COMPUTER SCIENCE

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

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

Each research group in Computer Science has systems specific to its research needs. These systems include workstations (PCs, Macs), multi-CPU computer clusters, and local mail and file servers. Servers and workstations running Linux or various versions of Windows are commonplace. Support for course work and instruction is provided on systems available through U (http://itservices.stanford.edu)niversity IT (https://uit.stanford.edu) (UIT) and the School of Engineering (http://engineering.stanford.edu) (SoE).

Mission of the Undergraduate Program in Computer Science

The mission of the undergraduate program in Computer Science is to develop students' breadth of knowledge across the subject areas of computer science, including their ability to apply the defining processes of computer science theory, abstraction, design, and implementation to solve problems in the discipline. Students take a set of core courses. After learning the essential programming techniques and the mathematical foundations of computer science, students take courses in areas such as programming techniques, automata and complexity theory, systems programming, computer architecture, analysis of algorithms, artificial intelligence, and applications. The program prepares students for careers in government, law, and the corporate sector, and for graduate study.

Learning Outcomes (Undergraduate)

The department expects undergraduate majors in the program to be able to demonstrate the following learning outcomes. These learning outcomes are used in evaluating students and the department's undergraduate program. Students are expected to be able to:

1. Apply the knowledge of mathematics, science, and engineering.
2. Design and conduct experiments, as well to analyze and interpret data.
3. Design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.
4. Function on multidisciplinary teams.
5. Identify, formulate, and solve engineering problems.
6. Understand professional and ethical responsibility.
7. Communicate effectively.
8. Understand the impact of engineering solutions in a global, economic, environmental, and societal context.
9. Demonstrate a working knowledge of contemporary issues.
10. Apply the techniques, skills, and modern engineering tools necessary for engineering practice.
11. Transition from engineering concepts and theory to real engineering applications.

Learning Outcomes (Graduate)

The purpose of the master's program is to provide students with the knowledge and skills necessary for a professional career or doctoral studies. This is done through course work in the foundational elements of the field and in at least one graduate specialization. Areas of specialization include artificial intelligence, biocomputation, computer and network security, human-computer interaction, information management and analytics, mobile and internet computing, real-world computing, software theory, systems, and theoretical computer science.

The Ph.D. is conferred upon candidates who have demonstrated substantial scholarship and the ability to conduct independent research. Through course work and guided research, the program prepares students to make original contributions in Computer Science and related fields.

Graduate Programs in Computer Science

The University's basic requirements for the M.S. and Ph.D. degrees are discussed in the "Graduate Degrees (http://exploredegrees.stanford.edu/graduatedegrees)" section of this bulletin.

Computer Science Course Catalog Numbering System

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

Digit	Description
001-099	Service courses for nontechnical majors
100-199	Other service courses, basic undergraduate
200-299	Advanced undergraduate/beginning graduate
300-399	Advanced graduate
400-499	Experimental
500-599	Graduate seminars

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

Digit	Description
00-09	Introductory, miscellaneous
10-19	Hardware and Software Systems
20-39	Artificial Intelligence
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

Bachelor of Science in Computer Science

The department offers both a major in Computer Science and a minor in Computer Science. Further information is available in the *Handbook for Undergraduate Engineering Programs* published by the School of Engineering. The Computer Science major offers a number of tracks (programs of study) from which students can choose, allowing them to focus their program on the areas of most interest. These tracks also reflect the broad diversity of areas in computing disciplines. The department has an honors program, which is described in the following section.

In addition to Computer Science itself, Stanford offers several interdisciplinary degrees with a substantial computer science component. The 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. The Mathematical and Computational Sciences major (also Humanities and Sciences) allows students to explore computer science along with more mathematics, statistics, and operations research.

Computer Science (CS)

Completion of the undergraduate program in Computer Science leads to the conferral of the Bachelor of Science in Computer Science.

Mission of the Undergraduate Program in Computer Science

The mission of the undergraduate program in Computer Science is to develop students' breadth of knowledge across the subject areas of computer science, including their ability to apply the defining processes of computer science theory, abstraction, design, and implementation to solve problems in the discipline. Students take a set of core courses. After learning the essential programming techniques and the mathematical foundations of computer science, students take courses in areas such as programming techniques, automata and complexity theory, systems programming, computer architecture, analysis of algorithms, artificial intelligence, and applications. The program prepares students for careers in government, law, and the corporate sector, and for graduate study.

Requirements

Mathematics (26 units minimum)—

CS 103	Mathematical Foundations of Computing	5
CS 109	Introduction to Probability for Computer Scientists	5
MATH 19 & MATH 20 & MATH 21	Calculus and Calculus and Calculus [1]	3, 3, 4
Plus two electives [2]		

Science (11 units minimum)—

PHYSICS 41	Mechanics	4
PHYSICS 43	Electricity and Magnetism	4
Science elective [3]		3

Technology in Society (3-5 units)—

One course; course chosen must be on the SoE Approved Courses list at <ughb.stanford.edu> the year taken; see Basic Requirements 4 in the School of Engineering section

Engineering Fundamentals (13 units minimum; see Basic Requirement 3 in the School of Engineering section)—

CS 106B	Programming Abstractions	5
or CS 106X	Programming Abstractions (Accelerated)	
ENGR 40A	Introductory Electronics (and ENGR 40B)	3
ENGR 40M	An Intro to Making: What is EE	3-5
Fundamentals Elective (may not be 70A, B, or X)		3-5

*Students who take ENGR 40A without also taking ENGR 40B or 40M for fewer than 5 units are required to take 1-2 additional units of ENGR Fundamentals (13 units minimum), or 1-2 additional units of Depth (27 units minimum for track and elective courses).

Writing in the Major—

Select one of the following:

CS 181W	Computers, Ethics, and Public Policy
CS 191W	Writing Intensive Senior Project
CS 194W	Software Project
CS 210B	Software Project Experience with Corporate Partners
CS 294W	Writing Intensive Research Project in Computer Science

Computer Science Core (15 units)—

CS 107	Computer Organization and Systems	5
or CS 107E	Computer Systems from the Ground Up	
CS 110	Principles of Computer Systems	5
CS 161	Design and Analysis of Algorithms	5

Senior Capstone Project (3 units minimum)

CS 191	Senior Project [6]
CS 191W	Writing Intensive Senior Project [6]
CS 194	Software Project
CS 194H	User Interface Design Project
CS 194W	Software Project
CS 210B	Software Project Experience with Corporate Partners
CS 294W	Writing Intensive Research Project in Computer Science

Senior Project (3 units)—

CS 191	Senior Project
CS 191W	Writing Intensive Senior Project
CS 194	Software Project
CS 194H	User Interface Design Project
CS 194W	Software Project
CS 210B	Software Project Experience with Corporate Partners
CS 294 or CS 294W	[6] Writing Intensive Research Project in Computer Science

Computer Science Depth B.S.

Choose one of the following ten CS degree tracks (a track must consist of at least 25 units and 7 classes):

Artificial Intelligence Track—

		Units
CS 221	Artificial Intelligence: Principles and Techniques	4
Select two of the following:		6-8
CS 223A	Introduction to Robotics	
CS 224N	Natural Language Processing with Deep Learning	
CS 228	Probabilistic Graphical Models: Principles and Techniques	
CS 229	Machine Learning	
CS 131	Computer Vision: Foundations and Applications	
or CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
One additional course from the list above or the following:		3-4
CS 124	From Languages to Information	
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	
CS 224S	Spoken Language Processing	
CS 224U	Natural Language Understanding	
CS 224W	Social and Information Network Analysis	
CS 225A	Experimental Robotics	

CS 227B	General Game Playing	
CS 231A	Computer Vision: From 3D Reconstruction to Recognition (If not taken for track requirement B)	
CS 231B		
CS 231M		
CS 231N	Convolutional Neural Networks for Visual Recognition	
CS 262		
CS 276	Information Retrieval and Web Search	
CS 277		
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	
CS 327A	Advanced Robotic Manipulation	
CS 329	Topics in Artificial Intelligence	
CS 331A		
CS 371	Computational Biology in Four Dimensions	
CS 374		
CS 379	Interdisciplinary Topics (with adviser consent)	
EE 263	Introduction to Linear Dynamical Systems	
EE 376A	Information Theory	
ENGR 205	Introduction to Control Design Techniques	
ENGR 209A	Analysis and Control of Nonlinear Systems	
MS&E 251	Stochastic Control	
MS&E 351	Dynamic Programming and Stochastic Control	
STATS 315A	Modern Applied Statistics: Learning	
STATS 315B	Modern Applied Statistics: Data Mining	
Track Electives (at least three additional courses from the above lists, the general CS electives list, or the following): [4]		9-13
CS 224E		
CS 238	Decision Making under Uncertainty	
CS 275	Translational Bioinformatics	
CS 334A	Convex Optimization I	
or EE 364A	Convex Optimization I	
EE 278	Introduction to Statistical Signal Processing	
EE 364B	Convex Optimization II	
ECON 286	Game Theory and Economic Applications	
MS&E 252	Decision Analysis I: Foundations of Decision Analysis	
MS&E 352	Decision Analysis II: Professional Decision Analysis	
MS&E 355	Influence Diagrams and Probabilistics Networks	
PHIL 152	Computability and Logic	
PSYCH 202	Cognitive Neuroscience	
PSYCH 204A	Human Neuroimaging Methods	
PSYCH 204B	Human Neuroimaging Methods	
STATS 200	Introduction to Statistical Inference	
STATS 202	Data Mining and Analysis	
STATS 205	Introduction to Nonparametric Statistics	

Biocomputation Track—

		Units
The Mathematics, Science, and Engineering Fundamentals requirements are non-standard for this track. See Handbook for Undergraduate Engineering Programs for details.		
Select one of the following:		3-4
CS 221	Artificial Intelligence: Principles and Techniques	
CS 228	Probabilistic Graphical Models: Principles and Techniques	
CS 229	Machine Learning	
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
Select one of the following:		
CS 262		
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	
CS 273A	A Computational Tour of the Human Genome	
CS 274	Representations and Algorithms for Computational Molecular Biology	
CS 275	Translational Bioinformatics	
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	
One additional course from the lists above or the following:		3-4
CS 124	From Languages to Information	
CS 145	Introduction to Databases	
CS 147	Introduction to Human-Computer Interaction Design	
CS 148	Introduction to Computer Graphics and Imaging	
CS 248	Interactive Computer Graphics	
One course selected from either the Biomedical Computation (BMC) 'Informatics' electives list (go to http://bmc.stanford.edu and select Informatics from the elective options), BIOE 101, or from the general CS electives list [4]		3-4
One course from the BMC Informatics elective list or CS 273B		3-4
One course from either the BMC Informatics, Cellular/Molecular, or Organs/Organisms electives lists		3-5
One course from either the BMC Cellular/Molecular or Organs/Organisms electives lists		3-5

Computer Engineering Track—

		Units
EE 108 & EE 180	Digital System Design and Digital Systems Architecture	6-8
Select two of the following:		8
EE 101A	Circuits I	
EE 101B	Circuits II	
EE 102A	Signal Processing and Linear Systems I	
EE 102B	Signal Processing and Linear Systems II	
Satisfy the requirements of one of the following concentrations:		
1) Digital Systems Concentration		
CS 140	Operating Systems and Systems Programming	
or CS 143	Compilers	
EE 109	Digital Systems Design Lab	
EE 271	Introduction to VLSI Systems	
Plus two of the following (6-8 units):		
CS 140	Operating Systems and Systems Programming (if not counted above)	
or CS 143	Compilers	
CS 144	Introduction to Computer Networking	
CS 149	Parallel Computing	
CS 240E		
CS 244	Advanced Topics in Networking	
EE 273	Digital Systems Engineering	
EE 282	Computer Systems Architecture	
2) Robotics and Mechatronics Concentration		
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	

CS 223A	Introduction to Robotics	
ME 210	Introduction to Mechatronics	
ENGR 105	Feedback Control Design	
Plus one of the following (3-4 units):		
CS 225A	Experimental Robotics	
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
ENGR 205	Introduction to Control Design Techniques	
ENGR 207A	Linear Control Systems I	
ENGR 207B	Linear Control Systems II	
3) Networking Concentration		
CS 140 & CS 144	Operating Systems and Systems Programming and Introduction to Computer Networking	
Plus three of the following (9-11 units):		
CS 240	Advanced Topics in Operating Systems	
CS 240E		
CS 241	Embedded Systems Workshop	
CS 244	Advanced Topics in Networking	
CS 244B		
CS 244E		
CS 249A		
EE 179	Analog and Digital Communication Systems	

Graphics Track—

		Units
CS 148 & CS 248	Introduction to Computer Graphics and Imaging and Interactive Computer Graphics	8
Select one of the following: [5]		3-5
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics (strongly recommended as a preferred choice)	
CME 104	Linear Algebra and Partial Differential Equations for Engineers (Note: students taking CME 104 are also required to take its prerequisite course, CME 102)	
CME 108	Introduction to Scientific Computing	
MATH 52	Integral Calculus of Several Variables	
MATH 113	Linear Algebra and Matrix Theory	
Select two of the following:		6-8
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
or CS 131	Computer Vision: Foundations and Applications	
CS 233	Geometric and Topological Data Analysis	
CS 268	Geometric Algorithms	
CS 348A	Computer Graphics: Geometric Modeling & Processing	
CS 348B	Computer Graphics: Image Synthesis Techniques	
CS 348C	Computer Graphics: Animation and Simulation	
CS 448	Topics in Computer Graphics	
Track Electives: at least two additional courses from the lists above, the general CS electives list, or the following: [4]		6-8
ARTSTUDI 160	Intro to Digital / Physical Design	
ARTSTUDI 170	PHOTOGRAPHY I: BLACK AND WHITE	
ARTSTUDI 179	Digital Art I	
CME 302	Numerical Linear Algebra	
CME 306	Numerical Solution of Partial Differential Equations	
EE 168	Introduction to Digital Image Processing	
EE 262	Two-Dimensional Imaging	
EE 264	Digital Signal Processing	
EE 278	Introduction to Statistical Signal Processing	
EE 368	Digital Image Processing	
ME 101	Visual Thinking	
PSYCH 30	Introduction to Perception	
PSYCH 221	Image Systems Engineering	

Human-Computer Interaction Track—

		Units
CS 147	Introduction to Human-Computer Interaction Design	4
CS 247	Human-Computer Interaction Design Studio	4
Any three of the following:		
CS 142	Web Applications	
CS 148	Introduction to Computer Graphics and Imaging	
CS 194H	User Interface Design Project	
CS 210A	Software Project Experience with Corporate Partners	
CS 376	Human-Computer Interaction Research	
Any CS 377 'Topics in HCI' of three or more units		
CS 448B	Data Visualization	
ME 216M	Introduction to the Design of Smart Products	
At least two additional courses from above list, the general CS electives list, or the following: [4]		3-6
any d.school class of 3+ units; any class of 3+ units at hci.stanford.edu under the 'courses' link		
Communication-		
COMM 121	Behavior and Social Media	
COMM 124	Lies, Trust, and Tech	
or COMM 224	Lies, Trust, and Tech	
COMM 140		
or COMM 240		
COMM 166	Virtual People	
COMM 169		
or COMM 269		
COMM 172	Media Psychology	
or COMM 272	Media Psychology	
COMM 182		
COMM 324	Language and Technology	
Art Studio-		
ARTSTUDI 160	Intro to Digital / Physical Design	
ARTSTUDI 162	Embodied Interfaces	
ARTSTUDI 163	Drawing with Code	
ARTSTUDI 164	DESIGN IN PUBLIC SPACES	
ARTSTUDI 165	Social Media and Performative Practices	
ARTSTUDI 168	Data as Material	
ARTSTUDI 264	Advanced Interaction Design	
ARTSTUDI 266	Sculptural Screens / Malleable Media	
ARTSTUDI 267	Emerging Technology Studio	
Sym Sys-		
SYMSYS 245	Cognition in Interaction Design	
Psychology-		
PSYCH 30	Introduction to Perception	
PSYCH 45	Introduction to Learning and Memory	
PSYCH 70	Self and Society: Introduction to Social Psychology	
PSYCH 75	Introduction to Cultural Psychology	

PSYCH 110	Research Methods and Experimental Design
PSYCH 131	Language and Thought
PSYCH 154	Judgment and Decision-Making
Empirical Methods-	
MS&E 125	Introduction to Applied Statistics
PSYCH 252	Statistical Methods for Behavioral and Social Sciences
PSYCH 254	Lab in Experimental Methods
PSYCH 110	Research Methods and Experimental Design
STATS 203	Introduction to Regression Models and Analysis of Variance
EDUC 191	Introduction to Survey Research
HUMBIO 82A	Qualitative Research Methodology
ME Design-	
ME 101	Visual Thinking
ME 115A	Introduction to Human Values in Design
ME 203	Design and Manufacturing
ME 210	Introduction to Mechatronics
ME 216A	Advanced Product Design: Needfinding
Learning Design + Tech-	
EDUC 236	Beyond Bits and Atoms: Designing Technological Tools
EDUC 281	Technology for Learners
EDUC 239	Educating Young STEM Thinkers
EDUC 338	Innovations in Education
EDUC 342	Child Development and New Technologies
MS&E-	
MS&E 185	Global Work
MS&E 331	
Computer Music-	
MUSIC 220A	Fundamentals of Computer-Generated Sound
MUSIC 220B	Compositional Algorithms, Psychoacoustics, and Computational Music
MUSIC 220C	Research Seminar in Computer-Generated Music
MUSIC 250A	Physical Interaction Design for Music
MUSIC 256A	Music, Computing, Design I: Art of Design for Computer Music
Optional Elective [4]	

Information Track—

		Units
CS 124	From Languages to Information	4
CS 145	Introduction to Databases	4
Two courses, from different areas:		6-9
1) Information-based AI applications		
CS 224N	Natural Language Processing with Deep Learning	
CS 224S	Spoken Language Processing	
CS 229	Machine Learning	
CS 233	Geometric and Topological Data Analysis	
2) Database and Information Systems		
CS 140	Operating Systems and Systems Programming	
CS 142	Web Applications	
CS 245	Database Systems Principles	
CS 246	Mining Massive Data Sets	
CS 341	Project in Mining Massive Data Sets	
CS 345	(Offered occasionally)	
CS 346		
CS 347		
3) Information Systems in Biology		
CS 262		
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	
CS 274	Representations and Algorithms for Computational Molecular Biology	
4) Information Systems on the Web		
CS 224W	Social and Information Network Analysis	
CS 276	Information Retrieval and Web Search	
At least three additional courses from the above areas or the general CS electives list. [4]		

Systems Track—

		Units
CS 140	Operating Systems and Systems Programming	4
Select one of the following:		3-4
CS 143	Compilers	
EE 180	Digital Systems Architecture	
Two additional courses from the list above or the following:		6-8
CS 144	Introduction to Computer Networking	
CS 145	Introduction to Databases	
CS 149	Parallel Computing	
CS 155	Computer and Network Security	
CS 240	Advanced Topics in Operating Systems	
CS 242	Programming Languages	
CS 243	Program Analysis and Optimizations	
CS 244	Advanced Topics in Networking	
CS 245	Database Systems Principles	
EE 271	Introduction to VLSI Systems	
EE 282	Computer Systems Architecture	
Track Electives: at least three additional courses selected from the list above, the general CS electives list, or the following: [4]		9-12
CS 240E		
CS 241	Embedded Systems Workshop	
CS 244E		
CS 316	Advanced Multi-Core Systems	
CS 341	Project in Mining Massive Data Sets	
CS 343	(Not given this year)	
CS 344	Topics in Computer Networks (3 or more units, any suffix)	
CS 345	(Advanced Topics in Database Systems - 3 or more units, any suffix)	
CS 346		
CS 347		
CS 349	Topics in Programming Systems (with permission of undergraduate advisor)	
CS 448	Topics in Computer Graphics (3 or more units, any suffix)	
EE 108	Digital System Design	
EE 382C	Interconnection Networks	
EE 384A	Internet Routing Protocols and Standards	
EE 384B	Multimedia Communication over the Internet	
EE 384C	Wireless Local and Wide Area Networks	
EE 384S	Performance Engineering of Computer Systems & Networks	
EE 384X	Packet Switch Architectures	

Theory Track—

		Units
CS 154	Introduction to Automata and Complexity Theory	4
Select one of the following:		3
CS 167		
CS 168	The Modern Algorithmic Toolbox	
CS 255	Introduction to Cryptography	
CS 261	Optimization and Algorithmic Paradigms	
CS 264	Beyond Worst-Case Analysis	
CS 265	Randomized Algorithms and Probabilistic Analysis	
CS 268	Geometric Algorithms	
Two additional courses from the list above or the following:		6-8
CS 143	Compilers	
CS 155	Computer and Network Security	
CS 157	Logic and Automated Reasoning	
or PHIL 151	Metalogic	
CS 166		
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	
CS 228	Probabilistic Graphical Models: Principles and Techniques	
CS 233	Geometric and Topological Data Analysis	
CS 242	Programming Languages	
CS 250	Error Correcting Codes: Theory and Applications	
CS 251	Bitcoin and Crypto Currencies	
CS 254	Computational Complexity	
CS 259	(With adviser consent, offered occasionally)	
CS 262		
CS 263	Algorithms for Modern Data Models	
CS 266		
CS 267	Graph Algorithms	
CS 354	(Not given this year)	
CS 355	(Not given this year)	
CS 357	(Not given this year)	
CS 358	Topics in Programming Language Theory	
CS 359	Topics in the Theory of Computation (with adviser consent)	
CS 364A		
CS 367	(Not given this year)	
CS 369	Topics in Analysis of Algorithms (with adviser consent)	
CS 374		
MS&E 310	Linear Programming	
Track Electives: at least three additional courses from the list above, the general CS electives list, or the following: [4]		9-12
CME 302	Numerical Linear Algebra	
CME 305	Discrete Mathematics and Algorithms	
PHIL 152	Computability and Logic	

Unspecialized Track—

		Units
CS 154	Introduction to Automata and Complexity Theory	4
Select one of the following:		4
CS 140	Operating Systems and Systems Programming	
CS 143	Compilers	
One additional course from the list above or the following:		3-4
CS 144	Introduction to Computer Networking	
CS 155	Computer and Network Security	
CS 242	Programming Languages	
CS 244	Advanced Topics in Networking	
EE 180	Digital Systems Architecture	
Select one of the following:		3-4
CS 221	Artificial Intelligence: Principles and Techniques	
CS 223A	Introduction to Robotics	
CS 228	Probabilistic Graphical Models: Principles and Techniques	
CS 229	Machine Learning	
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
Select one of the following:		3-4
CS 145	Introduction to Databases	
CS 147	Introduction to Human-Computer Interaction Design	
CS 148	Introduction to Computer Graphics and Imaging	
CS 248	Interactive Computer Graphics	
CS 262		
At least two courses from the general CS electives list [4]		

Individually Designed Track—

Students may propose an individually designed track. Proposals should include a minimum of seven courses, at least four of which must be CS courses numbered 100 or above. See Handbook for Undergraduate Engineering Programs for further information.

