COMPUTER SCIENCE (CS)

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. 3-4 Units.
Aimed at non-CS undergraduate and graduate students who want to learn the basics of big data tools and techniques 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 collecting and analyzing large volumes of data. At the same time, it is surprisingly easy to make errors or come to false conclusions from data analysis alone. This course provides a broad and practical introduction to big data: data analysis techniques including databases, data mining, and machine learning; data analysis tools including spreadsheets, relational databases and SQL, Python, and R; data visualization techniques and tools; pitfalls in data collection and analysis. Tools and techniques are hands-on but at a cursory level, providing a basis for future exploration and application. Prerequisites: comfort with basic logic and mathematical concepts, along with high school AP computer science, CS106A, or other equivalent programming experience.

CS 103. Mathematical Foundations of Computing. 3-5 Units.
What are the theoretical limits of computing power? What problems can be solved with computers? Which ones cannot? And how can we reason about the answers to these questions with mathematical certainty? This course explores the answers to these questions and serves as an introduction to discrete mathematics, computability theory, and complexity theory. At the completion of the course, students will feel comfortable writing mathematical proofs, reasoning about discrete structures, reading and writing statements in first-order logic, and working with mathematical models of computing devices. Throughout the course, students will gain exposure to some of the most exciting mathematical and philosophical ideas of the late nineteenth and twentieth centuries. Specific topics covered include formal mathematical proofwriting, propositional and first-order logic, set theory, binary relations, functions (injections, surjections, and bijections), cardinality, basic graph theory, the pigeonhole principle, mathematical induction, finite automata, regular expressions, the Myhill-Nerode theorem, context-free grammars, Turing machines, decidable and recognizable languages, self-reference and undecidability, verifiers, and the P versus NP question. Students with significant proofwriting experience are encouraged to instead take CS154. Students interested in extra practice and support with the course are encouraged to concurrently enroll in CS103A. Prerequisite: CS106B or equivalent. CS106B may be taken concurrently with CS103.

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. Emphasis is on good programming style and the built-in facilities of respective languages. No prior programming experience required. Summer quarter enrollment is limited. Alternative versions of CS106A may be available which cover most of the same material but in different programming languages.
Same as: ENGR 70A

CS 106AJ. 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. Uses the JavaScript programming language. Emphasis is on good programming style and the built-in facilities of the JavaScript language. No prior programming experience required. This course covers most of the same material as CS106A Section 1 in Java and CS 106A Section 3 in Python, but this course uses the JavaScript programming language. To enroll in this class, enroll in CS 106A Section 2 for Fall Qtr. May be taken for 3 units by grad students.

CS 106AP. Programming Methodology in Python. 3-5 Units.
Introduction to the engineering of computer applications in Python, emphasizing modern software engineering principles: decomposition, abstraction, testing and good programming style. This course covers most of the same material as the other versions of CS106A, but using the Python programming language which is popular for general engineering and web development. Required readings will all be available for free on the web. Students are encouraged to bring a laptop to lecture to do the live exercises which are integrated with lecture. No prior programming experience required. To enroll in this class, enroll in CS 106A Section 3.

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 106E. Practical Exploration of Computing. 3-4 Units.
A follow up class to CS106A for non-majors which will both provide practical web programming skills and cover essential computing topics including computer security and privacy. Additional topics will include digital representation of images and music, an exploration of how the Internet works, and a look at the internals of the computer. Students taking the course for 4 units will be required to carry out supplementary programming assignments in addition to the course's regular assignments. Prerequisite: 106A or equivalent.

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. Coding for Social Good. 1 Unit.
Survey course on applications of fundamental computer science concepts from CS 106B/X to problems in the social good space (such as health, government, education, and environment). Each week consists of in-class activities designed by student groups, local tech companies, and nonprofits. Introduces students to JavaScript and the basics of web development. Topics have included mental health chatbots, tumor classification with basic machine learning, sentiment analysis of tweets on refugees, and storytelling through virtual reality. 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. Significant amount of additional advanced material and substantially more challenging projects. Some projects may relate to CS department research. 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. There is a $50 required lab fee.

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 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 11SI. How to Make VR: Introduction to Virtual Reality Design and Development. 2 Units.
In this hands-on, experiential course, students will design and develop virtual reality applications. You'll learn how to use the Unity game engine, the most popular platform for creating immersive applications. The class will teach the design best-practices and the creation pipeline for VR applications, and will include tangents that explore sister fields such as augmented reality and 360 video. Students will work in groups to present a final project in building an application for the Oculus Go headset. Enrollment is limited and by rolling application only. Prerequisite: CS 106A or equivalent.

CS 124. From Languages to Information. 3-4 Units.
Extracting meaning, information, and structure from human language text, speech, web pages, social networks. Methods include: string algorithms, edit distance, language modeling, the noisy channel, machine learning classifiers, inverted indices, collaborative filtering, neural embeddings, PageRank. Applications such as question answering, sentiment analysis, information retrieval, text classification, social network models, spell checking, recommender systems, chatbots. Prerequisites: 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 projects such that they will get to implement cutting-edge computer vision algorithms. Prerequisites: Students should be familiar with Python (i.e. have programmed in Python 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 140E. Operating systems design and implementation. 3-4 Units.
This is an experimental course offering. Students will implement a simple, clean operating system (virtual memory, processes, file system) on a rasberry pi computer and use the result to run a variety of devices. Enrollment is limited, and students should expect the course to have rough edges since it is the first offering.

