DR:4)
3 units, Aut, Win (Johnson)

103B. Discrete Structures—Continuation of 103A. Topics: analysis of algorithms, mathematical formulations of basic data models (linear models, trees, graphs, and sets), regular expressions, grammers. Corequisite: 106B or 106X.
3 units, Win, Spr (Johnson)

103X. Discrete Structures (Accelerated)—Covers the material in 103A and B in a single quarter. Students who take 103X feel comfortable with mathematical formalism. GER:2c (DR:4)
**4 units, Win (Dill, Mitchell)*

105. Introduction to Computers—For non-technical majors. What computers are and how they work. Practical experience in programming. Construction of computer programs and basic design techniques. A survey of Internet technology and the basics of computer hardware. Students in technical fields and students looking to acquire programming skills should take 106A or 106X. Prerequisite: minimal math skills. GER:2b (DR:6)
**5 units, Aut, Spr (Young)*

106A. Programming Methodology—For students in technical disciplines. Broad introduction to the engineering of computer applications, emphasizing software engineering principles: design, decomposition, information hiding, procedural abstraction, testing, and reusable software components. Uses the programming language C, and concentrates on the development of good programming style and on understanding the

* May be taken for 3 units by graduate students.

basic facilities provided by the language. Alternatives: 105, 106X. GER:2b (DR:6)

**5 units, Aut (Plummer)*
Win (Staff)
Spr (Roberts)

106B. Programming Abstractions—Abstraction and its relation to programming. The software engineering principles of data abstraction, modules, certain fundamental data structures (e.g., stacks and queues), and data-directed design. Recursion and recursive data structures (linked lists and binary trees). Brief introduction to time and space complexity analysis. Prerequisite: 106A or consent of the instructor, based on prior exposure to ANSI C. GER:2b (DR:6)

**5 units, Aut (Staff)*
Win (Plummer)
Spr (Zelenski)

106X. Programming Methodology and Abstractions (Accelerated)—Covers most of the material in 106A,B. Students are expected to have previous programming experience at a level that allows them to understand the concepts presented in 106A, usually in a language other than C. First three weeks focus on understanding how the concepts are expressed in ANSI C. 106B material is covered for the balance. Students who complete 106A should enroll in 106B. 106X can be taken after 106A only with consent of instructor. GER:2b (DR:6)

**5 units, Aut (Staff)*
Win (Plummer)
Spr (Zelenski)

107. Programming Paradigms—Intense introduction to a variety of programming paradigms, programming languages, and language implementations. Topics: the compile-time languages, using advanced memory management features of imperative and multithreaded C; the functional paradigm, using LISP (or possibly ML); the object-oriented paradigm, using Java (and possibly Python); and the generic programming paradigm, introducing C++ and templates. Substantial programming projects. Prerequisite: 106B, 106X, or equivalent.

**5 units, Aut, Spr (Cain)*

108. Object-Oriented Systems Design—Software design and construction in the context of large OOP libraries. May be taught in C++ or Java. Topics: review of OOP, the structure of Graphical User Interface (GUI) OOP libraries, GUI application design and construction, OOP software engineering strategies, approaches to programming in teams. Prerequisite: 107.

**4 units, Aut, Win (Parlante)*

110. Introduction to Computer Systems and Assembly Language Programming—Organization of digital computers, buses, registers, processors, I/O, memory systems, and paged memory. Data representation, data structures, and computer arithmetic. Instruction sets and execution; addressing modes. Assembly language programming, including subroutines, co-routines, interrupts, and traps. Operating systems issues and principles of storage management; combines general principles and practice in implementations. Prerequisite: 106B or 106X.

**4 units, Spr (Staff)*

112. Computer Organization and Design—(Enroll in Electrical Engineering 182.)

4 units, Aut, Spr (Staff)

121. Artificial Intelligence—(Only one of 121/221 counts towards CS degree requirements.) Introduction to the key concepts, representations, and techniques used in building practical computational systems ("agents") that appear to display artificial intelligence (AI), through the use of sophisticated adaptive information processing algorithms. Topics: history of AI, reactive systems, heuristic search, planning, constraint satisfaction, knowledge representation and uncertain reasoning, machine learning, classification, applications to language, and vision. Prerequisites: 103B or 103X and basic facility with differential calculus, vector algebra, and probability theory.

3 units, Win (Manning)

137. Introduction to Scientific Computing—The fundamental issues of numerical computation for the mathematical, computational, and physical sciences, and engineering. Emphasis is from the perspective of the computer scientist. Use of numerical algorithms in engineering practice. Problems of accurately computing solutions in the presence of rounding errors and of computing discrete approximations of solutions which are defined on the continuum. The taxonomy of problem classes with methods for their solution and principles useful for analysis of performance and algorithmic development. Topics: error analysis, the solution of linear and nonlinear equations, interpolation and numerical differentiation, the approximation of integrals, and the solution of differential equations. Prerequisites: 106A; Mathematics 103 or 113 or equivalents.

**4 units, Aut (Fedkiw)*

138. Matlab and Maple for Science and Engineering Applications—Introduction to use of Matlab and Maple in engineering applications. Emphasis is on the use of software to solve real problems. How the algorithms work, primarily so user may understand their possible limitations. How to use packages to solve a variety of introductory but important problems in: linear systems, eigenvalue problems, ordinary differential equations, elementary statistics, elementary signal processing (Fourier transforms, wavelets), computer algebra, graphical interfaces. Applications for the engineering and physical sciences. Prerequisites: undergraduate linear algebra and a willingness to program.

**4 units, Win (Staff)*

140. Operating Systems and Systems Programming—The fundamentals of operating systems design and implementation. Basic structure; synchronization and communication mechanisms; implementation of processes, process management, scheduling, and protection; memory organization and management, including virtual memory; I/O device management, secondary storage, and file systems. Prerequisite: 108. Recommended: Electrical Engineering 182.

**4 units, Aut, Win (Engler)*

143. Compilers—Principles and practices in the design of programming language compilers. Topics: lexical analysis; parsing theory (LL, LR, and LALR parsing); symbol tables; type checking; common representations for records, arrays, and pointers; runtime conventions for procedure calls; storage allocation for variables; and generation of unoptimized code. Students construct simple compiler as programming project. Prerequisites: 103B or X, 107.

**4 units, Aut, Win (Zelenski)*

145. Introduction to Databases—Entity-relationship and relational data models and approaches to database design. Relational and object-relational query languages, with substantial coverage of SQL including SQL3. Algebraic query languages and some database theory. Integrity constraints, triggers, and views; functional dependencies and normal forms. Object-oriented database design and query languages including ODMG standard. Database transactions and security from the application perspective. Designing a database for an application. Interactive and programmatic interfaces to database systems. Introduction to advanced topics, e.g., data warehouses, data mining, XML, and Web/database interactions. Individual database application programming project with extensive use of SQL. Prerequisites: 103B or X, 107.