For additional information and sample programs see the Handbook for Undergraduate Engineering Programs (UGHB) (http://ughb.stanford.edu)

1 MATH 19, MATH 20, and MATH 21 may be taken instead of MATH 41 and MATH 42 as long as at least 26 MATH units are taken. AP Calculus must be approved by the School of Engineering.

2 The math electives list consists of: MATH 51, MATH 104, MATH 108, MATH 109, MATH 110, MATH 113; CS 157, CS 205A; PHIL 151; CME 100, CME 102, CME 104. Completion of MATH 52 and MATH 53 counts as one math elective. Restrictions: CS 157 and PHIL 151 may not be used in combination to satisfy the math electives requirement. Students who have taken both MATH 51 and MATH 52 may not count CME 100 as an elective. Courses counted as math electives cannot also count as CS electives, and vice versa.

3 The science elective may be any course of 3 or more units from the School of Engineering Science list plus PSYCH 30; AP Chemistry may be used to meet this requirement. Either of the PHYSICS sequences 61/63 or 21/23 may be substituted for 41/43 as long as at least 11 science units are taken. AP Physics must be approved by the School of Engineering.

4 General CS Electives: CS 108,CS 124, CS 131, CS 140, CS 142, CS 143 CS 144, CS 145, CS 147, CS 148, CS 149, CS 154, CS 155, CS 157(or PHIL 151), CS 164, CS 166, CS 167, CS 168, CS 190, CS 205A, CS 205B, CS 210A, CS 221, CS 223A, CS 224N, CS 224S, CS 224U, CS 224W, CS 225A, CS 227B, CS 228, CS 229, CS 229T, CS 231A, CS 231B, CS 231M, CS 231N, CS 232, CS 233, CS 240, CS 240H, CS 242, CS 243, CS 244, CS 244B, CS 245, CS 246, CS 247, CS 248, CS 249A, CS 251, CS 254, CS 255, CS 261, CS 262, CS 263, CS 264, CS 265, CS 266, CS 267, CS 270, CS 272, CS 273A, CS 273B, CS 274, CS 276, CS 279, CS 348B, CS 348C; CME 108; EE 180, EE 282, EE 364A.

5 CS 205A Mathematical Methods for Robotics, Vision, and Graphics is recommended in this list for the Graphics track. Students taking CME 104 Linear Algebra and Partial Differential Equations for Engineers are also required to take its prerequisite, CME 102 Ordinary Differential Equations for Engineers.

[6] Independent study projects (CS 191 Senior Projector CS 191W Writing Intensive Senior Project) require faculty sponsorship and must be approved by the adviser, faculty sponsor, and the CS senior project adviser (P. Young). A signed approval form, along with a brief description of the proposed project, should be filed the quarter before work on the project is begun. Further details can be found in the *Handbook for Undergraduate Engineering Programs*.

Honors Program

The Department of Computer Science (CS) offers an honors program for undergraduates whose academic records and personal initiative indicate that they have the necessary skills to undertake high-quality research in computer science. Admission to the program is by application only. To apply for the honors program, students must be majoring in Computer Science, have a grade point average (GPA) of at least 3.6 in courses that count toward the major, and achieve senior standing (135 or more units) by the end of the academic year in which they apply. Coterminal master's students are eligible to apply as long as they have not already received their undergraduate degree. Beyond these requirements, students who apply for the honors program must 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 CS 191W under the direction of their project sponsor.
2. Attend a weekly honors seminar Winter and Spring quarters.
3. Complete an honors thesis deemed acceptable by the thesis adviser and at least one additional faculty member.
4. Present the thesis at a public colloquium sponsored by the department.
5. Maintain the 3.6 GPA required for admission to the honors program.

Guide to Choosing Introductory Courses

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

For students whose principal interest is an exposure to the fundamental ideas behind computer science and programming, CS 101 or CS 105 are the most appropriate courses. They are 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 101 and CS 105 meet the new Ways of Thinking Ways of Doing breadth requirements in Formal Reasoning and include 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 CS 106A followed by CS 106B. Students in CS 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 CS 106X or may start directly in CS 106B. CS 106A uses Java as its programming language; CS 106B and X use C++. No prior knowledge of these languages is assumed, and the prior programming experience required for CS 106B or X may be in any language. In all cases, students are encouraged to discuss their background with the instructors responsible for these courses.

After the introductory sequence, Computer Science majors and those who need a significant background in computer science for related majors in engineering should take CS 103, CS 107 and CS 110. CS 103 offers an introduction to the mathematical and theoretical foundations of computer science. CS 107 exposes students to a variety of programming concepts that illustrate critical strategies used in systems development; CS 110 builds on this material, focusing on the development of larger-scale software making use of systems and networking abstractions.

In summary:

For exposure:	
CS 1C	Introduction to Computing at Stanford
For nontechnical use:	
CS 101	Introduction to Computing Principles
or CS 105	Introduction to Computers
For scientific use:	
CS 106A	Programming Methodology
For a technical introduction:	
CS 106A	Programming Methodology
For significant use:	
CS 106A & CS 106B	Programming Methodology and Programming Abstractions
or CS 106X	Programming Abstractions (Accelerated)
CS 103	Mathematical Foundations of Computing
CS 107	Computer Organization and Systems
CS 110	Principles of Computer Systems

Overseas Studies Courses in Computer Science

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

Joint Major Program: Computer Science and a Humanities Major

The joint major program (JMP), authorized by the Academic Senate for a pilot period of six years beginning in 2014-15, permits students to major in both Computer Science and one of ten Humanities majors. See the "Joint Major Program (http://exploredegrees.stanford.edu/undergraduatedegreesandprograms/#jointmajortext)" section of this bulletin for a description of University requirements for the JMP. See also the Undergraduate Advising and Research JMP web site and its associated FAQs.

Students completing the JMP receive a B.A.S. (Bachelor of Arts and Science).

Because the JMP is new and experimental, changes to procedures may occur; students are advised to check the relevant section of the bulletin periodically.

Mission

The Joint Major provides a unique opportunity to gain mastery in two disciplines: Computer Science and a selected humanities field. Unlike the double major or dual major, the Joint Major emphasizes integration of the two fields through a cohesive, transdisciplinary course of study and integrated capstone experience. The Joint Major not only blends the intellectual traditions of two Stanford departments-it does so in a way that reduces the total unit requirement for each major.

Computer Science Major Requirements in the Joint Major Program

(See the respective humanities department Joint Major Program section of this bulletin for details on humanities major requirements.)

The CS requirements for the Joint Major follow the CS requirements for the CS-BS degree with the following exceptions:

1. Two of the depth electives are waived. The waived depth electives are listed below for each CS track.
2. The Senior Project is fulfilled with a joint capstone project. The student enrolls in CS191 or 191W (3 units) during the senior year. Depending on the X department, enrollment in an additional Humanities capstone course may also be required. But, at a minimum, 3 units of CS191 or 191W must be completed.
3. There is no double-counting of units between majors. If a course is required for both the CS and Humanities majors, the student will work with one of the departments to identify an additional course - one which will benefit the academic plan - to apply to that major's total units requirement.
4. For CS, WIM can be satisfied with CS181W or CS191W.

Depth Electives for CS Tracks for students completing a Joint Major:

Artificial Intelligence Track:
One Track Elective (rather than three).

Biocomputation Track:
One course from Note 3 of the Department Program Sheet, plus one course from Note 4 of the Program Sheet..

Computer Engineering Track:
- EE 108A and 108B
- One of the following: EE 101A, 101B, 102A, 102B
- Satisfy the requirements of one of the following concentrations:

    1. Digital Systems Concentration: CS 140 or 143; EE 109, 271; plus one of CS 140 or 143 (if not counted above), 144, 149, 240E, 244: EE 273, 282
    2. Robotics and Mechatronics Concentration: CS 205A, 223A; ME 210; ENGR 105
    3. Networking Concentration: CS 140, 144; plus two of the following, CS 240, 240E, 244, 244B, 244E, 249A, 249B, EE 179, EE 276

Graphics Track:
No Track Electives required (rather than two)

HCI Track:
No Interdisciplinary HCI Electives required

Information Track:

One Track Elective (rather than three)

Systems Track:
One Track Elective (rather than three)

Theory Track:
One Track Elective (rather than three)

Unspecialized Track:
No Track Electives required (rather than two)

Individually Designed Track:
Proposals should include a minimum of five (rather than seven) courses, at least four of which must be CS courses numbered 100 or above.

Declaring a Joint Major Program

To declare the joint major, students must first declare each major through Axess, and then submit the Declaration or Change of Undergraduate Major, Minor, Honors, or Degree Program. (https://stanford.box.com/change-UG-program) The Major-Minor and Multiple Major Course Approval Form (https://stanford.box.com/MajMin-MultMaj) is required for graduation for students with a joint major.

Dropping a Joint Major Program

To drop the joint major, students must submit the Declaration or Change of Undergraduate Major, Minor, Honors, or Degree Program. (https://stanford.box.com/change-UG-program) . Students may also consult the Student Services Center (http://studentservicescenter.stanford.edu) with questions concerning dropping the joint major.

Transcript and Diploma

Students completing a joint major graduate with a B.A.S. degree. The two majors are identified on one diploma separated by a hyphen. There will be a notation indicating that the student has completed a "Joint Major". The two majors are identified on the transcript with a notation indicating that the student has completed a "Joint Major".

Computer Science (CS) Minor

The following core courses fulfill the minor requirements. Prerequisites include the standard mathematics sequence through MATH 51.

		Units
Introductory Programming (AP Credit may be used to fulfill this requirement):		
CS 106B	Programming Abstractions	5
or CS 106X	Programming Abstractions (Accelerated)	
Core:		
CS 103	Mathematical Foundations of Computing	5
CS 107	Computer Organization and Systems	5
or CS 107E	Computer Systems from the Ground Up	
CS 109	Introduction to Probability for Computer Scientists	5
Electives (choose two courses from different areas):		
Artificial Intelligence—		
CS 124	From Languages to Information	4
CS 221	Artificial Intelligence: Principles and Techniques	4
CS 229	Machine Learning	3-4
Human-Computer Interaction—		
CS 147	Introduction to Human-Computer Interaction Design	4
Software—		
CS 108	Object-Oriented Systems Design	4

CS 110	Principles of Computer Systems	5
Systems—		
CS 140	Operating Systems and Systems Programming	4
CS 143	Compilers	4
CS 144	Introduction to Computer Networking	4
CS 145	Introduction to Databases	4
CS 148	Introduction to Computer Graphics and Imaging	4
Theory—		
CS 154	Introduction to Automata and Complexity Theory	4
CS 157	Logic and Automated Reasoning	3
CS 161	Design and Analysis of Algorithms	5

Note: for students with no programming background and who begin with CS 106A, the minor consists of seven courses.

Master of Science in Computer Science

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

Admission

Applications to the M.S. program and all supporting documents must be submitted and received online by the published deadline. Information on admission requirements (http://cs.stanford.edu/admissions) is available on the department's web site; see also the department's deadlines page (https://cs.stanford.edu/admissions/deadlines). Exceptions are made for applicants who are already students at Stanford and are applying to the coterminal program (https://cs.stanford.edu/admissions/current-stanford-students/coterminal-program).

University Coterminal Requirements

Coterminal master's degree candidates are expected to complete all master's degree requirements as described in this bulletin. University requirements for the coterminal master's degree are described in the "Coterminal Master's Program (http://exploredegrees.stanford.edu/cotermdegrees)" section. University requirements for the master's degree are described in the "Graduate Degrees (http://exploredegrees.stanford.edu/graduatedegrees/#masterstext)" section of this bulletin.

After accepting admission to this coterminal master's degree program, students may request transfer of courses from the undergraduate to the graduate career to satisfy requirements for the master's degree. Transfer of courses to the graduate career requires review and approval of both the undergraduate and graduate programs on a case by case basis.

In this master's program, courses taken during or after the first quarter of the sophomore year are eligible for consideration for transfer to the graduate career; the timing of the first graduate quarter is not a factor. No courses taken prior to the first quarter of the sophomore year may be used to meet master's degree requirements.

Course transfers are not possible after the bachelor's degree has been conferred.

The University requires that the graduate adviser be assigned in the student's first graduate quarter even though the undergraduate career may still be open. The University also requires that the Master's Degree Program Proposal be completed by the student and approved by the department by the end of the student's first graduate quarter.

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 10 units of courses from those listed below in Requirement 1. Thus, students needing to take more than two of the courses listed in Requirement 1 actually complete more than 45 units of course work in the program. Only well-prepared students may expect to finish the program in one year; most students 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 prior course work related to their specialization area.

Requirement 1: Foundations—

Students must complete the following courses, or waive out of them by providing evidence to their advisers that similar or more advanced courses have been taken, either at Stanford or another institution (total units used to satisfy foundations requirement may not exceed 10):

Logic, Automata, and Computability	
CS 103	Mathematical Foundations of Computing
Probability	
Select one of the following:	
CS 109	Introduction to Probability for Computer Scientists
STATS 116	Theory of Probability
MS&E 220	Probabilistic Analysis
CME 106	Introduction to Probability and Statistics for Engineers
Algorithms	
CS 161	Design and Analysis of Algorithms
Computer Organization and Systems	
CS 107	Computer Organization and Systems
or CS 107E	Computer Systems from the Ground Up
Principles of Computer Systems	
CS 110	Principles of Computer Systems

Requirement 2: Significant Software Implementation—

Students must complete at least one course designated as having a significant software implementation component. The list of such courses includes:

CS 140	Operating Systems and Systems Programming	3-4
CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
CS 145	Introduction to Databases	3-4
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 210B	Software Project Experience with Corporate Partners	3-4
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 227B	General Game Playing	3
CS 243	Program Analysis and Optimizations	3-4
CS 248	Interactive Computer Graphics	3-4
CS 341	Project in Mining Massive Data Sets	3
CS 346	(Offered occasionally)	3-5

Requirement 3: Specialization—

Students may choose to satisfy this requirement through one of two options, Single Depth or Dual Depth, outlined following. All courses taken for this requirement must be taken on a letter grade basis for three or more units.

Option 1—Single Depth

- A program of 27 units in a single area of specialization must be completed. A maximum of 9 units of independent study (CS 393, CS 395, CS 399) may be counted toward the specialization.
- Additionally, students must complete three breadth courses from the list of approved breadth courses associated with their chosen specialization. Individual specializations explicitly have different breadth requirements; see the individual specialization sheets at http://cs.stanford.edu/degrees/mscs/programsheets for details.
- Breadth courses may not be waived, must be taken for at least 3 units each, and must be completed for a letter grade.

Option 2—Dual Depth

- Students select distinct primary and secondary areas.
- A program of 21 units in the primary area of specialization must be completed. A maximum of 9 units of independent study (CS 393, CS 395, CS 399) may be counted toward the primary specialization.
- Students must also complete a program of five courses satisfying the requirements for their secondary area of specialization.
- Breadth courses are not required.

Specialization Areas—

Ten approved specialization areas which may be used to satisfy Requirement 3 are listed following. Students may propose to the M.S. program committee other coherent programs that meet their goals and satisfy the basic requirements.

Courses marked with an asterisk (*) require consent of the faculty adviser. Courses marked with a double asterisk (**) may be waived by students with equivalent course work and with the approval of their adviser.

1. Artificial Intelligence—

A.

CS 221	Artificial Intelligence: Principles and Techniques **

B. Select at least four of the following:

CS 223A	Introduction to Robotics
CS 224N	Natural Language Processing with Deep Learning
CS 224S	Spoken Language Processing
CS 224U	Natural Language Understanding
CS 224W	Social and Information Network Analysis
CS 228	Probabilistic Graphical Models: Principles and Techniques
CS 229	Machine Learning
CS 231A	Computer Vision: From 3D Reconstruction to Recognition

C. Sufficient depth units from category (b) and the following:

CS 205A	Mathematical Methods for Robotics, Vision, and Graphics
CS 225A	Experimental Robotics
CS 225B	Robot Programming Laboratory
CS 227B	General Game Playing
CS 229A	(Not given this year)
CS 229T	Statistical Learning Theory
CS 231B	
CS 231M	
CS 231N	Convolutional Neural Networks for Visual Recognition
CS 232	Digital Image Processing
CS 233	Geometric and Topological Data Analysis
CS 238	Decision Making under Uncertainty
CS 239	Advanced Topics in Sequential Decision Making
CS 246	Mining Massive Data Sets
CS 262	
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 273A	A Computational Tour of the Human Genome
CS 273B	Deep Learning in Genomics and Biomedicine
CS 274	Representations and Algorithms for Computational Molecular Biology
CS 275	Translational Bioinformatics
CS 276	Information Retrieval and Web Search
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells
CS 294A	Research Project in Artificial Intelligence *
CS 323	Automated Reasoning: Theory and Applications
CS 325	
CS 327A	Advanced Robotic Manipulation (Not given this year)
CS 328	Topics in Computer Vision
CS 329	Topics in Artificial Intelligence
CS 331A	
CS 331B	Representation Learning in Computer Vision
CS 334A	Convex Optimization I
or EE 364A	Convex Optimization I
CS 341	Project in Mining Massive Data Sets
CS 345	(Offered occasionally)
CS 362	(Not given this year)
CS 364A	
CS 371	Computational Biology in Four Dimensions
CS 373	Statistical and Machine Learning Methods for Genomics
CS 374	(not given this year)
CS 377	Topics in Human-Computer Interaction (CS 377 with any suffix) *
CS 379	Interdisciplinary Topics (CS 379 with any suffix) *
CS 393	Computer Laboratory *
CS 395	Independent Database Project *
CS 399	Independent Project *
EE 263	Introduction to Linear Dynamical Systems
EE 278	Introduction to Statistical Signal Processing
EE 364B	Convex Optimization II
EE 376A	Information Theory
EE 377	Information Theory and Statistics
EE 378B	Inference, Estimation, and Information Processing
ENGR 205	Introduction to Control Design Techniques
ENGR 209A	Analysis and Control of Nonlinear Systems
MS&E 226	"Small" Data
MS&E 251	Stochastic Control
MS&E 252	Decision Analysis I: Foundations of Decision Analysis
MS&E 351	Dynamic Programming and Stochastic Control
MS&E 352	Decision Analysis II: Professional Decision Analysis
MS&E 353	Decision Analysis III: Frontiers of Decision Analysis
PSYCH 202	Cognitive Neuroscience
STATS 202	Data Mining and Analysis
STATS 315A	Modern Applied Statistics: Learning
STATS 315B	Modern Applied Statistics: Data Mining
BIOE 332	

APPPHYS 293	Theoretical Neuroscience

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Artificial Intelligence must take five total courses satisfying the area (a) and (b) requirements above.
- Those students who have waived out of CS 221 may take an additional course in either area (b) or (c).