CS 141. Introduction to Computer Sound. 3 Units.
Core mathematics and methods for computer sound with applications to computer science. Background on digital signal processing; time- and frequency-domain methods. Project-focussed exploration of computer sound areas: fundamentals of sound analysis & synthesis, robotics and learning (sound features, filterbanks & deep learning, perception, localization, tracking, manipulation), speech (recognition, synthesis), virtual and augmented reality (3D auralization, HRTFs, reverberation), computational acoustics (wave simulation, physics-based modeling, animation sound), computer music (music synthesis, instrument modeling, audio effects, historical aspects), games (game audio, music and sound design, middleware), hardware acceleration (architectures, codecs, synthesizers). Prerequisite: CS 106A or equivalent programming experience.

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. Data Management and Data Systems. 3-4 Units.
Introduction to the use, design, and implementation of database and data-intensive systems, including data models; schema design; data storage; query processing, query optimization, and cost estimation; concurrency control, transactions, and failure recovery; distributed and parallel execution; semi-structured databases; and data system support for advanced analytics and machine learning. Prerequisites: 103 and 107 (or equivalent).

CS 146. Introduction to Game Design and Development. 3 Units.
This project-based course provides an introduction to game design covering topics like 2D/3D Art, Audio, User Interfaces, Production, Narrative Design, Marketing, and Publishing. Speakers from the profession will provide relevant context during a weekly seminar. Weekly assignments include in-depth materials and require students to independently create small video games. Classroom meetings will be used to foster student project discussions, and deepen understanding of material. The course culminates with students forming project teams to create a final video game. Assignments will be completed within the Unity game development engine; prior Unity experience is not required. Given class size limitations, an online survey will be distributed before class starts and students will be selected so to achieve a diverse class composition. Prerequisite: CS 106A or equivalent programming experience.

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, MapReduce, Apache Spark, 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.

CS 151. Logic Programming. 3 Units.
Logic Programming is a style of programming based on symbolic logic. In writing a logic program, the programmer describes the application area of the program (as a set of logical sentences) without reference to the internal data structures or operations of the system executing the program. In this regard, a logic program is more of a specification than an implementation; and logic programs are often called runnable specifications. This course introduces basic logic programming theory, current technology, and examples of common applications, notably deductive databases, logical spreadsheets, enterprise management, computational law, and game playing. Work in the course takes the form of readings and exercises, weekly programming assignments, and a term-long project. Prerequisite: CS 106B or equivalent.

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. Computational Logic. 3 Units.
Rigorous introduction to Symbolic Logic from a computational perspective. Encoding information in the form of logical sentences. Reasoning with information in this form. Overview of logic technology and its applications - in mathematics, science, engineering, business, law, and so forth. Topics include the syntax and semantics of Propositional Logic, Relational Logic, and Herbrand Logic, validity, contingency, unsatisfiability, logical equivalence, entailment, consistency, natural deduction (Fitch), mathematical induction, resolution, compactness, soundness, completeness.

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 166. Data Structures. 3-4 Units.
Techniques in the design, analysis, and implementation of data structures. Isometries between data structures (including red/black trees and 2-3-4 trees), amortized analysis (including Fibonacci heaps and splay trees), and randomization (including count-min sketches and dynamic perfect hash tables). Data structures for integers and strings (including van Emde Boas trees and suffix trees). Possible additional topics include functional data structures, concurrent data structures, and spatial data structures. Prerequisites: CS107 and CS161.

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. 1-5 Unit.
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. May be repeat for credit.
Same as: MUSIC 128

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. To take this course, students need permission of instructor and may need to complete an assignment due at the first day of class.

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. To take this course, students need permission of instructor and may need to complete an assignment due at the first day of class.
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. Enrollment limited and application required for admittance.

CS 190. Software Design Studio. 3 Units.
This course teaches the art of software design: how to decompose large complex systems into classes that can be implemented and maintained easily. Topics include the causes of complexity, modular design, techniques for creating deep classes, minimizing the complexity associated with exceptions, in-code documentation, and name selection. The class involves significant system software implementation and uses an iterative approach consisting of implementation, review, and revision. The course is taught in a studio format with in-class discussions and code reviews in addition to lectures. Apply at: https://web.stanford.edu/class/cs190.

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.

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. iOS Application Development. 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, 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<https://explorecourses.stanford.edu/search?view=catalog&filter-coursestatus-Active=on&page=0&q=CS107>, 108 (preferred) or 110. Recommended: UNIX, graphics, databases.

CS 194. Software Project. 3 Units.
Design, specification, coding, and testing of a significant team programming project under faculty supervision. Documentation includes capture of project rationale, design and discussion of key performance indicators, a weekly progress log and a software architecture diagram. Public demonstration of the project at the end of the quarter. Preference given to seniors. May be repeated 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 195. Supervised Undergraduate Research. 3-4 Units.
Directed research under faculty supervision. Students are required to submit a written report and give a public presentation on their work.