**4 units, Aut (Ullman)*
Spr (Widom)

147. Introduction to Human-Computer Interaction Design—Introduction to the concepts underlying the design of human-computer interaction: usability and affordances, direct manipulation, systematic

* May be taken for 3 units by graduate students

design methods, user conceptual models and interface metaphors, design languages and genres, human cognitive and physical ergonomics, information and interactivity structures, design tools and environments. Structured around a set of case studies in which notable interface designs and/or projects are analyzed as illustrative of underlying principles. Students participate in discussions of cases and do interface analysis and design exercises.
3-4 units, Aut (Winograd)

148. Introductory Computer Graphics—(For undergraduates; M.S. students or students with a strong interest in continuing in graphics should take 248. Only one of 148 or 248 counts towords CS degree requirements.) Introduction to two- and three-dimensional computer graphics. Topics: fundamentals of input and display devices, scan conversion of geometric primitives, two- and three-dimensional transformations and clipping, windowing techniques, curves and curved surfaces, three-dimensional viewing and perspective, hidden surface removal, illumination and color models, OpenGL, VRML, and 3-D modeling tools. Emphasis is on the development of practical skills in using graphics libraries and tools. Programming on Macintosh using C, OpenGL, and VRML, with demos in SoftImage. Prerequisites: 107, Mathematics 103.
3 units, Aut (Staff)

154. Introduction to Automata and Complexity Theory—Regular sets: finite automata, regular expressions, equivalences among notations, methods of proving a language not to be regular. Context free languages: grammars, pushdown automata, normal forms for grammars, proving languages non-context free. Turing machines: equivalent forms, undecidability. Nondeterministic Turing machines: properties, the class NP, complete problems for NP, Cook's theorem, reducibilities among problems. Prerequisite: 103B or X.
**4 units, Win (Staff)*
Spr (Motwani)

154N. Introduction to NP Completeness—Turing machines: equivalent forms, undecidability. Nondeterministic Turing machines: properties, the class NP, complete problems for NP, Cook's theorem, reducibilities among problems. Students participate in approximately the last half of 154. Prerequisite: a knowledge of formal languages and automata as in the first part of 154.
2 units, Win (Staff)
Spr (Motwani)

156. Introduction to Verification and Concurrency—A taste of logic: propositional, predicate, temporal. Specification and verification of sequential programs: correctness and termination. Concurrent programming: communication and synchronization, principles and algorithms. Specification of concurrent programs: safety and progress. Verification of safety properties: invariants. Prerequisite: 103B or X.
3 units (Manna)

157. Logic and Automated Reasoning—Introduction to logic for computer scientists. An elementary exposition from a computational point of view, of propositional logic, predicate logic, axiomatic theories, and theories with equality and induction. Interpretations, models, validity, proof. Automated deduction: polarity, skolemization, unification, resolution, equality. Strategies. Applications. Prerequisite: 103B or X.
**4 units, Aut (Manna)*
Spr (Genesereth)

157L. Logic and Automated Reasoning Laboratory
1 unit

161. Design and Analysis of Algorithms—Efficient algorithms for sorting, searching, and selection. Algorithm analysis: worst and average case analysis. Recurrences and asymptotics. Data structures: balanced trees, heaps, etc. Algorithm design techniques: divide-and-conquer, dynamic programming, greedy algorithms, amortized analysis. Algorithms for fundamental graph problems, e.g., depth-first search, connected components, topological sort, shortest paths. Possible topics: network flow, string searching, parallel computation. Prerequisite: 103B or X; Statistics 116.
**4 units, Aut (Plotkin)*
Win (Staff)

* May be taken for 3 units by graduate students

162. Introduction to Combinatorics and its Applications—(Enroll in Mathematics 108.)
3 units, Aut (Diaconis)

163. Symmetric Functions and Algebraic Combinatorics—(Enroll in Mathematics 112.)
3 units, Win (Diaconis)

191. Senior Project—Restricted to Computer Science students. Group or individual projects under faculty direction. Register using the section number associated with the instructor.
1-6 units, any quarter (Staff)

191W. Writing Intensive Senior Project—Restricted to Computer Science students. Group or individual projects under faculty direction. Register using the section number of an Academic Council member. (WIM)
1-6 units, any quarter (Staff)

192. Programming Service Project—Restricted to Computer Science students. Appropriate academic credit (without financial support) is given for volunteer computer programming work of public benefit and educational value.
1-3 units, any quarter (Staff)

193D. C++ and Object-Oriented Programming—C++ programming language and object-oriented programming paradigm. The major features of C++ 3.0 and the object design principles which apply generally in Object Oriented Languages. Intensive programming assignments. Prerequisites: knowledge of C and basic programming methodology as developed in 106B or 106X.
3 units, Win (Cain)

193H. Human-Computer Interface Tools—Design-practitioner oriented survey of tools for building interactive interfaces, including Director, Flash, and Basic Visual. Emphasis is on understanding interaction issues, exploiting tool capabilities, rapid prototyping, and extending design skills. Prerequisites: programming fundamentals at the level of 106B or 106X.
3 units, Aut (Staff)

193I. Internet Technologies—Programmer-oriented survey of the authoring, distributing, and browsing technologies. The role, use, and implementation of current Internet tools. Topics: TCP/IP; namespace, connections, and protocols. Client/server structures. Web/HTTP/HTML techniques for text, images, links, and forms. Server side programming, CGI scripts. Security and privacy issues. Programming projects on client- and server-side may be in Perl or Java. Languages are introduced as needed. Emphasis is on understanding, exploiting, and extending Internet technologies. Prerequisites: programming fundamentals at the level of 106B or 106X, and the ability to build and debug programs in a Unix environment.
3 units, Spr (Parlante)

193J. Programming in Java—Hands-on experience to gain practical Java programming skills. Topics: object-oriented programming (classes, objects, messaging, inheritance), Java language features (interfaces, exceptions, packages, concurrency, garbage collection), use of the built-in packages (lang, util, io, networking, awt, swing), applets and servlets, security and verification, Java implementation and the virtual machine.

Intensive programming assignments. Prerequisite: knowledge of C language and programming experience at the level of 106B, 106X.
3 units, Win (Cain)

193K. Advanced Java Applications—Tour of the advanced applications possible in Java. Possible topics: portable GUIs in Swing and distributed applications with RMI, and the various supporting technologies, e.g., concurrency, reflection, and serialization. Prerequisite: mastery of Java, e.g., 193J.
2 units, Spr (Parlante)

193L. Programming in LISP—Introduction to problem solving in the LISP language, focusing on the functional programming paradigm. Topics: recursion, list manipulation, mapping, functional arguments, destructive processing, macros, I/O, Lisp implementation, environments, packages, efficiency, object-oriented programming, classes, and methods. Term project. Prerequisite: 106B or 106X, or equivalent.
3 units (McCarthy)

193W. Microsoft Windows Programming—The fundamentals of programming on the Microsoft Windows platform, focusing on the use of Microsoft Foundation Class (MFC) framework. Other aspects of Windows programming including Microsoft's COM and OLE object models. Requires a significant amount of programming. Prerequisite: knowledge of C++ at the level of 108 or 193D.
3 units, Win (Young)

194. Software Project—Student teams complete a significant programming project through the phases of design, specification, coding, and testing under faculty supervision. Lectures on software engineering methodologies. Implementation; well-written proposal, specification, and software design document; demonstration of a prototype design and the final product. Prerequisite: 108. (WIM)
3 units, Win (Young)
Spr (Plummer)

196. Microcomputer Consulting—Overview of computer consulting, focusing on Macintosh and IBM-compatible systems. Topics: operating systems, networks, troubleshooting, and consulting methodology. Biweekly lectures emphasize on-campus computing environments. Students work as consultants in campus computer clusters and in residences. Prerequisite: 1C.
2 units, Aut, Spr (Staff)

197. Mainframe and Workstation Computer Consulting—Computer consulting in a workstation and server environment, focusing on the UNIX operating system under the SUN, HP, and SGI hardware systems. Topics: UNIX fundamentals, consulting tips, networking, and systems administration. Students work as on-duty consultants at the Sweet Hall and Terman computer clusters. Pre- or corequisite: 1U.
2 units, Win, Spr (Staff)

198. Teaching of Computer Science—Teach a small discussion section of 106A while learning the fundamentals of teaching a programming language at the introductory level. Two workshops/one meeting weekly on introductory material in general, 106 specifically, and teaching techniques. Application and interview required; see the 198 coordinator in CS for information. Prerequisite: 106B or 106X.
4 units, Aut, Win, Spr (Roberts, Chong, Burgess)

199. Independent Work—Special study under faculty direction, usually leading to a written report. Letter grade given; if this is not appropriate, enroll in 199P. Register using the section number associated with the instructor.
any quarter (Staff)

199P. Independent Work—Like 199, but graded Satisfactory/No Credit.
any quarter (Staff)

UNDERGRADUATE AND GRADUATE

200. Undergraduate Colloquium—Strongly recommended for junior-year CS majors as a way to build contacts with faculty. Weekly presentations by faculty and people from industry informally describing their views of computer science as a field and their experience as computer scientists. (AU)
1 unit, Aut (Plummer)