Artificial Intelligence Breadth Courses
Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 140	Operating Systems and Systems Programming	3-4
CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
or EE 284	Introduction to Computer Networks	
CS 145	Introduction to Databases	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 240	Advanced Topics in Operating Systems	3
CS 240E		
CS 240H		3-4
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 244	Advanced Topics in Networking	3-4
CS 244B		3
CS 244E		
CS 249A		3
CS 255	Introduction to Cryptography	3
CS 261	Optimization and Algorithmic Paradigms	3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 268	Geometric Algorithms	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

2. Biocomputation—

A. Select at least four of the following:	
CS 262	
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 272	Introduction to Biomedical Informatics Research Methodology
CS 273A	A Computational Tour of the Human Genome
CS 274	Representations and Algorithms for Computational Molecular Biology
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells
B. Sufficient depth units from category (a) and the following:	
CS 228	Probabilistic Graphical Models: Principles and Techniques
CS 229	Machine Learning
CS 231N	Convolutional Neural Networks for Visual Recognition
CS 233	Geometric and Topological Data Analysis
CS 245	Database Systems Principles
CS 246	Mining Massive Data Sets
CS 261	Optimization and Algorithmic Paradigms
CS 264	Beyond Worst-Case Analysis
CS 265	Randomized Algorithms and Probabilistic Analysis
CS 268	Geometric Algorithms
CS 273B	Deep Learning in Genomics and Biomedicine
CS 275	Translational Bioinformatics
CS 325	
CS 341	Project in Mining Massive Data Sets
CS 345	(Offered occasionally)
CS 346	
CS 362	(Not given this year)
CS 371	Computational Biology in Four Dimensions
CS 373	Statistical and Machine Learning Methods for Genomics
CS 374	
CS 393	Computer Laboratory *
CS 395	Independent Database Project *
CS 399	Independent Project *
APPPHYS 293	Theoretical Neuroscience
BIOC 218	
BIOE 332	
GENE 203	
GENE 211	Genomics
SBIO 228	

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) and (b) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Biocomputation must take five total courses, three courses of which must come from area (a) and the remaining two courses may come from either area (a) or (b).

Biocomputation Breadth Courses
Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 140	Operating Systems and Systems Programming	3-4
CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
or EE 284	Introduction to Computer Networks	
CS 145	Introduction to Databases	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3

CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 224M		3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E		
CS 240H		3-4
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 244	Advanced Topics in Networking	3-4
CS 244B		3
CS 244E		
CS 249A		3
CS 255	Introduction to Cryptography	3
CS 276	Information Retrieval and Web Search	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

3. Computer and Network Security—

A.

CS 140	Operating Systems and Systems Programming [**]	
CS 144	Introduction to Computer Networking [**]	
CS 155	Computer and Network Security	
CS 244	Advanced Topics in Networking	
CS 255	Introduction to Cryptography	

B. Select at least three of the following:

CS 142	Web Applications	
CS 240	Advanced Topics in Operating Systems	
CS 244B		
CS 261	Optimization and Algorithmic Paradigms	
CS 265	Randomized Algorithms and Probabilistic Analysis	
CS 340	Topics in Computer Systems	
CS 344	Topics in Computer Networks (CS 344 with any suffix)	
CS 355	(Not given this year)	

C. Sufficient depth units from category (b) and the following:

CS 240E		
CS 241	Embedded Systems Workshop	
CS 244E		
CS 245	Database Systems Principles	
CS 251	Bitcoin and Crypto Currencies	
CS 264	Beyond Worst-Case Analysis	
CS 294S	Research Project in Software Systems and Security (Not given this year) [*]	
CS 341	Project in Mining Massive Data Sets	
CS 345	(Offered occasionally)	
CS 347		
CS 393	Computer Laboratory [*]	
CS 395	Independent Database Project [*]	
CS 399	Independent Project [*]	
EE 384A	Internet Routing Protocols and Standards	
EE 384C	Wireless Local and Wide Area Networks	
EE 384S	Performance Engineering of Computer Systems & Networks	

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Computer and Network Security must take five courses; those five courses must satisfy the area (a) requirement and additional courses from area (b) should be taken if any area (a) requirements are waived.

Computer and Network Security Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 143	Compilers	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 228	Probabilistic Graphical Models: Principles and Techniques	3-4
CS 229	Machine Learning	3-4
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 233	Geometric and Topological Data Analysis	3
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 246	Mining Massive Data Sets	3-4
CS 249A		3
CS 262		3
CS 268	Geometric Algorithms	3
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 276	Information Retrieval and Web Search	3

CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

4. Human-Computer Interaction—

A.

CS 147	Introduction to Human-Computer Interaction Design **	
CS 247	Human-Computer Interaction Design Studio **	

B. Select any three of the following:

CS 142	Web Applications	
CS 148	Introduction to Computer Graphics and Imaging	
CS 194H	User Interface Design Project	
CS 210A	Software Project Experience with Corporate Partners	
CS 248	Interactive Computer Graphics	
CS 376	Human-Computer Interaction Research	
CS 377	Topics in Human-Computer Interaction (CS 377 with any suffix)	
CS 448B	Data Visualization	
ME 216M	Introduction to the Design of Smart Products	

C. A total of at least 27 units from categories (a), (b), and the following:

a. Broader CS		
CS 221	Artificial Intelligence: Principles and Techniques	
CS 224N	Natural Language Processing with Deep Learning	
CS 224U	Natural Language Understanding	
CS 224W	Social and Information Network Analysis	
CS 229	Machine Learning	
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	
CS 231B		
CS 242	Programming Languages	
CS 246	Mining Massive Data Sets	
CS 341	Project in Mining Massive Data Sets	
CS 393	Computer Laboratory *	
CS 395	Independent Database Project *	
CS 399	Independent Project *	
b. Art Studio		
ARTSTUDI 160	Intro to Digital / Physical Design	
ARTSTUDI 162	Embodied Interfaces	
ARTSTUDI 163	Drawing with Code	
ARTSTUDI 164	DESIGN IN PUBLIC SPACES	
ARTSTUDI 165	Social Media and Performative Practices	
ARTSTUDI 168	Data as Material	
ARTSTUDI 264	Advanced Interaction Design	
ARTSTUDI 266	Sculptural Screens / Malleable Media	
ARTSTUDI 267	Emerging Technology Studio	
c. Communication		
COMM 224	Lies, Trust, and Tech	
COMM 240		
COMM 266	Virtual People	
COMM 269		
COMM 272	Media Psychology	
Comm 282		
COMM 324	Language and Technology	
d. Empirical Methods		
COMM 314	Ethnographic Methods	
EDUC 200B	Introduction to Qualitative Research Methods	
EDUC 291	Learning Sciences and Technology Design Research Seminar and Colloquium	
MS&E 125	Introduction to Applied Statistics	
PSYCH 252	Statistical Methods for Behavioral and Social Sciences	
PSYCH 254	Lab in Experimental Methods	
STATS 203	Introduction to Regression Models and Analysis of Variance	
e. Learning Design & Tech		
EDUC 236	Beyond Bits and Atoms: Designing Technological Tools	
EDUC 239	Educating Young STEM Thinkers	
EDUC 281	Technology for Learners	
EDUC 338	Innovations in Education	
EDUC 342	Child Development and New Technologies	
f. Man Sci & Engr		
MS&E 185	Global Work	
MS&E 331		
MS&E 334	The Structure of Social Data	
g. Mechanical Engr		
ME 203	Design and Manufacturing	
ME 210	Introduction to Mechatronics	
ME 216A	Advanced Product Design: Needfinding	
h. Music		
MUSIC 220A	Fundamentals of Computer-Generated Sound	
MUSIC 220B	Compositional Algorithms, Psychoacoustics, and Computational Music	
MUSIC 220C	Research Seminar in Computer-Generated Music	
MUSIC 250A	Physical Interaction Design for Music	
MUSIC 256A	Music, Computing, Design I: Art of Design for Computer Music	
i. Psych		
PSYCH 204	Computation and cognition: the probabilistic approach	
PSYCH 209	Neural Network Models of Cognition: Principles and Applications	
j. Sym Sys		
SYMSYS 245	Cognition in Interaction Design	
Any d.school class listed at http://dschool.stanford.edu, or any HCI class listed at http://hci.stanford.edu/courses/; such courses must be numbered 100 or above and be taken for at least 3 units to count for this requirement		

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) through (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Human-Computer Interaction must take five courses satisfying the areas (a) through (c).

Human-Computer Interaction Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 140	Operating Systems and Systems Programming	3-4

CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
or EE 284	Introduction to Computer Networks	
CS 145	Introduction to Databases	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 223A	Introduction to Robotics	3
CS 224S	Spoken Language Processing	2-4
CS 227B	General Game Playing	3
CS 228	Probabilistic Graphical Models: Principles and Techniques	3-4
CS 233	Geometric and Topological Data Analysis	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E		
CS 240H		3-4
CS 243	Program Analysis and Optimizations	3-4
CS 244	Advanced Topics in Networking	3-4
CS 244B		3
CS 244E		
CS 249A		3
CS 255	Introduction to Cryptography	3
CS 261	Optimization and Algorithmic Paradigms	3
CS 262		3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 268	Geometric Algorithms	3
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 276	Information Retrieval and Web Search	3
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

5. Information Management and Analytics—

A.

CS 145	Introduction to Databases [**]	3-4

B. Select at least four of the following:

CS 224N	Natural Language Processing with Deep Learning
CS 224W	Social and Information Network Analysis
CS 229	Machine Learning
CS 245	Database Systems Principles
CS 246	Mining Massive Data Sets
CS 276	Information Retrieval and Web Search
CS 345	(Offered occasionally)
CS 346	(no longer offered)
CS 347	

C. Sufficient depth units from category (b) and the following:

CS 144	Introduction to Computer Networking
CS 224S	Spoken Language Processing
CS 224U	Natural Language Understanding
CS 228	Probabilistic Graphical Models: Principles and Techniques
CS 229T	Statistical Learning Theory
CS 231A	Computer Vision: From 3D Reconstruction to Recognition
CS 231N	Convolutional Neural Networks for Visual Recognition
CS 233	Geometric and Topological Data Analysis
CS 240	Advanced Topics in Operating Systems
CS 242	Programming Languages
CS 243	Program Analysis and Optimizations
CS 244	Advanced Topics in Networking
CS 244B	
CS 249A	
CS 251	Bitcoin and Crypto Currencies
CS 255	Introduction to Cryptography
CS 262	
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 272	Introduction to Biomedical Informatics Research Methodology
CS 273A	A Computational Tour of the Human Genome
CS 274	Representations and Algorithms for Computational Molecular Biology
CS 275	Translational Bioinformatics
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells
CS 316	Advanced Multi-Core Systems
CS 325	
CS 341	Project in Mining Massive Data Sets
CS 344	Topics in Computer Networks (CS 344 with any suffix)
CS 362	(Not given this year)
CS 374	
CS 393	Computer Laboratory [*]
CS 395	Independent Database Project [*]
CS 399	Independent Project [*]
MS&E 226	"Small" Data
STATS 315A	Modern Applied Statistics: Learning
STATS 315B	Modern Applied Statistics: Data Mining

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Information Management and Analytics must take five courses satisfying the area (a) and (b) requirements above. Note that if CS145 was waived in area (a), students should take an additional course from either area (b) or (c) in its place.

Information Management and Analytics Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4

CS 140	Operating Systems and Systems Programming	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 227B	General Game Playing	3
CS 240E		
CS 244E		
CS 261	Optimization and Algorithmic Paradigms	3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 268	Geometric Algorithms	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

6. Mobile and Internet Computing—

A. Select two of the following:		
CS 140	Operating Systems and Systems Programming **	
CS 144	Introduction to Computer Networking	
CS 244	Advanced Topics in Networking	
B. Select one of the following:		
CS 142	Web Applications	
CS 147	Introduction to Human-Computer Interaction Design	
CS 247	Human-Computer Interaction Design Studio	
C. Select one of the following:		
CS 155	Computer and Network Security	
CS 255	Introduction to Cryptography	
D.		
CS 294S	Research Project in Software Systems and Security	
E. Sufficient depth units from categories (a) through (d) and the following:		
CS 224W	Social and Information Network Analysis	
CS 241	Embedded Systems Workshop	
CS 244E		
CS 246	Mining Massive Data Sets	
CS 251	Bitcoin and Crypto Currencies	
CS 344	Topics in Computer Networks (CS 344 with any suffix)	
CS 364A		
CS 376	Human-Computer Interaction Research	
CS 393	Computer Laboratory *	
CS 395	Independent Database Project *	
CS 399	Independent Project *	
EE 359	Wireless Communications	
EE 384A	Internet Routing Protocols and Standards	
EE 384B	Multimedia Communication over the Internet (not given this year)	
EE 384C	Wireless Local and Wide Area Networks	
EE 384E	Networked Wireless Systems	
EE 384S	Performance Engineering of Computer Systems & Networks	
COMM 268		
PSYCH 252	Statistical Methods for Behavioral and Social Sciences	

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) through (e) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Mobile and Internet Computing must take five courses satisfying the area (a) through (d) requirements above.

Mobile and Internet Computing Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 143	Compilers	3-4
CS 145	Introduction to Databases	3-4
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 227B	General Game Playing	3
CS 228	Probabilistic Graphical Models: Principles and Techniques	3-4
CS 229	Machine Learning	3-4
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 233	Geometric and Topological Data Analysis	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E	(no longer offered)	
CS 240H		3-4
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 244B		3
CS 249A		3
CS 261	Optimization and Algorithmic Paradigms	3
CS 262		3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 268	Geometric Algorithms	3

CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 276	Information Retrieval and Web Search	3
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

7. Real-World Computing—

A. Select at least three of the following:

CS 148	Introduction to Computer Graphics and Imaging
CS 223A	Introduction to Robotics
CS 231A	Computer Vision: From 3D Reconstruction to Recognition
CS 248	Interactive Computer Graphics

B. Select at least three of the following:

CS 205A	Mathematical Methods for Robotics, Vision, and Graphics
CS 233	Geometric and Topological Data Analysis
CS 249A	
CS 262	
CS 268	Geometric Algorithms
CS 348A	Computer Graphics: Geometric Modeling & Processing
CS 348B	Computer Graphics: Image Synthesis Techniques
CS 348C	Computer Graphics: Animation and Simulation
CS 374	
CME 302	Numerical Linear Algebra
CME 306	Numerical Solution of Partial Differential Equations

C. Sufficient additional units chosen from the above and from the following:

CS 225A	Experimental Robotics
CS 228	Probabilistic Graphical Models: Principles and Techniques
CS 229	Machine Learning
CS 231B	
CS 231M	
CS 232	Digital Image Processing
or EE 368	Digital Image Processing
CS 247	Human-Computer Interaction Design Studio
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 272	Introduction to Biomedical Informatics Research Methodology
CS 273A	A Computational Tour of the Human Genome
CS 274	Representations and Algorithms for Computational Molecular Biology
CS 294A	Research Project in Artificial Intelligence *
CS 327A	Advanced Robotic Manipulation (Not given this year)
CS 328	Topics in Computer Vision
CS 331A	
CS 331B	Representation Learning in Computer Vision
CS 393	Computer Laboratory *
CS 395	Independent Database Project *
CS 399	Independent Project *
CS 448	Topics in Computer Graphics (CS 448 with any suffix)
EE 267	Virtual Reality

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Real-World Computing must take five total courses satisfying area (a) and two of the three courses in the area (b) requirements above (i.e., three courses in area (a) and two courses in area (b).

Real-World Computing Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 140	Operating Systems and Systems Programming	3-4
CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
or EE 284	Introduction to Computer Networks	
CS 145	Introduction to Databases	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E	(no longer offered)	
CS 240H		3-4
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 244	Advanced Topics in Networking	3-4
CS 244B		3
CS 244E		
CS 246	Mining Massive Data Sets	3
CS 255	Introduction to Cryptography	3
CS 261	Optimization and Algorithmic Paradigms	3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 276	Information Retrieval and Web Search	3
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
EE 180	Digital Systems Architecture	3-4

EE 282	Computer Systems Architecture	3

8. Software Theory—

A.

CS 243	Program Analysis and Optimizations	

B. Select at least one of the following:

CS 244	Advanced Topics in Networking	
CS 245	Database Systems Principles	
CS 341	Project in Mining Massive Data Sets	
CS 343	(Offered occasionally)	
CS 345	(Offered occasionally)	

C. Select at least two courses from the following:

CS 242	Programming Languages	
CS 255	Introduction to Cryptography	
CS 261	Optimization and Algorithmic Paradigms	
CS 263	Algorithms for Modern Data Models	
CS 264	Beyond Worst-Case Analysis	
CS 265	Randomized Algorithms and Probabilistic Analysis	
CS 266		
CS 267	Graph Algorithms	
CS 268	Geometric Algorithms	
CS 355	(Not given this year)	
CS 361B		
CS 367	(Not given this year)	

D. Sufficient depth units from (b), (c), or the following:

CS 250	Error Correcting Codes: Theory and Applications	
CS 251	Bitcoin and Crypto Currencies	
CS 294S	Research Project in Software Systems and Security (Not given this year)	
CS 346		
CS 362	(Not given this year)	
CS 393	Computer Laboratory *	
CS 395	Independent Database Project *	
CS 399	Independent Project *	

- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a)-(d) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Software Theory need to take 5 total courses satisfying the area (a) through (d) requirements above.

Software Theory Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 140	Operating Systems and Systems Programming	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 228	Probabilistic Graphical Models: Principles and Techniques	3-4
CS 229	Machine Learning	3-4
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 233	Geometric and Topological Data Analysis	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E	(no longer offered)	
CS 240H		3-4
CS 244B		3
CS 244E		
CS 246	Mining Massive Data Sets	3-4
CS 249A		3
CS 262		3
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 276	Information Retrieval and Web Search	3
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

9. Systems—

A.

CS 140	Operating Systems and Systems Programming **	
CS 144	Introduction to Computer Networking **	
CS 240	Advanced Topics in Operating Systems	

B. Select at least four of the following:

CS 242	Programming Languages	
CS 243	Program Analysis and Optimizations	
CS 244	Advanced Topics in Networking	
CS 245	Database Systems Principles	
CS 248	Interactive Computer Graphics	
CS 348B	Computer Graphics: Image Synthesis Techniques	
EE 271	Introduction to VLSI Systems	
EE 282	Computer Systems Architecture	

C. At least two additional courses chosen from category (b) and the following:

CS 240E	(no longer offered)	
CS 240H		
CS 241	Embedded Systems Workshop	
CS 244B		
CS 244E		
CS 246	Mining Massive Data Sets	
CS 249A		
CS 249B		
CS 251	Bitcoin and Crypto Currencies	
CS 255	Introduction to Cryptography	
CS 262		

CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 272	Introduction to Biomedical Informatics Research Methodology
CS 276	Information Retrieval and Web Search
CS 294S	Research Project in Software Systems and Security (Not given this year) *
CS 315B	Parallel Computing Research Project
CS 316	Advanced Multi-Core Systems
CS 340	Topics in Computer Systems
CS 341	Project in Mining Massive Data Sets
CS 343	(Not given this year)
CS 344	Topics in Computer Networks (CS 344 with any suffix)
CS 345	(Offered occasionally)
CS 346	
CS 347	
CS 348A	Computer Graphics: Geometric Modeling & Processing
CS 348C	Computer Graphics: Animation and Simulation
CS 349	Topics in Programming Systems
CS 374	
CS 393	Computer Laboratory *
CS 395	Independent Database Project *
CS 399	Independent Project *
CS 448	Topics in Computer Graphics (CS 448 with any suffix)
EE 267	Virtual Reality
EE 273	Digital Systems Engineering
EE 382C	Interconnection Networks
EE 384A	Internet Routing Protocols and Standards
EE 384B	Multimedia Communication over the Internet (not given this year)
EE 384C	Wireless Local and Wide Area Networks
EE 384S	Performance Engineering of Computer Systems & Networks

- Students with a 27-unit depth option (Option 1 above) must take 27 units subject to satisfying the area (a), (b), and (c) requirements above.
- Students with a 21-unit depth option (Option 2 above) must take that many units subject to satisfying the area (a) and (b) requirements above, and additional courses may be taken from area (c) if any courses in the area (a) requirement are waived.
- Students with a secondary area of specialization (per Option 2 above) in Systems need to take five courses; those courses must satisfy the area (a) requirement and additional courses may be taken from area (b).

