SCHOOL OF ENGINEERING

COMPUTER SCIENCE

Emeriti: (Professors) Tom Binford, Edward Feigenbaum, Richard Fikes, Donald E. Knuth,* John McCarthy, Edward J. McCluskey, William F. Miller, Nils J. Nilsson, Vaughan Pratt,* Jeffrey D. Ullman, Gio Wiederhold*
Chair: Jennifer Widom
Associate Chair for Education: Mehran Sahami
Professors: Alex Aiken, Dan Boneh, David Cheriton, William J. Dally, David Dill, Hector Garcia-Molina, Leonidas J. Guibas, Patrick Hanrahan, John Hennessy, Mark A. Horowitz, Oussama Khatib, Daphne Koller, Monica Lam, Jean-Claude Latombe, Marc Levoy, Zohar Manna, Teresa Meng, John Mitchell, Kunle Olukotun, Yoav Shoham, Sebastian Thrun, Jennifer Widom, Terry Winograd
Associate Professors: Serafim Batzoglou, Dawson Engler, Ronald P. Fedkiw, Michael Genesereth, Christoforos Kozyrakis, Christopher Manning, David Mazieres, Nick McKeown, Andrew Ng, Serge A. Plotkin, Balaji Prabhakar, Mendel Rosenblum
Assistant Professors: Gill Bejerano, Jeffrey Heer, Sachin Katti, Scott Klemmer, Vladlen Koltun, Jure Leskovec, Philip Levis, Fei-Fei Li, Tim Roughgarden
Professors (Research): John Ousterhout, John K. Salisbury
Professor (Teaching): Eric S. Roberts
Associate Professor (Teaching): Mehran Sahami
Courtesy Professors: Russ Altman, Martin Fischer, Bernd Girod, Michael Levitt, Clifford J. Nass, Roy Pea, Fouad A. Tobagi
Courtesy Associate Professors: Ashish Goel, Dan Jurafsky, Vijay Pande, Benjamin Van Roy
Courtesy Assistant Professors: Paulo Blikstein, Atul Butte, Ramesh Johari, Ge Wang
Lecturers: Gerald Cain, Nicholas J. Parlante, Robert Plummer, Patrick Young, Julie Zelenski
Consulting Professors: Gary Bradski, Kathleen Fisher, Prabhakar Raghavan
Consulting Associate Professor: Federico Barbagli, Pei Cao
Consulting Assistant Professors: Kurt Akeley
Visiting Professor: Martin Abadi

* Recalled to active duty.

Mail Code: 94305-9025
Phone: (650) 723-2273
Web Site: http:// www.cs.stanford.edu

Courses offered by the Department of Computer Science are listed under the subject code CS on the *Stanford Bulletin's* ExploreCourses web site.

The Department of Computer Science (CS) operates and supports computing facilities for departmental education, research, and administration needs. All CS students have access to the departmental student machine for general use (mail, news, etc.), as well as computer labs with public workstations located in the Gates Building. In addition, most students have access to systems located in their research areas.

Each research group in Computer Science has systems specific to its research needs. These systems include workstations (PCs, Macs), multi-CPU computer clusters, and local mail and file servers. Servers and workstations running Linux or various versions of Windows are commonplace. Support for course work and instruction is provided on systems available through Information Technology Services (ITS) and the School of Engineering (SoE).

COMPUTER SCIENCES COURSE CATALOG NUMBERING SYSTEM

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

001-099	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	Computational Biology and Interdisciplinary Topics
90-99	Independent Study and Practicum

UNDERGRADUATE PROGRAMS IN COMPUTER SCIENCE

The mission of Stanford's undergraduate program in Computer Science is to provide a foundation of mathematics, science, and engineering knowledge. Building on Stanford's core ideals of liberal education, the program combines fundamentals with practical experience in problem solving, programming, communication, and collaboration, allowing each student to realize his or her individual potential.

Graduates of the program are prepared to pursue graduate study at the highest academic level, or advance into leadership positions in industry. The program creates an atmosphere that promotes innovative thinking, values mutual respect and diversity, supports scholarship and research, instills ethical behavior, and cultivates lifelong learning.

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

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 Computer Science major offers a number of tracks (programs of study) from which students can choose, allowing them to focus their program on the areas of most interest. These tracks also reflect the broad diversity of areas in computing disciplines. 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 areas requiring a knowledge 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 an opportunity 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 PROGRAM

The Department of Computer Science (CS) offers an honors program for 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 3.6 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 Computer Science 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 CS 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 selects 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. Attend a weekly honors seminar Winter and Spring quarters.
3. Complete an honors thesis deemed acceptable by 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 3.6 GPA required for admission to the honors program.

GUIDE TO CHOOSING 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 General Education Disciplinary Breadth Requirement in Engineering and Applied Sciences 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.

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 or may start directly in 106B. CS106A uses Java as its programming language; CS106B and X use C++. No prior knowledge of these languages is assumed, and the prior programming experience required for 103B or X 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 110. CS 103 offers an introduction to the mathematical and theoretical foundations of computer science. CS 107 exposes students to a variety of programming concepts that illustrate critical strategies used in systems development; 110 builds on this material, focusing on the development of larger-scale software making use of systems and networking abstractions.

In summary:
For exposure: 1C
For nontechnical use: 105
For scientific use: 106A
For a technical introduction: 106A
For significant use: 106A,B or 106X, plus 103, 107, and 110

GRADUATE PROGRAMS IN COMPUTER SCIENCE

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 COMPUTER 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 program leading to a master's degree with distinction in research. This program 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 by December 8, 2009. Exceptions are made for applicants who are already students at Stanford and are applying to the coterminal program. Information on these deadlines is available from the department.

For University coterminal degree program rules and University application forms, see http://registrar.stanford.edu/shared/ publications.htm#Coterm.

REQUIREMENTS

A candidate is required to complete a program of 45 units. At least 36 of these must be graded units, passed with a grade point average (GPA) of 3.0 (B) 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 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 103, or 103A, B, or X, 106A, B, or X, 107, 108, 110; MATH 51.

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 (CS 109 *or* STATS 116 *or* MS&E 220 *or* CME 106)
      2. Algorithms (CS 161)
      3. Automata (CS 154)
   b. Choose one of:
      1. Numerical Analysis (CME 108 *or* 302)
      2. Logic (CS 156, 157, 258, *or* PHIL 251)
      3. Mathematical Methods (CS 205A)
2. Area B: Computer Systems
   a. Required: Architecture (EE 108B *or* 282)
   b. Choose two of:
      1. Operating Systems (CS 140)
      2. Compilers (CS 143 or 243)
      3. Introduction to Computer Networks (CS 144 *or* EE 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 may narrow the set of choices in specific areas of the breadth requirement; see the individual specialization sheets at http://cs.stanford.edu/degrees/mscs /programsheets 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. Ten 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.

1. Artificial Intelligence
   a. at least four of: CS 223A, 223B, 224M, 224N, 224S, 224U, 226, 227, 228, 229
   b. a total of 21 units from category (a) and the following: CS 124, 205A, 222, 225A, 225B, 227B, 228T, 262, 270, 273A, 274, 275, 276, 277, 278, 279, 294A, 321, 322, 323, 327A, 328, 329, 374, 377,* 379*; ECON 286; EE 263, 363, 364A, 364B, 376A; ENGR 205, 209A; MS&E 251, 252, 339, 351, 352, 353; PSYCH 202, 205; STATS 202, 315A, 315B
2. Biocomputation
   a. at least four of: CS 262, 270, 272, 273A, 274, 278, 279
   b. a total of 21 units from category (a) and the following: CS 228, 229, 245, 261, 268, 275, 277, 345, 346, 365, 374; BIOC 218; BIOMEDIN 234; GENE 203, 211; SBIO 228
3. Computer and Network Security
   a. CS 155, 244, 255
   b. at least three of: CS 142, 240, 241, 244B, 244C, 259, 261, 340, 344, 365
   c. at least one additional course chosen from (b) and the following: CS 240E, 244E, 245, 295, 344B, 345, 347, 355, 361A; EE 384A, 384B, 384C, 384M, 384S
4. Database Systems
   a. CS 245
   b. at least two of: CS 345, 346, 347
   c. at least four additional courses from category (b) and the following: CS 240, 242, 243, 244, 244B, 244C, 249A, 249B, 255, 262, 270, 271, 272, 275, 276, 315A, 321, 344, 364B, 374
5. Human-Computer Interaction
   a. CS 147, 247
   b. at least one of: ME 313, 377; MS&E 27*, 28* (where 27* and 28* are any of the MS&E courses beginning with those digits that are offered in 2009/10): Psych 205, 252
   c. at least one of: CS124, 142, 148, 294H, 376, 377 (may be repeated for credit), 378, 448
   d. a total of 21 units from categories (a), (b), (c), and the following: CS 221, 223B, 229, 242, 248, 249A, 249B, 276, 379L: COMM 268, 269, 272; ENGR 231; ME 206A, 206B, 312, 314, 377; EDUC 124; MUSIC 205A: SYMBSYS 145
6. Numerical Analysis/Scientific Computation
   a. CME 302, 306; CS 205A
   b. at least two of: CME 326; CS 205B; MS&E 121; MATH 131, 132, 220; STATS 200
   c. at least two of: CS 223A, 327A, 339; AA 214A, 214B; CME 324, 342
7. Real-World Computing
   a. at least two of: CS 223A, 223B, 248
   b. at least three of: CS 205A, 205B, 226, 249A, 249B, 262, 268, 277, 348A, 348B, 374; CME 302, 306, 326
   c. a total of 21 units from the above and from the following: CS 225A, 225B, 228, 229, 247, 270, 271, 272, 273A, 274, 294A, 327A, 328, 448; CME 324
8. Software Theory
   a. CS 242, 243
   b. CS 241 or 258 or 259
   c. at least one of: CS 244, 245, 295, 343, 345
   d. at least one course from the following: CS 255, 261, 268, 355, 361A, 361B, 365
   e. at least two additional courses from (b), (c), and CS 346
9. Systems
   a. CS 240, 242
   b. at least three of: CS 243, 244, 245, 248, 348B; EE 271
   c. at least two additional courses chosen from (b) and the following: CS 194, 240C, 240D, 240E, 240X, 244B, 244C, 244E, 249A, 249B, 255, 259, 262, 270, 271, 272, 276, 294S, 295, 315A, 315B, 340, 343, 344, 344B, 344E, 345, 346, 347, 348A, 349, 374, 448; EE 384A, 384B, 384C, 384S, 384X, 384Y
10. Theoretical Computer Science
    a. CS 241 or 258 or 259, 261 (361A, 361B, or 365 may be substituted for 261)
    b. at least five additional courses chosen from CS 228, 241, 255, 258, 259, 262, 268, 345, 355, 357, 358, 359,* 361A, 361B, 364A, 364B, 365, 369,* 374; MS&E 310

* With consent of faculty adviser.