201. Computers, Ethics, and Social Responsibility—Primarily for majors entering computer-related fields. Analysis of the ethical and social issues related to the development and use of computer technology. Introduction to the relevant background in ethical theory, and the social, political, and legal considerations. Analysis of scenarios in specific problem areas: privacy, reliability and risks of complex systems, and the responsibility of professionals for the applications and consequences of their work. Prerequisite: 106B or 106X. (WIM)
**4 units, Win (Roberts)*

205. Mathematical Methods for Robotics and Vision—Overview of some of the mathematical background necessary for research in robotics and vision. Possible topics: geometric meaning of linear algebra concepts; Singular Value Decomposition; Schur Decomposition; differential equations; dynamic systems and stochastic estimation (Kalman filtering); vector and tensor calculus; calculus of variations. Prerequisites: 106B or 106X; Mathematics 51 and 113; or equivalents.
3 units, Aut (Tomasi)

206. Technical Foundations of Electronic Commerce—As the internet and wide-area networks are increasingly used to conduct commerce, computer scientists need to understand the nature of economic mechanisms, e.g., auctions, and devise the ways to implement them efficiently. Relevant economic theories. Lab to design and implement a substantial application in small groups. Prerequisites: sufficient mathematical maturity to follow basic combinatorial and probabilistic arguments, and ability to code in either C++ or Java.
3 units, Spr (Shoham, Boneh, Ullman)

211. Logic Design—(Enroll in Electrical Engineering 275.)
3 units, Aut, Win (McCluskey)

212. Computer Architecture and Organization—(Enroll in Electrical Engineering 282.)
3 units, Aut, Win (Staff)

221. Artificial Intelligence: Principles and Techniques—(Only one of 121 or 221 counts towards CS degree requirements.) Broad technical introduction to core concepts and techniques in artificial intelligence. Topics: search, planning, constraint satisfaction. knowledge representation, probabilistic models, machine learning, neural networks, vision, robotics, and natural language understanding. Prerequisites: 103B or 106X, or 109 and 157, or Philosophy 160A and exposure to basic concepts in probability. Recommended: facility with basic differential calculus.
**4 units, Aut (Koller)*

222. Knowledge Representation—Declarative knowledge representation methods in artificial intelligence. Topics: time and action, defaults, compositional modeling, object-oriented representation, inheritance, ontologies, knowledge on the Web, knowledge servers, multiple views, qualitative modeling. Prerequisite: basic familiarity with logic. Recommended: prior exposure to artificial intelligence as in 121/221.
3 units, Win (Fikes)

223A. Introduction to Robotics—Topics: manipulator kinematics and inverse kinematics; manipulator dynamics, motion, and force control;

* May be taken for 3 units by graduate students

motion planning and robot programming. Recommended: knowledge of matrix algebra.
3 units, Win (Khatib)

223B. Introduction to Computer Vision—Fundamental issues and techniques of computer vision. Image formation, edge detection and image segmentation, stereo, motion, shape representation, recognition. Project or final. Prerequisite: 205 or equivalent.
3 units, Win (Tomasi)

224M. Multi-Agent Systems—Aimed at advanced undergraduate, master's levels, and interested Ph.D. students. Various aspects of extending AI theories and techniques from the single-agent case to the multi-agent (MA) case. Topics: MA knowledge representation, planning, reasoning under uncertainty, learning, coordination mechanisms, and automated negotiation. Emphasis is on representation techniques and algorithms, the former drawn from logic, decision theory, and game theory. There are no programming assignments or textbooks on the topic. Limited enrollment. Prerequisites: knowledge of basic probability theory, first-order logic, and algorithms.
3 units, Win (Shoham)

224N. Natural Language Processing—(Enroll in Linguistics 237.)
3 units, Spr (Manning)

225A. Experimental Robotics—Hands-on experience with robotic manipulation and navigation systems. Topics: kinematic and dynamic control of motion, compliant motion and force control, sensor-based collision avoidance, motion planning, assembly planning, task specifications, and robot-human interfaces. Limited enrollment. Prerequisite: 223A.
3 units, Spr (Khatib)

225B. Robot Programming Laboratory—Hands-on introduction to the techniques of robot programming for robotics and non-robotics students. Series of guided exercises in which students program mobile robots to exhibit increasingly complex behavior (simple dead reckoning and reactivity, planning and map building, communication and cooperation). Topics: basics of motor control and sensor characteristics; sensor fusion, model construction, and robust estimation; control regimes (fuzzy control and potential fields); active perception; reactive planning architectures; various topics in sensor-based control, including vision-guided navigation. Student programmed robot contest. Programming is in C on Unix or Windows machines, done in teams. Prerequisites: Basic programming skills at the level of 106B, 106X, 205, or equivalent.
**4 units, Aut (Konolige)*

226. Knowledge-Based Systems and Applications—Knowledge-based (expert) system technology is the most widely-used application technology to emerge from AI. Topics: basics of KBS and ES; tech transfer from research to industry; knowledge engineering, KB programming, knowledge acquisition methodology; evolution of the technology as applied to business and government problems, current and future impact. Case studies, readings. System building project possible. Some guest lectures.
3 units, not given 2000-01

227. Reasoning Methods in AI—Technical presentation of algorithmic techniques for problem solving in AI. Combines formal algorithmic analysis with description of recent applications. Topics: propositional satisfiability, constraint satisfaction, planning and scheduling, diagnosis and repair. Focus is on recent results. Prerequisites: familiarity with the basic notions in data structures and design and with techniques in the design and analysis of algorithms. Recommended: previous or concurrent course in AI.
3 units, Spr (Nayak)

228. Probabilistic Models in Artificial Intelligence—Probabilistic modeling languages suitable for representing complex domains, algorithms for reasoning and decision making using these representations, and learning these representations from data. Focus is on graphical modeling languages such as Bayesian belief networks, extensions to temporal modeling using hidden Markov models and dynamic Bayesian networks, and extensions to decision making using influence diagrams and Markov decision processes. Recent applications to domains (speech recognition, medical diagnosis, data mining, statistical text modeling, and robot motion planning). Prerequisites: understanding of basic concepts in probability theory and in design and analysis.
3 units, Win (Koller)

229. Statistical Learning—Survey of major research areas in pattern recognition and statistical learning. Topics: foundations of statistical pattern recognition, parametric and non-parametric density estimation, linear and nonlinear classifiers, decision trees, Bayesian and neural networks, reinforcement learning, learning theory, and recent trends (e.g., boosting and support vector machines). Focus is on the underlying concepts and their application to various problems in vision, speech, language processing, animation, control, etc. Prerequisites: background in linear algebra, basic probability theory, and statistics; ability to write computer programs in one or more commonly used languages.
3 units, Win (Bregler)

237. Advanced Numerical Analysis—Three-quarter graduate sequence designed to acquaint students in mathematical and physical sciences and engineering with the fundamental theory of numerical analysis. Examples from applications.

