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 Weiderhold*
Chair: William J. Dally
Associate Chair for Education: Mehran Sahami
Professors: Alex Aiken, David Cheriton, William J. Dally, David Dill, Hector Garcia-Molina, Gene H. Golub, Leonidas J. Guibas, Patrick Hanrahan, John Hennessy, Mark A. Horowitz, Oussama Khatib, Daphne Koller, Monica Lam, Jean-Claude Latombe, Marc Levoy, Zohar Manna, John Mitchell, Rajeev Motwani, Yoav Shoham, Sebastian Thrun, Jennifer Widom, Terry Winograd
Associate Professors: Dan Boneh, Dawson Engler, Ronald P. Fedkiw, Michael Genesereth, Christopher Manning, David Mazieres, Nick McKeown, Serge A. Plotkin, Balaji Prabhakar, Mendel Rosenblum
Assistant Professors: Serafim Batzoglou, Gill Bejerano, Scott Klemmer, Vladlen Koltun, Christoforos Kozyrakis, Philip Levis, Andrew Ng, Tim Roughgarden
Professor (Research): John K. Salisbury
Professor (Teaching): Eric S. Roberts
Associate Professor (Teaching): Mehran Sahami
Courtesy Professors: Russ Altman, Bernd Girod, Teresa Meng, Mark Musen, Kunle Olukotun, Fouad A. Tobagi
Courtesy Associate Professors: Ashish Goel, Dan Jurafsky
Courtesy Assistant Professors: Ramesh Johari, Benjamin Van Roy
Lecturers: Gerald Cain, Nicholas J. Parlante, Robert Plummer, Patrick Young, Julie Zelenski
Consulting Professor: Gary Bradski
Consulting Assistant Professor: Federico Barbagli
Visiting Professor: Martin Abadi

* Recalled to active duty.

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

Courses given in Computer Science have the subject code CS. For a complete list of subject codes, see Appendix.

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, a two CPU Dell PowerEdge 2850 Xeon, 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 PCs, Macs, multi-CPU computer clusters, and file servers. Servers and workstations manufactured by SUN, Dell, and Apple are commonplace. Support for course work and instruction is provided on systems available through Information Technology Systems (ITS) and the School of Engineering (SoE).

UNDERGRADUATE PROGRAMS

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.

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

In addition to Computer Science itself, Stanford offers several interdisciplinary degrees with a substantial computer science component. The Computer Systems Engineering major (also in Engineering) allows the study of 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

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.

GRADUATE PROGRAMS

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

MASTER OF SCIENCE

In general, the M.S. degree in Computer Science is intended as a terminal professional degree and does not lead to the Ph.D. degree. Most students planning to obtain the Ph.D. degree should apply directly for admission to the Ph.D. program. Some students, however, may wish to complete the master's program before deciding whether to pursue the Ph.D. To give such students a greater opportunity to become familiar with research, the department has instituted a 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 11, 2007. 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 103A, B, or X, 106A, B, or X, 107, 108; MATH 103.

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 (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)
      3) Introduction to Computer Networks (CS 244A 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 205A, 222, 225A, 225B, 227B, 256, 262, 270, 273A, 274, 275, 276, 277, 278, 279, 294A, 321, 327A, 328, 329, 374, 377,* 379*; ECON 286; EE 263, 376A; ENGR 205, 209A; LINGUIST 180; 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, 244A, 255
   b) at least three of: CS 240, 244B, 244C, 259, 261, 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, 244A, 244B, 244C, 249A, 249B, 255, 262, 270, 271, 272, 276, 315A, 344, 374
5. Human-Computer Interaction
   a) CS 147, 247; MS&E 430
   b) at least two of: CS 148 or 248, 376, 377 (may be repeated for credit), 378, 447; COMM 207, 268, 269; EDUC 124; MUSIC 250A; SYMBSYS 145
   c) a total of 21 units from categories (a), (b), and the following: CS 221, 223B, 229, 242, 249A, 249B, 276, 448; COMM 272; LINGUIST 180; MS&E 234; ME 101, 115, 313, 314; PSYCH 205, 221, 252
6. Numerical Analysis/Scientific Computation
   a) CME 302, 306, 326
   b) at least two of: CS 205A; 205B; MS&E 121; MATH 131, 132, 220A, 220B, 220C; STATS 200
   c) at least two of: CS 223A, 327A, 328, 339; AA 214A, 214B; CME 324, 342; STATS 227
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, 256, 258
   b) at least one of: CS 244A, 245, 295, 343, 345
   c) at least one course from the following: CS 255, 259, 261, 268, 355, 356, 361A, 361B, 365
   d) at least one additional course chosen from (b), (c), and CS 346
9. Systems
   a) CS 240, 242
   b) at least three of: CS 243, 244A, 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, 343, 344, 344B, 344E, 345, 346, 347, 348A, 349, 374, 448; EE 384A, 384B, 384C, 384S, 384X, 384Y
10. Theoretical Computer Science
   a) CS 256, 258, 261 (361A, 361B, or 365 may be substituted for 261)
   b) at least four additional courses chosen from CS 228, 255, 259, 262, 268, 345, 355, 356, 357, 358, 359,* 361A, 361B, 364A, 364B, 365, 369,* 374; MS&E 310