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 is 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 the writing and research units described in parts (1) and (2) must be taken in addition to the 21 units required for the specialization, although they do 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. A copy 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.

JOINT M.S. AND LAW DEGREE

Law students interested in pursuing an M.S. in Computer Science must apply for admission to the Computer Science Department either (i) concurrently with applying to the Law School; or (ii) after being admitted to the Law School, but no later than the earlier of: (a) the end of the second year of law school; or (b) the Computer Science Department's admission deadline for the year following that second year of law school. In addition to being admitted separately to the Law School and the Computer Science Department, students must secure permission from both academic units to pursue degrees in those units as part of a joint degree program. J.D./M.S. students may elect to begin their course of study in either the Law School or the Computer Science Department. Faculty advisors from each academic unit will participate in the planning and supervising of the student's joint program. Students must be enrolled full time in the Law School for the first year of law school. Otherwise, enrollment may be in the graduate school or the Law School and students may choose courses from either program regardless of where enrolled. Students must satisfy the requirements for both the J.D. and the M.S. degrees as specified in the Stanford Bulletin or elsewhere.

The Law School approves courses from the Computer Science Department that may count toward the J.D. degree, and the Computer Science Department shall approve courses from the Law School that may count toward the M.S. degree in Computer Science. In either case, approval may consist of a list applicable to all join degree students or may be tailored to each individual student program. No more than 45 units of approved courses may be counted toward both degrees. No more than 36 units of courses that originate outside the Law School may count toward the law degree. To the extent that courses under this joint degree program originate outside of the Law School but count toward the law degree, the Law School credits permitted under Section 17(1) of the Law School Regulations shall be reduced on a unit-per-unit basis, but not below zero. The maximum number of Law School credits that may be counted toward the M.S. in Computer Science is the greater of: (i) 12 units; or (ii) the maximum number of units from courses outside of the department that M.S. candidates in Computer Science are permitted to count toward the M.S. in the case of a particular student's individual program. Tuition and financial aid arrangements will normally be through the school in which the student is then enrolled.

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

Applications to the Ph.D. program and all supporting documents must be submitted and received online by December 8, 2009. See http://cs.stanford.edu/wiki/admissions for complete information. Changes or updates to the admission process are posted in September and October, 2009. The following are general department requirements; contact the Computer Science Ph.D. administrator for details.

1. A student should plan and 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. A student who fulfills six of thirteen exams in the breadth requirement may apply for candidacy prior to the second year in the program. The student must completely satisfy the breadth requirement by the end of nine quarters (excluding Summer Quarters), 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 course assistant or instructor for courses in Computer Science numbered 100 or above.
4. The most important requirement is the dissertation. After passing the required 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, typically held after all or a substantial portion of 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 IN COMPUTER SCIENCE

For a minor in Computer Science, a candidate must complete 20 unduplicated units of computer science course work numbered 200 or above. At least three of the courses must be master's core courses to provide breadth and one course numbered 300 or above to provide depth. One of the courses taken must include a significant programming project to demonstrate programming efficiency. Courses must be taken for a letter grade and passed with a grade of 'B' or better. Applications for a minor in Computer Science are submitted at the same time as admission to candidacy.

TEACHING AND RESEARCH ASSISTANTSHIPS IN COMPUTER SCIENCE

Graduate student assistantships are available. Half-time assistants receive a tuition scholarship for 8, 9, or 10 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. Course assistants (CAs) help an instructor teach a course by conducting discussion sections, consulting with students, and grading examinations. Research assistants (RAs) help faculty and senior staff members with research in computer science. Most course and research assistantships are held by Ph.D. students. If there is an insufficient number of Ph.D. students to staff teaching and research assistantships, then these positions are open to master's students. 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.

OVERSEAS STUDIES COURSES IN COMPUTER SCIENCE

For course descriptions and additional offerings, see the listings in the *Stanford Bulletin's* ExploreCourses web site (http://explorecourses.stanford.edu) or the Bing Overseas Studies web site (http://bosp.stanford.edu). Students should consult their department or program's student services office for applicability of Overseas Studies courses to a major or minor program.

SPRING QUARTER

BERLIN

OSPBER 45. Computers, Ethics, and Public Policy. 3-4 units, Eric Roberts, WIM, GER:EC:EthicReas

COMPUTER SCIENCE (CS)

UNDERGRADUATE COURSES IN COMPUTER SCIENCE

CS 1C. Introduction to Computing at Stanford
For those with limited experience with computers or who want to learn more about Stanford's computing environment. Topics include: computer maintenance and security, computing resources, Internet privacy, and copyright law. One-hour lecture/demonstration in dormitory clusters prepared and administered weekly by the Resident Computer Consultant (RCC). Final project. Not a programming course.
1 unit, Aut (Staff)

CS 2C. Multimedia Production
Sound, image and video editing techniques and applications, including understanding file formats and publishing multimedia online. Topics: GarageBand, Photoshop, iMovie, Final Cut Pro, and iDVD. Weekly lecture followed by lab section. Second unit for additional creative production assignments completed out of class time and extensive Final Project. Not a programming course, but will use computer multimedia applications heavily for editing.
1-2 units, Aut (Chan, K), Win (Chan, K)

CS 21N. Can Machines Know? Can Machines Feel?
(F,Sem) Stanford Introductory Seminar. Preference to freshmen. Can mental attitudes attributed to people and sometimes to animals, including knowledge, belief, desire, and intention, also be ascribed to machines? Can light sensors have a belief? Can a pool cleaning robot or tax-preparation software have an intention? If not, why not? If yes, what are the rules of such ascription, and do they vary between human beings and machines? Sources include philosophy, neuroscience, computer science, and artificial intelligence. Topics: logic, probability theory, and elements of computation. Students present a paper. GER:DB-EngrAppSci
3 units, Aut (Shoham, Y)

CS 26N. Motion Planning for Robots, Digital Actors, and Other Moving Objects
(F,Sem) Stanford Introductory Seminar. Preference to freshmen. Motion planning theory and computational approaches: how to represent, simulate, and plan motions in a computer. Intriguing algorithms, representations, and applications: terminology and concepts for reading motion planning research literature. Problems include: how a robot arm manipulates parts without colliding with its environment; how many maneuvers are required to park a car in a tight spot; how characters in computer games avoid running into obstacles; how molecules change shapes to perform biological functions; how to assemble a product from individual parts; how a multi-limbed robot can navigate on rough terrain; how robots can perform surgical procedures. Prerequisite: some computer programming experience in any language. GER:DB-EngrAppSci
3 units, Spr (Latombe, J)

CS 47N. Computers and the Open Society
(F,Sem) Stanford Introductory Seminar. How online technologies change our lives and the social structure that we live in. Course emphasizes critical analyses of current trends i.e. blogging, social networks, and instant mobile communication. Readings include case studies and analyses of basic principles i.e. privacy, equity and sustainability. Guest speakers who have participated in development of computers and the net will share their experiences and enter into debates on current issues. Students work individually and in small groups to research issues, develop the capacity for critical thinking about them, and use the results as the basis for writing and discussions both in class and on-line.
3 units, Aut (Winograd, T)

CS 73N. Business on the Information Highways
(F,Sem) Stanford Introductory Seminar. Preference to freshmen. The capabilities of the Internet and its services. Writing for the web. The effect on commerce, education, government, and health care. Technical and business alternatives. Who is hurt and who benefits from the changes? Participants develop web publications. GER:DB-EngrAppSci
3 units, Spr (Wiederhold, G; Barr, A; Tessler, S)

CS 74N. Digital Dilemmas
(F,Sem) Stanford Introductory Seminar. Preference to freshmen. Issues where policy decision making requires understanding computer and communications technology. Technology basics taught in non-technology terms. Topics include consumer privacy, government surveillance, file sharing and intellectual property, and electronic voting. GER:DB-EngrAppSci
3 units, Aut (Dill, D)

CS 103. Mathematical Foundations of Computing
Mathematical foundations required for computer science, including propositional predicate logic, induction, sets, functions, and relations. Formal language theory, including regular expressions, grammars, finite automata, Turing machines, and NP-completeness. Mathematical rigor, proof techniques, and applications. May not be taken by students who have completed 103A,B or 103X. Prerequisite: 106A or equivalent. GER:DB-Math
3-5 units, Aut (Plummer, R), Spr (Plummer, R)

CS 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. Students with prior computer science experience at the level of 106 or above require consent of instructor. Prerequisite: minimal math skills. GER:DB-EngrAppSci
3-5 units, Aut (Young, P), Spr (Young, P)

CS 106A. Programming Methodology
(Same as ENGR 70A) Introduction to the engineering of computer applications emphasizing modern software engineering principles: object-oriented design, decomposition, encapsulation, abstraction, and testing. Uses the Java programming language. Emphasis is on good programming style and the built-in facilities of the Java language. No prior programming experience required. GER:DB-EngrAppSci
3-5 units, Aut (Sahami, M), Win (Roberts, E), Spr (Cain, G), Sum (Staff)

CS 106B. Programming Abstractions
(Same as ENGR 70B) Abstraction and its relation to programming. Software engineering principles of data abstraction and modularity. Object-oriented programming, fundamental data structures (such as stacks, queues, sets) and data-directed design. Recursion and recursive data structures (linked lists, trees, graphs). Introduction to time and space complexity analysis. Uses the programming language C++ covering its basic facilities. Prerequisite: 106A or equivalent. GER:DB-EngrAppSci
3-5 units, Aut (Roberts, E), Win (Cain, G), Spr (Cain, G), Sum (Staff)

CS 106L. Standard C++ Programming Laboratory
Supplemental lab to 106B and 106X. Additional features of standard C++ programming practice. Possible topics include advanced C++ language features, standard libraries, STL containers and algorithms, object memory management, operator overloading, and inheritance. Prerequisite: consent of instructor. Corequisite: 106B or 106X.
1 unit, not given this year