Systems Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 166		3-4
CS 168	The Modern Algorithmic Toolbox	3-4
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 233	Geometric and Topological Data Analysis	3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 228	Probabilistic Graphical Models: Principles and Techniques	3-4
CS 229	Machine Learning	3-4
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 233	Geometric and Topological Data Analysis	3
CS 261	Optimization and Algorithmic Paradigms	3
CS 264	Beyond Worst-Case Analysis	3
CS 265	Randomized Algorithms and Probabilistic Analysis	3
CS 266		3
CS 267	Graph Algorithms	3
CS 268	Geometric Algorithms	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3

10. Theoretical Computer Science—

A.

CS 154	Introduction to Automata and Complexity Theory **
CS 261	Optimization and Algorithmic Paradigms

B. Sufficient additional units chosen from:

CS 166	
CS 168	The Modern Algorithmic Toolbox
CS 228	Probabilistic Graphical Models: Principles and Techniques
CS 233	Geometric and Topological Data Analysis
CS 246	Mining Massive Data Sets
CS 250	Error Correcting Codes: Theory and Applications
CS 251	Bitcoin and Crypto Currencies
CS 254	Computational Complexity
CS 255	Introduction to Cryptography
CS 262	
CS 263	Algorithms for Modern Data Models
CS 264	Beyond Worst-Case Analysis
CS 265	Randomized Algorithms and Probabilistic Analysis
CS 266	
CS 267	Graph Algorithms
CS 268	Geometric Algorithms
CS 334A	Convex Optimization I
or EE 364A	Convex Optimization I
CS 341	Project in Mining Massive Data Sets
CS 345	(Offered occasionally)
CS 354	(Not given this year)
CS 355	(Not given this year)

CS 357	(Not given this year)
CS 358	Topics in Programming Language Theory
CS 359	Topics in the Theory of Computation *
CS 362	(Not given this year)
CS 364A	
CS 366	(Not given this year)
CS 367	(Not given this year)
CS 369	Topics in Analysis of Algorithms *
CS 374	(not given this year)
CS 393	Computer Laboratory *
CS 395	Independent Database Project *
CS 399	Independent Project *
CS 468	Topics in Geometric Algorithms: Machine Learning for 3D Data *
MS&E 310	Linear Programming

- Multiple CS 359, CS 369, and/or CS 468 courses may be taken as long as they are each on different topics, denoted by different letter suffixes for the courses.
- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) and (b) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Theoretical Computer Science need to take 5 total courses satisfying the area (a) and (b) requirements above.

Theoretical Computer Science Breadth Courses

Students in the single depth specialization must complete three of the following breadth courses and receive a letter grade for each.

CS 124	From Languages to Information	3-4
CS 140	Operating Systems and Systems Programming	3-4
CS 143	Compilers	3-4
CS 144	Introduction to Computer Networking	3-4
or EE 284	Introduction to Computer Networks	
CS 145	Introduction to Databases	3-4
CS 147	Introduction to Human-Computer Interaction Design	3-5
CS 148	Introduction to Computer Graphics and Imaging	3-4
CS 149	Parallel Computing	3-4
CS 154	Introduction to Automata and Complexity Theory	3-4
CS 155	Computer and Network Security	3
CS 157	Logic and Automated Reasoning	3
CS 205A	Mathematical Methods for Robotics, Vision, and Graphics	3
CS 221	Artificial Intelligence: Principles and Techniques	3-4
CS 223A	Introduction to Robotics	3
CS 224N	Natural Language Processing with Deep Learning	3-4
CS 224S	Spoken Language Processing	2-4
CS 224U	Natural Language Understanding	3-4
CS 224W	Social and Information Network Analysis	3
CS 227B	General Game Playing	3
CS 229	Machine Learning	3-4
CS 229A	(Not given this year)	3-4
CS 231A	Computer Vision: From 3D Reconstruction to Recognition	3
CS 240	Advanced Topics in Operating Systems	3
CS 240E		
CS 240H		3-4
CS 242	Programming Languages	3
CS 243	Program Analysis and Optimizations	3-4
CS 244	Advanced Topics in Networking	3-4
CS 244B		3
CS 244E		
CS 249A		3
CS 270	Modeling Biomedical Systems: Ontology, Terminology, Problem Solving	3
CS 273A	A Computational Tour of the Human Genome	3
CS 274	Representations and Algorithms for Computational Molecular Biology	3-4
CS 276	Information Retrieval and Web Search	3
CS 279	Computational Biology: Structure and Organization of Biomolecules and Cells	3
CME 108	Introduction to Scientific Computing	3-4
CME 302	Numerical Linear Algebra	3
EE 180	Digital Systems Architecture	3-4
EE 282	Computer Systems Architecture	3

* With consent of faculty adviser.

** Students with equivalent course work may waive with approval of their adviser.

Requirement 4

Additional elective units must be technical courses (numbered 100 or above) related to the degree program and approved by the adviser and MS program administrator. All CS courses numbered above 110 (with the exception of CS 196 and 198) taken for 3 or more units are pre-approved as elective courses. Additionally, up to a maximum of 3 units of 500-level CS seminars, CS 300, EE 380, EE 385A, or other 1-2 unit seminars offered in the School of Engineering may be counted as electives. 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. in 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. in 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, CS 395, or CS 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, CS 395, or CS 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. The writing and research units described in parts (1) and (2) may be counted toward the 45 units required for the degree.

3. All independent study units (CS 393, CS 395, CS 399) must be taken for letter grades and a GPA of 3.0 (B) or better must be maintained.
4. **Research Report**—Students must complete a significant report describing their research and its conclusions. The research report represents work that is publishable in a journal or at a high-quality conference, although it is presumably longer and more expansive in scope than a typical conference paper. A copy of the research report must be submitted to the student services office in the department three weeks before the beginning of the examination period in the student's final quarter. Both the primary and secondary adviser must approve the research report before the distinction-in-research designation can be conferred.

Master of Science in Computer Science Education

Candidates for the MS specialization in Computer Science Education will be admitted from a separate pool of applicants and will be eligible only for this specialization. The qualifications for admission are:

- A doctorate in an academic discipline other than Computer Science
- Experience and evidence of excellence in college-level teaching
- Successful completion of a standard introductory programming sequence (CS 106B or equivalent)

Admitted candidates will complete the following courses (45 units) over the course of four quarters:

CS 103, Mathematical Foundations of Computing

CS 107, Computer Organization and Systems

CS 108, Object-oriented Systems Design

CS 109, Introduction to Probability for Computer Scientists

CS 110, Principles of Computer Systems

CS 161, Design and Analysis of Algorithms

CS 198, Teaching Computer Science

CS 208E, Great Ideas in Computer Science

Two CS elective courses and a final project

Admission: In addition to the strong academic preparation required for any graduate program at Stanford, we are looking for candidates who are excellent teachers and who are able to supply evidence of successful university teaching. Applicants should submit a list of courses taught along with the associated teaching evaluations. In addition, applicants should ask their recommenders to concentrate on teaching experience and expertise; recommendation letters that focus on research strengths will carry relatively little weight.

Applications to the MS specialization in Computer Science Education and all supporting documents must be submitted and received online by the published deadline. Information on admission requirements and deadlines is available at http://cs.stanford.edu/admissions/.

Joint M.S. and MBA Degree

The joint MS in Computer Science/MBA degree links two of Stanford University's world-class programs. This joint degree offers students an opportunity to develop advanced technical and managerial skills for a broader perspective on both existing technologies and new technology ventures.

Admission to the joint MSCS/MBA program requires that students apply and be accepted independently to both the Computer Science Department in the School of Engineering and the Graduate School of Business. Students may apply concurrently, or elect to begin their course of study in CS and apply to the GSB during their first year.

Additional information on the MS in Computer Science/MBA Joint Degree Program and its requirements is available on the web at: https://cs/academics/current-masters/joint-cs-msmba-degree

Joint M.S. and Law Degree

Law students interested in pursuing an M.S. in Computer Science must apply for admission to the Computer Science Department either (i) concurrently with applying to the Law School; or (ii) after being admitted to the Law School, but no later than the earlier of: (a) the end of the second year of Law School; or (b) the Computer Science Department's admission deadline for the year following that second year of Law School.

In addition to being admitted separately to the Law School and the Computer Science Department, students must secure permission from both academic units to pursue degrees in those units as part of a joint degree program.

J.D./M.S. students may elect to begin their course of study in either the Law School or the Computer Science Department. Faculty advisors from each academic unit participate in the planning and supervising of the student's joint program. Students must be enrolled full-time in the Law School for the first year of law studies. Otherwise, enrollment may be in the graduate school or the Law School, and students may choose courses from either program regardless of where enrolled. Students must satisfy the requirements for both the J.D. degree as specified by the Law School and the M.S. degree as specified in this Bulletin.

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

Teaching and Research Assistantships in Computer Science

Graduate student assistantships are available. Half-time assistants receive a tuition scholarship for 8, 9, or 10 units per quarter during the academic year, and in addition receive a monthly stipend.

Duties for half-time assistants during the academic year involve approximately 20 hours of work per week. Course assistants (CAs) help an instructor teach a course by conducting discussion sections, consulting with students, and grading examinations. Research assistants (RAs) help faculty and senior staff members with research in computer science. Many MS students are hired to staff teaching and research assistantships. However, MS 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.

Doctor of Philosophy in Computer Science

The University's basic requirements for the Ph.D. degree are outlined in the "Graduate Degrees (http://exploredegrees.stanford.edu/graduatedegrees)" section of this bulletin. Department requirements are stated below.

Requirements

Applications to the Ph.D. program and all supporting documents must be submitted and received online by the published deadline. See the department's web site for admissions requirements and the application deadline (https://cs.stanford.edu/admissions/general-information). Changes or updates to the admission process are posted in September.

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 Student Services Office.
2. The first year of the Ph.D. program is spent working with 1-3 different professors on a rotating basis. The intent is to allow the first-year Ph.D. student to work with a variety of professors before aligning with a permanent program adviser. Students who don't need the full year to find a professor to align with will have the option of aligning within the first or second quarter.
3. The CS 300 Departmental Lecture Series seminar gives faculty the opportunity to explain their research to first year CS Ph.D. students. First year CS Ph.D. students are required to attend 2/3 of the classes to receive credit.
4. A student must complete 135 course units for graduation. Computer Science Ph.D. students take 8-10 units per quarter. Credit for coursework done elsewhere (up to the maximum of 45 course units) may be applied to graduation requirements. Students must also take at least three units of coursework from four different faculty members. There are NO courses specifically required by the CS Ph.D. program except for the 1 unit CS 300 Departmental Lecture Series and CS 499 Advanced Reading and Research or its equivalent. At least one course must be taken for a letter grade. A 3.0 GPA must be maintained.
5. 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 must fulfill two breadth-area requirements in each of three general areas by the end of the second year in the program. If students have fulfilled the six breadth-area requirements, and taken courses from at least four different faculty members, they are eligible to apply for candidacy prior to the second year in the program. An up-to-date list of courses that satisfy the breadth requirements (http://cs.stanford.edu/education/phd) can be found on the department's web site. The student must completely satisfy the breadth requirement by the end of the second year in the program and must pass a qualifying exam in the general area of their expected dissertation by the end of the third year in the program.
6. University policy requires that all doctoral students declare candidacy by the end of the sixth quarter in residence, excluding summers. However, after aligning with a permanent adviser, passing six breadth requirements, and taking classes with four different faculty, a student is eligible to file for candidacy prior to the sixth quarter. The candidacy form serves as a "contract" between the department and the student. The department acknowledges that the student is a *bona fide* candidate for the Ph.D. and agrees that the program submitted by the student is sufficient to warrant granting the Ph.D. upon completion. Candidacy expires five years from the date of submission of the candidacy form, rounded to the end of the quarter. In special cases, the department may extend a student's candidacy, but is under no obligation to do so.
7. Each student is required to pass a qualifying exam in their area by the end of their third year in the program. A student may only take the qualifying exam twice. If the student fails the qualifying exam a second time, the Ph.D. program committee is convened to discuss the student's lack of reasonable academic progress. Failing the exam a second time is cause for dismissal from the Computer Science Ph.D. program and the committee meets to discuss the final outcome for the student.
8. As part of the training for the Ph.D., the student is also required to complete at least four units (a unit is ten hours per week for one quarter) as a course assistant or instructor for courses in Computer Science numbered 100 or above.
9. The Reading Committee form and Oral Thesis Proposal must be submitted within one year of passing the qualifying exam.
10. The Oral Thesis Proposal must be submitted nine months before the oral defense date.
11. 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. The dissertation adviser is often the student's program adviser.
12. 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.
13. The student is expected to demonstrate the ability to present scholarly material orally in the dissertation defense.
14. 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 or outside of the University. The department requires at least two committee members to be affiliated with the Computer Science department. The principal adviser and at least one of the other committee members must be Academic Council members.

Guidelines for Reasonable Progress

By the end of the first academic year, a student should be aligned with a permanent research advisor.

By Spring Quarter of the second year, a student should complete all six breadth area requirements, two breadth area requirements in each of three areas, and file for candidacy.

By Spring Quarter of the third year, a student should pass a Qualifying Examination (https://cs.stanford.edu/academics/phd/qualifying-exams) in the area of his or her intended dissertation.

Within one year of passing the Qualifying Examination, a student should submit a signed Reading Committee Form (https://stanford.app.box.com/v/docdiss-reading-committee-form).

The teaching requirement may be satisfied at any time. The research requirement is routinely satisfied by participation in research throughout the student's career.

Ph.D. Minor in Computer Science

For a minor in Computer Science, a candidate must complete 20 units of Computer Science coursework numbered 200 or above, except for the 100-level courses listed on the Ph.D. Minor Worksheet (http://cs.stanford.edu/degrees/phd/admissions/Worksheet.pdf) (pdf). At least three of the courses must be master's core courses to provide breadth and one course numbered 300 or above to provide depth. One

of the courses taken must include a significant programming project to demonstrate programming efficiency. Courses must be taken for a letter grade and passed with a grade of 'B' or better. Applications for a minor in Computer Science are submitted at the same time as admission to candidacy.

Emeriti (Professors): Tom Binford, Edward Feigenbaum (http://ksl-web.stanford.edu/people/eaf), Richard Fikes (http://www.stanford.edu/~fikes), Donald E. Knuth (http://www-cs-faculty.stanford.edu/~knuth)*, Jean-Claude Latombe (http://robotics.stanford.edu/~latombe), Marc Levoy (http://graphics.stanford.edu/~levoy)*, Zohar Manna, Teresa Meng (http://dualist.stanford.edu/~thm), William F. Miller, Nils J. Nilsson (http://robotics.stanford.edu/~nilsson), Serge Plotkin (http://troll-w.stanford.edu/plotkin), Vaughan Pratt (http://boole.stanford.edu/pratt.html)*, Eric Roberts (http://cs.stanford.edu/people/eroberts)*, Yoav Shoham (http://robotics.stanford.edu/~shoham), Jeffrey D. Ullman (http://infolab.stanford.edu/~ullman), Gio Wiederhold (http://infolab.stanford.edu/people/gio.html), Terry Winograd (http://hci.stanford.edu/winograd)

Chair: Alex Aiken (http://theory.stanford.edu/~aiken)

Associate Chair for Education: Mehran Sahami (http://robotics.stanford.edu/users/sahami/bio.html)

Professors: Maneesh Agrawala (http://graphics.stanford.edu/~maneesh), Alex Aiken (http://theory.stanford.edu/~aiken), Serafim Batzoglou (http://www.serafimb.org/people.html), Dan Boneh (http://crypto.stanford.edu/~dabo), Moses Charikar, David Cheriton (http://www.stanford.edu/~cheriton), David Dill (http://verify.stanford.edu/dill), Ronald P. Fedkiw (http://physbam.stanford.edu/~fedkiw), Hector Garcia-Molina (http://infolab.stanford.edu/people/hector.html), Leonidas J. Guibas (http://geometry.stanford.edu/member/guibas), Patrick Hanrahan (http://www-graphics.stanford.edu/~hanrahan), John Hennessy, Mark A. Horowitz (http://www-vlsi.stanford.edu/~horowitz), Doug James (http://www.cs.cornell.edu/~djames), Dan Jurafsky (http://web.stanford.edu/~jurafsky), Oussama Khatib (http://robotics.stanford.edu/~ok), Monica Lam (http://suif.stanford.edu/~lam), James Landay (https://profiles.stanford.edu/james-landay), Nick McKeown (http://tiny-tera.stanford.edu/~nickm), Christopher Manning (http://nlp.stanford.edu/~manning), David Mazieres (http://www.scs.stanford.edu/~dm), John Mitchell (http://theory.stanford.edu/people/jcm/home.html), Kunle Olukotun (http://ogun.stanford.edu/~kunle), John Ousterhout (http://www.stanford.edu/~ouster/cgi-bin/home.php), Balaji Prabhakar (http://www.stanford.edu/~balaji), Omer Reingold, Mendel Rosenblum (http://web.stanford.edu/~mendel), Jennifer Widom (http://infolab.stanford.edu/~widom)

Associate Professors: Gill Bejerano (http://bejerano.stanford.edu), Ron Dror (http://cs.stanford.edu/people/rondror), Dawson Engler (http://www.stanford.edu/~engler), Michael Genesereth (http://logic.stanford.edu/people/genesereth/genesereth.html), Christoforos Kozyrakis (http://csl.stanford.edu/~christos), Jure Leskovec (http://cs.stanford.edu/people/jure), Philip Levis (http://csl.stanford.edu/~pal), Fei-Fei Li (http://vision.stanford.edu), Subhasish Mitra (http://www.stanford.edu/~subh), Tim Roughgarden (http://theory.stanford.edu/~tim)

Assistant Professors: Michael Bernstein (http://people.csail.mit.edu/msbernst), Stefano Ermon, Sachin Katti (http://www.stanford.edu/~skatti), Anshul Kundaje (https://sites.google.com/site/anshulkundaje), Percy Liang, Christopher Re (http://cs.stanford.edu/people/chrismre), Silvio Savarese (http://cvgl.stanford.edu/silvio), Greg Valiant (http://theory.stanford.edu/~valiant), Ryan Williams (http://web.stanford.edu/~rrwill), Virginia Williams (http://theory.stanford.edu/~virgi), Keith Winstein (http://web.mit.edu/keithw), Mary Wootters, Matei Zaharia

Professors (Research): Clark Barrett (http://www.cs.nyu.edu/~barrett), William J. Dally (http://cva.stanford.edu/billd_webpage_new.html), John K. Salisbury (http://robotics.stanford.edu/~jks)

Professor (Teaching): Mehran Sahami (http://robotics.stanford.edu/users/sahami/bio.html)

Associate Professor (Teaching):

Courtesy Professors: Russ Altman (http://bmir.stanford.edu/people/view.php/russ_b_altman), Stephen Boyd (http://www.stanford.edu/~boyd), Patrick Hayden, Michael Levitt, Roy Pea

Courtesy Associate Professors: Ashish Goel (http://www.stanford.edu/~ashishg), Noah Goodman (http://cocolab.stanford.edu/ndg.html), Justin Grimmer, Allison Okamura, Chris Potts

Courtesy Assistant Professors: John Duchi, Sean Follmer, Sharad Goel, Thomas Icard, Ramesh Johari, Mykel Kochenderfer (http://mykel.kochenderfer.com), Lester Mackey (http://web.stanford.edu/~lmackey), Stephen Montgomery (http://montgomerylab.stanford.edu), Camille Utterback, Ge Wang (https://ccrma.stanford.edu/~ge), Gordon Wetzstein

Lecturers: Gerald Cain, Chris Gregg, Victoria Kirst, Cynthia Lee, Nicholas J. Parlante (http://www-cs-faculty.stanford.edu/~nick), Chris Piech, Keith Schwarz, Marty Stepp (http://www.martystepp.com), Patrick Young (http://www.stanford.edu/~psyoung), Julie Zelenski (http://www-cs-faculty.stanford.edu/~zelenski)

Adjunct Professors: Stuart Card, Tom Dean, Kurt Konolige, P. Pandurang Nayak, Andrew Ng (http://www.andrewng.org), Bill MacCartney (http://nlp.stanford.edu/~wcmac), Prabhakar Raghavan (http://theory.stanford.edu/people/pragh), Vishal Sikka, Sebastian Thrun (http://robots.stanford.edu)

Visiting Professors: Gregory Niemeyer, Corrado Priami

Visiting Assistant Professors: Matthias Niessner, Soren Pirk

Secondary Appointment in CS: Anshul Kundaje

* Recalled to active duty.