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.
Same as: VPTL 196

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 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.
Same as: VPTL 1

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. 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 202. Law for Computer Science Professionals. 1 Unit.
An overview of intellectual property law as it relates to computer science and other disciplines, including discussions of patents, trademarks, copyrights, trade secrets, computer fraud litigation and interesting historical tidbits. Emphasis on topics of current interest such as software and business method patents, copyright issues concerning software, music, art and artificial intelligence, and current disputes of note including the recently-settled Waymo v. Uber lawsuit and the ongoing Oracle v. Google, Apple v. Samsung and hiQ v. LinkedIn sagas. Guest lectures typically have covered open source and the free software movement, practical issues for business founders (including corporate formation issues and non-disclosure, non-compete, work-made-for-hire and license agreements), and other pertinent topics. Classes are presented in an open discussion format broadly directed to students with both technical and non-technical backgrounds.

CS 203. Cybersecurity: A Legal and Technical Perspective. 2 Units.
(Formerly IPS 251) This class will use the case method to teach basic computer, network, and information security from technology, law, policy, and business perspectives. Using real world topics, we will study the technical, legal, policy, and business aspects of an incident or issue and its potential solutions. The case studies will be organized around the following topics: vulnerability disclosure, 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, security risks and benefits of cloud data storage, wiretapping on the Internet, and digital forensics. Students taking the class will learn about the techniques attackers use, applicable legal prohibitions, rights, and remedies, the policy context, and strategies in law, policy and business for managing risk. Grades will be based on class participation, two reflection papers, and a final exam. Special Instructions: This class is limited to 65 students, with an effort made to have students from Stanford Law School (30 students will be selected by lottery) and students from Computer Science (30 students) and International Policy Studies (5 students). Elements used in grading: Class Participation (20%), Written Assignments (40%), Final Exam (40%). Cross-listed with the Law School (Law 4004) and International Policy Studies (IPS course number TBD).
Same as: INTLPOL 251

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 205L. Continuous Mathematical Methods with an Emphasis on Machine Learning. 3 Units.
A survey of numerical approaches to the continuous mathematics used in computer vision and robotics with emphasis on machine and deep learning. Although motivated from the standpoint of machine learning, the course will focus on the underlying mathematical methods including computational linear algebra and optimization, as well as special topics such as automatic differentiation via backward propagation, momentum methods from ordinary differential equations, CNNs, RNNs, etc. (Replaces CS205A, and satisfies all similar requirements.) Prerequisites: Math 51; Math 104 or 113 or equivalent or comfortable with the associated material.

CS 206. Exploring Computational Journalism. 3 Units.
This project-based course will explore the field of computational journalism, including the use of Data Science, Info Visualization, AI, and emerging technologies to help journalists discover and tell stories, understand their audience, advance free speech, and build trust. Admission by application; please email R.B. Brenner at rbbrenner@stanford.edu to request 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.

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 217. Hardware Accelerators for Machine Learning. 3-4 Units.
This course provides in-depth coverage of the architectural techniques used to design accelerators for training and inference in machine learning systems. This course will cover classical ML algorithms such as linear regression and support vector machines as well as DNN models such as convolutional neural nets, and recurrent neural nets. We will consider both training and inference for these models and discuss the impact of parameters such as batch size, precision, sparsity and compression on the accuracy of these models. We will cover the design of accelerators for ML model inference and training. Students will become familiar with hardware implementation techniques for using parallelism, locality, and low precision to implement the core computational kernels used in ML. To design energy-efficient accelerators, students will develop the intuition to make trade-offs between ML model parameters and hardware implementation techniques. Students will read recent research papers and complete a design project. Prerequisites: CS 149 or EE 180. CS 229 is ideal, but not required.

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 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. Deep learning and other methods for automatic speech recognition, speech synthesis, affect detection, dialogue management, and applications to digital assistants and spoken language understanding systems. 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, LINGUIST 288

CS 224W. Analysis of Networks. 3-4 Units.
Networks are a fundamental tool for modeling complex social, technological, and biological systems. Coupled with emergence of online social networks and large-scale data availability in biological sciences, this course focuses on the analysis of massive networks which provide many computational, algorithmic, and modeling challenges. This course develops computational tools that reveal how the social, technological, and natural worlds are connected, and how the study of networks sheds light on these connections. nTopics include: how information spreads through society; robustness and fragility of food webs and financial markets; algorithms for the World Wide Web; friend prediction in online social networks; identification of functional modules in biological networks; disease outbreak detection.

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 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 229A. Applied Machine Learning. 3-4 Units.
You will learn to implement and apply machine learning algorithms. This course emphasizes practical skills, and focuses on giving you skills to make these algorithms work. You will learn about commonly used learning techniques including supervised learning algorithms (logistic regression, linear regression, SVM, neural networks/deep learning), unsupervised learning algorithms (k-means), as well as learn about specific applications such as anomaly detection and building recommender systems. This class is taught in the flipped-classroom format. You will watch videos and complete in-depth programming assignments and online quizzes at home, then come to class for discussion sections. This class will culminate in an open-ended final project, which the teaching team will help you on. Prerequisites: Programming at the level of CS106B or 106X, and basic linear algebra such as Math 51.