CS 106X. Programming Abstractions (Accelerated)
(Same as ENGR 70X) Intensive version of 106B for students with a strong programming background interested in a rigorous treatment of the topics at an accelerated pace. Additional advanced material and more challenging projects. Prerequisite: excellence in 106A or equivalent, or consent of instructor. GER:DB-EngrAppSci
3-5 units, Aut (Cain, G)

CS 107. Computer Organization and Systems
Introduction to the fundamental concepts of computer systems. Explores how computer systems execute programs and manipulate data, working from the C programming language down to the microprocessor. Topics covered include: the C programming language, data representation, machine-level code, computer arithmetic, elements of code compilation, performance evaluation and optimization, memory organization and management, and concurrency and threading. Prerequisites: 106B or X, or consent of instructor. GER:DB-EngrAppSci
3-5 units, Aut (Zelenski, J), Spr (Zelenski, J)

CS 108. Object-Oriented Systems Design
Software design and construction in the context of large OOP libraries. Taught in Java. Topics: OOP design, design patterns, testing, graphical user interface (GUI) OOP libraries, software engineering strategies, approaches to programming in teams. Prerequisite: 107. GER:DB-EngrAppSci
3-4 units, Aut (Young, P), Win (Young, P)

CS 109. Introduction to Probability for Computer Scientists
Topics include: counting and combinatorics, random variables, conditional probability, independence, distributions, expectation, point estimation, and limit theorems. Applications of probability in computer science including machine learning and the use of probability in the analysis of algorithms. Prerequisites: 103, 106B or X, 109 and MATH 51 or equivalent. GER:DB-EngrAppSci
3-5 units, Win (Sahami, M), Spr (Sahami, M)

CS 110. Principles of Computer Systems
Principles and practice of engineering of computer software and hardware systems. Topics include: techniques for controlling complexity; strong modularity using client-server design, virtual memory, and threads; networks; atomicity and coordination of parallel activities; security, and encryption; and performance optimizations. Prerequisite: 107. GER:DB-EngrAppSci
3-5 units, Win (Rosenblum, M), Spr (Rosenblum, M)

CS 121. Introduction to Artificial Intelligence
(Only one of 121 or 221 counts towards any CS degree program.) Concepts, representations, and techniques used in building practical computational systems (agents) that appear to display artificial intelligence (AI), through the use of 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: 103 or 103B, and facility with differential calculus, vector algebra, and probability theory. GER:DB-EngrAppSci
3 units, Spr (Latombe, J), Sum (Staff)

CS 124. From Languages to Information
(Same as LINGUIST 180) Automated processing of less structured information: human language text and speech, web pages, social networks, genome sequences, with goal of automatically extracting meaning and structure. Methods include: string algorithms, automata and transducers, hidden Markov models, graph algorithms, XML processing. Applications such as information retrieval, text classification, social network models, machine translation, genomic sequence alignment, word meaning extraction, and speech recognition. Prerequisite: CS103, CS107, CS109.
3-4 units, Win (Jurafsky, D)

CS 140. Operating Systems and Systems Programming
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: CS 110. GER:DB-EngrAppSci
3-4 units, Win (Mazieres, D), Spr (Ousterhout, J)

CS 142. Web Applications
Concepts and techniques used in constructing interactive web applications. Browser-side web facilities such as HTML, cascading stylesheets, javascript, and the document object model. Server-side technologies such as sessions, templates, relational databases, and object-relational mapping. Issues in web security and application scalability. New models of web application deployment. Prerequisites: CS 107 and CS 108.
3 units, Aut (Ousterhout, J)

CS 143. Compilers
Principles and practices for design and implementation of compilers and interpreters. Topics: lexical analysis; parsing theory; symbol tables; type systems; scope; semantic analysis; intermediate representations; runtime environments; code generation; and basic program analysis and optimization. Students construct a compiler for a simple object-oriented language during course programming projects. Prerequisites: 103 or 103B, and 107. GER:DB-EngrAppSci
3-4 units, Aut (Aiken, A), Sum (Staff)

CS 144. Introduction to Computer Networking
Principles and practice. Structure and components of computer networks, packet switching, layered architectures. Applications: web/http, voice-over-IP, p2p file sharing and socket programming. Reliable transport: TCP/IP, reliable transfer, flow control, and congestion control. The network layer: names and addresses, routing. Local area networks: ethernet and switches. Wireless networks and network security. Prerequisite: CS 108 GER:DB-EngrAppSci
3-4 units, Aut (Levis, P; Mazieres, D)

CS 145. Introduction to Databases
Database design and use of database management systems for applications. The relational model, relational algebra, and SQL, the standard language for creating, querying, and modifying relational databases. XML data including DTDs and XML Schema for validation, and the query and transformation languages XPath, XQuery and XSLT. UML database design, and relational design principles based on functional dependencies and normal forms. Indexes, views, transactions, authorization, integrity contraints, and triggers. Advanced topics may include data warehousing, data mining, web data management, and data integration. Prerequisites: 103 or 103B, and 107. GER:DB-EngrAppSci
3-4 units, Aut (Widom, J)

CS 147. Introduction to Human-Computer Interaction Design
Usability and affordances, direct manipulation, systematic design methods, user conceptual models and interface metaphors, human cognitive and physical ergonomics, information and interactivity structures, and design tools and environments. Team project in interaction design. Prerequisite: 106B or X or equivalent programming experience.
3-4 units, Aut (Klemmer, S)

CS 147L. Human-Computer Interaction Technology Laboratory
Hands-on introduction to building mobile web applications with html, css, and php. Corequisite: 147. Concurrent enrollment in CS147 required.
1 unit, Aut (Krieger, M)

CS 148. Introduction Computer Graphics and Imaging
Topics: Image input and output devices such as cameras and displays, graphics hardware and software, input technologies and interactive techniques, typography and page layout, light and color representations, exposure and tone reproduction, image composition and imaging models, digital signal processing, sampling, aliasing and antialiasing, compression, two- and three-dimensional geometry and formations, modeling techniques including curves and surfaces, reflection models and illumination algorithms, and basic methods of animation. Progamming asssignments using C++ and OpenGL. Prerequisites: CS 107, MATH 51. GER:DB-EngrAppSci
3 units, Aut (Hanrahan, P), Sum (Staff)

CS 149. Parallel Computing
Course is an introduction to parallelism and parallel programming. Most new computer architectures are parallel; programming these machines requires knowlege of the basic issues of and techniques for writing parallel software. Topics: varieties of parallelism in current hardware (e.g., fast networks, multicore, accelerators such as GPUs, vector instruction sets), importance of locality, implicit vs. explicit parallelism, shared vs. non-shared memory, synchronization mechanisms (locking, atomicity, transactions, barriers), and parallel programming models (threads, data parallel/streaming, futures, SPMD, message passing, SIMT, transactions, and nested parallelism). Significant parallel programming assignments will be given as homework. Course is open to students who have completed the introductory CS course sequence through 110 and have taken at least one of CS 140, 143, 144, or 145. GER:DB-EngrAppSci
3-4 units, Win (Aiken, A; Olukotun, O)

CS 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. Prerequisites: 103 or 103B. GER:DB-EngrAppSci
3-4 units, Aut (Dill, D), Spr (Ullman, J)

CS 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: formal languages and automata as in first part of 154.
2 units, Aut (Dill, D), Spr (Ullman, J)

CS 155. Computer and Network Security
For seniors and first-year graduate students. Principles of computer systems security. Attack techniques and how to defend against them. Topics include: network attacks and defenses, operating system holes, application security (web, email, databases), viruses, social engineering attacks, privacy, and digital rights management. Course projects focus on building reliable code. Prerequisite: 140. Recommended: basic Unix. GER:DB-EngrAppSci
3 units, Spr (Boneh, D; Mitchell, J)

CS 156. Calculus of Computation
Decision procedures with applications to analyzing and developing robust software. Logic review. Propositional and first-order logic; induction. Verification: methods for proving correctness of sequential programs using first-order reasoning; need for decision procedures. Decision procedures: algorithms that decide the validity of logical formulas for common theories including SAT, equality, arithmetic, recursive data structures, and arrays. Combination theories and combination of decision procedures. Static analysis: algorithms for deducing program properties. Projects include writing verified programs. Prerequisites: 103, 106, or equivalents. GER:DB-EngrAppSci
3-4 units, Win (Manna, Z)

CS 157. Logic and Automated Reasoning
An elementary exposition from a computational point of view of propositional and predicate logic, axiomatic theories, and theories with equality and induction. Interpretations, models, validity, proof, strategies, and applications. Automated deduction: polarity, skolemization, unification, resolution, equality. Prerequisite: 103 or 103B. GER:DB-EngrAppSci
3 units, Aut (Genesereth, M)

CS 161. Design and Analysis of Algorithms
Worst and average case analysis. Recurrences and asymptotics. Efficient algorithms for sorting, searching, and selection. Data structures: binary search trees, heaps, hash tables. Algorithm design techniques: divide-and-conquer, dynamic programming, greedy algorithms, amortized analysis, randomization. Algorithms for fundamental graph problems: minimum-cost spanning tree, connected components, topological sort, and shortest paths. Possible additional topics: network flow, string searching. Prerequisite: 103 or 103B; 109 or STATS 116. GER:DB-EngrAppSci
3-5 units, Aut (Plotkin, S), Win (Roughgarden, T), Sum (Staff)

CS 164. Computing with Physical Objects: Algorithms for Shape and Motion
Algorithms and data structures deadling with the representation and manipulation of physical objects and entities in the computer. Computational structures for shape and motion, shape fitting and matching, triangulations and other spatial subdivisions, and low-dimensional search and optimization. Examples relevant to computer graphics, computer vision, robotics and geometric computation emphasizing algorithmic paradigms applicable to multidimensional data. Prerequisites: CS 103 or 103B, and CS 109 or STATS 116, and CS 106B/X or consent of instructor. GER:DB-EngrAppSci
3 units, Spr (Guibas, L)