Courses

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

CS 1U. Practical Unix. 1 Unit.
A practical introduction to using the Unix operating system with a focus on Linux command line skills. Class will consist of video tutorials and weekly hands-on lab sections. The time listed on AXESS is for the first week's logistical meeting only. Topics include: grep and regular expressions, ZSH, Vim and Emacs, basic and advanced GDB features, permissions, working with the file system, revision control, Unix utilities, environment customization, and using Python for shell scripts. Topics may be added, given sufficient interest. Course website: http://cs1u.stanford.edu.

CS 9. Problem-Solving for the CS Technical Interview. 1 Unit.
This course will prepare students to interview for software engineering and related internships and full-time positions in industry. Drawing on multiple sources of actual interview questions, students will learn key problem-solving strategies specific to the technical/coding interview. Students will be encouraged to synthesize information they have learned across different courses in the major. Emphasis will be on the oral and combination written-oral modes of communication common in coding interviews, but which are unfamiliar settings for problem solving for many students. Prerequisites: CS 106B or X.

CS 20SI. Tensorflow for Deep Learning Research. 2 Units.
This course will cover the fundamentals and contemporary usage of the Tensorflow library for deep learning research. Through the course, students will use Tensorflow to build models of different complexity, from simple linear/logistic regression to convolutional neural network and recurrent neural networks with LSTM to solve tasks such as word embeddings, translation, optical character recognition. Students will also learn best practices to structure a model and manage research experiments. Prerequisites: CS229 or CS224D/N.

CS 22A. The Social & Economic Impact of Artificial Intelligence. 1 Unit.
Recent advances in computing may place us at the threshold of a unique turning point in human history. Soon we are likely to entrust management of our environment, economy, security, infrastructure, food production, healthcare, and to a large degree even our personal activities, to artificially intelligent computer systems. The prospect of "turning over the keys" to increasingly autonomous systems raises many complex and troubling questions. How will society respond as versatile robots and machine-learning systems displace an ever-expanding spectrum of blue- and white-collar workers? Will the benefits of this technological revolution be broadly distributed or accrue to a lucky few? How can we ensure that these systems respect our ethical principles when they make decisions at speeds and for rationales that exceed our ability to comprehend? What, if any, legal rights and responsibilities should we grant them? And should we regard them merely as sophisticated tools or as a newly emerging form of life? The goal of CS22 is to equip students with the intellectual tools, ethical foundation, and psychological framework to successfully navigate the coming age of intelligent machines.

CS 27. Literature and Social Online Learning. 3-5 Units.
Study, develop, and test new digital methods, games, apps, interactive social media uses to innovate how the humanities can engage and educate students and the public today. Exploring well-known literary texts, digital storytelling forms and literary communities online, students work individually and in interdisciplinary teams to develop innovative projects aimed at bringing literature to life. Tasks include literary role-plays on Twitter; researching existing digital pedagogy and literary projects, games, and apps; reading and coding challenges; collaborative social events mediated by new technology. Minimal prerequisites which vary for students in CS and the humanities; please check with instructors. Same as: COMPLIT 239B, ENGLISH 239B

CS 41. Hap.py Code: The Python Programming Language. 2 Units.
The fundamentals and contemporary usage of the Python programming language. Primary focus on developing best practices in writing Python and exploring the extensible and unique parts of Python that make it such a powerful language. Topics include: data structures (e.g. lists and dictionaries) and characteristic pythonic conventions such as anonymous functions, iterables, and powerful built-ins (e.g. map, filter, zip). We will also cover object-oriented design, the standard library, and common third-party packages (e.g. requests, pillow). Time permitting, we will explore modern Python-based web frameworks and project distribution. Prerequisite: 106B/X or equivalent. Application required.

CS 42. Callback Me Maybe: Contemporary Javascript. 2 Units.
Introduction to the JavaScript programming language with a focus on building contemporary applications. Course consists of in-class activities and programming assignments that challenge students to create functional web apps (e.g. Yelp, Piazza, Instagram). Topics include syntax/semantics, event-based programming, document object model (DOM), application programming interfaces (APIs), asynchronous JavaScript and XML (AJAX), jQuery, Node.js, and MongoDB. Prerequisite: CS 107.

CS 45N. Computers and Photography: From Capture to Sharing. 3-4 Units.
Preference to freshmen with experience in photography and use of computers. Elements of photography, such as lighting, focus, depth of field, aperture, and composition. How a photographer makes photos available for computer viewing, reliably stores them, organizes them, tags them, searches them, and distributes them online. No programming experience required. Digital SLRs and editing software will be provided to those students who do not wish to use their own.

CS 50. Using Tech for Good. 2 Units.
Students in the class will work in small teams to implement high-impact projects for partner organizations. Taught by the CS+Social Good team, the aim of the class is to empower you to leverage technology for social good by inspiring action, facilitating collaboration, and forging pathways towards global change. Recommended: CS 106B, CS 42 or 142. Class is open to students of all years.May be repeat for credit.

CS 51. CS + Social Good Studio: Building Social Impact Projects for Change. 2 Units.
Get real-world experience launching and developing your own social impact projects! Students will work in small teams to develop high-impact projects around problem domains provided by partner organizations, under the guidance and support of design/technical coaches from industry and nonprofit domain experts. The class aims to provide an outlet, along with the resources, for students to create social change through CS, while providing students with experience engaging in the full product development cycle on real-world projects. Prerequisite: CS 147, equivalent experience, or consent of instructors.

CS 52. CS + Social Good: Implementing Sustainable Social Impact Projects. 2 Units.
Continuation of CS51 (Building Social Impact Projects for Change). Teams enter the quarter having completed and tested a minimal viable product (MVP) with a well-defined target user, and a community partner. Students will learn to apply scalable technical frameworks, methods to measure social impact, tools for deployment, user acquisition techniques and growth/exit strategies. The purpose of the class is to facilitate students to build a sustainable infrastructure around their product idea. CS52 will host mentors, guest speakers and industry experts for various workshops and coaching-sessions. The class culminates in a showcase where students share their projects with stakeholders and the public. Prerequisite: CS 51, or consent of instructor.

CS 54N. Great Ideas in Computer Science. 3 Units.
Stanford Introductory Seminar. Preference to freshmen. Covers the intellectual tradition of computer science emphasizing ideas that reflect the most important milestones in the history of the discipline. No prior experience with programming is assumed. Topics include programming and problem solving; implementing computation in hardware; algorithmic efficiency; the theoretical limits of computation; cryptography and security; and the philosophy behind artificial intelligence.

CS 82. Social Impacts of Media Innovation. 1 Unit.
Media innovation merges technical and cultural development and benefits diverse social groups in different ways. Considering historic media innovations such as cinema, hip-hop, and the works of innovator in residence Paul D. Miller aka DJ Spooky, the course focuses on what ideas benefit whom. Lectures and workshops underscore the need to innovate to survive and get heard, and offer know-how for radical innovation in the arts and entertainment industry. Course projects will be considered for inclusion in the Stanford Humanities Showcase. Open to both undergraduate and graduate students.

CS 95SI. Functional Programming in Clojure. 2 Units.
Clojure is a dialect of Lisp that runs on the JVM, CLR, or Javascript engine. This course explores the fundamentals and philosophy of Clojure, with emphasis on the benefits of immutability and functional programming that make it such a powerful and fun language. Topics include: immutability, functional programming (function composition, higher order functions), concurrency (atoms, promises, futures, actors, Software Transactional Memory, etc), LISP (REPL-driven development, homoiconicity, macros), and interop (between Clojure code and code native to the host VM). The course also explores design paradigms and looks at the differences between functional programing and object-oriented programing, as well as bottom-up versus top-down design.

CS 96SI. Mobilizing Healthcare - iOS Development for Mobile Health. 2 Units.
How can mobile technology can be leveraged to tackle pressing problems in healthcare? Our class will feature guest lecturers from Verily (formerly Google Life Sciences), Apple Health, and mobile health companies in developing countries and in the Bay Area. This class will give an overview of the fundamentals and contemporary usage of iOS development with a Mobile Health focus. Primary focus on developing best practices for Apple HealthKit and ResearchKit among other tools for iOS application development. Students will complete a project in the mobile health space sponsored and advised by professionals and student TAs. Recommended: CS193P or iOS development at a similar level. Apply at https://enrollcs96si.typeform.com/to/FGGHVl by Sept 30.

CS 101. Introduction to Computing Principles. 3-5 Units.
Introduces the essential ideas of computing: data representation, algorithms, programming "code", computer hardware, networking, security, and social issues. Students learn how computers work and what they can do through hands-on exercises. In particular, students will see the capabilities and weaknesses of computer systems so they are not mysterious or intimidating. Course features many small programming exercises, although no prior programming experience is assumed or required. CS101 is not a complete programming course such as CS106A. CS101 is effectively an alternative to CS105. A laptop computer is recommended for the in-class exercises.

CS 102. Big Data: Tools and Techniques, Discoveries and Pitfalls. 3-4 Units.
Aimed primarily at students who may not major in CS but want to learn about big data and apply that knowledge in their areas of study. Many of the world's biggest discoveries and decisions in science, technology, business, medicine, politics, and society as a whole, are now being made on the basis of analyzing massive data sets, but it is surprisingly easy to come to false conclusions from data analysis alone, and privacy of data connected to individuals can be a major concern. This course provides a broad introduction to big data: historical context and case studies; privacy issues; data analysis techniques including databases, data mining, and machine learning; sampling and statistical significance; data analysis tools including spreadsheets, SQL, Python, R; data visualization techniques and tools. Tools and techniques are hands-on but at a cursory level, providing a basis for future exploration and application. Prerequisites: high school AP computer science, CS106A, or other equivalent programming experience; comfort with statistics and spreadsheets helpful but not required.

CS 103. Mathematical Foundations of Computing. 3-5 Units.
Mathematical foundations required for computer science, including propositional predicate logic, induction, sets, functions, and relations. Formal language theory, including regular expressions, grammars, finite automata, Turing machines, and NP-completeness. Mathematical rigor, proof techniques, and applications. Prerequisite: 106A or equivalent.

CS 103A. Mathematical Problem-solving Strategies. 1 Unit.
Problem solving strategies and techniques in discrete mathematics and computer science. Additional problem solving practice for CS103. In-class participation required. Prerequisite: consent of instructor. Co-requisite: CS103.

CS 105. Introduction to Computers. 3-5 Units.
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.

CS 106A. Programming Methodology. 3-5 Units.
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. Summer quarter enrollment is limited.
Same as: ENGR 70A

CS 106B. Programming Abstractions. 3-5 Units.
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. Summer quarter enrollment is limited.
Same as: ENGR 70B

CS 106J. Programming Methodology in JavaScript. 3-5 Units.
Introduction to the engineering of computer applications emphasizing modern software engineering principles: object-oriented design, decomposition, encapsulation, abstraction, and testing. Emphasis is on good programming style. This course covers the same material as CS 106A but does so using JavaScript, the most common language for implementing interactive web pages, instead of Java. No prior programming experience required. Enrollment limited to 100.

CS 106L. Standard C++ Programming Laboratory. 1 Unit.
Supplemental lab to 106B and 106X. Additional features of standard C++ programming practice. Possible topics include advanced C++ language features, standard libraries, STL containers and algorithms, object memory management, operator overloading, and inheritance. Prerequisite: consent of instructor. Corequisite: 106B or 106X.

CS 106S. Programming Abstractions and Social Good. 1 Unit.
Supplemental lab to CS 106B and CS 106X. Students will apply fundamental computer science concepts learned in 106B/X to problems in the social good space (such as health, government, education, and environment). Course consists of in-class activities designed by local tech companies and nonprofits. Corequisite: 106B or 106X.

CS 106X. Programming Abstractions (Accelerated). 3-5 Units.
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.
Same as: ENGR 70X

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

CS 107E. Computer Systems from the Ground Up. 3-5 Units.
Introduction to the fundamental concepts of computer systems through bare metal programming on the Raspberry Pi. Explores how five concepts come together in computer systems: hardware, architecture, assembly code, the C language, and software development tools. Students do all programming with a Raspberry Pi kit and several add-ons (LEDs, buttons). Topics covered include: the C programming language, data representation, machine-level code, computer arithmetic, compilation, memory organization and management, debugging, hardware, and I/O. Prerequisite: 106B or X, and consent of instructor.

CS 108. Object-Oriented Systems Design. 3-4 Units.
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.

CS 109. Introduction to Probability for Computer Scientists. 3-5 Units.
Topics include: counting and combinatorics, random variables, conditional probability, independence, distributions, expectation, point estimation, and limit theorems. Applications of probability in computer science including machine learning and the use of probability in the analysis of algorithms. Prerequisites: 103, 106B or X, multivariate calculus at the level of MATH 51 or CME 100 or equivalent.

CS 109L. Statistical Computing with R Laboratory. 1 Unit.
Supplemental lab to CS109. Introduces the R programming language for statistical computing. Topics include basic facilities of R including mathematical, graphical, and probability functions, building simulations, introductory data fitting and machine learning. Provides exposure to the functional programming paradigm. Corequisite: CS109.

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

CS 124. From Languages to Information. 3-4 Units.
Extracting meaning, information, and structure from human language text, speech, web pages, genome sequences, social networks. Methods include: string algorithms, edit distance, language modeling, the noisy channel, naive Bayes, inverted indices, collaborative filtering, PageRank. Applications such as question answering, sentiment analysis, information retrieval, text classification, social network models, chatbots, genomic sequence alignment, spell checking, speech processing, recommender systems. Prerequisite: CS103, CS107, CS109.
Same as: LINGUIST 180, LINGUIST 280

CS 131. Computer Vision: Foundations and Applications. 3-4 Units.
Robots that can navigate space and perform duties, search engines that can index billions of images and videos, algorithms that can diagnose medical images for diseases, or smart cars that can see and drive safely: Lying in the heart of these modern AI applications are computer vision technologies that can perceive, understand and reconstruct the complex visual world. This course is designed for students who are interested in learning about the fundamental principles and important applications of computer vision. Course will introduce a number of fundamental concepts in computer vision and expose students to a number of real-world applications, plus guide students through a series of well designed projects such that they will get to implement cutting-edge computer vision algorithms. Prerequisites: Students should be familiar with Matlab (i.e. have programmed in Matlab before) and Linux; plus Calculus & Linear Algebra.

CS 140. Operating Systems and Systems Programming. 3-4 Units.
Operating systems design and implementation. Basic structure; synchronization and communication mechanisms; implementation of processes, process management, scheduling, and protection; memory organization and management, including virtual memory; I/O device management, secondary storage, and file systems. Prerequisite: CS 110.

CS 142. Web Applications. 3 Units.
Concepts and techniques used in constructing interactive web applications. Browser-side web facilities such as HTML, cascading stylesheets, the document object model, and JavaScript frameworks and Server-side technologies such as server-side JavaScript, sessions, and object-oriented databases. Issues in web security and application scalability. New models of web application deployment. Prerequisites: CS 107 and CS 108.

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

CS 144. Introduction to Computer Networking. 3-4 Units.
Principles and practice. Structure and components of computer networks, packet switching, layered architectures. Applications: web/http, voice-over-IP, p2p file sharing and socket programming. Reliable transport: TCP/IP, reliable transfer, flow control, and congestion control. The network layer: names and addresses, routing. Local area networks: ethernet and switches. Wireless networks and network security. Prerequisite: CS 110.

CS 145. Introduction to Databases. 3-4 Units.
The course covers database design and the use of database management systems for applications. It includes extensive coverage of the relational model, relational algebra, and SQL.The course includes database design and relational design principles based on dependencies and normal forms. Many additional key database topics from the design and application-building perspective are also covered: indexes, views, transactions, authorization, integrity constraints, triggers, on-line analytical processing (OLAP), JSON, and emerging NoSQL systems. Class time will include guest speakers from industry and additional advanced topics as time and class interest permits. Prerequisites: 103 and 107 (or equivalent).

CS 147. Introduction to Human-Computer Interaction Design. 3-5 Units.
Introduces fundamental methods and principles for designing, implementing, and evaluating user interfaces. Topics: user-centered design, rapid prototyping, experimentation, direct manipulation, cognitive principles, visual design, social software, software tools. Learn by doing: work with a team on a quarter-long design project, supported by lectures, readings, and studios. Prerequisite: 106B or X or equivalent programming experience. Recommended that CS Majors have also taken one of 142, 193P, or 193A.

CS 148. Introduction to Computer Graphics and Imaging. 3-4 Units.
Introductory prerequisite course in the computer graphics sequence introducing students to the technical concepts behind creating synthetic computer generated images. Focuses on using OpenGL to create visual imagery, as well as an understanding of the underlying mathematical concepts including triangles, normals, interpolation, texture mapping, bump mapping, etc. Course will cover fundamental understanding of light and color, as well as how it impacts computer displays and printers. Class will discuss more thoroughly how light interacts with the environment, constructing engineering models such as the BRDF, plus various simplifications into more basic lighting and shading models. Also covers ray tracing technology for creating virtual images, while drawing parallels between ray tracers and real world cameras to illustrate various concepts. Anti-aliasing and acceleration structures are also discussed. The final class mini-project consists of building out a ray tracer to create visually compelling images. Starter codes and code bits will be provided to aid in development, but this class focuses on what you can do with the code as opposed to what the code itself looks like. Therefore grading is weighted toward in person "demos" of the code in action - creativity and the production of impressive visual imagery are highly encouraged. Prerequisites: CS 107, MATH 51.

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

CS 154. Introduction to Automata and Complexity Theory. 3-4 Units.
This course provides a mathematical introduction to the following questions: What is computation? Given a computational model, what problems can we hope to solve in principle with this model? Besides those solvable in principle, what problems can we hope to efficiently solve? In many cases we can give completely rigorous answers; in other cases, these questions have become major open problems in computer science and mathematics. By the end of this course, students will be able to classify computational problems in terms of their computational complexity (Is the problem regular? Not regular? Decidable? Recognizable? Neither? Solvable in P? NP-complete? PSPACE-complete?, etc.). Students will gain a deeper appreciation for some of the fundamental issues in computing that are independent of trends of technology, such as the Church-Turing Thesis and the P versus NP problem. Prerequisites: CS 103 or 103B.

CS 155. Computer and Network Security. 3 Units.
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 security, application security (web, apps, databases), malware, privacy, and security for mobile devices. Course projects focus on building reliable code. Prerequisite: 110. Recommended: basic Unix.

CS 157. Logic and Automated Reasoning. 3 Units.
An elementary exposition from a computational point of view of propositional and predicate logic, axiomatic theories, and theories with equality and induction. Interpretations, models, validity, proof, strategies, and applications. Automated deduction: polarity, skolemization, unification, resolution, equality. Prerequisite: 103 or 103B.

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

CS 168. The Modern Algorithmic Toolbox. 3-4 Units.
This course will provide a rigorous and hands-on introduction to the central ideas and algorithms that constitute the core of the modern algorithms toolkit. Emphasis will be on understanding the high-level theoretical intuitions and principles underlying the algorithms we discuss, as well as developing a concrete understanding of when and how to implement and apply the algorithms. The course will be structured as a sequence of one-week investigations; each week will introduce one algorithmic idea, and discuss the motivation, theoretical underpinning, and practical applications of that algorithmic idea. Each topic will be accompanied by a mini-project in which students will be guided through a practical application of the ideas of the week. Topics include hashing, dimension reduction and LSH, boosting, linear programming, gradient descent, sampling and estimation, and an introduction to spectral techniques. Prerequisites: CS107 and CS161, or permission from the instructor.