CS 229T. Statistical Learning Theory. 3 Units.
How do we formalize what it means for an algorithm to learn from data? How do we use mathematical thinking to design better machine learning methods? This course focuses on developing mathematical tools for answering these questions. We will present various learning algorithms and prove theoretical guarantees about them. Topics include generalization bounds, implicit regularization, the theory of deep learning, spectral methods, and online learning and bandits problems. Prerequisites: A solid background in linear algebra and probability theory, statistics and machine learning (STATS 315A or CS 229).
Same as: STATS 231

CS 22A. The Social & Economic Impact of Artificial Intelligence. 1 Unit.
(Formerly IPS 200.) 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.
Same as: INTLPOL 200

CS 230. Deep Learning. 3-4 Units.
Deep Learning is one of the most highly sought after skills in AI. We will help you become good at Deep Learning. In this course, you will learn the foundations of Deep Learning, understand how to build neural networks, and learn how to lead successful machine learning projects. You will learn about Convolutional networks, RNNs, LSTM, Adam, Dropout, BatchNorm, Xavier/He initialization, and more. You will work on case studies from healthcare, autonomous driving, sign language reading, music generation, and natural language processing. You will master not only the theory, but also see how it is applied in industry. You will practice all these ideas in Python and in TensorFlow, which we will teach. AI is transforming multiple industries. After this course, you will likely find creative ways to apply it to your work. This class is taught in the flipped-classroom format. You will watch videos and complete in-depth programming assignments and online quizzes at home, then come in to class for advanced discussions and work on projects. This class will culminate in an open-ended final project, which the teaching team will help you on. Prerequisites: Familiarity with programming in Python and Linear Algebra (matrix / vector multiplications). CS 229 may be taken concurrently.

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 236. Deep Generative Models. 3 Units.
Generative models are widely used in many subfields of AI and Machine Learning. Recent advances in parameterizing these models using neural networks, combined with progress in stochastic optimization methods, have enabled scalable modeling of complex, high-dimensional data including images, text, and speech. In this course, we will study the probabilistic foundations and learning algorithms for deep generative models, including Variational Autoencoders (VAE), Generative Adversarial Networks (GAN), and flow models. The course will also discuss application areas that have benefitted from deep generative models, including computer vision, speech and natural language processing, and reinforcement learning. Prerequisites: Basic knowledge about machine learning from at least one of CS 221, 228, 229 or 230. Students will work with computational and mathematical models and should have a basic knowledge of probabilities and calculus. Proficiency in some programming language, preferably Python, required.

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. 2 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. 3-4 Units.
This course explores models of computation, both old, like functional programming with the lambda calculus (circa 1930), and new, like memory-safe systems programming with Rust (circa 2010). Topics include type systems (polymorphism, algebraic data types, static vs. dynamic), control flow (exceptions, continuations), concurrency/parallelism, metaprogramming, and the semantic gap between computational models and modern hardware. The study of programming languages is equal parts systems and theory, looking at how a rigorous understanding of the syntax, structure, and semantics of computation enables formal reasoning about the behavior and properties of complex real-world systems. In light of today's Cambrian explosion of new programming languages, this course also seeks to provide a conceptual clarity on how to compare and contrast the multitude of programming languages, models, and paradigms in the modern programming landscape. Prerequisites: 103, 110.

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 244B. Distributed Systems. 3 Units.
Distributed operating systems and applications issues, emphasizing high-level protocols and distributed state sharing as the key technologies. Topics: distributed shared memory, object-oriented distributed system design, distributed directory services, atomic transactions and time synchronization, application-sufficient consistency, file access, process scheduling, process migration, and storage/communication abstractions on distribution, scale, robustness in the face of failure, and security. Prerequisites: CS 144.

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.
Availability of massive datasets is revolutionizing science and industry. This course discusses data mining and machine learning algorithms for analyzing very large amounts of data. The focus is on algorithms and systems for mining big data. nTopics include: Big data systems (Hadoop, Spark, Hive); Link Analysis (PageRank, spam detection, hubs-and-authorities); Similarity search (locality-sensitive hashing, shingling, minhashing, random hyperplanes); Stream data processing; Analysis of social-network graphs; Association rules; Dimensionality reduction (UV, SVD, and CUR decompositions); Algorithms for very-large-scale mining (clustering, nearest-neighbor search); Large-scale machine learning (gradient descent, support-vector machines, classification, and regression); Submodular function optimization; Computational advertising. Prerequisites: At least one of CS107 or CS145.

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.

CS 248. Interactive Computer Graphics. 3-4 Units.
This course provides a comprehensive introduction to interactive computer graphics, focusing on fundamental concepts and techniques, as well as their cross-cutting relationship to multiple problem domains in interactive graphics (such as rendering, animation, geometry, image processing). Topics include: 2D and 3D drawing, sampling theory, interpolation, rasterization, image compositing, the real-time GPU graphics pipeline (and parallel rendering), VR rendering, geometric transformations, curves and surfaces, geometric data structures, subdivision, meshing, spatial hierarchies, image processing, time integration, physically-based animation, and inverse kinematics. The course will involve several in-depth programming assignments and a self-selected final project that explores concepts covered in the class. Prerequisite: CS 107, MATH 51.

CS 250. Algebraic Error Correcting Codes. 3 Units.
Introduction to the theory of error correcting codes, emphasizing algebraic constructions, and diverse applications throughout computer science and engineering. Topics include basic bounds on error correcting codes; Reed-Solomon and Reed-Muller codes; list-decoding, list-recovery and locality. Applications may include communication, storage, complexity theory, pseudorandomness, cryptography, streaming algorithms, group testing, and compressed 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. Cryptocurrencies and blockchain technologies. 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 cryptocurrencies, 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 cryptocurrencies. Prerequisite: CS110. Recommended: CS255.