* With consent of specialization chair.

Requirement 5—Additional elective units must be technical courses (numbered 100 or above) related to the degree program and approved by the adviser. Elective courses may be taken on a satisfactory/no credit basis provided that a minimum of 36 graded units 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. Two copies of the research report must be submitted to the Student Services office in the department three weeks before the beginning of the examination period in the student's final quarter. Both the primary and secondary adviser must approve the research report before the distinction-in-research designation can be conferred.

DOCTOR OF PHILOSOPHY

Applications to the Ph.D. program and all supporting documents must be submitted and received online by December 11, 2007. See http://cs.stanford.edu/wiki/admissions/ for complete information. Changes or updates to the admission process are posted in September and October, 2007. 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. This is typically held after all or a substantial portion of the dissertation research has been completed.
6. The student is expected to demonstrate the ability to present scholarly material orally, both in the dissertation defense and by a lecture in a department seminar.
7. The dissertation must be accepted by a reading committee composed of the principal dissertation adviser, a second member from within the department, and a third member chosen from within the University. The principal adviser and at least one of the other committee members must be Academic Council members.

PH.D. MINOR

For a minor in Computer Science, a candidate must complete 20 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. All courses must be taken for a letter grade and passed with a grade 'B' or better. Applications for a minor in Computer Science are submitted at the same time as admission to candidacy.

TEACHING AND RESEARCH ASSISTANTSHIPS

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.

COURSES

WIM indicates that the course satisfies the Writing in the Major requirement. (AU) indicates that the course is subject to the University Activity Unit limitations (8 units maximum).

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, which covers most of the material in 106A and B in a single quarter. 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 106X may be in any language. In all cases, students are encouraged to discuss their background with the instructors responsible for these courses.

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

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

NUMBERING SYSTEM

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

1- 99	Service courses for nontechnical majors
100-199	Other service courses, basic undergraduate
200-299	Advanced undergraduate/beginning graduate
300-399	Advanced graduate
400-499	Experimental
500-599	Graduate seminars

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

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

NON-MAJOR

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. Basics including security breach precautions and media labs for class presentations. Computer labs and resources, email and printing, AFS space, and computer maintenance and security. One-hour lecture/demonstration in dormitory clusters prepared and administered weekly by the Resident Computer Coordinator (RCC). Final project. Not a programming course.
1 unit, Aut (Ly, J)

CS 2C. Intermediate Computing at Stanford—Continuation of 1C. Sound, image and video editing, and publishing media. Applications include: Audacity, Dreamweaver, Photoshop, and Powerpoint. One-hour lecture/demonstration in dormitory clusters prepared and administered weekly by the Resident Computer Coordinator (RCC). Final project. Not a programming course.
1 unit, Win (Ly, J)

CS 12N. The Coming Revolution in Computer Architecture—Stanford Introductory Seminar. Preference to freshmen. Classic architecture and technology trends that have driven its performance growth, and factors leading to the end of this growth; characteristics of technology that can be exploited for future growth and driving application areas; alternative organizations and programming systems that are candidates to replace the status quo. GER:DB-EngrAppSci
3 units, Spr (Dally, W)

CS 20N. The Role of Information Technology in Global Conflict Resolution—Stanford Introductory Seminar. Preference to freshmen. Larger aspects of the impact of the Internet such as war and peace. Online activities of peace NGOs. Group project to build a portal for NGO activities concerning the Israeli-Palestinian conflict. Web site design and group project management. GER:DB-EngrAppSci
3 units, Spr (Shoham, Y)