CS 170. Composition, Coding, and Performance with SLOrc
(Same as MUSIC 128) Classroom instantiation of the Stanford Laptop Orchestra (SLOrk) which includes public performances. An ensemble of more than 20 humans, laptops, controllers, and special speaker arrays designed to provide each computer-mediated instrument with its sonic identity and presence. Topics and activities include issues of composing for laptop orchestras, instrument design, sound synthesis, programming, and live performance. May be repeated four times for credit.
1-5 units, Spr (Wang, G)

CS 178. Digital Photography
Scientific, artistic, and computing aspects of digital photography. Topics: lenses and optics, light and sensors, optical effects in nature, perspective and depth of field, sampling and noise, the camera as a computing platform, image processing and editing, history of photography, computational photography. Counts as a CS elective in the Graphics track. Prerequisites: introductory calculus; students must have a digital camera with manual control over shutter speed and aperture. Loaner cameras may be available. No programming experience required. Enrollment limited; see cs178.stanford.edu on March 1 for enrollment procedure. GER:DB-EngrAppSci
3-5 units, Spr (Levoy, M)

CS 181. Computers, Ethics, and Public Policy
(Formerly 201.) Primarily for majors entering computer-related fields. Ethical and social issues related to the development and use of computer technology. Ethical theory, and social, political, and legal considerations. Scenarios in problem areas: privacy, reliability and risks of complex systems, and responsibility of professionals for applications and consequences of their work. Prerequisite: 106B or X. GER:EC-EthicReas
3-4 units, Win (Johnson, M)

CS 191. Senior Project
Restricted to Computer Science and Computer Systems Engineering students. Group or individual projects under faculty direction. A project can be either a significant software application or publishable research. Software application projects include substantial programming and modern user-interface technologies and are comparable in scale to shareware programs or commercial applications. Research projects may result in a paper publishable in an academic journal or presentable at a conference. Required public presentation of final application or research results.
1-6 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 191W. Writing Intensive Senior Project
Restricted to Computer Science and Computer Systems Engineering students. Writing-intensive version of CS191. Register using the section number of an Academic Council member.
3-6 units, Aut (Staff), Win (Staff), Spr (Staff)

CS 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.
1-4 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 193C. Client-Side Internet Technologies
Client-side technologies used to create web sites such as sophisticated Web 2.0 interfaces similar to Google maps. XHTML, CSS, JavaScript, document object model (DOM), AJAX, and Flash. Prerequisite: programming experience at the level of 106A.
3 units, Sum (Staff)

CS 193D. Professional Software Development with C++
Programming techniques and methodologies. Language concepts including object-oriented design, memory management, and the standard library. Modern software development concepts such as design patterns, test-driven development, extreme programming, and XML. Prerequisites: basic C++ or significant experience in C or Java. GER:DB-EngrAppSci
3 units, not given this year

CS 193P. iPhone Application Programming
Tools and APIs required to build applications for the iPhone platform using the iPhone SDK. User interface designs for mobile devices and unique user interactions using multitouch technologies. Object-oriented design using model-view-controller pattern, memory management, Objective-C programming language. iPhone APIs and tools including Xcode, Interface Builder and Instruments on Mac OS X. Other topics include: core animation, bonjour networking, mobile device power management and performance considerations. Prerequisites: C language and programming experience at the level of 106B or X. Recommended: UNIX, object-oriented programming, graphical toolkits.
3 units, not given this year

CS 194. Software Project
Design, specification, coding, and testing of a significant team programming project under faculty supervision. Documentation includes a detailed proposal. Public demonstration of the project at the end of the quarter. Prerequisites: CS 110 and CS 161.
3 units, Spr (Plummer, R)

CS 196. Computer Consulting
Focus is on Macintosh and Windows operating system maintenance and troubleshooting through hardware and software foundation and concepts. Topics include operating systems, networking, security, troubleshooting methodology with emphasis on Stanford's computing environment. Not a programming course. Prerequisite: 1C or equivalent.
2 units, Win (Ly, J), Spr (Ly, J)

CS 198. Teaching Computer Science
Students lead a discussion section of 106A while learning how to teach a programming language at the introductory level. Focus is on teaching skills, techniques, and course specifics. Application and interview required; see http://cs198.stanford.edu.
3-4 units, Aut (Sahami, M; Wang, L; Ruth, E), Win (Sahami, M; Wang, L), Spr (Sahami, M; Wang, L; Ruth, E)

CS 199. Independent Work
Special study under faculty direction, usually leading to a written report. Letter grade; if not appropriate, enroll in 199P.
1-6 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 199P. Independent Work
1-6 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 202. Law for Computer Science Professionals
Intellectual property law as it relates to computer science including copyright registration, patents, and trade secrets; contract issues such as non-disclosure/non-compete agreements, license agreements, and works-made-for-hire; dispute resolution; and principles of business formation and ownership. Emphasis is on topics of current interest such as open source and the free software movement, peer-to-peer sharing, encryption, data mining, and spam.
1 unit, Aut (Hansen, D)

CS 204. Computational Law
Legal informatics based on representation of regulations in computable form. Encoding regulations facilitate creation of legal information systems with significant practical value. Convergence of technological trends, growth of the Internet, advent of semantic web technology, and progress in computational logic make computational law prospects better. Topics: current state of computational law, prospects and problems, philosophical and legal implications. Prerequisite: basic concepts of programming.
3 units, not given this year

CS 210A. Software Project Experience with Corporate Partners
Two quarter project course. Focus is on real world software development. Corporate partners provide loosely defined challenges from their R&D labs for which they are seeking innovative solutions and ideas. Student teams function as small startup companies with a technical advisory board comprised of the instructional staff. Exposure to: current practices in software engineering; exploration of the design space; significant development experience with creative freedoms; working in groups; real world software engineering challenges; public presentation of technical work; creating written descriptions of technical work. Prerequisite: CS 108 or CS110.
3-4 units, Win (Borenstein, J)

CS 210B. Software Project Experience with Corporate Partners
Continuation of CS210A. Focus is on real world software development. Corporate partners provide loosely defined challenges from their R&D labs for which they are seeking innovative solutions and ideas. Student teams function as a small startup companies with a technical advisory board comprised of the instructional staff. Exposure to: current practices in software engineering; exploration of the design space; significant development experience with creative freedoms; working in groups; real world software engineering challenges; public presentation of technical work; creating written descriptions of technical work. Prerequisite: CS 210A.
1-4 units, Spr (Borenstein, J)

CS 226. Statistical Techniques in Robotics
Theory and practice of statistical techniques used in robotics and large-scale sensor-based systems. Probabilistic state estimation, Bayes, Kalman, information and particle filters. Simultaneous localization and mapping techniques, and multi-robot sensor fusion. Markov techniques for making decisions under uncertainty, and probabilistic control algorithms and exploration.
3 units, Win (Thrun, S)

GRADUATE COURSES IN COMPUTER SCIENCE

CS 205A. Mathematical Methods for Robotics, Vision, and Graphics
Continuous mathematics background necessary for research in robotics, vision, and graphics. Possible topics: linear algebra; the conjugate gradient method; ordinary and partial differential equations; vector and tensor calculus. Prerequisites: 106B or X; MATH 51 and 113; or equivalents.
3 units, Aut (Fedkiw, R)

CS 205B. Mathematical Methods for Fluids, Solids, and Interfaces
Numerical methods for simulation of problems involving solid mechanics and fluid dynamics. Focus is on practical tools needed for simulation, and continuous mathematics involving nonlinear hyperbolic partial differential equations. Possible topics: finite element method, highly deformable elastic bodies, plasticity, fracture, level set method, Burgers' equation, compressible and incompressible Navier-Stokes equations, smoke, water, fire, and solid-fluid coupling. Prerequisite: 205A or equivalent.
3 units, Spr (Fedkiw, R)

CS 207. The Economics of Software
How software products are moved into the marketplace and how the resulting intellectual property is exploited. Concepts that are outside of the common knowledge of computer scientists such as business terms and spreadsheet computations to quantitatively compare alternatives. Goal is to contribute to informed decision making in high-tech product design, acquisition, production, marketing, selection of business structures, outsourcing, and impact of taxation policies.No specific background required.
1 unit, Aut (Wiederhold, G)

CS 208. The Canon of Computer Science
Analysis and discussion of influential and inspiring writings in computer science. Original works by Turing, von Neumann, Shannon, Bush, Engelbart, Licklider, Kay, Feynman, and others. Emphasis on writings that shaped the field and continue to provoke and stimulate. The visions that inspired personal computing, computer networks, the GUI, the Web, and other key developments.
3-4 units, Spr (Koltun, V)

CS 209. Introduction to Functional Programming
Functional programming offers insights and advanced programming techniques not found in other programming languages. Topics: lambda calculus (an alternative to Turing machines), higher-order functions, lazy evaluation, type-oriented programming, syntactic extension, and advanced control abstractions known as monads and continuations. Functional programming languages to be studied include Scheme, an eagerly evaluated, dynamically typed language, and Haskell, a lazily evaluated, statically typed language with type inferencing. Prerequisites: CS 107 and CS 161.
3 units, not given this year

CS 221. Artificial Intelligence: Principles and Techniques
(Only one of 121 or 221 counts towards any CS degree program.) Topics: search, constraint satisfaction, knowledge representation, probabilistic models, Bayesian networks, machine learning, neural networks, vision, robotics, and natural language processing. Prerequisites: 103 or 103B/X; 106B or 106X; and exposure to probability. Recommended: 107 and facility with basic differential calculus.
3-4 units, Aut (Ng, A)

CS 222. Rational Agency and Intelligent Interaction
(Same as PHIL 358) For advanced undergraduates, and M.S. and beginning Ph.D. students. Logic-based methods for knowledge representation, information change, and games in artificial intelligence and philosophy. Topics: knowledge, certainty, and belief; time and action; belief dynamics; preference and social choice; games; and desire and intention. Prerequisite: propositional and first-order logic. Recommended: modal logic; game theory.
3 units, Spr (Shoham, Y)

CS 223A. Introduction to Robotics
Topics: robotics foundations in kinematics, dynamics, control, motion planning, trajectory generation, programming and design. Recommended: matrix algebra.
3 units, Win (Khatib, O)

CS 223B. Introduction to Computer Vision
Fundamental issues and techniques of computer vision. Image formation, edge detection and image segmentation, stereo, motion, shape representation, recognition.
3 units, Win (Li, F)