237A. Numerical Linear Algebra—First in a three-quarter graduate sequence. Solution of systems of linear equations: direct methods, error analysis, structured matrices; iterative methods and least squares. Parallel techniques. Prerequisites: 106A, 137, Mathematics 103 or 113.
3 units, Aut (Golub)

237B. Numerical Solution of Initial Value Problems—Linear multistep methods and Runge-Kutta methods for ordinary differential equations: zero-stability, A-stability, and convergence. Differential algebraic equations. Parabolic partial differential equations: stability, convergence, and qualitative properties. Hyperbolic partial differential equations: stability convergence and qualitative properties. Prerequisites: Mathematics 130, 131.
3 units, Win (Staff)

237C. Numerical Solution of Boundary Value Problems—Elliptic partial differential equations: finite difference, finite element, and spectral methods. Iterative methods for solution of resulting algebraic equations: SOR, fast Poisson solvers, domain decomposition, multigrid methods, and Newton iteration. Prerequisites: Mathematics 130, 131.
3 units, Spr (Staff)

238. Parallel Methods in Numerical Analysis—Recent developments in parallel computer technology have made it necessary to reformulate numerical algorithms to exploit the full potential of this technology. Emphasis is on the different techniques for obtaining maximum parallelism in various numerical algorithms, especially those occurring when solving matrix problems and partial differential equations, and the subsequent mapping onto the computer. Implementation issues on parallel computers. Topics: parallel architecture, programming models, matrix computations, FFT, fast multiple methods, domain decomposition, graph partitioning. Prerequisite: 237A or Mechanical Engineering 200A, or consent of instructor. Recommended: familiarity with differential equations, and experience in advanced programming language such as F90, C, C++.
3 units, Win (Staff)

240. Advanced Topics in Operating Systems—Advanced study in OS topics and exposure to recent developments in OS research. Readings/ lectures on classic and new papers. Topics: virtual memory management, synchronization and communication, file systems, protection and secu-

* May be taken for 3 units by graduate students.

rity, operating system extension techniques, fault tolerance, and the history and experience of systems programming. Prerequisite: 140 or equivalent.
3 units, Win (Staff)
Spr (Rosenblum)

241. Advanced Topics in Internet Technologies and Systems—Architecture, design, and implementation of Internet-scale software infrastructure (services and applications). Achieving scalability, high availability, and robustness through modular software structure and performance tradeoffs, including harvest vs. yield and consistency vs. availability. Cluster-based runtime systems for Internet workloads, implementation and deployment challenges, economics of deploying and operating a service. Extending Internet services to mobile and post-PC computing devices. Service-centric view of the Internet, including composition of services and mass customization. Research agenda for Internet-scale services. Project: build and deploy an Internet-scale service prototype. Limited enrollment. Prerequisites: 193I or equivalent experience; 240 and 244A.
3 units, Win (Fox)

242. Programming Languages—The basic elements of programming languages and programming paradigms: functional, imperative, and object-oriented. Introduction to formal semantic methods. Modern type systems, higher-order functions and closure, exceptions and continuations. Runtime support for different language features. Emphasis is on separating the different elements of programming languages and styles. First half uses Lisp and ML to illustrate concepts; second half a selection of object-oriented languages. Prerequisite: 107, or experience with Lisp, C and some object-oriented language.
3 units, Aut (Mitchell)

243. Advanced Compiling Techniques—The theoretical and practical aspects of building modern compilers. Topics: intermediate representations, basic blocks and flow-graphs, dataflow analysis, register allocation, global code optimizations, and interprocedural analysis. Prerequisite: 143 or equivalent.
**4 units, Win (Lam)*

244A. Introduction to Computer Networks—The principles and practice of computer networking, with emphasis on the Internet. The structure and components of computer networks, packet switching, layered architectures, TCP/IP, physical layer, error control, window flow control, local area networks (Ethernet, Token Ring; FDDI), network layer, congestion control, quality of service, multicast. Students enrolling in 244A Winter Quarter must have completed 140, or equivalent.
**4 units, Aut (Tobagi) (enroll in Electrical Engineering 284)*
Win (McKeown)

244B. Distributed Systems—Distributed operating systems and applications issues, emphasizing high-level protocols and distributed state sharing as the key technologies. Topics: distributed shared memory, object-oriented distributed system design, distributed directory services, atomic transactions and time synchronization, file access, process scheduling, process migration and remote procedure call focusing on distribution, scale, robustness in the face of failure, and security. Prerequisites: 240, 244A.
3 units, Spr (Staff)

244C. Distributed Systems Project—Companion project option for students taking 244B. Corequisite: 244B.
3-6 units, Spr (Staff)

245. Database System Principles—File organization and access, buffer management, performance analysis, and storage management. Database system architecture, query optimization, transaction management, recovery, concurrency control. Reliability, protection, and integrity. Design and management issues. Prerequisites: 145, 161.
3 units, Win (Garcia-Molina)

247A. Human-Computer Interaction: Interaction Design Studio—Intended as preparation for project-based courses, e.g., 377 and 447/ Mechanical Engineering 293. Systematic presentation and experience with the methods used in interaction design, including needs analysis, user observation, idea sketching, concept generation, scenario-building, storyboards, user character stereotypes, usability analysis, and market strategies. Prerequisite: 147 or Mechanical Engineering 101.
3-4 units, Aut, Win, Spr (Staff)

247B. Contextual and Organizational Issues in Human-Computer Interaction—(Enroll in Management Science and Engineering 430.)
3-4 units, Spr (Hinds)

248. Introduction to Computer Graphics—(Only one of 148 or 248 counts towards CS degree requirements.) The fundamentals of input, display, and hardcopy devices, scan conversion of geometric primitives, 2D and 3D geometric transformations, clipping and windowing, scene modeling and animation, algorithms for visible surface determination, introduction to local and global shading models, color, and photorealistic image synthesis. Written assignments and programming projects. Prerequisites: 107, Mathematics 103 or equivalent.
**5 units, Aut (Levoy)*

248V. Introduction to Scientific Visualization—Alternative introduction to computer graphics, emphasizing techniques for visualizing multidimensional data, especially continuous functions of two and three dimensions. Topics: the graphics pipeline, visualization of scalar, vector, and tensor fields, volume rendering, isosurface extraction, display of level sets and polygon meshes, and programming toolkits and interactive systems for data visualization; the design of visual metaphors for medical imaging, fluid flow, geophysics, meteorology, and other applications. Written assignments and programming projects. Not a substitute for 248 as a prerequisite for upper-level graphics courses. Prerequisites: basic calculus, linear algebra, and ability to program in C or C++.
**4 units, Spr (Fedkiw, Levoy)*

249. Object-Oriented Programming from a Modeling and Simulation Perspective—Object-oriented programming techniques and issues, emphasizing programming as modeling and simulation. Topics: large-scale software development approaches, encapsulation, use of inheritance and dynamic dispatch, design of interfaces and interface/ implementation separation, exception handling, design patterns, minimalizing dependencies and value-oriented programming. The role of programming conventions/style/restrictions in surviving object-oriented programming for class libraries, frameworks, and programming-in-the-large; general techniques for object-oriented programming. Prerequisites: knowledge of C and basic programming methodology as developed in 106B or 106X, 107, basic knowledge of C++ (may be taken concurrently). Recommended: 193D.
3-5 units, Win (Staff)

255. Introduction to Cryptography and Computer Security—Intended for advanced undergraduates and graduate students. Introduction to the basic theory and practice of cryptographic techniques used in computer security. Topics: encryption (single and double-key), digital signatures, pseudo-random bit generation, authentication, electronic commerce (anonymous cash, micropayments), key management, zero-knowledge protocols. Prerequisite: basic understanding of probability theory.
3 units, Win (Boneh)

256. Formal Methods for Reactive Systems—Formal methods for specification, verification, and development of concurrent and reactive programs. Reactive systems: syntax and semantics, fairness requirements. Specification language: temporal formulas (state, future, and

* May be taken for 3 units by graduate students.

past) and w-automata. Hierarchy of program properties: safety, guarantee, obligation, response, persistence, and reactivity. Deductive verification of programs: verification diagrams and rules, completeness. Modularity. Parameterized programs. Algorithmic verification of finite-state programs. Prerequisite: 157 or Philosophy 160A, or equivalent.
3 units, Win (Manna)

256L. Formal Methods for Concurrent and Reactive Systems Laboratory
2 units, Win (Manna)

257. Automated Deduction and its Applications—Proving theorems and extracting information from proofs. Uses in software engineering (program specification, synthesis, and verification) and artificial intelligence (commonsense and robotic planning, natural-language understanding). The foundations of logic programming. Deductive tableaux, nonclausal resolution, skolemization, building theories into unification and inference rules, term rewriting, inductive theorem proving. The design of theorem provers. Prerequisite: 157.
3 units (Staff)

258. Introduction to Programming Language Theory—Syntactic, operational, and semantic issues in the mathematical analysis of programming languages. Type systems and non-context-free syntax. Universal algebra and algebraic data types. Operational semantics given by rewrite rules; confluence and termination. Scott-semantics for languages with higher-type functions and recursion. Treatment of side-effects. Prerequisites: 154, 157 or Philosophy 160A.
3 units, Win (Mitchell)