CS 170. Stanford Laptop Orchestra: Composition, Coding, and Performance. 3-4 Units.
Classroom instantiation of the Stanford Laptop Orchestra (SLOrk) which includes public performances. An ensemble of more than 20 humans, laptops, controllers, and special speaker arrays designed to provide each computer-mediated instrument with its sonic identity and presence. Topics and activities include issues of composing for laptop orchestras, instrument design, sound synthesis, programming, and live performance. May be repeated four times for credit. Space is limited; see https://ccrma.stanford.edu/courses/128 for information about the application and enrollment process.
Same as: MUSIC 128

CS 173. A Computational Tour of the Human Genome. 3 Units.
(Only one of 173 or 273A counts toward any CS degree program.) Introduction to computational biology through an informatic exploration of the human genome. Topics include: genome sequencing; functional landscape of the human genome (genes, gene regulation, repeats, RNA genes, epigenetics); genome evolution (comparative genomics, ultraconservation, co-option). Additional topics may include population genetics, personalized genomics, and ancient DNA. Course includes primers on molecular biology, the UCSC Genome Browser, and text processing languages. Guest lectures on current genomic research topics. Class will be similar in spirit to CS273A, which will not be offered this year. Prerequisites: CS107 or equivalent background in programming.

CS 181. Computers, Ethics, and Public Policy. 4 Units.
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.

CS 181W. Computers, Ethics, and Public Policy. 4 Units.
Writing-intensive version of CS181. Satisfies the WIM requirement for Computer Science, Engineering Physics, STS, and Math/Comp Sci undergraduates.
Same as: WIM

CS 183E. Effective Leadership in High-Tech. 1 Unit.
You will undoubtedly leave Stanford with the technical skills to excel in your first few jobs. But non-technical skills are just as critical to making a difference. This seminar is taught by two industry veterans in engineering leadership and product management. In a small group setting, we will explore how you can be a great individual contributor (communicating with clarity, getting traction for your ideas, resolving conflict, and delivering your best work) and how you can transition into leadership roles (finding leadership opportunities, creating a great team culture, hiring and onboarding new team members). We will end by turning back to your career (picking your first job and negotiating your offer, managing your career changes, building a great network, and succeeding with mentors). Prerequisites: Preference given to seniors and co-terms in Computer Science and related majors.

CS 183F. Startup School: The First 100 Days. 2 Units.
Starting a company is hard. Starting a company and building it into a successful, lasting business is even more so. This course aims to teach the fundamentals of starting a startup through a 10-week interactive class, with the goal of decreasing the barrier to entry for new entrepreneurs. Instruction will focus primarily on the first 100 days, from ideation to execution, covering the minutiae of company structure, product design, core metric evaluation, ethics and so on. Lectures will be taught by experts in the startup space, accompanied by small group learning sessions with active entrepreneurs.

CS 191. Senior Project. 1-6 Unit.
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. Prerequisite: Completion of at least 135 units.

CS 191W. Writing Intensive Senior Project. 3-6 Units.
Restricted to Computer Science and Computer Systems Engineering students. Writing-intensive version of CS191. Register using the section number of an Academic Council member. Prerequisite: Completion of at least 135 units.
Same as: WIM

CS 192. Programming Service Project. 1-4 Unit.
Restricted to Computer Science students. Appropriate academic credit (without financial support) is given for volunteer computer programming work of public benefit and educational value.

CS 193A. Android Programming. 3 Units.
Introduction to building applications for Android platform. Examines key concepts of Android programming: tool chain, application life-cycle, views, controls, intents, designing mobile UIs, networking, threading, and more. Features weekly lectures and a series of small programming projects. Phone not required, but a phone makes the projects more engaging. Prerequisites: 106B or Java experience at 106B level. Enrollment limited and application required: https://goo.gl/forms/ACKDYowYfOZD4vNd2.

CS 193C. Client-Side Internet Technologies. 3 Units.
Client-side technologies used to create web sites such as Google maps or Gmail. Includes HTML5, CSS, JavaScript, the Document Object Model (DOM), and Ajax. Prerequisite: programming experience at the level of CS106A.

CS 193P. iPhone and iPad Application Programming. 3 Units.
Tools and APIs required to build applications for the iPhone and iPad platforms using the iOS SDK. User interface design for mobile devices and unique user interactions using multi-touch technologies. Object-oriented design using model-view-controller paradigm, memory management, Swift programming language. Other topics include: object-oriented database API, animation, mobile device power management, multi-threading, networking and performance considerations. Prerequisites: C language and object-oriented programming experience exceeding 106B or X level. Previous completion of any one of the following is required: CS 107, 108 (preferred) or 110. Recommended: UNIX, graphics, databases.

CS 193S. Scalability Engineering. 3 Units.
Learn to solve real world engineering challenges in this programming project course. Scale projects not just from the coding and engineering perspective, but use those same techniques to increase usability, popularity, development velocity and maintainability. Discover how engineering applies to project ideation, self and team development, customer acquisition, user experience. As we build applications, we will cover tools and practices for scalable programming including: the javascript ecosystem, containers and cloud platforms, agile development, growth hacking. We focus on rapid feedback loops to build better systems faster. In one quarter, develop scalable habits to build apps designed to grow. Application required. Prerequisites: one or more of CS 140, 142, 145, 147.

CS 193X. Web Programming Fundamentals. 3-5 Units.
Introduction to full-stack web development with an emphasis on fundamentals. Client-side topics include layout and rendering through HTML and CSS, event-driven programming through JavaScript, and single-threaded asynchronous programming techniques including Promises. Focus on modern standardized APIs and best practices. Server-side topics include the development of RESTful APIs, JSON services, and basic server-side storage techniques. Covers desktop and mobile web development. Enrollment is limited. Application required. Apply at https://goo.gl/forms/nlQte0OqVpU6maRq2 byn11:59pm Tuesday, March 28. nPrerequisite: 106B or equivalent.

CS 194. Software Project. 3 Units.
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. Preference given to seniors. May be repeat for credit. Prerequisites: CS 110 and CS 161.

CS 194H. User Interface Design Project. 3-4 Units.
Advanced methods for designing, prototyping, and evaluating user interfaces to computing applications. Novel interface technology, advanced interface design methods, and prototyping tools. Substantial, quarter-long course project that will be presented in a public presentation. Prerequisites: CS 147, or permission of instructor.

CS 194W. Software Project. 3 Units.
Restricted to Computer Science and Electrical Engineering undergraduates. Writing-intensive version of CS194. Preference given to seniors.
Same as: WIM

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

CS 198. Teaching Computer Science. 3-4 Units.
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.

CS 198B. Additional Topics in Teaching Computer Science. 1 Unit.
Students build on the teaching skills developed in CS198. Focus is on techniques used to teach topics covered in CS106B. Prerequisite: successful completion of CS198.

CS 199. Independent Work. 1-6 Unit.
Special study under faculty direction, usually leading to a written report. Letter grade; if not appropriate, enroll in 199P.

CS 199P. Independent Work. 1-6 Unit.
(Staff).

CS 202. Law for Computer Science Professionals. 1 Unit.
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.

CS 203. Cybersecurity: A Legal and Technical Perspective. 2 Units.
This class will use the case method to teach basic computer, network, and information security from a technology, law, policy, and business perspective. Using recent security incidents from the news, we will discuss the technical aspects of the incident, the legal and policy aspects of the problem, and business approaches to managing breaches. The case studies will be organized around the following topics: tracking political dissidents, state sponsored sabotage, corporate and government espionage, credit card theft, theft of embarrassing personal data, phishing and social engineering attacks, denial of service attacks, attacks on weak session management and URLs, cloud data storage as a security risk, wiretapping on the Internet, and digital forensics.nStudents taking the class will learn about the techniques attackers use, applicable legal prohibitions, rights, and remedies, and approaches to managing the risk and aftermath of an attack. Grades will be based on class participation (25%) and on a student term paper explaining the technical and legal concepts relevant to a recent cybersecurity breach of the student's choice, with instructor approval (75%). The class will be co-taught by Stanford Professor of Computer Science and Electrical Engineering and co-director of the Stanford Computer Security Lab Dan Boneh and Director of Civil Liberties at the Law School's Center for Internet and Society Jennifer Granick. Special Instructions: This class is limited to 80 students, with an effort made to have students from SLS (40 students will be selected by lottery) and students from Computer Science (40 students). Elements used in grading: Class Participation, Final Paper. Cross-listed with Computer Science (Same as Law 4004).

CS 204. Legal Informatics. 2-3 Units.
Legal informatics based on representation of regulations in computable form. Encoding regulations facilitate creation of legal information systems with significant practical value. Convergence of technological trends, growth of the Internet, advent of semantic web technology, and progress in computational logic make computational law prospects better. Topics: current state of computational law, prospects and problems, philosophical and legal implications. This course is *Cross* listed with LAW 4019. Prerequisite: basic concepts of programming.

CS 205A. Mathematical Methods for Robotics, Vision, and Graphics. 3 Units.
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; or equivalents.

CS 205B. Mathematical Methods for Fluids, Solids, and Interfaces. 3 Units.
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.

CS 206. Exploring Computational Journalism. 3 Units.
This course will explore the evolving field of computational journalism. Students will research and discuss the state of field in five areas where computation is affecting journalism: AI, Data Science, and Info Viz; Emerging Hardware Tech, including Drones, Sensors, and VR; Audience Participation and Diverse Viewpoints; Free Speech and Democracy; and News Ecosystems and Business Models. Admission by application; please email James Hamilton at jayth@stanford.edu to request an application.
Same as: COMM 281

CS 208E. Great Ideas in Computer Science. 3 Units.
Great Ideas in Computer Science Covers the intellectual tradition of computer science emphasizing ideas that reflect the most important milestones in the history of the discipline. Topics include programming and problem solving; implementing computation in hardware; algorithmic efficiency; the theoretical limits of computation; cryptography and security; computer networks; machine learning; and the philosophy behind artificial intelligence. Readings will include classic papers along with additional explanatory material. Enrollment limited to students in the Master's program in Computer Science Education.

CS 210A. Software Project Experience with Corporate Partners. 3-4 Units.
Two-quarter project course. Focus is on real-world software development. Corporate partners seed projects with loosely defined challenges from their R&D labs; students innovate to build their own compelling software solutions. Student teams are treated as start-up companies with a budget and a technical advisory board comprised of instructional staff and corporate liaisons. Teams will typically travel to the corporate headquarters of their collaborating partner, meaning some teams will travel internationally. Open loft classroom format such as found in Silicon Valley software companies. Exposure to: current practices in software engineering; techniques for stimulating innovation; significant development experience with creative freedoms; working in groups; real-world software engineering challenges; public presentation of technical work; creating written descriptions of technical work. Prerequisites: CS 109 and 110.

CS 210B. Software Project Experience with Corporate Partners. 3-4 Units.
Continuation of CS210A. Focus is on real-world software development. Corporate partners seed projects with loosely defined challenges from their R&D labs; students innovate to build their own compelling software solutions. Student teams are treated as start-up companies with a budget and a technical advisory board comprised of the instructional staff and corporate liaisons. Teams will typically travel to the corporate headquarters of their collaborating partner, meaning some teams will travel internationally. Open loft classroom format such as found in Silicon Valley software companies. Exposure to: current practices in software engineering; techniques for stimulating innovation; significant development experience with creative freedoms; working in groups; real world software engineering challenges; public presentation of technical work; creating written descriptions of technical work. Prerequisites: CS 210A.

CS 213. Creating Great VR: From Ideation to Monetization. 1 Unit.
Covering everything from VR fundamentals to futurecasting to launch management, this course will expose you to best practices and guidance from VR leaders that helps positions you to build great VR experiences.

CS 221. Artificial Intelligence: Principles and Techniques. 3-4 Units.
Artificial intelligence (AI) has had a huge impact in many areas, including medical diagnosis, speech recognition, robotics, web search, advertising, and scheduling. This course focuses on the foundational concepts that drive these applications. In short, AI is the mathematics of making good decisions given incomplete information (hence the need for probability) and limited computation (hence the need for algorithms). Specific topics include search, constraint satisfaction, game playing, Markov decision processes, graphical models, machine learning, and logic. Prerequisites: CS 103 or CS 103B/X, CS 106B or CS 106X, CS 107, and CS 109 (algorithms, probability, and programming experience).

CS 223A. Introduction to Robotics. 3 Units.
Robotics foundations in modeling, design, planning, and control. Class covers relevant results from geometry, kinematics, statics, dynamics, motion planning, and control, providing the basic methodologies and tools in robotics research and applications. Concepts and models are illustrated through physical robot platforms, interactive robot simulations, and video segments relevant to historical research developments or to emerging application areas in the field. Recommended: matrix algebra.
Same as: ME 320

CS 224D. Deep Learning for Natural Language Processing. 3-4 Units.
Deep learning approaches have obtained very high performance across many different natural language processing tasks. In this class, students will learn to understand, implement, train, debug, visualize and potentially invent their own neural network models for a variety of language understanding tasks. The course provides a deep excursion from early models to cutting-edge research. Applications will range across a broad spectrum: from simple tasks like part of speech tagging, over sentiment analysis to question answering and machine translation. The final project will involve implementing a complex neural network model and applying it to a large scale NLP problem. We will introduce a common programming framework for deep learning for the problem sets.Prerequisites: programming abilities (python), linear algebra, Math 21 or equivalent, machine learning background (CS 229 or similar) Recommended: CS 224N, EE364a (convex optimization), CS 231N.

CS 224N. Natural Language Processing with Deep Learning. 3-4 Units.
Methods for processing human language information and the underlying computational properties of natural languages. Focus on deep learning approaches: understanding, implementing, training, debugging, visualizing, and extending neural network models for a variety of language understanding tasks. Exploration of natural language tasks ranging from simple word level and syntactic processing to coreference, question answering, and machine translation. Examination of representative papers and systems and completion of a final project applying a complex neural network model to a large-scale NLP problem. Prerequisites: calculus and linear algebra; CS124 or CS121/221.
Same as: LINGUIST 284

CS 224S. Spoken Language Processing. 2-4 Units.
Introduction to spoken language technology with an emphasis on dialogue and conversational systems. Automatic speech recognition, speech synthesis, dialogue management, and applications to digital assistants, search, and spoken language understanding systems. Covers state-of-the-art approaches based on deep learning as well as traditional methods. Prerequisites: CS 124, 221, 224N, or 229.
Same as: LINGUIST 285

CS 224U. Natural Language Understanding. 3-4 Units.
Project-oriented class focused on developing systems and algorithms for robust machine understanding of human language. Draws on theoretical concepts from linguistics, natural language processing, and machine learning. Topics include lexical semantics, distributed representations of meaning, relation extraction, semantic parsing, sentiment analysis, and dialogue agents, with special lectures on developing projects, presenting research results, and making connections with industry. Prerequisites: one of LINGUIST 180, CS 124, CS 224N, CS224S, or CS221; and logical/semantics such as LINGUIST 130A or B, CS 157, or PHIL150.
Same as: LINGUIST 188

CS 224W. Social and Information Network Analysis. 3-4 Units.
How do diseases spread? Who are the influencers? How can we predict friends and enemies in a social network? How information flows and mutates as it is passed through networks? Behind each of these questions there is an intricate wiring diagram, a network, that defines the interactions between the components. And we will never understand these questions unless we understand the networks behind them. The course will cover recent research on the structure and analysis of such large social and information networks and on models and algorithms that abstract their basic properties. Class will explore how to practically analyze large-scale network data and how to reason about it through models for network structure and evolution. Topics include methods for link analysis and network community detection, diffusion and information propagation on the web, virus outbreak detection in networks, and connections with work in the social sciences and economics.

CS 225A. Experimental Robotics. 3 Units.
Hands-on laboratory course experience in robotic manipulation. Topics include robot kinematics, dynamics, control, compliance, sensor-based collision avoidance, and human-robot interfaces. Second half of class is devoted to final projects using various robotic platforms to build and demonstrate new robot task capabilities. Previous projects include the development of autonomous robot behaviors of drawing, painting, playing air hocket, yoyo, basketball, ping-pong or xylophone. Prerequisites: 223A or equivalent.

CS 225B. Robot Programming Laboratory. 3-4 Units.
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.

CS 227B. General Game Playing. 3 Units.
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.

CS 228. Probabilistic Graphical Models: Principles and Techniques. 3-4 Units.
Probabilistic graphical modeling languages for representing complex domains, algorithms for reasoning using these representations, and learning these representations from data. Topics include: Bayesian and Markov networks, extensions to temporal modeling such as hidden Markov models and dynamic Bayesian networks, exact and approximate probabilistic inference algorithms, and methods for learning models from data. Also included are sample applications to various 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.

CS 229. Machine Learning. 3-4 Units.
Topics: statistical pattern recognition, linear and non-linear regression, non-parametric methods, exponential family, GLMs, support vector machines, kernel methods, model/feature selection, learning theory, VC dimension, clustering, density estimation, EM, dimensionality reduction, ICA, PCA, reinforcement learning and adaptive control, Markov decision processes, approximate dynamic programming, and policy search. Prerequisites: linear algebra, and basic probability and statistics.
Same as: STATS 229

CS 229T. Statistical Learning Theory. 3 Units.
How do we formalize what it means for an algorithm to learn from data? This course focuses on developing mathematical tools for answering this question. We will present various common learning algorithms and prove theoretical guarantees about them. Topics include classical asymptotics, method of moments, generalization bounds via uniform convergence, kernel methods, online learning, and multi-armed bandits. Prerequisites: A solid background in linear algebra and probability theory, statistics and machine learning (STATS 315A or CS 229). Convex optimization (EE 364A) is helpful but not required.
Same as: STATS 231

CS 231A. Computer Vision: From 3D Reconstruction to Recognition. 3-4 Units.
(Formerly 223B) An introduction to the concepts and applications in computer vision. Topics include: cameras and projection models, low-level image processing methods such as filtering and edge detection; mid-level vision topics such as segmentation and clustering; shape reconstruction from stereo, as well as high-level vision tasks such as object recognition, scene recognition, face detection and human motion categorization. Prerequisites: linear algebra, basic probability and statistics.

CS 231N. Convolutional Neural Networks for Visual Recognition. 3-4 Units.
Computer Vision has become ubiquitous in our society, with applications innsearch, image understanding, apps, mapping, medicine, drones, andnself-driving cars. Core to many of these applications are the tasks of image classification, localization and detection. This course is a deep dive into details of neural network architectures with a focus on learning end-to-end models for these tasks, particularly image classification. During the 10-week course, students will learn to implement, train and debug their own neural networks and gain a detailed understanding of cutting-edge research in computer vision. The final assignment will involve training a multi-million parameter convolutional neural network and applying it on the largest image classification dataset (ImageNet). We will focus on teaching how to set up the problem of image recognition, the learning algorithms (e.g. backpropagation), practical engineering tricks for training and fine-tuning the networks and guide the students through hands-on assignments and a final course project. Much of the background and materials of this course will be drawn from the ImageNet Challenge: http://image-net.org/challenges/LSVRC/2014/index. Prerequisites: Proficiency in Python; familiarity with C/C++; CS 131 and CS 229 or equivalents; Math 21 or equivalent, linear algebra.

CS 232. Digital Image Processing. 3 Units.
Image sampling and quantization color, point operations, segmentation, morphological image processing, linear image filtering and correlation, image transforms, eigenimages, multiresolution image processing, noise reduction and restoration, feature extraction and recognition tasks, image registration. Emphasis is on the general principles of image processing. Students learn to apply material by implementing and investigating image processing algorithms in Matlab and optionally on Android mobile devices. Term project. Recommended: EE261, EE278.
Same as: EE 368

CS 233. Geometric and Topological Data Analysis. 3 Units.
Mathematical computational tools for the analysis of data with geometric content, such images, videos, 3D scans, GPS traces -- as well as for other data embedded into geometric spaces. Global and local geometry descriptors allowing for various kinds of invariances. The rudiments of computational topology and persistent homology on sampled spaces. Clustering and other unsupervised techniques. Spectral methods for geometric data analysis. Non-linear dimensionality reduction. Alignment, matching, and map computation between geometric data sets. Function spaces and functional maps.Networks of data sets and joint analysis for segmentation and labeling. The emergence of abstractions or concepts from data. Prerequisites: discrete algorithms at the level of 161; linear algebra at the level of CME103.
Same as: CME 251

CS 234. Reinforcement Learning. 3 Units.
To realize the dreams and impact of AI requires autonomous systems that learn to make good decisions. Reinforcement learning is one powerful paradigm for doing so, and it is relevant to an enormous range of tasks, including robotics, game playing, consumer modeling and healthcare. This class will briefly cover background on Markov decision processes and reinforcement learning, before focusing on some of the central problems, including scaling up to large domains and the exploration challenge. One key tool for tackling complex RL domains is deep learning and this class will include at least one homework on deep reinforcement learning. Prerequisites: proficiency in python, CS 229 or equivalents or permission of the instructor; linear algebra, basic probability.