CS 224M. Multi-Agent Systems
For advanced undergraduates, and M.S. and beginning Ph.D. students. Topics: logics of knowledge and belief, other logics of mental state, theories of belief change, multi-agent probabilities, essentials of game theory, social choice and mechanism design, multi-agent learning, communication. Applications discussed as appropriate; emphasis is on conceptual matters and theoretical foundations. Prerequisites: basic probability theory and first-order logic.
3 units, Aut (Shoham, Y)

CS 224N. Natural Language Processing
(Same as LINGUIST 280) Methods for processing human language information and the underlying computational properties of natural languages. Syntactic and semantic processing from linguistic and algorithmic perspectives. Focus is on modern quantitative techniques in NLP: using large corpora, statistical models for acquisition, translation, and interpretation; and representative systems. Prerequisites: CS124 or CS121/221.
3-4 units, Spr (Manning, C)

CS 224S. Speech Recognition and Synthesis
(Same as LINGUIST 285) Automatic speech recognition, speech synthesis, and dialogue systems. Focus is on key algorithms including noisy channel model, hidden Markov models (HMMs), Viterbi decoding, N-gram language modeling, unit selection synthesis, and roles of linguistic knowledge. Prerequisite: programming experience. Recommended: CS 221 or 229.
2-4 units, not given this year

CS 224U. Natural Language Understanding
(Same as LINGUIST 188, LINGUIST 288) Machine understanding of human language. Computational semantics (determination of word sense and synonymy, event structure and thematic roles, time, aspect, causation, compositional semantics, scopal operators), and computational pragmatics and discourse (coherence, coreference resolution, information packaging, dialogue structure). Theoretical issues, online resources, and relevance to applications including question answering and summarization. Prerequisites: one of LINGUIST 180 / CS 124 / CS 224N,S: and logic such as LINGUIST 130A or B, CS 157, or PHIL150).
3-4 units, Win (Jurafsky, D; MacCartney, W)

CS 225A. Experimental Robotics
Hands-on. Topics: kinematic and dynamic control of motion, compliant motion and force control, sensor-based collision avoidance, motion planning, dynamic skills, and robot-human interfaces. Limited enrollment. Prerequisite: 223A.
3 units, Spr (Khatib, O)

CS 225B. Robot Programming Laboratory
For robotics and non-robotics students. Students program mobile robots to exhibit increasingly complex behavior (simple dead reckoning and reactivity, goal-directed motion, localization, complex tasks). Topics: motor control and sensor characteristics; sensor fusion, model construction, and robust estimation; control regimes (subsumption, potential fields); probabalistic methods, including Markov localization and particle filters. Student programmed robot contest. Programming is in C++ on Unix machines, done in teams. Prerequisite: programming at the level of 106B, 106X, 205, or equivalent.
3-4 units, Aut (Konolige, K)

CS 227. Reasoning Methods in Artificial Intelligence
Technical presentation of logical algorithmic techniques for problem solving in AI. Combines formal algorithmic analysis with a description of recent applications. Topics: representation and modelling, propositional satisfiability, constraint satisfaction, planning and scheduling, advanced topics. Focus is on recent results. Prerequisites: familiarity with basic notions in data structures and with techniques in algorithm design and analysis. Recommended: previous or concurrent course in AI.
3 units, Spr (Staff)

CS 227B. General Game Playing
A general game playing system accepts a formal description of a game to play it without human intervention or algorithms designed for specific games. Hands-on introduction to these systems and artificial intelligence techniques such as knowledge representation, reasoning, learning, and rational behavior. Students create GGP systems to compete with each other and in external competitions. Prerequisite: programming experience. Recommended: 103 or equivalent.
3 units, Spr (Genesereth, M)

CS 228. Structured Probabilistic Models: Principles and Techniques
Probabilistic modeling languages for representing complex domains, algorithms for reasoning and decision making using these representations, and learning these representations from data. Focus is on probabilistic graphic models, including Bayesian and Markov networks, extensions to temporal modeling such as hidden Markov models and dynamic Bayesian networks, and extensions to decision making such as influence diagrams. Basic techniques and their applications to domains including speech recognition, biological modeling and discovery, medical diagnosis, message encoding, vision, and robot motion planning. Prerequisites: basic probability theory and algorithm design and analysis.
3 units, Win (Koller, D)

CS 228T. Structured Probabalistic Models: Theoretical Foundations
For students interested in advanced methods in machine learning and probabilistic AI. Theoretical foundations and extension for the ideas and algorithms covered in CS 228. Topics include theory and advanced algorithms for approximate inference in graphical models, representation and inference in continuous processes, and theory and algorithms for learning with missing data and hidden variables. Pre- or corequisites: CS 228; strong mathematical foundation.
3 units, Win (Koller, D)

CS 229. Machine Learning
Topics: statistical pattern recognition, linear and non-linear regression, non-parametric methods, exponential family, GLMs, support vector machines, kernel methods, model/feature selection, learning theory, VC dimension, clustering, density estimation, EM, dimensionality reduction, ICA, PCA, reinforcement learning and adaptive control, Markov decision processes, approximate dynamic programming, and policy search. Prerequisites: linear algebra, and basic probability and statistics.
3 units, Aut (Ng, A)

CS 240. Advanced Topics in Operating Systems
Recent research. Classic and new papers. Topics: virtual memory management, synchronization and communication, file systems, protection and security, operating system extension techniques, fault tolerance, and the history and experience of systems programming. Prerequisite: 140 or equivalent.
3 units, Spr (Engler, D)

CS 240C. Advanced Operating Systems Implementation
Operating system techniques for meeting the performance, security, flexibility, and robustness needs of demanding applications. Review of hardware/software interface and traditional operating system concepts. Recent operating systems research. Lab to apply concepts. Students work with a minimal operating system capable of running on standard PC hardware. Operating system written in C with some assembly. Prerequisite: 140 or consent of instructor.
3 units, not given this year

CS 240D. Distributed Storage Systems
File system implementation, low-level database storage techniques, and distributed programming. File system structures, journaling and logging, I/O system performance, RAID (redundant arrays of inexpensive disks), remote procedure call abstraction, and systems illustrating these concepts. File systems, distributed computing, replication and consistency, fault tolerance, and crash recovery. Programming assignments. Final project to build a functioning Unix file system. Prerequisites: C++ and familiarity with Unix; 140 or consent of instructor.
3 units, not given this year

CS 240E. Low Power Wireless System Software
The structure and implementation of software systems for low power embedded sensors; how to build software that can run unattended for years on small batteries. Topics: hardware trends, energy profiles, execution models, aggregation, storage, application requirements, allocation, power management, resource management, scheduling, time synchronization, programming models, software design, and fault tolerance. Students build working systems on TinyOS, a low-power embedded operation system.
3 units, not given this year

CS 240X. Advanced Operating Systems II
Same content as 240, with expanded topics focusing on more difficult and specialized papers. Recent topics in systems research.
3 units, not given this year

CS 242. Programming Languages
Central concepts in modern programming languages, impact on software development, language design trade-offs, and implementation considerations. Functional, imperative, and object-oriented paradigms. Formal semantic methods and program analysis. Modern type systems, higher order functions and closures, exceptions and continuations. Modularity, object-oriented languages, and concurrency. Runtime support for language features, interoperability, and security issues. Prerequisite: 107, or experience with Lisp, C, and an object-oriented language.
3 units, Aut (Mitchell, J)

CS 243. Program Analysis and Optimizations
Program analysis techniques used in compilers and software development tools to improve productivity, reliability, and security. The methodology of applying mathematical abstractions such as graphs, fixpoint computations, binary decision diagrams in writing complex software, using compilers as an example. Topics include data flow analysis, instruction scheduling, register allocation, parallelism, data locality, interprocedural analysis, and garbage collection. Prerequisites: 103 or 103B, and 107.
3-4 units, Win (Lam, M)

CS 244. Advanced Topics in Networking
Classic papers, new ideas, and research papers in networking. Architectural principles: naming, addressing, routing; congestion control, traffic management, QoS; wireless and mobility; overlay networks and virtualization; network security; switching and routing; content distribution; and proposals for future Internet structures. Prerequisite: 144 or equivalent.
3-4 units, Win (McKeown, N)

CS 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, application-sufficient consistency, file access, process scheduling, process migration, and storage/communication abstractions on distribution, scale, robustness in the face of failure, and security. Prerequisites: CS 144 and CS 249A.
3 units, Spr (Cheriton, D)

CS 244C. Readings and Projects in Distributed Systems
Companion project option for 244B. Corequisite: 244B.
3-6 units, Spr (Cheriton, D)

CS 244E. Wireless Networking
Challenges of low power wireless networking protocols and applications. Topics: the OSI model, 802.11, Bluetooth, 802.15.4, WiMAX, hardware considerations, media access, radio propagation models, flooding, dissemination, gossip, link behavior, opportunistic reception, network coding, modulation, TCP. Students read papers and build working protocols on the 100-mode Stanford wireless testbed.
3 units, Spr (Levis, P)

CS 245. Database Systems 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, H), Sum (Staff)

CS 247. Human-Computer Interaction Design Studio
Project-based. 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. Prerequisites: 147 and 106A or equivalent background in programming.
3-4 units, Win (Winograd, T)

CS 247L. Human Computer Interaction Technology Lab
Hands-on introduction to contemporary HCI technologies. Interaction design with Adobe Flash, mobile development, physical computing, and web applications. Corequisite: 247.
1 unit, Win (Winograd, T)

CS 248. Three-Dimensional Computer Graphics
Rendering, animation and modeling for interactive computer graphics. Rasterization, graphics pipeline, graphics hardware; texture mapping and its applications; lighting and surface shading; rendering optimization; keyframing; physics simulation. Programming projects and final project. Prerequisite: CS148.
3-5 units, Win (Koltun, V)

CS 249A. Object-Oriented Programming from a Modeling and Simulation Perspective
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: C, C++, and programming methodology as developed in 106B or X, and 107 (107 may be taken concurrently). Recommended: 193D.
3 units, Aut (Cheriton, D)

CS 249B. Advanced Object-Oriented Programming
How to produce reasonable-cost, high quality software such as next-stage, large-scale systems that handle life-critical systems. Software process, people, practice, and audit: integrating invariant checks with production software; collection implementation; generic programming and templates; design of value types; named descriptions for large value types; memory management; controlling placement; locality and consumption; concurrency with modular object-oriented programming. Inheritance: when and why multiple inheritance naming, directories, manager, and other design patterns.
3 units, Win (Cheriton, D)