260. Concrete Mathematics—Mathematics for the analysis of algorithms: recurrences, summations, generating functions, asymptotics. Elementary combinatorics, discrete probability, and number theory. Prerequisites: 103B or 103X, Mathematics 42, or equivalent.
3 units (Staff)

261. Optimization and Algorithmic Paradigms—Algorithms for network optimization: max-flow, min-cost flow, matching, assignment, and min-cut problems. Introduction to linear programming. Use of LP duality for design and analysis of algorithms. Approximation algorithms for NP-complete problems such as Steiner Trees, Traveling Salesman, and scheduling problems. Randomized algorithms. Introduction to on-line algorithms.
3 units, Win (Plotkin)

270A. Introduction to Medical Informatics: Fundamental Methods—(Same as Biomedical Informatices 210A.) Issues in the modeling, design, and implementation of computational systems for use in biomedicine. Topics: controlled terminologies in medicine and biological science, ontologies, fundamental algorithms, basic knowledge representation, information dissemination and retrieval. Emphasis is on the principles of modeling data and knowledge in biomedicine and on the translation of resulting models into useful automated systems.
3 units, Aut (Musen, Altman)

270B. Introduction to Medical Informatics: Systems and Requirements—(Same as Biomedical Informatics 210B.) Survey of the major application areas in medical informatics, including clinical information systems, imaging systems, bioinformatics, public policy, decision support, and signal processing. Emphasis is on the system requirements, relevant data, algorithms, and implementation issues in each area. Prerequisite: 270A.
3 units, Win (Shahar, Dev)

271. Decision-Making Methods for Biomedicine—For undergraduates or graduate students, building on concepts introduced in 270B. Intermediate biomedical decision making and survey of the methods for the implementation of such concepts in computer-based decision-support tools. Emphasis is on Bayesian statistics, decision analysis, cost-benefit analysis, neural networks, artificial intelligence/expert systems, belief networks, influence diagrams, and the synergies among such approaches. Prerequisites: 270B and at least one programming course.
3 units (Higgens, Garber, Owens, Sanders) not given 2000-01

272. Medical Informatics Project Course—(Same as Biomedical Informatics 212.) For students who have completed 270A, 270B, 271 or 274, and who wish to implement those ideas in a computer program. Students may take 274 concurrently and complete a project that is coordinated between the two courses. Prerequisites: programming experience, 270B.
3 units, Sum (Altman, Koza)

274. Representations and Algorithms for Computational Molecular Biology—(Same as Biomedical Informatics 214.) Introduction to basic computational issues and methods used in bioinformatics, including access and use of biological data sources on the Internet. Topics: basic algorithms for alignment of biological sequences and structures, computing with strings, phylogenetic tree construction, hidden Markov models, computing with networks of genes, basic structural computations on proteins, protein structure prediction, protein threading techniques, homology modeling, molecular dynamics and energy minimization, statistical analysis of 3D biological data, integration of diverse data sources, knowledge representation and controlled terminologies for molecular biology, graphical display of biological data, genetic algorithms and genetic programming applied to biological problems. See instructor for unit options. Prerequisites: programming skills and understanding of matrix algebra.
1-4 units, Spr (Altman, Koza)

275A. Musical Information: An Introduction—(Enroll in Music 253.)
1-4 units, Win (Selfridge-Field)

275B. Seminar: Musical Representation and Computer Analysis—(Enroll in Music 254.)
1-4 units, Spr (Selfridge-Field)

298. Seminar on Teaching Introductory Computer Science—Opportunity for faculty and undergraduate and graduate students who are interested in teaching to discuss specific topics raised by teaching computer science at the introductory level. Prerequisite: consent of instructor.
1-3 units, Aut, Win, Spr (Roberts)

PRIMARILY FOR GRADUATE STUDENTS

300. Departmental Lecture Series—For first-year Computer Science Ph.D. students. Presentations by members of the department faculty, each describing informally his or her current research interests and views of computer science as a whole. (AU)
1 unit, Aut (Staff)

309. Industrial Lectureships in Computer Science—The department invites an outstanding computer scientist to give a course in his/her specialty. Lecturers and topics change yearly; courses may be taken repeatedly. See *Time Schedule* for offerings.
3 units

312. Processor Design—(Enroll in Electrical Engineering 382.)
3 units (Staff) given 2001-02

315A. Parallel Computer Architecture and Programming—The principles and tradeoffs in the design of parallel architectures. Emphasis is on naming, latency, bandwidth, and synchronization in parallel machines. Case studies on shared-memory, message-passing, dataflow, and data-parallel machines illustrate techniques. Architectural studies and lectures on techniques for programming parallel computers. Programming assignments on one or more commercial multiprocessors. Prereq-

uisites: Electrical Engineering 282, and reasonable programming experience.
3 units (Staff) not given 2000-01

315B. Parallel Programming Project—Continuation of 315A. A significant parallel programming project is required using shared-memory, message-passing, or data-parallel machines. Lectures on parallel programming languages and their implementation, performance debugging of parallel programs, parallel data structures and algorithms. Prerequisite: 315A or consent of instructor.
3 units (Staff)

316A. Logic Synthesis of VLSI Circuits—(Enroll in Electrical Engineering 318.)
3 units, Win (DeMicheli)

316B. Computer-Aided System Design Laboratory—(Enroll in Electrical Engineering 319.)
3 units, Spr (DeMicheli)

317. Fault Tolerant Computing Systems—(Enroll in Electrical Engineering 489.)
3 units (Staff) alternate years, given 2001-02

318. Testing Aspects of Computer Systems—(Enroll in Electrical Engineering 488.)
3 units, alternate years, given 2001-02

319. Topics in Digital Systems—Advanced material is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number. See *Time Schedule* for topics currently being offered.
by arrangement

323. Common Sense Reasoning in Logic—Formalizing common sense knowledge and reasoning using situation calculus with nonmonotonic logics, especially circumscription. Variations of situation calculus. Formalizing context. Formalizing facts about knowledge. Prerequisite: basic knowledge of logic such as 157, or Philosophy 160A.
3 units, Aut (McCarthy)

326A. Motion Planning—For students interested in computer graphics, geometrical computing, robotics, and/or artificial intelligence. Computing object motions is central to many application domains (e.g., design, manufacturing, robotics, animated graphics, medical surgery, drug design). Basic path planning methods generate collision-free paths among static obstacles. Extensions include uncertainty, mobile obstacles, manipulating movable objects, and maneuvering with kinematic constraints. Configuration space is a unifying concept, geometric arrangements are a basic combinatorial structure. Theoretical methods with applications in various domains: assembly planning, radiosurgery, graphic animation of human figures.
3 units (Latombe) not given 2000-01

327A. Advanced Robotic Manipulation—Topics: redundant manipulators, robot motion/force control; kinematic singularities; inertial properties, dynamic performance, and robot design; macro/mini manipulator systems; mobile manipulator platforms; cooperative robots; sensor-based primitives, artificial potential field and force strategies. Prerequisites: 223A, consent of instructor.
3 units, Spr (Khatib)

328. Topics in Computer Vision—Fundamental issues of, and mathematical models for, computer vision. Sample topics: camera calibration, texture, stereo, motion, shape representation, image retrieval, experimental techniques. Student papers and project. Prerequisites: 205, 223B, or equivalents.
3 units, Apr (Tomasi) alternate years, not given 2001-02

329. Topics in Artificial Intelligence—Advanced material is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number.
1-3 units

336. Advanced Methods in Matrix Computation—Eigenvalue problems: perturbation theory, Lanczos method, Jacobi method. Parallel implementation. Singular value problems. Generalized eigenvalue problems. Polynomial equations. Prerequisite: 237A.
3 units (Golub)

337. Numerical Methods for Initial Boundary Value Problems—Initial boundary value problems are solved in different areas of engineering and science modeling phenomena, e.g., wave propagation and vibration, fluid flow, etc. Numerical techniques for such simulations are discussed in the context of applications. Emphasis is on stability and convergence theory for methods for hyperbolic and parabolic initial boundary value problems, and the development of efficient methods for these problems.
3 units, Spr (Staff)