CS 238. Decision Making under Uncertainty. 3-4 Units.
This course is designed to increase awareness and appreciation for why uncertainty matters, particularly for aerospace applications. Introduces decision making under uncertainty from a computational perspective and provides an overview of the necessary tools for building autonomous and decision-support systems. Following an introduction to probabilistic models and decision theory, the course will cover computational methods for solving decision problems with stochastic dynamics, model uncertainty, and imperfect state information. Topics include: Bayesian networks, influence diagrams, dynamic programming, reinforcement learning, and partially observable Markov decision processes. Applications cover: air traffic control, aviation surveillance systems, autonomous vehicles, and robotic planetary exploration. Prerequisites: basic probability and fluency in a high-level programming language.
Same as: AA 228

CS 239. Advanced Topics in Sequential Decision Making. 3-4 Units.
Survey of recent research advances in intelligent decision making for dynamic environments from a computational perspective. Efficient algorithms for single and multiagent planning in situations where a model of the environment may or may not be known. Partially observable Markov decision processes, approximate dynamic programming, and reinforcement learning. New approaches for overcoming challenges in generalization from experience, exploration of the environment, and model representation so that these methods can scale to real problems in a variety of domains including aerospace, air traffic control, and robotics. Students are expected to produce an original research paper on a relevant topic. Prerequisites: AA 228/CS 238 or CS 221.
Same as: AA 229

CS 240. Advanced Topics in Operating Systems. 3 Units.
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.

CS 241. Embedded Systems Workshop. 3 Units.
Project-centric building hardware and software for embedded computing systems. Students work on an existing project of their own or join one of these projects. Syllabus topics will be determined by the needs of the enrolled students and projects. Examples of topics include: interrupts and concurrent programming, deterministic timing and synchronization, state-based programming models, filters, frequency response, and high-frequency signals, low power operation, system and PCB design, security, and networked communication. Prerequisite: CS107 (or equivalent).
Same as: EE 285

CS 242. Programming Languages. 3 Units.
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.

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

CS 244. Advanced Topics in Networking. 3-4 Units.
Classic papers, new ideas, and research papers in networking. Architectural principles: why the Internet was designed this way? Congestion control. Wireless and mobility; software-defined networks (SDN) and network virtualization; content distribution networks; packet switching; data-center networks. Prerequisite: 144 or equivalent.

CS 244C. Readings and Projects in Distributed Systems. 3-6 Units.
Companion project option for 244B. Corequisite: 244B.

CS 245. Database Systems Principles. 3 Units.
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.

CS 246. Mining Massive Data Sets. 3-4 Units.
The course will discuss data mining and machine learning algorithms for analyzing very large amounts of data. The emphasis will be on Map Reduce as a tool for creating parallel algorithms that can process very large amounts of data. Topics include: Frequent itemsets and Association rules, Near Neighbor Search in High Dimensional Data, Locality Sensitive Hashing (LSH), Dimensionality reduction, Recommender Systems, Clustering, Link Analysis, Large-scale machine learning, Data streams, Analysis of Social-network Graphs, and Web Advertising. Prerequisites: At lease one of CS107 or CS145; At least one of CS109 or STAT116, or equivalent.

CS 246H. Mining Massive Data Sets Hadoop Lab. 1 Unit.
Supplement to CS 246 providing additional material on Hadoop. Students will learn how to implement data mining algorithms using Hadoop, how to implement and debug complex MapReduce jobs in Hadoop, and how to use some of the tools in the Hadoop ecosystem for data mining and machine learning. Topics: Hadoop, MapReduce, HDFS, combiners, secondary sort, distributed cache, SQL on Hadoop, Hive, Cloudera ML/ Oryx, Mahout, Hadoop streaming, implementing Hadoop jobs, debugging Hadoop jobs, TF-IDF, Pig, Sqoop, Oozie, HBase, Impala. Prerequisite: CS 107 or equivalent.

CS 247. Human-Computer Interaction Design Studio. 3-4 Units.
Project-based focus on interaction design process, especially early-stage design and rapid prototyping. Methods used in interaction design including needs analysis, user observation, sketching, concept generation, scenario building, and evaluation. Prerequisites: 147 or equivalent background in design thinking; 106B or equivalent background in programming. Recommended: CS 142 or equivalent background in web programming. Enrollment limited to 40 students based on an application to be given out the first day of class.

CS 247L. Human Computer Interaction Technology Laboratory. 1 Unit.
Hands-on introduction to contemporary HCI technologies. Interaction design with Adobe Flash, mobile development, physical computing, and web applications. Corequisite: 247.

CS 248. Interactive Computer Graphics. 3-4 Units.
This is the second course in the computer graphics sequence, and as such it assumes a strong familiarity with rendering and image creation. The course has a strong focus on computational geometry, animation, and simulation. Topics include splines, implicit surfaces, geometric modeling, collision detection, animation curves, particle systems and crowds, character animation, articulation, skinning, motion capture and editing, rigid and deformable bodies, and fluid simulation. As a final project, students implement an interactive video game utilizing various concepts covered in the class. Games may be designed on mobile devices, in a client/server/browser environment, or on a standard personal computer. Prerequisite: CS148.

CS 250. Error Correcting Codes: Theory and Applications. 3 Units.
Introduction to the theory of error correcting codes, emphasizing diverse applications throughout computer science and engineering. Topics include basic bounds on error correcting codes; constructions like Reed-Solomon, Reed-Muller, and expander codes; list-decoding, list-recovery and locality. Applications include communication, storage, complexity theory, pseudorandomness, cryptography, streaming algorithms, group testing, and com-pressed sensing. Prerequisites: Linear algebra, basic probability (at the level of, say, CS109, CME106 or EE178), and ¿mathematical maturity¿ (students will be asked to write proofs). Familiarity with finite fields will be helpful but not required.
Same as: EE 387

CS 251. Bitcoin and Crypto Currencies. 3 Units.
For advanced undergraduates and for graduate students. The potential applications for Bitcoin-like technologies is enormous. The course will cover the technical aspects of crypto-currencies, blockchain technologies, and distributed consensus. Students will learn how these systems work and how to engineer secure software that interacts with the Bitcoin network and other crypto currencies. Prerequisite: CS110. Recommended: CS255.

CS 254. Computational Complexity. 3 Units.
An introduction to computational complexity theory. Topics include the P versus NP problem; diagonalization; space complexity: PSPACE, Savitch's theorem, and NL=coNL; counting problems and #P-completeness; circuit complexity; pseudorandomness and derandomization; complexity of approximation; quantum computing; complexity barriers. Prerequisites: 154 or equivalent; mathematical maturity.

CS 255. Introduction to Cryptography. 3 Units.
For advanced undergraduates and graduate students. Theory and practice of cryptographic techniques used in computer security. Topics: encryption (symmetric and public key), digital signatures, data integrity, authentication, key management, PKI, zero-knowledge protocols, and real-world applications. Prerequisite: basic probability theory.

CS 257. Logic and Artificial Intelligence. 2-4 Units.
This is a course at the intersection of philosophical logic and artificial intelligence. The goal of the course is to understand the role that expressive logical frameworks might play in AI, and to gain a deeper understanding of how different logical systems relate, and what features of a logical system could make it useful for representation and/or reasoning. Specific themes may include: 1. Tradeoff between complexity and expressivity 2. Capturing subtle reasoning about agent mental states 3. Defeasibility, causality, and the relation between logic and probability 4. Logical formalizations of legal and normative reasoning 5. Combining statistical learning and inference with rich logical structure 6. Logical systems close to the structure of natural language ("natural logics"). Prerequisites: It is expected that students already have a solid background in logic. Phil 151 is ideal, but Phil 150 or CS 157 would be acceptable, with the understanding that there may be some catching up to do. 2 unit option for PhD students only.
Same as: PHIL 356C

CS 261. Optimization and Algorithmic Paradigms. 3 Units.
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.

CS 263. Algorithms for Modern Data Models. 3 Units.
We traditionally think of algorithms as running on data available in a single location, typically main memory. In many modern applications including web analytics, search and data mining, computational biology, finance, and scientific computing, the data is often too large to reside in a single location, is arriving incrementally over time, is noisy/uncertain, or all of the above. Paradigms such as map-reduce, streaming, sketching, Distributed Hash Tables, Bulk Synchronous Processing, and random walks have proved useful for these applications. This course will provide an introduction to the design and analysis of algorithms for these modern data models. Prerequisite: Algorithms at the level of CS 261.
Same as: MS&E 317

CS 264. Beyond Worst-Case Analysis. 3 Units.
This course is motivated by problems for which the traditional worst-case analysis of algorithms fails to differentiate meaningfully between different solutions, or recommends an intuitively "wrong" solution over the "right" one. This course studies systematically alternatives to traditional worst-case analysis that nevertheless enable rigorous and robust guarantees on the performance of an algorithm. Topics include: instance optimality; smoothed analysis; parameterized analysis and condition numbers; models of data (pseudorandomness, locality, diffuse adversaries, etc.); average-case analysis; robust distributional analysis; resource augmentation; planted and semi-random graph models. Motivating problems will be drawn from online algorithms, online learning, constraint satisfaction problems, graph partitioning, scheduling, linear programming, hashing, machine learning, and auction theory. Prerequisites: CS161 (required). CS261 is recommended but not required.

CS 265. Randomized Algorithms and Probabilistic Analysis. 3 Units.
Randomness pervades the natural processes around us, from the formation of networks, to genetic recombination, to quantum physics. Randomness is also a powerful tool that can be leveraged to create algorithms and data structures which, in many cases, are more efficient and simpler than their deterministic counterparts. This course covers the key tools of probabilistic analysis, and application of these tools to understand the behaviors of random processes and algorithms. Emphasis is on theoretical foundations, though we will apply this theory broadly, discussing applications in machine learning and data analysis, networking, and systems. Topics include tail bounds, the probabilistic method, Markov chains, and martingales, with applications to analyzing random graphs, metric embeddings, random walks, and a host of powerful and elegant randomized algorithms. Prerequisites: CS 161 and STAT 116, or equivalents and instructor consent.
Same as: CME 309

CS 267. Graph Algorithms. 3 Units.
An introduction to advanced topics in graph algorithms. Focusing on a variety of graph problems, the course will explore topics such as small space graph data structures, approximation algorithms, dynamic algorithms, and algorithms for special graph classes. Topics include: approximation algorithms for shortest paths and graph matching, distance oracles, graph spanners, cliques and graph patterns, dynamic algorithms, graph coloring, algorithms for planar graphs. Prerequisites: 161 or the equivalent mathematical maturity.

CS 268. Geometric Algorithms. 3 Units.
Techniques for design and analysis of efficient geometric algorithms for objects in 2-, 3-, and higher dimensions. Topics: convexity, triangulations and simplicial complexes, sweeping, partitioning, and point location. Voronoi/Delaunay diagrams and their properties. Arrangements of curves and surfaces. Intersection and visibility problems. Geometric searching and optimization. Random sampling methods. Range searching. Impact of numerical issues in geometric computation. Example applications to robotic motion planning, visibility preprocessing and rendering in graphics, and model-based recognition in computer vision. Prerequisite: discrete algorithms at the level of 161. Recommended: 164.

CS 269G. Almost Linear Time Graph Algorithms. 3 Units.
Over the past decade there has been an explosion in activity in designing new provably efficient fast graph algorithms. Leveraging techniques from disparate areas of computer science and optimization researchers have made great strides on improving upon the best known running times for fundamental optimization problems on graphs, in many cases breaking long-standing barriers to efficient algorithm design. In this course we will survey these results and cover the key algorithmic tools they leverage to achieve these breakthroughs. Possible topics include but are not limited to, spectral graph theory, sparsification, oblivious routing, local partitioning, Laplacian system solving, and maximum flow. Prerequisites: calculus and linear algebra.
Same as: MS&E 313

CS 269I. Incentives in Computer Science. 3 Units.
Many 21st-century computer science applications require the design of software or systems that interact with multiple self-interested participants. This course will provide students with the vocabulary and modeling tools to reason about such design problems. Emphasis will be on understanding basic economic and game theoretic concepts that are relevant across many application domains, and on case studies that demonstrate how to apply these concepts to real-world design problems. Topics include auction and contest design, equilibrium analysis, cryptocurrencies, design of networks and network protocols, reputation systems, social choice, and social network analysis. Case studies include BGP routing, Bitcoin, eBay's reputation system, Facebook's advertising mechanism, Mechanical Turk, and dynamic pricing in Uber/Lyft. Prerequisites: CS106B/X and CS161, or permission from the instructor.

CS 269O. Introduction to Optimization Theory. 3 Units.
Introduction of core algorithmic techniques and proof strategies that underlie the best known provable guarantees for minimizing high dimensional convex functions. Focus on broad canonical optimization problems and survey results for efficiently solving them, ultimately providing the theoretical foundation for further study in optimization. In particular, focus will be on first-order methods for both smooth and non-smooth convex function minimization as well as methods for structured convex function minimization, discussing algorithms such as gradient descent, accelerated gradient descent, mirror descent, Newton's method, interior point methods, and more. Prerequisite: multivariable calculus and linear algebra.
Same as: MS&E 213

CS 270. Modeling Biomedical Systems: Ontology, Terminology, Problem Solving. 3 Units.
Methods for modeling biomedical systems and for building model-based software systems. Emphasis is on intelligent systems for decision support and Semantic Web applications. Topics: knowledge representation, controlled terminologies, ontologies, reusable problem solvers, and knowledge acquisition. Students learn about current trends in the development of advanced biomedical software systems and acquire hands-on experience with several systems and tools. Prerequisites: CS106A, basic familiarity with biology.
Same as: BIOMEDIN 210

CS 272. Introduction to Biomedical Informatics Research Methodology. 3-5 Units.
Capstone Biomedical Informatics (BMI) experience. 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. Issues related to research reproducibility. Guest lectures from professional biomedical informatics systems builders on issues related to the process of project management. Software engineering basics. Because the team projects start in the first week of class, attendance that week is strongly recommended. Prerequisites: BIOMEDIN 210 or 211 or 214 or 217. Preference to BMI graduate students. Consent of instructor required.
Same as: BIOE 212, BIOMEDIN 212, GENE 212

CS 273A. A Computational Tour of the Human Genome. 3 Units.
Introduction to computational biology through an informatic exploration of the human genome. Topics include: genome sequencing (technologies, assembly, personalized sequencing); functional landscape (genes, gene regulation, repeats, RNA genes, epigenetics); genome evolution (comparative genomics, ultraconservation, co-option). Additional topics may include population genetics, personalized genomics, and ancient DNA. Course includes primers on molecular biology, the UCSC Genome Browser, and text processing languages. Guest lectures from genomic researchers. No prerequisites. Seehttp://cs273a.stanford.edu/.
Same as: BIOMEDIN 273A, DBIO 273A

CS 273B. Deep Learning in Genomics and Biomedicine. 3 Units.
Recent breakthroughs in high-throughput genomic and biomedical data are transforming biological sciences into "big data" disciplines. In parallel, progress in deep neural networks are revolutionizing fields such as image recognition, natural language processing and, more broadly, AI. This course explores the exciting intersection between these two advances. The course will start with an introduction to deep learning and overview the relevant background in genomics and high-throughput biotechnology, focusing on the available data and their relevance. It will then cover the ongoing developments in deep learning (supervised, unsupervised and generative models) with the focus on the applications of these methods to biomedical data, which are beginning to produced dramatic results. In addition to predictive modeling, the course emphasizes how to visualize and extract interpretable, biological insights from such models. Recent papers from the literature will be presented and discussed. Students will be introduced to and work with popular deep learning software frameworks. Students will work in groups on a final class project using real world datasets. Prerequisites: College calculus, linear algebra, basic probability and statistics such as CS109, and basic machine learning such as CS229. No prior knowledge of genomics is necessary.
Same as: BIODS 237, BIOMEDIN 273B, GENE 236

CS 274. Representations and Algorithms for Computational Molecular Biology. 3-4 Units.
Topics: introduction to bioinformatics and computational biology, algorithms for alignment of biological sequences and structures, computing with strings, phylogenetic tree construction, hidden Markov models, basic structural computations on proteins, protein structure prediction, protein threading techniques, homology modeling, molecular dynamics and energy minimization, statistical analysis of 3D biological data, integration of data sources, knowledge representation and controlled terminologies for molecular biology, microarray analysis, machine learning (clustering and classification), and natural language text processing. Prerequisite: CS 106B; recommended: CS161; consent of instructor for 3 units.
Same as: BIOE 214, BIOMEDIN 214, GENE 214

CS 275. Translational Bioinformatics. 4 Units.
Computational methods for the translation of biomedical data into diagnostic, prognostic, and therapeutic applications in medicine. Topics: multi-scale omics data generation and analysis, utility and limitations of public biomedical resources, machine learning and data mining, issues and opportunities in drug discovery, and mobile/digital health solutions. Case studies and course project. Prerequisites: programming ability at the level of CS 106A and familiarity with biology and statistics.
Same as: BIOE 217, BIOMEDIN 217

CS 275A. Symbolic Musical Information. 2-4 Units.
Focus on symbolic data for music applications including advanced notation systems, optical music recognition, musical data conversion, and internal structure of MIDI files.
Same as: MUSIC 253

CS 275B. Music Query, Analysis, and Style Simulation. 2-4 Units.
Leveraging off three synchronized sets of symbolic data resources for notation and analysis, the lab portion introduces students to the open-source Humdrum Toolkit for music representation and analysis. Issues of data content and quality as well as methods of information retrieval, visualization, and summarization are considered in class. Grading based primarily on student projects. Prerequisite: 253 or consent of instructor.
Same as: MUSIC 254

CS 276. Information Retrieval and Web Search. 3 Units.
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; Web search engines including crawling and indexing, link-based algorithms, web metadata, and question answering; distributed word representations. Prerequisites: CS 107, CS 109, CS 161.
Same as: LINGUIST 286

CS 279. Computational Biology: Structure and Organization of Biomolecules and Cells. 3 Units.
Computational techniques for investigating and designing the three-dimensional structure and dynamics of biomolecules and cells. These computational methods play an increasingly important role in drug discovery, medicine, bioengineering, and molecular biology. Course topics include protein structure prediction, protein design, drug screening, molecular simulation, cellular-level simulation, image analysis for microscopy, and methods for solving structures from crystallography and electron microscopy data. Prerequisites: elementary programming background (CS 106A or equivalent) and an introductory course in biology or biochemistry.
Same as: BIOE 279, BIOMEDIN 279, BIOPHYS 279, CME 279

CS 294A. Research Project in Artificial Intelligence. 3 Units.
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.

CS 294H. Research Project in Human-Computer Interaction. 3 Units.
Student teams under faculty supervision work on research and implementationnof a large project in HCI. State-of-the-art methods related to the problemndomain. Prerequisites CS 377, 147, 247, or permission from instructor.

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

CS 294W. Writing Intensive Research Project in Computer Science. 3 Units.
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.

CS 295. Software Engineering. 3 Units.
Software specification, testing, and verification. Emphasis is on modern technology for developing reliable software at a reasonable cost. Assignments focus on applying these techniques to realistic software systems. Prerequisites: 108. Recommended a project course such as 140, 143, or 145.

CS 298. Seminar on Teaching Introductory Computer Science. 1 Unit.
Faculty, undergraduates, and graduate students interested in teaching discuss topics raised by teaching computer science at the introductory level. Prerequisite: consent of instructor.
Same as: EDUC 298

CS 300. Departmental Lecture Series. 1 Unit.
Priority given to first-year Computer Science Ph.D. students. CS Masters students admitted if space is available. Presentations by members of the department faculty, each describing informally his or her current research interests and views of computer science as a whole.

CS 309. Industrial Lectureships in Computer Science. 1 Unit.
Guest computer scientist. By arrangement. May be repeated for credit.

CS 309A. Cloud Computing. 1 Unit.
For science, engineering, computer science, business, education, medicine, and law students. Cloud computing is bringing information systems out of the back office and making it core to the entire economy. Furthermore with the advent of smarter machines cloud computing will be integral to building a more precision planet. This class is intended for all students who want to begin to understand the implications of this technology. Guest industry experts are public company CEOs who are either delivering cloud services or using cloud services to transform their businesses.

CS 315A. Parallel Computer Architecture and Programming. 3 Units.
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.

CS 315B. Parallel Computing Research Project. 3 Units.
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.