CS 252. Analysis of Boolean Functions. 3 Units.
Boolean functions are among the most basic objects of study in theoretical computer science. This course is about the study of boolean functions from a complexity-theoretic perspective, with an emphasis on analytic methods. We will cover fundamental concepts and techniques in this area, including influence and noise sensitivity, polynomial approximation, hypercontractivity, probabilistic invariance principles, and Gaussian analysis. We will see connections to various areas of theoretical computer science, including circuit complexity, pseudorandomness, classical and quantum query complexity, learning theory, and property testing. Prerequisites: CS 103 and CS 109 or equivalents. CS 154 and CS 161 recommended.

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. After reviewing recent work in AI that has leveraged ideas from logic, we will slow down and study in more detail various components of high-level intelligence and the tools that have been designed to capture those components. Specific areas will include: reasoning about belief and action, causality and counterfactuals, legal and normative reasoning, natural language inference, and Turing-complete logical formalisms including (probabilistic) logic programming and lambda calculus. Our main concern will be understanding the logical tools themselves, including their formal properties and how they relate to other tools such as probability and statistics. At the end, students should expect to have learned a lot more about logic, and also to have a sense for how logic has been and can be used in AI applications. Prerequisites: A background in logic, at least at the level of Phil 151, will be expected. In case a student is willing to put in the extra work to catch up, it may be possible to take the course with background equivalent to Phil 150 or CS 157. A background in AI, at the level of CS 221, would also be very helpful and will at times be expected. 2 unit option only for PhD students past the second year. Course website: http://web.stanford.edu/class/cs257/.
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 sub-linear algorithms and decision making under uncertainty. 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 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, probability, and logic.
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 214 or 215 or 217 or 260. Preference to BMI graduate students. Consent of instructor required.
Same as: BIOE 212, BIOMEDIN 212, GENE 212

CS 273A. The Human Genome Source Code. 3 Units.
A computational introduction to the most amazing programming language on the planet: your genome. Topics include genome sequencing (assembling source code from code fragments); the human genome functional landscape: variable assignments (genes), control-flow logic (gene regulation) and run-time stack (epigenomics); human disease and personalized genomics (as a hunt for bugs in the human code); genome editing (code injection) to cure the incurable; and the source code behind amazing animal adaptations. Algorithmic approaches will introduce ideas from computational genomics, machine learning and natural language processing. Course includes primers on molecular biology, and text processing languages. No prerequisites.
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, GENE 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 278. Social Computing. 3 Units.
Today we interact with our friends and enemies, our team partners and romantic partners, and our organizations and societies, all through computational systems. How do we design these social computing systems to be effective? This course covers design patterns for social computing and crowdsourcing systems, and the foundational ideas that underpin them. Students will engage in the creation of new computationally-mediated social environments.

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 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 Seminar. 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 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) and EE 282. Recommended: CS 149.

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 325B. Data for Sustainable Development. 3-5 Units.
The sustainable development goals (SDGs) encompass many important aspects of human and ecosystem well-being that are traditionally difficult to measure. This project-based course will focus on ways to use inexpensive, unconventional data streams to measure outcomes relevant to SDGs, including poverty, hunger, health, governance, and economic activity. Students will apply machine learning techniques to various projects outlined at the beginning of the quarter. The main learning goals are to gain experience conducting and communicating original research. Prior knowledge of machine learning techniques, such as from CS 221, CS 229, CS 231N, STATS 202, or STATS 216 is required. Open to both undergraduate and graduate students. Enrollment limited to 24. Students must apply for the class by filling out the form at https://goo.gl/forms/9LSZF7lPkHadix5D3. A permission code will be given to admitted students to register for the class.
Same as: EARTHSYS 162, EARTHSYS 262

CS 326. Topics in Advanced Robotic Manipulation. 3-4 Units.
This course provides a survey of the most important and influential concepts in autonomous robotic manipulation. It includes classical concepts that are still widely used and recent approaches that have changed the way we look autonomous manipulation. We cover approaches towards motion planning and control using visual and tactile perception as well as machine learning. This course is especially concerned with new approaches for overcoming challenges in generalization from experience, exploration of the environment, and learning representation so that these methods can scale to real problems. Students are expected to present one paper in a tutorial, debate a paper once from the Pro and once from the Con side. They are also expected to propose an original research project and work on it towards a research paper. Recommended: CS 131, 223A, 229 or equivalents.

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.
A representation performs the task of converting an observation in the real world (e.g. an image, a recorded speech signal, a word in a sentence) into a mathematical form (e.g. a vector). This mathematical form is then used by subsequent steps (e.g. a classifier) to produce the outcome, such as classifying an image or recognizing a spoken word. Forming the proper representation for a task is an essential problem in modern AI. In this course, we focus on 1) establishing why representations matter, 2) classical and moderns methods of forming representations in Computer Vision, 3) methods of analyzing and probing representations, 4) portraying the future landscape of representations with generic and comprehensive AI/vision systems over the horizon, and finally 5) going beyond computer vision by talking about non-visual representations, such as the ones used in NLP or neuroscience. The course will heavily feature systems based on deep learning and convolutional neural networks. We will have several teaching lectures, a number of prominent external guest speakers, as well as presentations by the students on recent papers and their projects. nnRequired Prerequisites: CS131A, CS231A, CS231B, or CS231N. If you do not have the required prerequisites, please contact a member of the course staff before enrolling in this course.