339. Topics in Numerical Analysis—Advanced material is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number. See *Time Schedule* for current topics.
2-3 units, Aut (Van Huffel) alternate years, not given 2001-02
Win (Golub)

341. Advanced Topics in Data Communication—Readings/discussion are combined with topical lectures to familiarize students with a core of classic and new papers in the field of data networking. Emphasis is on understanding and applying existing work to new problems in the field, especially high-speed networking. Topics: network theory (the end-to-end argument), transport protocol performance (header prediction, checksum efficiency), cell relay (e.g., ATM and SONET), congestion control (Parekh's thesis, leaky bucket, fair queueing) and high-speed switching (input vs. output queueing, crossbars and banyans). Prerequisite: 244A.
3 units (Partridge)

342. Programming Language Design—Problems of programming language design and comparison of traditional solutions. Possible topics: formal semantics, implementation considerations, extensibility, very high level languages, evaluation of language designs, the innovative features of a variety of modern programming languages. Prerequisites: 242, 243.
3 units (Mitchell)

343. Topics in Compilers—Advanced topics in compilers. Topics change every quarter; course may be taken repeatedly for credit. Prerequisite: 243.
3 units, Spr (Lam)

344. Projects in Computer Networks—For students with a strong interest in computer networks from novel applications to physical layer coding schemes; software to hardware; theory to design-and-build. Teams of two or more complete a small research project of sufficient quality and interest to merit presentation at a conference, or to form the basis of a new business, e.g., studies of network traces, network traffic visualization tools, home-networking, analysis of performance of cable-modems, novel web applications, or novel router architecture. Enrollment limited to 20. Prerequisites: 244A; or Electrical Engineering 284 and 384A. Recommended: 244B; and Electrical Engineering 384B or 384C.
3 units, Aut (McKeown) alternate years, not given 2001-02

345. Advanced Topics in Database System—Advanced topics in the area of database and information systems. Content differs in each

offering; may be taken multiple times for credit. Prerequisite: 145.
3 units, Spr (Staff)

346. Database System Implementation—A major database system implementation project realizes the principles and techniques covered in earlier courses. Students independently build a complete database management system, from file structures through query processing, with a personally designed feature or extension. Lectures on project details and advanced techniques in database system implementation, focusing on query processing and optimization. Guest speakers from industry on commercial DBMS implementation techniques. Prerequisites: 145, 245. Recommended: programming experience in C++.
**5 units, Aut (Widom)*

347. Transaction Processing and Distributed Databases—The principles and system organization of distributed databases. Data fragmentation and distribution, distributed database design, query processing and optimization, distributed concurrency control, reliability and commit protocols, and replicated data management. Distributed algorithms for data management: clocks, deadlock detection, and mutual exclusion. Heterogeneous and federated distributed database systems. Overview of commercial systems and research prototypes. Prerequisites: 145, 245.
3 units, Spr (Garcia-Molina)

348A. Computer Graphics: Mathematical Foundations—The mathematical tools needed for the geometrical aspects of computer graphics. Fundamentals: homogeneous coordinates, transformations, and perspective. Theory of parametric and implicit curve and surface models: polar forms, Bezier arcs and de Casteljau subdivision, continuity constraints, B-splines, tensor product, and triangular patch surfaces. Representations of solids and conversions among them. Subdivision surfaces and multiple-solution representations of geometry. Mesh generation, simplification, and compressions. Geometric algorithms for graphics problems, with applications to ray tracing, hidden surface elimination, etc. Prerequisites: linear algebra and discrete algorithms.
**4 units, Win (Guibas)*

348B. Computer Graphics: Image Synthesis Techniques—Intermediate level, emphasizing the sampling, shading, and display aspects of computer graphics. Topics: local and global illumination methods including radiosity and distributed ray tracing, texture generation and rendering, volume rendering, strategies for anti-aliasing and photo-realism, human vision and color science as they relate to computer displays, and high-performance architectures for graphics. Written assignments and programming projects. Prerequisite: 248 or equivalent. Recommended: exposure to Fourier analysis or digital signal processing.
**4 units, Spr (Hanrahan)*

348C. Computer Graphics: Animation Techniques—Overview of computer animation techniques. Topics: traditional principles of animation, physical simulation, procedural methods, and motion capture based animation. Focus is on computer science aspects of animation. The basics, e.g., kinematic and dynamic modeling techniques to exploration of current research topics such as motion re-targeting, learning movements and behaviors, and video based modeling and animation). Hands-on animation experience through class projects.
3 units, Spr (Bregler)

348D. Vision and Image Processing—(Enroll in Psychology 267.)
1-3 units (Heeger) alternate years, given 2001-02

349. Topics in Programming Systems—Advanced material often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may enroll repeatedly in a course with this number. See Time Schedule for topics currently being offered.
3 units, Spr (Prabhakar)

351. Topics in Complexity Theory and Lower Bounds—Focus is on one of: basic machine models and complexity measures—their properties and relationships, complexity classes and their properties, reductions and complete problems, concrete representative problems from important complexity classes. Techniques for establishing limits on the possible efficiency of algorithms, and concrete lower bounds based on the following models of computation: decision trees, straight line programs, communication games, branching programs, PRAMs, boolean circuits. Approximation algorithms and the complexity of approximations. Pseudo-randomness and cryptography. Prerequisite: 154, or equivalent.
3 units (Motwani) alternate years, given 2001-02

352. Foundations of Control Structures—Theory of constructs for controlling program execution. Theories of serial control: verification conditions, partial correctness assertions, weakest preconditions, dynamic logic. Models of serial control: state functions and relations, regular expressions, dynamic algebras. Theories of parallel control: temporal logic, process algebra, CCS, CSP. Models of parallel control: state trajectories, synchronization trees, execution traces, partial orders, Petri nets, event structures, metric spaces, non-well-founded sets. Notions of time: ordered, real, probabilistic, linear, branching. Semantic equivalences. Structural operational semantics. Related soundness, completeness, and complexity issues. Prerequisite: 258 or consent of instructor.
3 units (Pratt)

353. Algebraic Logic—Algebraic methods relevant to computer science. Lattice theory: partial orders, monoids, closure systems, topologies, fixpoint theorems. Universal algebra: HSP, free algebras, equational theories, Birkhoff's theorem, completeness of equational logic. Algebras for logic: Boolean, Heyting, cylindric. Categories: limits, adjunctions, algebraic theories, enriched categories. Prerequisite(s): 157; or Philosophy 160A, 161; or equivalents.
3 units, Aut (Pratt)

354. Probabilistic Reasoning in Computing—The basics of (Bayesian) probability theory as applied to computing and intelligence systems. Emphasis is on working through applications and understanding relevant theory. Relevant probability theory and techniques: interpretations, graphical and network models, information theory, decision theory, inference, and "alternative" approaches. Probabilistic aspects of computational problems in learning, search, data analysis, neural, and dynamic systems. Some topics by guest lecturers. Prerequisites: 106B or 106X, 221, a knowledge of basic statistical measures as in Psychology 60, and basic math.
3 units (Staff)

355. Advanced Topics in Cryptography—For graduate students. Topics: pseudo-random generation, zero knowledge protocols, elliptic curve systems, threshold cryptography, security analysis using random oracles, lower and upper bounds on factoring and discrete log. Prerequisite: 255.
3 units, Spr (Boneh)

356. Automatic Formal Verification Techniques—Automatic methods for formally verifying hardware, protocol, and software system designs. Topics: state graph and automata models of system behavior. Automata on infinite strings. Linear and branching-time temporal logic. Model-checking. Modeling real-time systems. Analysis methods based on Boolean formulas, and other ways of coping with the "state explosion problem." Exploiting abstractions. Applications to circuits, algorithms, and protocols. Case studies use a variety of verification tools. Prerequisite: 154 or 254. Recommended: good understanding of basic automata and complexity theory, and undergraduate-level background in computer science.
3 units (Dill) alternate years, given 2001-02

* May be taken for 3 units by graduate students.