CS 255. Introduction to Cryptography
For advanced undergraduates and graduate students. 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, PKI, zero-knowledge protocols. Prerequisite: basic probability theory.
3 units, Win (Boneh, D)

CS 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 past) and omega-automata. Hierarchy of program properties: safety, guarantee, obligation, response, persistence, and reactivity. Invariant generation. Deductive verification of programs: verification diagrams and rules, completeness. Modularity. Parameterized programs. Algorithmic verification of finite-state programs (model checking). Prerequisite: 154, 156, 157, or equivalent.
3 units, not given this year

CS 256L. Formal Methods for Reactive Systems Laboratory
Practical application of CS 256. Individual projects include implementation of verification methods, verification case studies, or tool evaluation, depending on student preference.
2 units, not given this year

CS 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. Denotational semantics and elementary domain theory for languages with higher-type functions and recursion. Treatment of side effects. Prerequisites: 154, 157 or PHIL 160A.
3 units, not given this year

CS 259. Security Analysis of Network Protocols
Hands-on experience in formal methods to verify and evaluate the security of network protocols and other systems. Common security protocols and their properties including secrecy, authentication, key establishment, and fairness. Topics: standard formal models and tools used in security protocol analysis; their advantages and limitations. Fully automated, finite-state, model-checking techniques. Constraint solving, process algebras, protocol logics, probabilistic model checking, and game theory. Students select a protocol or secure system to analyze, specify it in the chosen model, use a formal analysis tool to verify its properties, and present findings.
3 units, not given this year

CS 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 online algorithms. Prerequisite: 161 or equivalent.
3 units, Win (Plotkin, S)

CS 262. Computational Genomics
(Same as BIOMEDIN 262) Applications of computer science to genomics, and concepts in genomics from a computer science point of view. Topics: dynamic programming, sequence alignments, hidden Markov models, Gibbs sampling, and probabilistic context-free grammars. Applications of these tools to sequence analysis: comparative genomics, DNA sequencing and assembly, genomic annotation of repeats, genes, and regulatory sequences, microarrays and gene expression, phylogeny and molecular evolution, and RNA structure. Prerequisites: 161 or familiarity with basic algorithmic concepts. Recommended: basic knowledge of genetics.
3 units, Win (Batzoglou, S)

CS 268. Geometric Algorithms
Techniques for design and analysis of efficient geometric algorithms for objects in 2-, 3-, and higher dimensions. Topics: convexity, triangulations and simplicial complexes, sweeping, partitioning, and point location. Voronoi/Delaunay diagrams and their properties. Arrangements of curves and surfaces. Intersection and visibility problems. Geometric searching and optimization. Random sampling methods. Impact of numerical issues in geometric computation. Example applications to robotic motion planning, visibility preprocessing and rendering in graphics, model-based recognition in computer vision, and structural molecular biology. Prerequisite: discrete algorithms at the level of 161. Recommended: 164.
3 units, not given this year

CS 270. Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
(Same as BIOMEDIN 210) Methods for modeling biomedical systems and for making those models explicit in the context of building software systems. Emphasis is on intelligent systems for decision support and Semantic Web applications. Topics: knowledge representation, controlled terminologies, ontologies, reusable problem solvers, and knowledge acquisition. Recommended: exposure to object-oriented systems, basic biology.
3 units, Aut (Musen, M)

CS 271. Effective Design in Clinical Informatics Systems
(Same as BIOMEDIN 211) Methods of designing and engineering software systems in complex clinical environments. Case studies illustrate factors leading to success or failure of systems. Project assignments involve focused team-based design work. Topics: user and organizational requirements, data and knowledge modeling, component-based system design, system prototyping, and human-systems interaction. Prerequisite: BIOMEDIN 210 recommended, or database or object-oriented programming course.
3 units, Win (Das, A)

CS 272. Introduction to Biomedical Informatics Research Methodology
(Same as BIOE 212, BIOMEDIN 212, GENE 212) Hands-on software building. Student teams conceive, design, specify, implement, evaluate, and report on a software project in the domain of biomedicine. Creating written proposals, peer review, providing status reports, and preparing final reports. Guest lectures from professional biomedical informatics systems builders on issues related to the process of project management. Software engineering basics. Prerequisites: BIOMEDIN 210, 211, 214, 217 or consent of instructor.
3 units, Aut (Altman, R; Cheng, B; Klein, T)

CS 273A. A Computational Tour of the Human Genome
(Same as BIOMEDIN 273A, DBIO 273A) Biology through an exploration of Human Genome. Key genomic and genetic concepts from an informatics perspective. Biomedical advances resulting from the Genomics revolution. Topics: genome sequencing: technologies, assembly, personalized sequencing. Functional landscape: genes, gene regulation, repeats, RNA genes. Genome evolution: comparative genomics, ultraconservation, co-option. Additional topics: population genetics, personalized genomics, and ancient DNA. Course starts with primer in Biology and text processing languages. Ends with guest lectures from forefront of genomic research.
3 units, Aut (Batzoglou, S; Bejerano, G)

CS 274. Representations and Algorithms for Computational Molecular Biology
(Same as BIOE 214, BIOMEDIN 214, GENE 214) Topics: introduction to bioinformatics and computational biology, algorithms for alignment of biological sequences and structures, computing with strings, phylogenetic tree construction, hidden Markov models, Gibbs Sampling, 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 data sources, knowledge representation and controlled terminologies for molecular biology, microarray analysis, machine learning (clustering and classification), and natural language text processing. Prerequisites: programming skills; consent of instructor for 3 units.
3-4 units, Spr (Staff)

CS 275. Translational Bioinformatics
(Same as BIOMEDIN 217) Analytic, storage, and interpretive methods to optimize the transformation of genetic, genomic, and biological data into diagnostics and therapeutics for medicine. Topics: access and utility of publicly available data sources; types of genome-scale measurements in molecular biology and genomic medicine; analysis of microarray data; analysis of polymorphisms, proteomics, and protein interactions; linking genome-scale data to clinical data and phenotypes; and new questions in biomedicine using bioinformatics. Case studies. Prerequisites: programming ability at the level of CS 106A and familiarity with statistics and biology.
4 units, Win (Butte, A)

CS 276. Information Retrieval and Web Search
(Same as LINGUIST 286) Text information retrieval systems; efficient text indexing; Boolean, vector space, and probabilistic retrieval models; ranking and rank aggregation; evaluating IR systems. Text clustering and classification: classification algorithms, latent semantic indexing, taxonomy induction; Web search engines including crawling and indexing, link-based algorithms, and web metadata. Prerequisites: CS 107, CS 109, CS 161.
3 units, Aut (Manning, C; Raghavan, P)

CS 277. Experimental Haptics
Haptics as it relates to creating touch feedback in simulated or virtualized environments. Goal is to develop virtual reality haptic simulators and applications. Theoretical topics: psychophysical issues, performance and design of haptic interfaces, haptic rendering methods for 3-D virtual environments, and haptic simulation and rendering of rigid and deformable solids. Applied topics: the CHAI haptic library; implementation of haptic rendering algorithms; collision detection in 3-D environments; design of real-time models for deformable objects. Guest speakers. Lab/programming exercises; a more open-ended final project. Enrollment limited to 20. Prerequisite: experience with C++. Recommended: 148 or 248, 223A.
3 units, Win (Barbagli, F; Salisbury, K)

CS 278. Systems Biology
(Same as BIOC 278, BIOE 310, CSB 278) Complex biological behaviors through the integration of computational modeling and molecular biology. Topics: reconstructing biological networks from high-throughput data and knowledge bases. Network properties. Computational modeling of network behaviors at the small and large scale. Using model predictions to guide an experimental program. Robustness, noise, and cellular variation. Prerequisites: background in biology and mathematical analysis.
3 units, not given this year

CS 279. Computational Methods for Analysis and Reconstruction of Biological Networks
Types of interactions, including: regulatory such as transcriptional, signaling, and chromatin modification; protein-protein interactions; and genetic. Biological network structure at scales such as single interaction, small subgraphs, and global organization. Methods for analyzing properties of biological networks. Techniques for reconstructing networks from biological data, including: DNA/protein sequence motifs and sequence conservation; gene expression data; and physical binding data such as protein-DNA, protein-RNA, and protein-protein. Network dynamics and evolution. Prerequisites: biology at the level of BIOSCI 41; computer science and data structures at the level of CS 103 and 106; and probability and statistics at the level of STATS 116 or CS 109.
3 units, not given this year

CS 294. Research Project in Computer Science
Student teams work under faculty supervision on research and implementation of a large project in some major sub-discipline in computer science. Lectures on state-of-the-art methods related to the particular problem domain. Prerequisites: consent of instructor.
3 units, not given this year

CS 294A. Research Project in Artificial Intelligence
Student teams under faculty supervision work on research and implementation of a large project in AI. State-of-the-art methods related to the problem domain. Prerequisites: AI course from 220 series, and consent of instructor.
3 units, Aut (Koller, D), Win (Ng, A)

CS 294H. Research Project in Human-Computer Interaction
Many of the most successful web applications are social, from personalized homepages to social networks. Students will learn the fundamental interface design, systems, and algorithms concepts in designing social software. The case-based syllabus will cover insights from both research and industry. As a student, you will contribute to this burgeoning field through a quarter-long, team-based project. Students are required to enter the class with an initial project idea.
3 units, Win (Heer, J)

CS 294S. Research Project in Software Systems and Security
Topics vary. Focus is on emerging research themes such as programmable open mobile Internet that spans multiple system topics such as human-computer interaction, programming systems, operating systems, networking, and security. May be repeated for credit. Prerequisites: CS 103 and 107.
3 units, Aut (Staff)

CS 294W. Writing Intensive Research Project in Computer Science
Restricted to Computer Science and Computer Systems Engineering undergraduates. Students enroll in the CS 294W section attached to the CS 294 project they have chosen.
3 units, Aut (Koller, D), Win (Ng, A)

CS 295. Software Engineering
Software specification, testing, and verification. Emphasis is on current best practices and technology for developing reliable software at reasonable cost. Assignments focus on applying these techniques to realistic software systems. Prerequisites: 108. Recommended a project course such as 140, 143, or 145.
2-3 units, Spr (Aiken, A)