CS 26N. Motion Planning for Robots, Digital Actors, and Other Moving Objects—Stanford Introductory Seminar. Preference to freshmen. Motion planning theory and computational approaches. Intriguing algorithms, representations, and applications. Terminology and concepts for reading motion planning research literature. Problems may 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; and how molecules change shapes to perform biological functions. GER:DB-EngrAppSci
3 units, Aut (Latombe, J)

CS 73N. Business on the Information Highways—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, Write-2
3 units, Spr (Wiederhold, G; Barr, A; Tessler, S)

CS 74N. Digital Dilemmas—Stanford Introductory Seminar. Preference to freshmen. Issues where policy decision making requires understanding computer and communications technology. Technology basics in non-technology terms. Topics include intellectual property, privacy, Internet neutrality, security and cryptography. Guest speakers. GER:DB-EngrAppSci
3 units, Aut (Dill, D)

UNDERGRADUATE

CS 103A. Discrete Mathematics for Computer Science—Mathematical foundations required for computer science. Topics: propositional and predicate logic, proof techniques, induction, recursion, combinatorics, and functions. Corequisite: 106A or X. GER:DB-Math
3 units, Aut, Win (Plummer, R)

CS 103B. Discrete Structures—Continuation of 103A. Topics: analysis of algorithms, recurrence relations, mathematical formulations of basic data models (sets, relations, linear models, trees and graphs), regular expressions, grammars, and finite automata. Corequisite: 106B or X. GER:DB-Math
3 units, Win, Spr (Sahami, M)

CS 103X. Discrete Structures (Accelerated)—Covers the material in 103A and B in a single quarter. The mathematical foundations of computer science. Introduction to set theory and logic, number theory, functions and relations, combinatorics, and graph theory. Prerequisite: background in mathematical formalism and mathematical proof. GER:DB-Math
3-4 units, Win (Koltun, V)

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, 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, Spr (Young, P), 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, Win (Zelenski, J), Spr, Sum (Staff)

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 (Zelenski, J), Win (Cain, G)

CS 107. Programming Paradigms—Advanced memory management features of C and C++; the differences between imperative and object-oriented paradigms. The functional paradigm (using LISP) and concurrent programming (using C and C++). Brief survey of other modern languages such as Python, Objective C, and C#. GER:DB-EngrAppSci
3-5 units, Aut, Spr (Cain, G)

CS 107L. Programming Paradigms Laboratory—Advanced C++ topics beyond the scope of 107. Topics: advanced memory management; placement new; manual destruction; operator overloading; STL template containers; algorithms; iterators; single and multiple inheritance; class hierarchy design; and C++ pitfalls.
1 unit, Aut, Spr (Cain, G)

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, Win (Parlante, N)

CS 121. Introduction to Artificial Intelligence—(Only one of 121/221 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: 103B or X, and facility with differential calculus, vector algebra, and probability theory. GER:DB-EngrAppSci
3 units, Spr (Latombe, J), Sum (Staff)

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: 107. GER:DB-EngrAppSci
3-4 units, Aut (Mazieres, D), Win (Rosenblum, M)

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: 103B or X, and 107. GER:DB-EngrAppSci
3-4 units, Aut (Cain, G), Sum (Staff)

CS 144. Introduction to Computer Networking—Only one of 144 or 244A counts towards any CS degree program. 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: 108 or equivalent. GER:DB-EngrAppSci
4 units, Spr (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 and object-relational databases. XML data including the query languages XPath and XQuery. UML database design, and relational design principles based on functional dependencies and normal forms. Other topics include indexes, views, transactions, authorization, integrity constraints, and triggers. Advanced topics from data warehousing, data mining, web data management, Datalog, data integration, data streams and continuous queries, and data-intensive web services. Prerequisites: 103B or X, and 107. GER:DB-EngrAppSci
3-4 units, Aut (Ullman, 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: 106A or equivalent background in programming.
3-4 units, Aut (Klemmer, S)

CS 148. Introductory Computer Graphics—(Only one of 148 or 248 counts towards any CS degree program.) For undergraduates; M.S. students and those interested in continuing in graphics, register for 248. Two- and three-dimensional computer graphics. Topics: input and display devices, scan conversion of geometric primitives, two- and three-dimensional transformations and clipping, windowing techniques, curves and curved surfaces, three-dimensional viewing and perspective, hidden surface removal, illumination and color models, OpenGL, and 3-D modeling tools. Emphasis is on practical skills in using graphics libraries and tools. Programming using C/C++ and OpenGL, with demos in SoftImage. Prerequisites: 107, MATH 103. GER:DB-EngrAppSci
3 units, Win (Hanrahan, P), Sum (Staff)