CS 332. Advanced Survey of Reinforcement Learning. 3 Units.
This class will provide a core overview of essential topics and new research frontiers in reinforcement learning. Planned topics include: model free and model based reinforcement learning, policy search, Monte Carlo Tree Search planning methods, off policy evaluation, exploration, imitation learning, temporal abstraction/hierarchical approaches, safety and risk sensitivity, human-in-the-loop RL, inverse reinforcement learning, learning to communicate, and insights from human learning. Students are expected to create an original research paper on a related topic. Prerequisites: CS221 or AA238/CS238 or CS234 or CS229 or similar experience.

CS 333. Safe and Interactive Robotics. 3-4 Units.
Once confined to the manufacturing floor, robots are quickly entering the public space at multiple levels: drones, surgical robots, service robots, and self-driving cars are becoming tangible technologies impacting the human experience. Our goal in this class is to learn about and design algorithms that enable robots to reason about their actions, interact with one another, the humans, and the environment they live in, as well as plan safe strategies that humans can trust and rely on.nThis is a project-based graduate course that studies algorithms in formal methods, control theory, and robotics, which can improve the state-of-the-art human-robot systems. We focus on designing new algorithms for enhancing safe and interactive autonomy. nnRecommended: Introductory course in AI and robotics.

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 and machine learning of very large-scale data, including the problem statement, implementation, and evaluation of a solution. Students work on real problems on real-world data. The course provides access to large real-world data and access to big data cloud computing infrastructure (Amazon EC2, Google Cloud Platform). Some lectures on relevant materials will be given (Hadoop, Spark, Hive, Amazon EC2) as well as other topics of relevance to projects.

CS 344. Topics in Computer Networks. 3 Units.
High-performance network system design. Students will work in teams of two to build a fully functioning Internet router. Students will design the control plane in C on a linux host and will design the data plane in the new P4 language on the NetFPGA 4 x 10GE switch. For the midterm milestone, teams must demonstrate that their routers can interoperate with the other teams by building a small scale datacenter topology. In the final 3-4 weeks of the class, teams will participate in an open-ended design challenge. Prerequisites: At least one student in each team must have taken CS144 at Stanford and completed Lab 3 (static router), Verilog experience for one member of each team is helpful but not required. 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 348K. Visual Computing Systems. 3-4 Units.
Visual computing tasks such as computational photography, image/video analysis, 3D reconstruction, and real-time 3D graphics are key responsibilities of modern computer systems ranging from sensor-rich smart phones, autonomous robots, and large data centers. These workloads demand exceptional system efficiency and this course examines the key ideas, techniques, and challenges associated with the design of parallel (and heterogeneous) systems that execute and accelerate visual computing applications. This course is intended for graduate and advanced undergraduate-level systems students interested in architecting efficient graphics, image processing, and computer vision platforms (both new hardware architectures and domain-optimized programming frameworks) and for students in graphics, vision, and ML that seek to understand throughput computing principles so they can develop scalablenalgorithms that map efficiently these future platforms. Students will perform daily research paper readings, complete simple programming assignments, and compete a self-selected term project. Prerequisites: CS 107 or equivalent. Recommended: Parallel computing or computer architecture (CS 149, EE 282), some background in techniques in either deep learning (CS 230, CS 231N), computer vision (CS 231A), digital image processing (CS 232).

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 349D. Cloud Computing Technology. 3 Units.
The largest change in the computer industry over the past five years has arguably been the emergence of cloud computing: organizations are increasingly moving their workloads to managed public clouds and using new, global-scale services that were simply not possible in private datacenters. However, both building and using cloud systems remains a black art with many difficult research challenges. This research seminar will cover industry and academic work on cloud computing and survey challenges including programming interfaces, cloud native applications, resource management, pricing, availability and reliability, privacy and security. Students will also propose and develop an original research project.n nPrerequisites: For graduate students, background in computer systems (CS 240, 244, 244B or 245) is strongly recommended. Undergrads will need instructor's approval.

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 354. Topics in Intractability: Unfulfilled Algorithmic Fantasies. 3 Units.
Over the past 45 years, understanding NP-hardness has been an amazingly useful tool for algorithm designers. This course will expose students to additional ways to reason about obstacles for designing efficient algorithms. Topics will include unconditional lower bounds (query- and communication-complexity), total problems, Unique Games, average-case complexity, and fine-grained complexity. Prerequisites: CS 161 or equivalent. CS 254 recommended but not required.

CS 355. Advanced Topics in Cryptography. 3 Units.
Topics: Pseudo randomness, multiparty computation, pairing-based and lattice-based cryptography, zero knowledge protocols, and new encryption and integrity paradigms. May be repeated for credit. Prerequisite: 255.

CS 356. Topics in Computer and Network Security. 3 Units.
Research seminar covering foundational work and current topics in computer and network security. Students will read and discuss published research papers as well as complete an original research project in small groups. Open to Ph.D. and masters students as well as advanced undergraduate students. Prerequisites: While the course has no official prerequisites, students need a mature understanding of software systems and networks to be successful. We strongly encourage students to first take CS155: Computer and Network Security.