CS 298. Seminar on Teaching Introductory Computer Science
Faculty, undergraduates, and graduate students interested in teaching discuss topics raised by teaching computer science at the introductory level. Prerequisite: consent of instructor.
1-3 units, not given this year

CS 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.
1 unit, Aut (Staff)

CS 302. Tech Law with Progressive Minds
How the advent of computing technologies is reflected in the confluence of law, public policy, and technology. Issues relating to civil liberties, consumer protection, e-voting, copyright law, patent law, international patent law, trade secrets, political processes, and litigation.
1 unit, not given this year

CS 303. Designing Computer Science Experiments
Introduction to empirical research in computer science. Learn how to design, execute, interpret, and report on computer science experiments. Conducting empirical work and using experiments to build theory is one of the major ways to move computer science forward, but these issues are often omitted from computer science curricula. Course features case studies drawn from artificial intelligence, systems, and human-computer interaction. Emphasizes the decision-making aspects of research and the logic behind research procedures.
3 units, Spr (Klemmer, S; Levis, P; Manning, C)

CS 309. Industrial Lectureships in Computer Science
Guest computer scientist. By arrangement. May be repeated for credit. (Staff)
1 unit, not given this year

CS 309A. Software as a Service
For technology and business students. The shift from traditional software model of disconnected development and CD-ROM deployment to engineering and delivery on the Internet as a service. Guest industry experts give first-hand view of changes in the software industry.
1 unit, Aut (Chou, T)

CS 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, data flow, and data parallel machines illustrate techniques. Architectural studies and lectures on techniques for programming parallel computers. Programming assignments on one or more commercial multiprocessors. Prerequisites: EE 282, and reasonable programming experience.
3 units, Spr (Olukotun, O)

CS 315B. Parallel Computing Research Project
Advanced topics and new paradigms in parallel computing including parallel algorithms, programming languages, runtime environments, library debugging/tuning tools, and scalable architectures. Research project. Prerequisite: consent of instructor.
3 units, not given this year

CS 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. May be repeated for credit.
3 units, offered occasionally

CS 322. Network Analylsis
The emergence of the web and large online computing applications can be seen as a convergence of social and technological networks, with systems such as the World Wide Web, blogging platforms and Facebook that can be characterized by the interplay between rich information content, the millions of individuals and organizations who create it, and the technology that supports it. Course will cover recent research on the structure and analysis of such large social and information networks and on models and algorithms that abstract their basic properties. Topics: probabilistic models for network structure and evolution, methods for link analysis and network community detection, search algorithms, diffusion and information propagation on the web, virus outbreak detection in networks, and connections with work in the social sciences and economics.
3 units, Aut (Leskovec, J)

CS 323. Understanding Images and Videos: Recognizing and Learning High-Level Visual Concepts
Field of computer vision has seen an explosive growth in past decade. Much of recent effort in vision research is towards developing algorithms that can perform high-level visual recogniztion tasks on real-world images and videos. With development of Internet, this task becomes particularly challenging and interesting given the heterogeneous data on the web. Course will focus on reading recent research papers that are focused on solving high-level visual recognition problems, such as object recognition and categorization, scene understanding, human motion understanding, etc. Project required. Prerequisite: some experience in research with one of the following fields: computer vision, image processing, computer graphics, machine learning.
3 units, Aut (Staff)

CS 326A. Motion Planning
Computing object motions in computer graphics, geometrical computing, robotics, or artificial intelligence for applications such as design, manufacturing, robotics, animated graphics, surgical planning, drug design, assembly planning, graphic animation of human figures, humanoid robots, inspection and surveillance, simulation of crowds, and biology. Path planning methods to generate collision-free paths among static obstacles. Extensions include uncertainty, mobile obstacles, manipulating moveable objects, maneuvering with kinematic constraints, and making and breaking contacts. Configuration space, geometric arrangements, and random sampling. Theoretical methods.
3 units, not given this year

CS 327A. Advanced Robotics
Emerging areas of human-centered robotics and interactive haptic simulation of virtual environments. Topics: redundancy; task-oriented dynamics and control, whole-body control-task and posture decomposition, cooperative robots, haptics and simulation, haptically augmented teleoperation, human-friendly robot design. Prerequisites: 223A or equivalent.
3 units, Spr (Khatib, O)

CS 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. May be repeated for credit.
3 units, offered occasionally

CS 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. May be repeated for credit.
3 units, offered occasionally

CS 340. Topics in Computer Systems
Topics vary every quarter, and may include advanced material being taught for the first time. May be repeated for credit.
3-4 units, not given this year

CS 340V. Networked Systems for Virtual Worlds
Open to graduate students and advanced undergraduates. Systems and networking aspects of building large, distributed virtual 3D environments, with a focus on scalability, consistency, security, fairness, and federation. Topics include existing architectures, naming, routing, caching, migration, interoperability, and attribution. Open-ended research project. Prerequisite: some systems and networking background. May be repeated for credit.
3-4 units, not given this year

CS 342. Programming Language Design
Tools for analysis and optimization of iterative coding systems. LDPC codes, Turbo codes, RA codes, optimized ensembles, message passing algorithms, density evolution, analytic techniques. Prerequisite: 376A.
3 units, not given this year

CS 343. Advanced Topics in Compilers
Topics change annually. May be repeated for credit. Prerequisite: 243.
3 units, Spr (Engler, D)

CS 344. Build an Internet Router
High-performance embedded system design. Student teams of two software engineers (C experience required) and one hardward engineer (Verilog experience required) build a fully functioning Internet router Work in team of three. How router interoperates with others in class. Open-ended design challenge judged by panel of industry experts. Prerequisites: CS 144, 244A, or network programming experience.
3 units, given next year

CS 344B. Advanced Topics in Distributed Systems
Continuation of 244B. The use of distributed systems research in practical systems. New applications due to the growth in high-bandwidth connections. Distributed systems knowledge and techniques from research and system implementations, and active research topics. Readings include research publications.
2 units, not given this year

CS 345. Advanced Topics in Database Systems
Content varies. May be repeated for credit with instructor consent. Prerequisite: 145. Recommended: 245.
3 units, offered occasionally

CS 345A. Data Mining
Algorithms for mining large-scale data, including data from the web and data maintained by web-based enterprises. Finding frequent itemsets; finding similar sets using minhashing, locality-sensitive hashing, and index-based methods; finding important web pages by PageRank; link-spam detection; collaborative filtering; stream mining; clustering; optimizing ad selection; virtual databases and extraction of relations from the web.
3 units, Win (Rajaraman, A; Ullman, J)

CS 345C. Data Integration
Techniques for integrating data from multiple heterogeneous data sources. Topics: semantic heterogeneity; languages for mediating between disparate data sources; techniques for automatic schema reconciliation and reference reconciliation; adaptive query processing; basics of XML and its relevance to data integration; peer-to-peer data sharing data exchange; combining structured and unstructured data; and dataspaces. Recommended: 145.
3 units, not given this year

CS 345L. Large-Scale Data Mining
(Same as CME 340) Focus is on very large scale data mining on the web and on social networks. Topics include network models, ranking algorithms, reputation, collaborative filtering, and supervised and unsupervised learning. Individual or group applications-oriented programming project. 1 unit without project; 3 units with final project. Prerequisites: programming at the level of CS 108; statistics at the level of MATH 103 and STATS 116. Recommended: machine learning at the level of CS 229; knowledge of Java.
1-3 units, not given this year

CS 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, programming experience in C++.
3-5 units, Spr (Agrawal, P; Park, H)

CS 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, H)

CS 348A. Computer Graphics: Geometric Modeling
The mathematical tools needed for the geometrical aspects of computer graphics and especially for modeling smooth shapes. 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. Subdivision surfaces and multiresolution representations of geometry. Representations of solids and conversions among them. Surface reconstruction from scattered data points. Geometry processing on meshes, including simplification. Prerequisite: linear algebra. Recommended: 164.
3-4 units, Aut (Guibas, L)

CS 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 antialiasing 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: Fourier analysis or digital signal processing.
3-4 units, Spr (Hanrahan, P)

CS 349. Topics in Programming Systems
Advanced material is often taught for the first time as a topics course, perhaps by a faculty member visiting from another institution. May be repeated for credit.
3 units, offered occasionally

CS 349C. Topics in Programming Systems: Readings in Distributed Systems
Discussion of research publications that are of current interest in distributed systems. Students are expected to read all papers, and sign up for presentation of one paper. The course itself is 1 unit. Those interested in working on a project along with the readings should enroll for 3 units.
1-3 units, Aut (Cao, P; Danzig, P)

CS 355. Advanced Topics in Cryptography
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. May be repeated for credit. Prerequisite: 255.
3 units, not given this year

CS 357. Advanced Topics in Formal Methods
Topics vary annually. Possible topics include automata on infinite words, static analysis methods, runtime analysis methods, verification of real-time and hybrid systems, and formalization of middleware services. May be repeated for credit. Prerequisite: 256.
3 units, offered occasionally

CS 359. Topics in the 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. May be repeated for credit.
3 units, offered occasionally

CS 359D. Hardness of Approximation
Results on and proof techniques for ruling out good approximation algorithms for NP-hard optimization problems. Topics: the PCP theorem; parallel repetition theorem; the unique games conjecture; applications to set cover, clique, max cut, network design, and problems. Prerequisites: 154 and 261, or equivalents.
3 units, not given this year

CS 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, online algorithms and competitive analysis, strings and pattern matching, heuristic and probabilistic analysis (TSP, satisfiability, cliques, colorings), local search algorithms. May be repeated for credit. Prerequisite: 161 or 261, or equivalent.
3 units, not given this year

CS 361B. Advanced Algorithms
Topics: fundamental techniques used in the development of exact and approximate algorithms for combinational optimization problems such as generalized flow, multicommodity flow, sparsest cuts, generalized Steiner trees, load balancing, and scheduling. Using linear programming, emphasis is on LP duality for design and analysis of approximation algorithms; interior point methods for LP. Techniques for development of strongly polynomial algorithms.
3 units, Spr (Plotkin, S)

CS 364A. Algorithmic Game Theory
Topics at the interface of theoretical computer science and game theory such as: algorithmic mechanism design; combinatorial and competitive auctions; congestion and potential games; cost sharing; existence, computation, and learning of equilibria; game theory and the Internet; network games; price of anarchy; and selfish routing. Prerequisites: 154N and 161, or equivalents.
3 units, not given this year