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: 103B or X. GER:DB-EngrAppSci
3-4 units, Aut (Dill, D), Spr (Motwani, R), Sum (Staff)

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 (Motwani, R)

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: 103B or X. GER:DB-EngrAppSci

3 units, Aut (Genesereth, M)

CS 161. Design and Analysis of Algorithms—Efficient algorithms for sorting, searching, and selection. Algorithm analysis: worst and average case analysis. Recurrences and asymptotics. Data structures: balanced trees, heaps, hash tables. Algorithm design techniques: divide-and-conquer, dynamic programming, greedy algorithms, amortized analysis. Algorithms for fundamental graph problems such as depth-first search, connected components, topological sort, and shortest paths. Possible additional topics: network flow, string searching, parallel computation. Prerequisite: 103B or X; STATS 116. GER:DB-EngrAppSci

3-4 units, Aut (Plotkin, S), Win (Roughgarden, T), Sum (Staff)

CS 191. Senior Project—Restricted to Computer Science and Computer Systems Engineering students. Group or individual projects under faculty direction. Register using instructor's section number. 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, Win, Spr, Sum (Staff)

CS 191W. Writing Intensive Senior Project—Restricted to Computer Science and Computer Systems Engineering students. Writing-intensive version of 191. Register using section number of an Academic Council member. WIM

3-6 units, Aut, Win, 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, Win, Spr, 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.

3 units, not given this year

CS 193E. Mac OS X Cocoa Programming—Hands-on project using the Cocoa frameworks for the Mac OS X platform. The essentials of designing and implementing graphical applications using Cocoa tools and APIs. Topics include: object-oriented event-driven programming; Objective-C language; development tools such as Interface Builder, XCode, and debugging and profiling tools; APIs for the foundation and application kits; and the Quartz graphic system. Requirements: C language and programming experience at the level of 106B/X. Recommended: UNIX, object-oriented programming, and graphical toolkits.

3 units, Win (Staff)

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. Prerequisite: 108. WIM

3 units, Spr (Plummer, R)

CS 196. Microcomputer Consulting—Focus is on Macintosh and PC systems. Computer use, maintenance, and troubleshooting. Topics: hardware, operating systems, networking, security, troubleshooting, and consulting methodology focusing on Stanford's computing environment. Final project. Not a programming course. Prerequisite: 1C or equivalent.

2 units, Win, 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. Prerequisite: 106B or X.

4 units, Aut, Win, Spr (Sahami, M; Jachowski, M; Kim, I)

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, Win, Spr, Sum (Staff)

CS 199P. Independent Work

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

UNDERGRADUATE AND GRADUATE

CS 201. Computers, Ethics, and Social Responsibility—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:DB-EthicReas, WIM

3-4 units, Win (Johnson, M)

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 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 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, machine learning, neural networks, vision, robotics, and natural language understanding. Prerequisites: 103B or 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; vanBenthem, J)

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 (Kosecka, J)

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, but emphasis is on conceptual matters and theoretical foundations. Prerequisites: basic probability theory and first-order logic.
3 units, Win (Shoham, Y)

CS 224N. Natural Language Processing—(Same as LINGUIST 280.) Methods for processing linguistic information and the underlying computational properties of natural languages. Syntactic and semantic processing from a linguistic and an algorithmic perspective. Focus is on modern quantitative techniques in NLP: using large corpora, statistical models for acquisition and interpretation, and representative systems. Prerequisites: CS 121/221 or LINGUIST 180, programming experience, familiarity with logic and probability.
3-4 units, Spr (Manning, C)

CS 224S. Speech Recognition and Synthesis—(Same as LINGUIST 281.) 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 188/288.) Machine understanding of human language. Computational semantics (determination of sense, event structure, thematic role, time, aspect, synonymy/meronymy, causation, compositional semantics, treatment of scopal operators), and computational pragmatics and discourse (coherence relations, anaphora resolution, information packaging, generation). Theoretical issues, online resources, and relevance to applications including question answering, summarization, and textual inference. Prerequisites: one of LINGUIST 180, CS 224N,S; and logic such as LINGUIST 130A or B, CS 157, or PHIL150).
2-4 units, Aut (Jurafsky, D; Manning, C)

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 226. Statistical Techniques in Robotics—For students seeking to develop robust robot software and those interested in real-world applications of statistical theory. Probabilistic state estimation, Bayes filters, Kalman filters, information filters, 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, not given this year