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 361. Engineering Design Optimization. 3-4 Units.
Design of engineering systems within a formal optimization framework. 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, automating differentiation, 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 368. Algorithmic Techniques for Big Data. 3 Units.
Designing algorithms for efficient processing of large data sets poses unique challenges. This course will discuss algorithmic paradigms that have been developed to efficiently process data sets that are much larger than available memory. We will cover streaming algorithms and sketching methods that produce compact datanstructures, dimension reduction methods that preserve geometric structure, efficient algorithms for numerical linear algebra, graph sparsification methods, as well as impossibility results for these techniques.

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 369L. Algorithmic Perspective on Machine Learning. 3 Units.
Many problems in machine learning are intractable in the worst case, andnpose a challenge for the design of algorithms with provable guarantees. In this course, we will discuss several success stories at the intersection of algorithm design and machine learning, focusing on devising appropriate models and mathematical tools to facilitate rigorous analysis.

CS 369M. Metric Embeddings and Algorithmic Applications. 3 Units.
Low distortion embeddings of finite metric spaces is a topic at the intersection of mathematics and theoretical computer science. Much progress in this area in recent years has been motivated by algorithmic applications. Mapping complicated metrics of interest to simpler metrics (normed spaces, trees, and so on) gives access to a powerful algorithmic toolkit for approximation algorithms, online algorithms as well as for efficient search and indexing of large data sets. In a different vein, convex relaxations are a useful tool for graph partitioning problems; central to the analysis are metric embedding questions for certainly computationally defined metrics. In this course, we will see several classical and recent results on metric embeddings with a focus on algorithmic applications. Students will be expected to have a strong background in algorithms and probability.

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 375. Large-Scale Neural Network Modeling for Neuroscience. 1-3 Unit.
Introduction to designing, building, and training neural networks for modeling brain and behavioral data, including: deep convolutional neural network models of sensory systems (vision, audition, somatosensation); recurrent neural networks for dynamics, memory and attention; integration of variational and generative methods for cognitive modeling; and methods and metrics for comparing such models to real-world neural data. Attention will be given both to established methods as well as cutting-edge techniques. Students will learn conceptual bases for deep neural network models, and will also implement learn to implement and train large-scale models in Tensorflow using GPUs. Requirements: Fluency in Unix shell and Python programming, familiarity with differential equations, linear algebra, and probability theory, and one or more courses in cognitive or systems neuroscience.
Same as: PSYCH 249

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 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 377G. Designing Serious Games. 3-4 Units.
Over the last few years we have seen the rise of "serious games" to promote understanding of complex social and ecological challenges, and to create passion for solving them. This project-based course provides an introduction to game design principals while applying them to games that teach. Run as a hands-on studio class, students will design and prototype games for social change and civic engagement. We will learn the fundamentals of games design via lecture and extensive reading in order to make effective games to explore issues facing society today. The course culminates in an end-of- quarter open house to showcase our games. Prerequisite: CS147 or equivalent. 247 recommended, but not required.

CS 377I. Designing for Complexity. 3-4 Units.
Complex problems require sophisticated approaches. In this project-based hands-on course, students explore the design of systems, information and interface for human use. We will model the flow of interactions, data and context, and crafting a design that is useful, appropriate and robust. Students will create utilities or games as a response to the challenges presented. We will also examine the ethical consequences of design decisions and explore current issues arising from unintended consequences. Prerequisite: CS 147 or equivalent. 247 recommended, but not required. May be repeat for credit.

CS 377J. Designing Systems for Collaboration, Cooperation, and Collective Action. 3 Units.
This project-based class focuses on the design of systems that support large groups to collaborate, cooperate, and act together. A large body of research in Human-Computer Interaction and Computer Supported Cooperative Work is devoted to the design of systems that assist large groups to come together and aggregate their efforts, whether in the form of information, code, or people power. Examples of these sociotechnical systems include Wikipedia, Facebook groups, and Change.org. Students will read papers in the HCI literature and participate in discussions that analyze the design of these systems, the various stakeholders, and how the systems play out in the real world. Prerequisites: CS 147; CS 376 recommended but not required.

CS 377P. Advanced User Interface Design Patterns. 3 Units.
User interface design is about creating the most effective, intuitive design possible to help users achieve a specific goal. While understanding users is one part of the equation, the other part is a strong understanding of user interface design rules and patterns that you can apply to solve their needs. This course will deep dive into user interface design across mobile, desktop, and wearable platforms covering common patterns, when to use them, and when to break them. Each week will cover a different user interface design challenge and explore the patterns in areas such as data input, search & filters, tables and lists, content organization, navigation, dark patterns and more. Through the use of in class exercises, integrated design challenges, and an exploration of examples, students will leave the class knowing how to integrate user interface patterns into their design work to create powerful, effective digital experiences. Prerequisite: CS 147 or equivalent. 247 recommended but not required.

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.
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 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. This course is for graduate students only. Undergraduate students should enroll in CS199.

CS 399P. Independent Project. 1-9 Unit.
Graded satisfactory/no credit. This course is for graduate students only. Undergraduate students should enroll in CS199P.

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. Interested students should complete the application at https://web.stanford.edu/class/educ211 by January 5, and come to the first class at 9am in CERAS 101.
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. Interested students should complete the application at https://web.stanford.edu/class/educ211 by January 5, and come to the first class at 9am in CERAS 101.
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 43. Functional Programming Abstractions. 2 Units.
This course explores the philosophy and fundamentals of functional programming, with a focus on the Haskell and Clojure programming languages. Topics include: functional abstractions (function composition, higher order functions), immutable data structures, type systems, Lisp macros, homoiconicity, and monads. The course interweaves a theoretical description of fundamentals with hands-on projects in Haskell and Clojure. Prerequisites: CS107 (or equivalent experience).