CS 364B. Topics in Algorithmic Game Theory
Topics on the interface of theoretical computer science and game theory. May be taken prior to 364A; may be repeated for credit. Prerequisites: 154N and 161, or equivalents.
3 units, not given this year

CS 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, online algorithms, number-theoretic algorithms. Prerequisites: CS 161 or 261, STATS 116 or CS 109, or equivalents.
3 units, not given this year

CS 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. May be repeated for credit.
3 units, offered occasionally

CS 369A. Advanced Geometric Algorithms
Approximate, randomized, and high-dimensional geometric algorithms. Topics of current interest: clustering; nearest-neighbor search; shortest paths; geometric random walks; shape fitting; geometric embeddings; coresets; geometric TSP; and linear programming. Prerequisites: 368 or equivalent.
3 units, not given this year

CS 369F. Topics in Analysis of Algorithms
Focus is on combinatorial optimization with emphasis on online algorithms.
3 units, not given this year

CS 369M. Algorithms for Modern Massive Data Set Analysis
Algorithmic and statistical methods for large-scale data analysis: matrix and graph algorithms; strengths and weaknesses of theoretical techniques for practical scientific and internet data analysis; overlap with related problems in statistics, optimization, numerical analysis, and machine learning. Representative topics: Matrix problems (numerical and statistical perpectives; algorithmic approaches, including Johnson-Lindenstrauss lemma and randomized projection and sampling algorithms; novel matrix factorizations); Graph problems (graph partitioning algorithms, including spectral methods, flow-based methods, and recent geometric methods; local graph algorithms and approximate eigenvector computation); and applications to machine learning and statistical data analysis (motivating applications; algorithmic basis of the RKHS method; geometric data analysis, regularization, and statistical inference; boosting and its relationships to conjugate gradient methods, duality, convexity, online
3 units, Aut (Mahoney, M)

CS 369N. Novel Paradigms for Algorithmic Analysis
May be repeated for credit.
3 units, Aut (Roughgarden, T)

CS 374. Algorithms in Biology (same as CS 374)
(Same as BIOMEDIN 374) Algorithms and computational models applied to molecular biology and genetics. Topics vary annually. Possible topics include biological sequence comparison, annotation of genes and other functional elements, molecular evolution, genome rearrangements, microarrays and gene regulation, protein folding and classification, molecular docking, RNA secondary structure, DNA computing, and self-assembly. May be repeated for credit. Prerequisites: 161, 262 or 274, or BIOCHEM 218, or equivalents.
2-3 units, Spr (Batzoglou, S)

CS 376. Research Topics in Human-Computer Interaction
Interactive systems, research areas in interaction techniques, and the design, prototyping, and evaluation of user interfaces. Topics: computer-supported cooperative work; audio, speech, and multimodal interfaces; user interface toolkits; design and evaluation methods; ubiquitous and context-aware computing; tangible interfaces, haptic interaction; and mobile interfaces.
3 units, Spr (Klemmer, S)

CS 377. Topics in Human-Computer Interaction
Contents change each quarter. May be repeated for credit. See http://hci.stanford.edu/academics for offerings.
2-3 units, offered occasionally

CS 377L. Learning in a Networked World
(Same as EDUC 298) Foundations, theories and empirical studies for interdisciplinary advances in how we conceive of the potentials and challenges associated with lifelong, lifewide and life-deep learning in a networked world given the growth of always-on cyberinfrastructure for supporting information and social networks across space and time with personal computers, netbooks, and mobiles.
3 units, Spr (Pea, R)

CS 377V. Persuasive Online Video: Methods and Metrics for Changing Behaviors
New methods for creating persuasive video, with a focus on metrics in guiding iterative design. Small teams create videos to achieve target behaviors of their own choosing. Goals go beyond viral distribution, which is just one potential target behavior (sharing the video with others). May be repeated for credit.
2 units, not given this year

CS 377W. Create Engaging Web Applications Using Metrics and Learning on Facebook
Experimental course. Students work in small, interdisciplinary teams to create, launch, and optimize web-based applications for social networks such as Facebook. Tools include Google Analytics. Online experiments and user responses to learn how to iterate and improve applications. Guest experts.
3-4 units, not given this year

CS 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, Aut (Winograd, T)

CS 379L. Designing Liberation Technology
(Same as POLISCI 337T) Small project teams work with NGOs to design new technologies for promoting development and democracy. Students conduct observations to identify needs, generate concepts, create prototypes, and test their appropriateness. Some projects may continue past the quarter towards full-scale implementation. Taught through the Hasso Plattner Institute of Design at Stanford (http://dschool.stanford.edu). Enrollment limited. Prerequisites: consent of instructors; application.
3 units, Spr (Cohen, J; Winograd, T)

CS 390A. Curricular Practical Training
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 complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. 390 A, B, and C may each be taken once.
1 unit, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 390B. Curricular Practical Training
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 complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. 390A,B,C may each be taken once.
1 unit, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 390C. Curricular Practical Training
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 complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. 390A,B,C may each be taken once.
1 unit, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 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.
1-9 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 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.
1-6 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 399. Independent Project
Letter grade only.
1-9 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 399P. Independent Project
Graded satisfactory/no credit.
1-9 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 402. Beyond Bits and Atoms: Designing Technologies for Thinking and Learning
(Same as EDUC 236X) Practicum in designing and building technology-enabled curricula and learning environments. Students use software toolkits and state-of-the-art fabrication machines to design educational software, educational toolkits, and tangible user interfaces. How to design low-cost technologies, particularly for urban school in the US and abroad. The constructionist learning design perspective, critical pedagogy, and the application of complexity sciences in education.
3-5 units, not given this year

CS 447. Software Design Experiences
Small teams develop technology prototypes combining product and interaction design. Focus is on software and hardware interfaces, interaction, design aesthetics, and underpinnings of successful design including a reflective, interactive design process, group dynamics of interdisciplinary teamwork, and working with users. Prerequisite: CS 247A.
3-4 units, alternate years, not given this year

CS 448. Topics in Computer Graphics
Topic changes each quarter. Recent topics: computational photography, data visualization, character animation, virtual worlds, graphics architectures, advanced rendering. See http://graphics.stanford.edu/courses for offererings and prerequisites. May be repeated for credit.
3-4 units, offered occasionally

CS 448A. Computational Photography
Sensing strategies and algorithmic techniques that extend traditional digital photography. Topics: high dynamic range imaging, flash-noflash, coded aperture, coded exposure, multi-perspective, panoramic stitching, digital photomontage, all-focus, and light field imaging. Lectures, readings, and project. Prerequisite: 148 or equivalent.
3-4 units, Win (Levoy, M)

CS 448B. Topics in Computer Graphics: Data Visualization
Techniques and algorithms for creating effective visualizations based on principles from graphic design, visual art, perceptual psychology, and cognitive science. Topics: graphical perception, data and image models, visual encoding, graph and tree layout, color, animation, interaction techniques, automated design. Lectures, reading, and project. Prerequisite: one of 147, 148, or equivalent.
3 units, Aut (Heer, J)

CS 448E. Research Topics in Computer Graphics: Virtual Worlds
Selected topics in current computer graphics research. Analysis of research publications, class discussions, quarter-long research project. Topics change each offering. Sample topics: procedural modeling, character animation, multimodal interfaces, perception and cognition. May be repeated for credit. Prerequisite: CS248.
1-4 units, Spr (Koltun, V)

CS 448F. Image Processing for Photography and Vision
Image processing with a focus on implementation of new techniques from the literature. Topics: sampling and reconstruction, linear and non-linear filters, features and alignment, compositing, gradient-domain techniques, and recent techniques from conferences such as SIGGRAPH and Eurographics. Prerequisites: Students should be comfortable coding in C++. An introductory graphics course such as CS148 is helpful but not necessary.
3 units, Aut (Adams, A)

CS 450. Introduction to Biotechnology
Academic and industrial experts discuss latest developments in fields such as bioenergy, green process technology, the production of industrial chemicals from renewable resources, protein pharmaceutical production, industrial enzyme production, stem cell applications, medical diagnostics, and medical imaging. Discussions of biotechnology ethics, business and patenting issues, and entrepreneurship in biotechnology.
3 units, not given this year

CS 468. Topics in Geometric Algorithms: Computational Topology
Focus on the connectivity of spaces (and ignoring, for example, metric information) one arrives at the study of topology. Concentrating mostly on the invariants arising from algebraic topology, course presents techniques for designing efficient algorithms to compute them. Alongside the algorithms, course presents the necessary background tools, both in topology and computer science, for their analysis. Topics: graphs, surfaces, simplicial complexes, (co)homology, topological data analysis, Morse functions.
3 units, Aut (Morozov, D)

CS 499. Advanced Reading and Research
For CS graduate students. Register using the section number associated with the instructor. Prerequisite: consent of instructor.
1-15 units, Aut (Staff), Win (Staff), Spr (Staff), Sum (Staff)

CS 523. The Future of the Automobile
(Same as ME 302) The concept of this course is to present, discuss, and envision the future of the automobile in terms of sustainability, safety, performance, and enjoyment. Invited speakers from academia and industry share their own visions, explain challenges, and present solutions regarding individual transportation. After each session the students research specific questions related to the lectures and present their findings in the following week. This course is offered by Stanford's Automotive Program with faculty involvement from mechanical engineering, computer science, and communications.
1 unit, Aut (Beiker, S; Gerdes, C; Thrun, S), Win, Spr (Staff)

CS 545. Database and Information Management Seminar
Current research and industrial innovation in database and information systems.
1 unit, Win (Widom, J)

CS 546. Seminar on Liberation Technologies
(Same as POLISCI 337S)
1 unit, Aut (Winograd, T; Cohen, J; Diamond, L)

CS 547. Human-Computer Interaction Seminar
Weekly speakers. May be repeated for credit.
1 unit, Aut (Winograd, T), Win (Winograd, T), Spr (Winograd)

CS 571. Surgical Robotics Seminar
Surgical robots developed and implemented clinically on varying scales. Goal is to expose students from engineering, medicine, and business to guest lecturers from academia and industry.engineering and clinical aspects connected to design and use of surgical robots, varying in degree of complexity and procedural role.
1 unit, Aut (Barbagli, F)