CS 227. Reasoning Methods in Artificial Intelligence—Technical presentation of algorithmic techniques for problem solving in AI. Combines formal algorithmic analysis with a description of recent applications. Topics: propositional satisfiability, constraint satisfaction, planning and scheduling, diagnosis and repair. Focus is on recent results. Prerequisites: familiarity with the basic notions in data structures and design and with techniques in the design and analysis of algorithms. Recommended: previous or concurrent course in AI.
3 units, not given this year

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. Topics: theoretical foundations and 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 229. Machine Learning—Topics: statistical pattern recognition, linear and non-linear regression, non-parametric methods, exponential family, GLIMs, 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, Win (Mazieres, D), 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. Advanced Compiling Techniques—The theoretical and practical aspects of building modem compilers. Topics: program optimizations including data flow analysis, instruction scheduling, register allocation, loop transforms for data locality and parallelism, interprocedural analysis, and garbage collection. Prerequisite: 143 or equivalent.
3-4 units, Win (Lam, M)

CS 244A. Introduction to Computer Networks—Only one of 144 or 244A counts towards any CS degree program. Packet switching; the Internet architecture; routing; router architecture; flow control algorithms; retransmission algorithms; congestion control, TCP/IP; detecting and recovering from errors; switching; Ethernet (wired and wireless) and local area networks; physical layers; clocking and synchronization. Assignments introduce network programming, including sockets, designing a router and implementing a transport layer. EE 284 is an alternate class, with less emphasis on programming; students may not take both EE 284 and CS 244A. Prerequisite: 140 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. Prerequisite: 249A. Corequisite: 244A.
3 units, Spr (Cheriton, D)

CS 244C. Distributed Systems Project—Companion project option for 244B. Corequisite: 244B.
3-6 units, Spr (Cheriton,D)

CS 244E. Low Power Wireless Networking—Challenges of low power wireless networking protocols and applications. Topics: the OSI model, 802.11, Bluetooth, 802.15.4, Zigbee, 6lowpan, hardware considerations, traffic patterns, media access (CSMA, TDMA, RTS/CTS, idle listening), DSSS, UWB, radio propagation models, cross-layer interactions, flooding, dissemination, gossip, epidemics, probabilistic approaches, global versus local communication, and in-network processing. Students read papers and build working protocols on TinyOS, a low-power embedded operating system.
3 units, not given this year

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)

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; Verplank, W)

CS 247L. Human Computer Interaction Technology Laboratory—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. Introduction to Computer Graphics—(Only one of 148 or 248 counts towards any CS degree program.) Input and display devices, scan conversion of geometric primitives, 2D and 3D geometric transformations, clipping and windowing, scene modeling and animation, algorithms for visible surface determination, local and global shading models, color, and real-time rendering methods. Written assignments and programming projects. Prerequisites: 108, MATH 103 or equivalent.
3-5 units, Aut (Akeley, K)

CS 249A. Object-Oriented Programming: A Modeling and Simulation Perspective—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. 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, Spr (Manna, Z)

CS 256L. Formal Methods for Reactive Systems Laboratory—Practical application of the specification and verification methods in 256. Individual projects include implementation of verification methods, verification case studies, or tool evaluation, depending on student preference.
2 units, Spr (Manna, Z)

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. Topics in Theoretical Computer Science—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, Win (Mitchell, J)

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, 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.
3 units, Spr (Staff)

CS 270. Introduction to Biomedical Informatics: Fundamental Methods—(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. Topics: knowledge representation, controlled terminologies, ontologies, reusable problem solvers, and knowledge acquisition. Recommended: exposure to object-oriented systems, basic knowledge of biology.
3 units, Aut (Musen, M)

CS 271. Introduction to Biomedical Informatics: Biomedical Systems Engineering—(Same as BIOMEDIN 211.) Focus is on undertaking design and implementation of computational and information systems for life scientists and healthcare providers. Case studies illustrate what design factors lead to success or failure in building systems in complex biomedical environments. Topics: requirements analysis, workflow and organizational factors, functional specification, knowledge modeling, data heterogeneity, component-based architectures, human-computer interaction, and system evaluation. Prerequisite: 210, or consent of instructor.
3 units, Win (Das, A; Islam, R; Levy, M)

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: 210, 211 or 214, 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.) Genomes as the ultimate biological information medium, carrying instructions for every organism's development, life cycle, and reproduction. Bioinformatics perspective. Advances in biology resulting from sequencing of human and related organisms. Genome sequencing: technologies, assembly, personalized sequencing. Functional landscape: genes, regulatory modules, repeats, RNA genes. Genome evolution: processes, comparative genomics, ultraconservation, exaptation. Topics may include population genetics and personalized genomics, ancient DNA, and metagenomics. Prerequisities: computational biology at the level of 262, 274, or BIOC 218.