357. Topics in Formal Methods—Formal methods for the specification, verification, analysis, and systematic development of real-time and hybrid systems. Hybrid systems involve continuous changes and discrete transitions. Computational models: timed and phase transition systems, timed and hybrid automata. Specification: timed and hybrid statecharts, metric and hybrid temporal logics, duration calculus. Statecharts. Structured specification. Verification rules and diagrams. Refinement techniques. Algorithmic verification of finite-state systems. Advanced research topics. Prerequisite: 256 or equivalent.
3-5 units, Spr (Manna, Sipma)

358. Topics in Programming Language Theory—Possible topics of current research interest in the mathematical analysis of programming languages: structured operational semantics, domain theory, semantics of concurrency, rich type disciplines, problems of representation independence, and full abstraction. May be repeated for credit. Prerequisites: 154, 157, 258, or equivalents.
3 units (Mitchell)

359. Topics in Theory of Computation—Advanced material is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number. See *Time Schedule* for topics currently being offered.

361A. Advanced Algorithms—Advanced data structures: union-find, self-adjusting data structures and amortized analysis, dynamic trees, Fibonacci heaps, universal hash function and sparse hash tables, persistent data structures. Advanced combinatorial algorithms: algebraic (matrix and polynomial) algorithms, number theoretic algorithms, group theoretic algorithms and graph isomorphism, on-line algorithms and competitive analysis, strings and pattern matching, heuristic and probabilistic analysis (TSP, satisfiability, cliques, colorings), local search algorithms. Prerequisite: 161 or 261, or equivalents.
3 units (Motwani) alternate years, given 2001-02

361B. Advanced Algorithms—Topics: exact and approximate algorithms for various combinational optimization problems, e.g., generalized and multicommodity flow, constrained forest problems, scheduling, and the max-cut problem multidimensional search. Linear programming; LP duality, ellipsoid dimension. Lattice reduction and strongly-polynomial algorithms.
3 units, Spr (Plotkin) alternate years, not given 2001-02

365. Randomized Algorithms—Design and analysis of algorithms that use randomness to guide their computations. Basic tools, from probability theory and probabilistic analysis, that are recurrent in algorithmic applications. Randomized complexity theory and game-theoretic techniques. Algebraic techniques. Probability amplification and derandomization. Applications: sorting and searching, data structures, combinatorial optimization and graph algorithms, geometric algorithms and linear programming, approximation and counting problems, parallel and distributed algorithms, on-line algorithms, number-theoretic algorithms. Prerequisites: 161 or 261, Statistics 116, or equivalents.
3 units (Motwani) alternate years, not given 2001-02

367A. Parallel Computation—Introduction to theoretical issues in parallel computation. Properties of parallel computation models and algorithm design techniques specific to each model, including systolic arrays, mesh-connected computers, hypercube-related networks, and PRAM. Topics: algorithms for sorting, connected components, shortest paths, and other basic problems. Upper and lower bounds for randomized and deterministic routing on hypercube and related networks. Techniques for reducing the processor-time product for PRAM algorithms.
3 units (Plotkin)

367B. Parallel Computation—Advanced parallel algorithms. Focus is on developing techniques for the design of parallel algorithms on the PRAM model of computation and its derivatives. Possible topics: efficient randomized parallel algorithms for symmetry-breaking and related problems. Derandomization techniques. Parallel sorting. Deterministic and randomized parallel algorithms for flows and related problems; assignment problem, matching in general graphs. Evaluation of straight-line code, P-complete problems.
3 units (Plotkin)

368. Geometric Algorithms—Graduate-level introduction to the basic techniques used in the design and analysis of efficient geometric algorithms including: convexity, triangulation, sweeping, partitioning, and point location. Voroni and Delaunay diagrams. Intersection and visibility problems. Recent developments using random sampling methods. Emphasizes data structures of general usefulness in geometric computing and the conceptual primitives appropriate for manipulating them. Impact of numerical issues in geometric computation. Applications to motion planning, visibility preprocessing, model-based recognition, and GIS. Prerequisite: 161.
3 units, Spr (Guibas)

369. Topics in Analysis of Algorithms—Advanced material is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number. See *Time Schedule* for topics currently being offered.
3 units

377. Topics in Human-Computer Interaction—Topics of current research interest in human-computer interaction. Contents change each quarter. May be repeated for credit.
3-4 units, Aut, Win, Spr

378. Phenomenological Foundations of Cognition, Language, and Computation—Critical analysis of theoretical foundations of the cognitive approach to language, thought, and computation. Contrasts of the rationalistic assumptions of current linguistics and artificial intelligence with alternatives from phenomenology, theoretical biology, critical literary theory, and socially-oriented speech act theory. Emphasis is on the relevance of theoretical orientation to the design, implementation, and impact of computer systems as it affects human-computer interaction.
3-4 units, Spr (Winograd)

379. Interdisciplinary Topics—Advanced material that relates computer science to other disciplines is often taught for the first time as a "topics" course, perhaps by a faculty member visiting from another institution. Students may therefore enroll repeatedly in a course with this number. See *Time Schedule* for topics being currently offered.
by arrangement

390A,B,C. Curricular Practical Training—Provides educational opportunities in high-technology research and development labs in the computing industry. Qualified computer science students engage in internship work and integrate that work into their academic program. Students register during the quarter they are employed and must complete a research report outlining their work activity, problems investigated, key results, and any follow-on projects they expect to perform. Meets the requirements for Curricular Practical Training for students on F-1 visas. 390 A, B and C may each be taken only once.
1 unit, any quarter (Motwani)

393. Computer Laboratory—For CS graduate students. A substantial computer program is designed and implemented; written report required. Recommended as a preparation for dissertation research. Register using the section number associated with the instructor. Prerequisite: consent of instructor.
any quarter (Staff)

394. Business Management for Computer Scientists and Electrical Engineers—Focus is on the functional areas necessary for making successful business decisions. Topics: corporate strategy, new product development, marketing, sales, distribution, customer service, and financial accounting. How to identify and analyze issues in each of these areas in a rapidly changing world. A framework and tool set is developed for formulating, evaluating, and recommending action from the general manager point of view and for communicating and defending ideas in a team environment. Enrollment limited to 60. See http://www.stanford.edu/class/cs394/ . Prerequisite: graduate student in Computer Science or Electrical Engineering.

3-4 units (Gibbons, Liddle) not given 2000-01

395. Independent Database Project—For graduate students in Computer Science. Use of database management or file systems for a substantial application or implementation of components of database management system. Written analysis and evaluation required. Register using the section number associated with the instructor. Prerequisite: consent of instructor.

any quarter (Staff)

399. Independent Project

1-9 units, any quarter (Staff)

399P. Independent Project—Graded satisfactory/no credit.

1-9 units, any quarter (Staff)

EXPERIMENTAL

409. Formal Software Development—Introduction to formal methods for software development and the automation of these methods. Focus is on the composition of large formal specifications, and the refinement of specifications into efficient code. The foundations for specification composition and refinement are provided by category-theoretic concepts, e.g., colimits and sheaves. Topics: application-specific domain theories, formal requirement specifications, representation and use of programming knowledge, software architectures, algorithm and data structure design, program optimization techniques, datatype refinement, code generation, and system support. Hands-on exercises with a working system.

3 units (Smith, Green) alternate years, given 2001-02

426. Genetic Algorithms and Genetic Programming—The genetic algorithm is a domain-independent algorithm for search, optimization, and machine learning patterned after Darwinian natural selection and naturally occurring genetic operators such as recombination; mutation; gene duplication, deletion, regulation; and embryonic development. Genetic programming is a domain-independent automatic programming technique that extends the genetic algorithm to the breeding of populations of computer programs capable of producing human-competitive results. Topics: introduction to genetic algorithms and genetic programming; the mathematical basis for genetic algorithms; implementation on parallel computers and field-programmable gate arrays; applications to problems of system identification, control, classification, analysis of genome and protein sequences; automatic synthesis of the design of topology, sizing, placement, and routing of analog electrical circuits; automatic synthesis of controllers; and automatic synthesis of other complex network structures.