CS 316. Advanced Multi-Core Systems. 3 Units.
In-depth coverage of the architectural techniques used in modern, multi-core chips for mobile and server systems. Advanced processor design techniques (superscalar cores, VLIW cores, multi-threaded cores, energy-efficient cores), cache coherence, memory consistency, vector processors, graphics processors, heterogeneous processors, and hardware support for security and parallel programming. Students will become familiar with complex trade-offs between performance-power-complexity and hardware-software interactions. A central part of CS316 is a project on an open research question on multi-core technologies. Prerequisites: EE 180 (formerly 108B). Recommended: CS 149, EE 282.

CS 319. Topics in Digital Systems. 3 Units.
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 323. Automated Reasoning: Theory and Applications. 3-4 Units.
Intelligent computer agents must reason about complex, uncertain, and dynamic environments. This course is a graduate level introduction to automated reasoning techniques and their applications, covering logical and probabilistic approaches. Topics include: logical and probabilistic foundations, backtracking strategies and algorithms behind modern SAT solvers, stochastic local search and Markov Chain Monte Carlo algorithms, variational techniques, classes of reasoning tasks and reductions, and applications.

CS 327A. Advanced Robotic Manipulation. 3 Units.
Advanced control methodologies and novel design techniques for complex human-like robotic and bio mechanical systems. Class covers the fundamentals in operational space dynamics and control, elastic planning, human motion synthesis. Topics include redundancy, inertial properties, haptics, simulation, robot cooperation, mobile manipulation, human-friendly robot design, humanoids and whole-body control. Additional topcs in emerging areas are presented by groups of students at the end-of-quarter mini-symposium. Prerequisites: 223A or equivalent.

CS 328. Topics in Computer Vision. 3 Units.
Fundamental issues of, and mathematical models for, computer vision. Sample topics: camera calibration, texture, stereo, motion, shape representation, image retrieval, experimental techniques. May be repeated for credit. Prerequisites: 205, 223B, or equivalents.

CS 329. Topics in Artificial Intelligence. 3 Units.
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 329M. Topics in Artificial Intelligence: Algorithms of Advanced Machine Learning. 3 Units.
This advanced graduate course explores in depth several important classes of algorithms in modern machine learning. We will focus on understanding the mathematical properties of these algorithms in order to gain deeper insights on when and why they perform well. We will also study applications of each algorithm on interesting, real-world settings. Topics include: spectral clustering, tensor decomposition, Hamiltonian Monte Carlo, adversarial training, and variational approximation. Students will learn mathematical techniques for analyzing these algorithms and hands-on experience in using them. We will supplement the lectures with latest papers and there will be a significant research project component to the class. Prerequisites: Probability (CS 109), linear algebra (Math 113), machine learning (CS 229), and some coding experience.

CS 331B. Representation Learning in Computer Vision. 3 Units.
This course surveys recent developments in representation learning that are relevant to visual recognition and understanding tasks. In particular we will examine: 1) why representations matter; 2) classical and moderns methods of forming and learning representations in 2D and 3D computer vision; 3) how to close the loop between sensing and action for perception robotics; 4) how to connect visual-based representations with language; 5) methods for analyzing and visualizing representations. In addition to regular lectures and talks by invited speakers, we will read advanced papers on this topic, and carry out in-depth discussions of these papers as well as the students' own research projects.

CS 334A. Convex Optimization I. 3 Units.
Convex sets, functions, and optimization problems. The basics of convex analysis and theory of convex programming: optimality conditions, duality theory, theorems of alternative, and applications. Least-squares, linear and quadratic programs, semidefinite programming, and geometric programming. Numerical algorithms for smooth and equality constrained problems; interior-point methods for inequality constrained problems. Applications to signal processing, communications, control, analog and digital circuit design, computational geometry, statistics, machine learning, and mechanical engineering. Prerequisite: linear algebra such as EE263, basic probability.
Same as: CME 364A, EE 364A

CS 340. Topics in Computer Systems. 3-4 Units.
Topics vary every quarter, and may include advanced material being taught for the first time. May be repeated for credit.

CS 341. Project in Mining Massive Data Sets. 3 Units.
Team project in data-mining of very large-scale data, including the problem statement and implementation and evaluation of a solution. Teams consist of three students each, and they will meet regularly with a "coach" chosen from participating staff. Early lectures will cover the use of Amazon EC2 and certain systems like Hadoop and Hive. Occasional lectures thereafter will feature outside speakers, special topics of interest, and progress reports by the teams. Prerequisite: CS 246.

CS 344. Topics in Computer Networks. 3 Units.
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 345S. Data-intensive Systems for the Next 1000x. 3-4 Units.
The last decade saw enormous shifts in the design of large-scale data-intensive systems due to the rise of Internet services, cloud computing, and Big Data processing. Where will we see the next 1000x increases in scale and data volume, and how should data-intensive systems accordingly evolve? This course will critically examine a range of trends, including the Internet of Things, drones, smart cities, and emerging hardware capabilities, through the lens of software systems research and design. Students will perform a comparative analysis by reading and discussing cutting-edge research while performing their own original research. Prerequisites: Strong background in software systems, especially databases (CS 245) and distributed systems (CS 244B), and/or machine learning (CS 229). Undergraduates who have completed CS 245 are strongly encouraged to attend.

CS 348A. Computer Graphics: Geometric Modeling & Processing. 3-4 Units.
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, Bézier arcs and de Casteljau subdivision, continuity constraints, B-splines, tensor product, and triangular patch surfaces. Subdivision surfaces and multi-resolution representations of geometry. Representations of solids and conversions among them. Surface reconstruction from scattered data points. Geometry processing on meshes, including simplification and parameterization. Prerequisite: linear algebra at the level of CME103. Recommended: 248.

CS 348B. Computer Graphics: Image Synthesis Techniques. 3-4 Units.
Intermediate level, emphasizing high-quality image synthesis algorithms and systems issues in rendering. Topics include: Reyes and advanced rasterization, including motion blur and depth of field; ray tracing and physically based rendering; Monte Carlo algorithms for rendering, including direct illumination and global illumination; path tracing and photon mapping; surface reflection and light source models; volume rendering and subsurface scattering; SIMD and multi-core parallelism for rendering. Written assignments and programming projects. Prerequisite: 248 or equivalent. Recommended: Fourier analysis or digital signal processing.

CS 348C. Computer Graphics: Animation and Simulation. 3 Units.
Core mathematics and methods for computer animation and motion simulation. Traditional animation techniques. Physics-based simulation methods for modeling shape and motion: particle systems, constraints, rigid bodies, deformable models, collisions and contact, fluids, and fracture. Animating natural phenomena. Methods for animating virtual characters and crowds. Additional topics selected from data-driven animation methods, realism and perception, animation systems, motion control, real-time and interactive methods, and multi-sensory feedback. Recommended: CS 148 and/or 205A. Prerequisite: linear algebra.

CS 349. Topics in Programming Systems. 3 Units.
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 352. Pseudo-Randomness. 3-4 Units.
Pseudorandomness is the widely applicable theory of efficiently generating objects that look random, despite being constructed using little or no randomness. Since psudorandom objects can replace uniformly distributed ones (in a well-defined sense), one may view pseudorandomness as an extension of our understanding of randomness through the computational lens. We will study the basic tools pseudorandomness, such as limited independence, randomness extractors, expander graphs, and pseudorandom generators. We will also discuss the applications of pseudrandomness to derandomization, cryptography and more. We will cover classic result as well as cutting-edge techniques. Prerequisites: CS 154 and CS 161, or equivalents.

CS 358. Topics in Programming Language Theory. 3 Units.
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 359. Topics in the Theory of Computation. 3 Units.
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 359C. Topics in Theory of Computation: Classics of Cryptography. 3 Units.
This course will review some of the greatest discoveries in modern cryptography: zero-knowledge proofs, factoring algorithms, elliptic-curve cryptography, post-quantum cryptography, and more. Some of the topics we will cover have immediate practical applications. Other topics we will study for their potential future applications. And yet others we will study for the theoretical insights they provide. The course readings will be a combination of the original "classic" papers and more modern treatments of the same topics. Prerequisite: CS 255, an equivalent course, or permission of instructors.

CS 361. Introduction to Multidisciplinary Design Optimization. 3-4 Units.
Design of engineering systems within a formal optimization framework. Engineering often involves the synthesis of several disciplines, such as fluids, structures, and controls in aerospace systems. These disciplines interact in complex ways, making the optimization of the system design challenging. This course covers the mathematical and algorithmic fundamentals of optimization, including derivative and derivative-free approaches for both linear and non-linear problems, with an emphasis on multidisciplinary design optimization. Topics will also include quantitative methodologies for addressing various challenges, such as accommodating multiple objectives, handling uncertainty in evaluations, selecting design points for experimentation, and principled methods for optimization when evaluations are expensive. Applications range from the design of aircraft to automated vehicles. Prerequisites: some familiarity with probability, programming, and multivariable calculus.
Same as: AA 222

CS 369. Topics in Analysis of Algorithms. 3 Units.
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 369H. Hierarchies of Integer Programming Relaxations. 3 Units.
Mathematical programming relaxations of integer programming formulations are a popular way to apply convex optimization techniques to hard combinatorial optimization problems. Such relaxations can be made closer to their integer programming counterparts by adding constraints; a systematic way to achieve this is via hierarchies of relaxations. Several such hierarchies are well-studied in the literature: Lovasz-Schrijver, Sherali-Adams and the Parrilo-Lasserre sum-of-squares (SoS) hierarchy. Recently, these hierarchies have received a lot of attention due to their potential to make progress on long standing algorithmic questions, and connections to various other areas such as computational complexity, combinatorial and polynomial optimization, quantum computing, proof complexity and so on. In this course we will cover recent research results in this area for problems arising from optimization, machine learning, computational complexity and more, discussing both lower and upper bounds. Prerequisites: Mathematical maturity (required), exposure to algorithms (strongly recommended), and optimization (recommended).

CS 371. Computational Biology in Four Dimensions. 3 Units.
Cutting-edge research on computational techniques for investigating and designing the three-dimensional structure and dynamics of biomolecules, cells, and everything in between. These techniques, which draw on approaches ranging from physics-based simulation to machine learning, play an increasingly important role in drug discovery, medicine, bioengineering, and molecular biology. Course is devoted primarily to reading, presentation, discussion, and critique of papers describing important recent research developments. Prerequisite: CS 106A or equivalent, and an introductory course in biology or biochemistry. Recommended: some experience in mathematical modeling (does not need to be a formal course).
Same as: BIOMEDIN 371, BIOPHYS 371, CME 371

CS 373. Statistical and Machine Learning Methods for Genomics. 3 Units.
Introduction to statistical and computational methods for genomics. Sample topics include: expectation maximization, hidden Markov model, Markov chain Monte Carlo, ensemble learning, probabilistic graphical models, kernel methods and other modern machine learning paradigms. Rationales and techniques illustrated with existing implementations used in population genetics, disease association, and functional regulatory genomics studies. Instruction includes lectures and discussion of readings from primary literature. Homework and projects require implementing some of the algorithms and using existing toolkits for analysis of genomic datasets.
Same as: BIO 268, BIOMEDIN 245, GENE 245, STATS 345

CS 376. Human-Computer Interaction Research. 3-4 Units.
Prepares students to conduct original HCI research by reading and discussing seminal and cutting-edge research papers. Main topics are ubiquitous computing, social computing, and design and creation; breadth topics include HCI methods, programming, visualization, and user modeling. Student pairs perform a quarter-long research project. Prerequisites: For CS and Symbolic Systems undergraduates/masters students, an A- or better in CS 147 or CS 247. No prerequisite for PhD students or students outside of CS and Symbolic Systems.

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

CS 377C. Topics in HCI: Crowdsourcing and Social Computing. 3-4 Units.
This project-based class focuses on the design of social computing and crowdsourcing systems. Students will learn how to engage large groups of people online, from microtask crowdsourcing to the design of online communities. The course will cover best practices for system design such as motivating participation, ethical guidelines, agreement measures, and gold standards. Advanced topics such as expert and team-based crowdsourcing, incentive design, and complex crowd workflows will also be discussed. Students will learn about the application of crowdsourcing to other areas of computer science, and how the field relates to social psychology and organizational behavior. Prerequisite: CS 147.

CS 377D. Topics in Learning and Technology: d.compress - Designing Calm. 3 Units.
Contents of the course change each year. The course can be repeated. Stress silently but steadily damages physical and emotional well-being, relationships, productivity, and our ability to learn and remember. This highly experiential and project-oriented class will focus on designing interactive technologies to enable calm states of cognition, emotion, and physiology for better human health, learning, creativity and productivity.
Same as: EDUC 328A

CS 377E. Designing Solutions to Global Grand Challenges. 3-4 Units.
In this course we creatively apply information technologies to collectively attack Global Grand Challenges (e.g., global warming, rising healthcare costs and declining access, and ensuring quality education for all). Interdisciplinary student teams will carry out need-finding within a target domain, followed by brainstorming to propose a quarter long project. Teams will spend the rest of the quarter applying user-centered design methods to rapidly iterate through design, prototyping, and testing of their solutions. This course will interweave a weekly lecture with a weekly studio session where students apply the techniques hands-on in a small-scale, supportive environment.

CS 377U. Understanding Users. 3-4 Units.
This project-based class focuses on understanding the use of technology in the world. Students will learn generative and evaluative research methods to explore how systems are appropriated into everyday life in a quarter-long project where they design, implement and evaluate a novel mobile application. Quantitative (e.g. A/B testing, instrumentation, analytics, surveys) and qualitative (e.g. diary studies, contextual inquiry, ethnography) methods and their combination will be covered along with practical experience applying these methods in their project. Prerequisites: CS 147, 193A/193P (or equivalent mobile programming experience).

CS 379. Interdisciplinary Topics. 3 Units.
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 379C. Computational Models of the Neocortex. 3 Units.
This course emphasizes approaches to scaling the technologies of computer science and systems neuroscience to take advantage of the exponential trend in computational power known as Moore's Law. Modern methods in signal processing and machine learning are combined with technologies for managing large datasets common in industry. Classes feature scientists presenting novel approaches for analyzing the structure and function of complex neural circuits. Grading is based on class participation (30%), a project proposal due at midterm (20%), and a final project demonstration and report due by the end of finals (50%). Team projects are encouraged, especially multi-disciplinary collaborations. Prerequisites are basic high-school biology, good math skills and familiarity with machine learning. Some background in computer vision and signal processing is important for projects in structural analysis. Familiarity with modern artificial neural network technologies is a plus for projects in functional analysis. For more detail, see http://www.stanford.edu/class/cs379c/ with special attention to the CALENDAR and DISCUSSION tabs from past classes available by following the ARCHIVES link.

CS 390A. Curricular Practical Training. 1 Unit.
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.

CS 390B. Curricular Practical Training. 1 Unit.
Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in internship work and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. 390A,B,C may each be taken once.

CS 390C. Curricular Practical Training. 1 Unit.
Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in internship work and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. 390A,B,C may each be taken once.

CS 390D. Part-time Curricular Practical Training. 1 Unit.
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. Students in F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT). 390A, B, C, D may each be taken once.

CS 390P. Part-time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390Q. Part-Time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390R. Part-Time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390S. Part-Time CPT. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390T. Part-Time CPT. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390U. Part-Time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390V. Part-time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390W. Part-time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research 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. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 393. Computer Laboratory. 1-9 Unit.
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.

CS 395. Independent Database Project. 1-6 Unit.
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.

CS 399. Independent Project. 1-9 Unit.
Letter grade only.

CS 399P. Independent Project. 1-9 Unit.
Graded satisfactory/no credit.

CS 402. Beyond Bits and Atoms: Designing Technological Tools. 3-4 Units.
Practicum in designing and building technology-enabled curricula and hands-on learning environments. Students use software toolkits and state-of-the-art fabrication machines to design educational software, educational toolkits, and tangible user interfaces. The course will focus on designing low-cost technologies, particularly for urban school in the US and abroad. We will explore theoretical and design frameworks from the constructionist learning perspective, critical pedagogy, interaction design for children.
Same as: EDUC 236

CS 402L. Beyond Bits and Atoms - Lab. 1-3 Unit.
This course is a hands-on lab in the prototyping and fabrication of tangible technologies, with a special focus in learning and education. We will learn how to use state-of-the-art fabrication machines (3D printers, 3D scanners, laser cutters, routers) to design educational toolkits, educational toys, science kits, and tangible user interfaces. A special focus of the course will be to design low-cost technologies, particularly for urban school in the US and abroad.
Same as: EDUC 211

CS 424M. Learning Analytics and Computational Modeling in Social Science. 3-4 Units.
Computational modeling and data-mining are dramatically changing the physical sciences, and more recently also the social and behavioral sciences. Traditional analysis techniques are insufficient to investigate complex dynamic social phenomena as social networks, online gaming, diffusion of innovation, opinion dynamics, classroom behavior, and other complex adaptive systems. In this course, we will learn about how modeling, network theory, and basic data-mining can support research in cognitive, and social sciences, in particular around issues of learning, cognitive development, and educational policy.
Same as: EDUC 390

CS 428. Computation and cognition: the probabilistic approach. 3 Units.
This course will introduce the probabilistic approach to cognitive science, in which learning and reasoning are understood as inference in complex probabilistic models. Examples will be drawn from areas including concept learning, causal reasoning, social cognition, and language understanding. Formal modeling ideas and techniques will be discussed in concert with relevant empirical phenomena.
Same as: PSYCH 204

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

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

CS 448H. Topics in Computer Graphics: Agile Hardware Design. 3 Units.
Topic changes each quarter. Recent topics: computational photography, data visualization, character animation, virtual worlds, graphics architectures, advanced rendering. See http://graphics.stanford.edu/courses for offerings and prerequisites. May be repeated for credit.

CS 448I. Computational Imaging and Display. 3 Units.
Spawned by rapid advances in optical fabrication and digital processing power, a new generation of imaging technology is emerging: computational cameras at the convergence of applied mathematics, optics, and high-performance computing. Similar trends are observed for modern displays pushing the boundaries of resolution, contrast, 3D capabilities, and immersive experiences through the co-design of optics, electronics, and computation. This course serves as an introduction to the emerging field of computational imaging and displays. Students will learn to master bits and photons.
Same as: EE 367

CS 468. Topics in Geometric Algorithms: Machine Learning for 3D Data. 3 Units.
Contents of this course change with each offering. Past offerings have included geometric matching, surface reconstruction, collision detection, computational topology. May be repeated for credit. Winter 2013/14 iteration will cover Computational Symmetry & Regularity and spring quarter 2013/14 will cover data-driven shape analysis. Prerequisites: Math 52 or equivalent, basic coding.

CS 476A. Music, Computing, Design I: Art of Design for Computer Music. 3-4 Units.
Creative design for computer music software. Programming, audiovisual design, as well as software design for musical tools, instruments, toys, and games. Provides paradigms and strategies for designing and building music software, with emphases on interactive systems, aesthetics, and artful product design. Course work includes several programming assignments and a "design+implement" final project. Prerequisite: experience in C/C++ and/or Java.See https://ccrma.stanford.edu/courses/256a/.
Same as: MUSIC 256A

CS 476B. Music, Computing, Design II: Virtual and Augmented Reality for Music. 3-4 Units.
Aesthetics, design, and exploration of creative musical applications of virtual reality (VR) and augmented reality (AR), centered around VR and mobile technologies. Comparison between AR, VR, and traditional software design paradigms for music. Topics include embodiment, interaction design, novel instruments, social experience, software design + prototyping. Prerequisite: MUSIC 256A / CS 476A.
Same as: MUSIC 256B

CS 499. Advanced Reading and Research. 1-15 Unit.
Letter grade only. Advanced reading and research for CS graduate students. Register using the section number associated with the instructor. Prerequisite: consent of instructor.

CS 499P. Advanced Reading and Research. 1-15 Unit.
Graded satisfactory/no credit. Advanced reading and research for CS graduate students. Register using the section number associated with the instructor. Prerequisite: consent of instructor.

CS 547. Human-Computer Interaction Seminar. 1 Unit.
Weekly speakers on human-computer interaction topics. May be repeated for credit.

CS 571. Surgical Robotics Seminar. 1 Unit.
Surgical robots developed and implemented clinically on varying scales. Seminar goal is to expose students from engineering, medicine, and business to guest lecturers from academia and industry. Engineering and clinical aspects connected to design and use of surgical robots, varying in degree of complexity and procedural role. May be repeated for credit.
Same as: ME 571

CS 801. TGR Project. 0 Units.
.

CS 802. TGR Dissertation. 0 Units.
.