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: algorithms for alignment of biological sequences and structures, computing with strings, phylogenetic tree construction, hidden Markov models, computing with networks of genes, basic structural computations on proteins, protein structure prediction, protein threading techniques, homology modeling, molecular dynamics and energy minimization, statistical analysis of 3D biological data, integration of data sources, knowledge representation and controlled terminologies for molecular biology, graphical display of biological data, and machine learning (clustering and classification), and natural language text processing. Consent of instructor required for 3 units. Prerequisites: programming skills, interest in biology.

3-4 units, Spr (Altman, R)

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; Hillenmeyer, M; Southworth, L)

CS 276. Text Retrieval and Web Search—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, cluster labeling; Web search engines including crawling and indexing, link-based algorithms, and web metadata.

3 units, not given this year

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, Spr (Salisbury, K)

CS 278. Introduction to Systems Biology—(Same as CSB 278.) For biologists, engineers, and computer scientists. Experimental and computational approaches to modeling and analysis of complex biological systems. Topics: biological noise; simple signaling circuits (cascades, feedback, and feed-forward circuits); bistability and oscillations; large scale models; synthetic biology; and analysis of omics-scale data sets. Computational approaches include ODE modeling, stochastic simulation, Boolean networks, Bayesian approaches, and hybrid modeling.

4 units, Spr (Dill, D; Brutlag, D; Koller, D; Covert, M; Ferrell, J)

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.

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.

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, Win (Ng, A), Spr (Koller, D)

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. WIM`

3 units, Win (Ng, A), Spr (Koller, D)

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 (Engler, D)

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

PRIMARILY FOR GRADUATE STUDENTS

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 (Motwani, R)

CS 309. Industrial Lectureships in Computer Science—Guest computer scientist. By arrangement. May be repeated for credit.

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, Win (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 321. Information Processing for Sensor Networks—Design and implementation of algorithms and protocols for performing information processing tasks in sensor networks, including routing, data dissemination and aggregation, information discovery and brokerage, service establishment (localization, time synchronization), sensor tasking and control, and distributed data storage. Techniques from signal processing, networking, energy-ware computing, distributed databases and algorithms, and embedded systems and platforms. Physical, networking, and application layers and design trade-offs across the layers. Prerequisites: linear algebra and elementary probability, networking background at the level of 244A or EE 284.
3 units, Aut (Guibas, L)

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, Aut (Latombe, J)

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 343. Advanced Topics in Compilers—Topics change every quarter. May be repeated for credit. Prerequisite: 243.
3 units, Spr (Lam, M)

CS 344. Projects in Computer Networks—Router implementation. A hardware and software student are paired to develop a functional Internet router. Course ends with open-ended design challenge judged by panel of expert network designers from industry. Prerequisites: 244A or network programming experience. Recommended: for those interested in hardware design, background in VHDL or Verilog; for those interested in software, C.
3 units, Spr (McKeown, N)

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 344E. Sensor Network Systems—(Formerly 344A.) Systems and networking research in wireless sensor networks; work from other areas such as databases. Topics include energy budgets, communication scheduling, application domains, protocols, technological trends, programming models, and fault tolerance. Students implement working systems on TinyOS, a sensor node OS.
3 units, Spr (Levis, P)

CS 345. Advanced Topics in Database Systems—Content varies. May be repeated for credit with instructor consent. Prerequisite: 145. Recommended: 245.

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 and correlated items; web crawling; finding important web pages; link-spam detection; collaborative filtering; stream mining; clustering; optimizing ad selection; and virtual databases and extraction of relations from the web.
3 units, Win (Ullman, J; Rajaraman, A)

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, Spr (Halevy, A)

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, not given this year

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.
3-4 units, Win (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 anti-aliasing and photo-realism, human vision and color science as they relate to computer displays, and high-performance architectures for graphics. Written assignments and programming projects. Prerequisite: 248 or equivalent. Recommended: 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 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, Aut (Boneh, D)

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, formalization of middleware services. May be repeated for credit. Prerequisite: 256.
3 units, offered occasionally