3 units, Spr (Koza)

444A. Software Development for Critical Applications—Introduction to current methods for developing safety-critical software (e.g., fly-by-wire avionics); and mission-critical software (e.g., Internet commerce). Topics: basic terminology, failure and fault taxonomies, hazard analysis techniques, failure mode analysis, fault tree analysis, software standards, formal methods, testing requirements, fault tolerance, probabilisitic models, and engineering techniques for critical systems from embedded systems to large-scale Internet applications. Students apply analysis techniques to example systems, use tools for specification, and implement example algorithms and applications.

3 units, Aut (Dill, Fox)

444N. Mobile and Wireless Networks and Applications—How mobility affects networks, systems, and applications. Mobility of devices and end-users has behavioral implications at all layers of the traditional Internet protocol stack, from the MAC layer up through the application layer. Handling mobility efficiently requires more information sharing between network layers than is typically considered. Topics: how mobility affects the layers of the protocol stack; and how it affects different functional aspects of systems, including security, privacy, file systems, resource discovery, resource management (including energy usage), personal on-line identities, and other areas. Emerging applications enabled by mobility. "Traditional" wireless networks, in which an underlying infrastructure is assumed; ad hoc mobile wireless networks, in which nodes may come and go and must form their own network infrastructure on the fly. Student groups design and implement mobile applications and system features of their choosing using network technologies such as WaveLan, Metricom's Ricochet network, the Palm-7, and Bluetooth. Prerequisites: 240, 244A, 244B, or equivalents.

3 units (Baker) not given 2000-01

446. Tools and Processes for Software—The fundamental concepts of software engineering: life-cycle models (waterfall, spiral, etc.), project and software metrics, quality assurance, software reuse. The development process: business process modeling, requirements engineering, analysis, design, implementation, testing, maintenance. Introduction to modeling techniques (UML and design patterns). Research challenges, with reviews of ongoing research by faculty and outside speakers on such topics as specification validation and software composition. Readings and modeling exercises. Focus throughout is on large-scale software development as seen in industry. UML and software development process may be taken for 1 unit. Prerequisites: prior software experience; graduate standing or consent of instructor.

1-3 units, not given 2000-01

447. Interdisciplinary Interaction Design—(Same as Mechanical Engineering 293.) Small teams develop innovative technology prototypes that combine product and interaction design. Focus is on software and hardware interfaces, interaction, design aesthetics, and some underpinnings of successful design: a reflective, interactive design process, group dynamics of effective interdisciplinary teamwork, and working with users. Prerequisite: 247A.

3-4 units, Spr (Winograd, Kelley)

448. Topics in Computer Graphics—In-depth study of an active research topic in computer graphics. Topic changes each quarter, e.g., exotic input and display technologies, graphics architectures, topics in modeling shape and motion, experiments in digital television, interactive workplaces, introduction to hand-drawn cartoon animation. Readings from literature and a project. May be taken repeatedly for credit. Prerequisite: 248 or consent of instructor.

448A. Experiments in Motion Capture

3 units, Aut (Bregler)

448B. Motion Study: An Introduction to Animation, Cartoon Physics, and Funny Walks—Preference to CS students with a graphics or animation specialization, and Art students from the Digital Arts program. Hands-on animation, providing a foundation for future work in computer graphics, digital art, and animation. The techniques, tools, and methods used by traditional animators. Through lectures, hands-on exercises, motion analysis, and screenings, students learn a variety of animation techniques and gain a basic control of timing, spacing, weight, and expressive motion. At the end of quarter, students have a short reel of their work plus new insight into the art of animation. Enrollment limited to 15. See http://graphics.stanford.edu/courses/cs448b/ .

3 units, Aut (Loeb)

448C. Interactive Workplaces

3 units, Aut (Hanrahan)

468. Topics in Geometric Algorithms—Advanced seminar covering different topics related to geometric computing. Recent offerings: shape matching, proximity and nearest-neighbor problems, visibility and motion planning, and collision detection. Readings from the literature and a presentation or a project required. May be taken multiple times for credit. Prerequisite: 368, or consent of instructor.
2 units, Aut, Win (Guibas)
Spr (Staff)

499. Advanced Reading and Research—For CS graduate students. Register using the section number associated with the instructor. Prerequisite: consent of instructor.
any quarter (Staff)

GRADUATE SEMINARS

510. Digital Systems Reliability Seminar—(Enroll in Electrical Engineering 385A.)
1-4 units, Aut, Win, Spr, Sum (McClusky)

523. Readings in Artificial Intelligence—Primarily for students planning to take the AI qualifying exam. A series of lectures and discussions on readings in all areas of artificial intelligence research. Prerequisite: 221.
3 units, Win (Staff)

525. Seminar on Knowledge Acquisition for Expert Systems—(Enroll in Biomedical Informatics 230.)
2 units, Spr (Musen) alternate years, not given 2001-02

528. Broad Area Colloquium for Artificial Intelligence, Geometry, Graphics, Robotics, and Vision—Weekly series of informal research talks on topics related to perceiving, modeling, manipulating, and displaying the physical world. The computational models and numerical methods underlying these topics. Brings together faculty and students in these five closely related areas. (AU)
1 unit, Aut, Win, Spr (Staff)

530. Applied Mathematics/Scientific Computing Seminar—(AU)
1 unit, Aut, Win, Spr (Staff)

531. Numerical Analysis/Scientific Computing Seminar—(AU)
1 unit, Aut, Win, Spr (Staff)

540. Seminar on Computer Systems—(Enroll in Electrical Engineering 380.)
1 unit, Aut, Win, Spr (Allison, Wharton)

545. Database Research Seminar—Presentations of current research and industrial innovation in information systems, sponsored by Infolab faculty. Topics: fundamental database technology, digital libraries, knowledge-based processing and advanced applications. Interaction with speakers. (AU)
1 unit, Aut, Spr (Wiederhold, Decker)

545I. Advanced Image Databases Seminar—Reading/demonstrations/analysis devoted to image and video databases as created by photographic, medical, and commercial sources. Emphasis is on combining image-derived and textual descriptors to retrieve on-line images. Issues: data structures and indexing schemes for real-time interaction, high-dimensional feature vectors for fast retrieval, metrics of closeness between query and stored vectors. Presentations by commercial and research image retrieval organizations illustrate the strengths and weaknesses of specific techniques. May be combined with a 395 project. (AU)
1 unit, Win (Firschein, Wiederhold)

547. Human-Computer Interaction Seminar—Weekly speakers on topics related to human-computer interaction design. (AU)
1 unit, Aut, Win, Spr (Winograd)

548. Distributed Systems Research Seminar—Recent research in distributed operating systems, computer communications, parallel machines, parallel programming, and distributed applications. Invited speakers from Stanford and elsewhere present topics and results of current interest. (AU)
1 unit, Spr (Staff)

559. Seminar on Mathematical Theory of Computation—Possible topics (vary each year): logic and its relation to computation, programming language analysis and design, specification and verification of software and hardware systems, theories of concurrency, approaches to static analysis and program state. Invited speakers present recent results and summaries of articles from the current literature. (AU)
1 unit, by arrangement (Mitchell)

579. Frontiers in Interdisciplinary Biosciences—(Cross-listed in multiple departments in the schools of Humanities and Sciences, Engineering, and Medicine; students should enroll directly through their affiliated department, if at all possible.) Introduction to cutting-edge research involving interdisciplinary approaches to bioscience and biotechnology; for specialists and non-specialists. Associated with Stanford's Clark Center for Interdisciplinary Bioscience, and held in conjunction with a seminar series meeting twice monthly during 2000-01. Leading investigators from Stanford and throughout the world speak on their research; students also meet separately to present and discuss the ever-changing subject matter, related literature, and future directions. Prerequisite: keen interest in all of science, with particular interest in life itself. Recommended: basic knowledge of biology, chemistry, and physics.
2 units, Aut, Win, Spr (S. Block)