CS 358. Topics in Programming Language Theory—Topics of current research interest in the mathematical analysis of programming languages, structured operational semantics, domain theory, semantics of concurrency, rich type disciplines, problems of representation independence, and full abstraction. May be repeated for credit. Prerequisites: 154, 157, 258, or equivalents.
3 units, 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 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. Game Theory in the Internet—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; Internet game theory; network games; price of anarchy and stability; pricing; and selfish routing. Minimal overlap with 224M and 324. Prerequisites: 154N and 161, or equivalents.
3 units, not given this year

CS 364B. Foundations of Sponsored Search—Further exploration of topics from 364A. Students work on a research problem related to the course. May be taken prior to 364A; may be repeated for credit. Prerequisites: 154N and 161, or equivalents.
3 units, Aut (Roughgarden, T)

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: 161 or 261, STATS 116, or equivalents.
3 units, Aut (Motwani, R)

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.

CS 369B. Advanced Graph Algorithms—Fast algorithms for graph optimization problems, including maximum flow, minimum s-t and global cuts, minimum spanning trees, nonbipartite matching, planar separators and applications, and shortest paths. Data structures including Fibonacci heaps, splay trees, and dynamic trees. Tools from linear programming, matroid theory, minmax theorems, polytope theory, and random sampling. Pre- or corequisite: 261 or equivalent.
3 units, Win (Roughgarden, T)

CS 374. Algorithms in Biology—(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.

CS 377A. Introduction to Cybernetics and the Design of Systems—Application of cybernetics to designing complex interactive systems, modeling human-computer interaction, and managing design processes for physical, virtual, social, or hybrid systems. Software applications and web services; environments for learning, business, and government; and collaboration systems for work or play. History and principles of cybernetics relative to CS and AI. Technical background not required.
3 units, Aut (Dubberly, H; Pangaro, P)

CS 377S. Designing Applications that See—Computer vision and image processing from an interaction design standpoint; application to challenges in human-computer interaction. How these methods can be used in real systems; how to use existing computer vision tools and libraries. Recommended: some programming experience.
3 units, Win (Maynes-Aminzade, D)

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, alternate years, not given this year

CS 379. Interdisciplinary Topics—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.

CS 379D. Computer Vision and Image Analysis in the Study of Art—Application of algorithms including to computer vision, image analysis, and two-dimensional Western art such as paintings, drawings, and etchings. Topics: multispectral image enhancement and color manipulation; geometric perspective and warped (anamorphic) perspective; visual metrology; view synthesis; statistical analysis of form; texture and brushstrokes; and shape-from-shading. These techniques, pattern classification, statistical estimation methods, and stylometry (quantification of artistic style) address art historical problems such as attribution, authentication, and dating to reveal artists' working methods.
3 units, Spr (Stork, D)

CS 379Y. Interdisciplinary Design for Agile Aging—(Same as HUMBIO 131, MED 279Y.) First of two quarter sequence; students may take 379Y without 379Z; offered by the d.school. Perspectives from computer science, design, social and behavioral sciences, physiology, geriatrics, and biodesign to develop projects that address the potential of people to maintain vitality and mobility as they age. New ways to integrate computer and device technologies with behavioral and social interventions. Focus is on small projects. Prerequisite: background in one of design, computing, medicine, behavioral sciences, communications, or business.
3-4 units, Win (Winograd, T; Winograd, C; Friedlander, A; Yock, P)

CS 379Z. Design Project for Agile Aging—(Same as MED 279Z.) Second of two quarter sequence; students may take 379Y without 379Z; offered by the d.school. Small teams develop projects that can have an impact in the world through products, programs, and practices that affect people's health on a broad scale. Technical interventions, social and contextual design, organizational contexts, and business and distribution issues. Limited enrollment. Prerequisites: CS379Y, and master's level skills in one of design, computing, medicine, behavioral sciences, communications, or business.
3-4 units, Spr (Winograd, T; Winograd, C; Friedlander, A; Yock, P)

CS 390A,B,C. 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, Win, Spr, 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, Win, Spr, 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, Win, Spr, Sum (Staff)

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

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

EXPERIMENTAL

CS 400. Future Faculty Seminar—How to enter and succeed in academia. Topics vary from year to year and may include the academic job search, time management for new faculty, grant writing, finding and advising students, designing courses, planning and delivering lectures, the service role of faculty, and the tenure process.
1 unit, Spr (Dally, W; Klingner, J; Beberg, A)

CS 447. Interdisciplinary Interaction Design—(Same as ME 325.) 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: exotic input and display technologies, graphics architectures, advanced rendering techniques, modeling shape and motion, data visualization, and computational photography. See http://graphics.stanford.edu/courses for offerings. May be repeated for credit. Prerequisite: 248 or consent of instructor.
3-4 units, Aut (Koltun, V)

CS 448A. Computational Photography—Capabilities unique to digital cameras. Sensors; in-camera processing systems; the ability to refocus photographs after they are taken or to combine views taken with different camera settings, aim, or placement. New technologies for creating efficient, controllable illumination such as pulsed LEDs or video projectors; the ability to selectively illuminate objects, recolor a scene, or extract shape information. How these developments relax notions of what constitutes a photograph, blur the distinction between photography and scene modeling, and lead to new photographic techniques, scientific tools, and art forms.
3-4 units, Spr (Levoy)

CS 448B. Topics in Computer Graphics—Topic changes each quarter. Recent topics: exotic input and display technologies, graphics architecures, advanced rendering techniques, modeling shape and motion, data visualization, and computational photography. See http://graphics.stanford.edu/courses/ for current offerings
1-3 units, Spr (Hanrahan)

CS 468. Topics in Geometric Algorithms—Recent offerings include: shape matching, proximity and nearest-neighbor problems, visibility and motion planning, collision detection, geometric sampling methods, shape interpolation, and computational topology. May be repeated for credit. Prerequisite: 268, 368, or consent of instructor.
3 units, Win (Koltun, V)

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, Win, Spr, Sum (Staff)

GRADUATE SEMINARS

CS 528. Broad Area Colloquium for Artificial Intelligence, Geometry, Graphics, Robotics, and Vision—Weekly series of informal research talks on topics related to perceiving, modeling, manipulating, and displaying the physical world. The computational models and numerical methods underlying these topics. May be repeated for credit.
1 unit, Aut, Spr (Staff)

CS 531. Numerical Analysis/Scientific Computing Seminar—Weekly research lectures by experts from academia, national laboratories, industry, and doctoral students. May be repeated for credit.
1 unit, not given this year

CS 541. Clean Slate Internet Research Seminar—Solving Internet deficiences. Focus is on unconventional, bold, and long-term research that tries to break the network's ossification. Given what is known today, how would a newly designed global communications infrastructure work? How should the Internet look in 15 years? Weekly speakers describe new work or propose new problems.
1 unit, not given this year

CS 545. Database and Information Management Seminar—Current research and industrial innovation in database and information systems.
1 unit, Win (Garcia-Molina, H)

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

CS 548. Internet and Distributed Systems Seminar—Guest speakers from academia and industry. May be repeated for credit.
1 unit, not given this year

COGNATE COURSES

CHEMENG 459. Frontiers in Interdisciplinary Biosciences—(Same as BIOC 459, BIOE 459, BIOSCI 459, CHEM 459, PSYCH 459.)
1 unit, Aut, Win, Spr (Robertson, C)

CME 108. Introduction to Scientific Computing
3-4 units, Win (Staff), Sum (Lambers, J)

CME 302. Numerical Linear Algebra
3 units, Aut (Golub, G)

CME 306. Numerical Solution of Partial Differential Equations
3 units, Spr (Farhat, C)

CME 324. Advanced Methods in Matrix Computation: Iterative Methods
3 units, not given this year

CME 326. Numerical Methods for Initial Boundary Value Problems
3 units, not given this year

CME 342. Parallel Methods in Numerical Analysis
3 units, Spr (Staff)

CME 352. Molecular Algorithms
3 units, Win (Goel, A)

CME 500. Numerical Analysis and Computational and Mathematical Engineering Seminar
1 unit, Aut, Win, Spr (Staff)

COMM 107/207. The First Amendment in the Digital Age
4-5 units, Spr (Noveck, B)

EE 282. Computer Systems Architecture
3 units, Aut (Kozyrakis, C)

EE 380. Seminar on Computer Systems
1 unit, Aut, Win, Spr, Sum (Allison, D; Long, E)

EE 382A. Advanced Processor Architecture
3 units, Spr (Kozyrakis, C)

EE 385A. Digital Systems Reliability Seminar
1-4 units, Aut, Win, Spr, Sum (McCluskey, E)

MS&E 319. Approximation Algorithms
3 units, alternate years, not given this year

MS&E 430. Tools for Experience Design
3-4 units, not given this year

MUSIC 253. Musical Information: An Introduction
1-4 units, Win (Selfridge-Field, E)

MUSIC 254. Applications of Musical Information: Query, Analysis, and Style Simulation
1-4 units, Spr (Selfridge-Field, E)