CS 431. High-level Vision: From Neurons to Deep Neural Networks. 1-3 Unit.
Interdisciplinary seminar focusing on understanding how computations in the brain enable rapid and efficient object perception. Covers topics from multiple perspectives drawing on recent research in Psychology, Neuroscience, and Computer Science. Emphasis on discussing recent empirical findings, methods and theoretical debates in the field.
Same as: PSYCH 250

CS 448. Topics in Computer Graphics. 3-4 Units.
Topic changes each quarter. Recent topics: computational photography, datanvisualization, 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 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 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 47. Cross-Platform Mobile Development. 2 Units.
The fundamentals of cross-platform mobile application development using the React Native framework (RN). Primary focus on developing best practices in creating apps for both iOS and Android by using Javascript and existing web + mobile development paradigms. Students will explore the unique aspects that made RN a primary tool for mobile development within Facebook, Instagram, Airbnb, Walmart, Tesla, and UberEats. Skills developed over the course will be consolidated by the completion of a final project. Required Prerequisites: at least one of the following: CS142, CS193P, CS193A (Web/Mobile development or heavy project-based class experience).

CS 476A. Music, Computing, Design I: The Art of Design. 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. This course is for graduate students only. Undergraduate students should enroll in CS199.

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. This course is for graduate students only. Undergraduate students should enroll in CS199P.

CS 49N. Using Bits to Control Atoms. 3 Units.
This is a crash course in how to use a stripped-down computer system about the size of a credit card (the rasberry pi computer) to control as many different sensors as we can implement in ten weeks, including LEDs, motion sensors, light controllers, and accelerometers. The ability to fearlessly grab a set of hardware devices, examine the data sheet to see how to use it, and stitch them together using simple code is a secret weapon that software-only people lack, and allows you to build many interesting gadgets. We will start with a "bare metal" system --- no operating system, no support --- and teach you how to read device data sheets describing sensors and write the minimal code needed to control them (including how to debug when things go wrong, as they always do). This course differs from most in that it is deliberately mostly about what and why rather than how --- our hope is that the things you are able at the end will inspire you to follow the rest of the CS curriculum to understand better how things you've used work. Prerequisites: knowledge of the C programming language. A Linux or Mac laptop that you are comfortable coding on.

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 repeated for credit. Cardinal Course certified by the Haas Center.

CS 51. CS + Social Good Studio: Designing Social Impact Projects. 2 Units.
Introduces students to the tech + social good space. Students 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 non-profit domain experts. Main class components are workshops, community discussions, guest speakers and mentorship. Studio provides an outlet for students to create social change through CS while engaging in the full product development cycle on real-world projects. The class culminates in a showcase where students share their project ideas and Minimum Viable Product prototypes with stakeholders and the public. Prerequisite: CS 147, equivalent experience, or consent of instructors.

CS 52. CS + Social Good Studio. 2 Units.
Continuation of CS51 (CS + Social Good Studio). 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 521. Seminar on AI Safety. 1 Unit.
In this seminar, we will focus on the challenges in the design of safe and verified AI-based systems. We will explore some of the major problems in this area from the viewpoint of industry and academia. We plan to have a weekly seminar speaker to discuss issues such as verification of AI systems, reward misalignment and hacking, secure and attack-resilient AI systems, diagnosis and repair, issues regarding policy and ethics, as well as the implications of AI safety in automotive industry. Prerequisites: There are no official prerequisites but an introductory course in artificial intelligence is recommended.

CS 522. Seminar in Artificial Intelligence in Healthcare. 1 Unit.
Artificial intelligence is poised to make radical changes in healthcare, transforming areas such as diagnosis, genomics, surgical robotics, and drug discovery. In the coming years, artificial intelligence has the potential to lower healthcare costs, identify more effective treatments, and facilitate prevention and early detection of diseases. This class is a seminar series featuring prominent researchers, physicians, entrepreneurs, and venture capitalists, all sharing their thoughts on the future of healthcare. We highly encourage students of all backgrounds to enroll (no AI/healthcare background necessary). Speakers and more at shift.stanford.edu/healthai.

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

CS 549. Human-Computer Interaction in the Real World. 1 Unit.
Intended for students who are pursuing a focus on HCI, this course focuses on showing students how HCI gets applied in industry across different types of companies. The course consists of on-site visits to large companies (for example Google, Yahoo, Square, Tesla) and to startups to talk to the HCI practitioners at these companies and learn first hand how HCI and design fits in at different companies. The objective of this class is to have students understand how HCI practitioners fit into organizations, the roles they play, and what skills they need in the real world to be able to do their magic.

CS 56N. Great Discoveries and Inventions in Computing. 3 Units.
This seminar will explore some of both the great discoveries that underlie computer science and the inventions that have produced the remarkable advances in computing technology. Key questions we will explore include: What is computable? How can information be securely communicated? How do computers fundamentally work? What makes computers fast? Our exploration will look both at the principles behind the discoveries and inventions, as well as the history and the people involved in those events. Some exposure to programming will be helpful, but it not strictly necessary.

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 581. Media Innovation. 1 Unit.
This course will introduce students interested in computer science, engineering, and media to what¿s possible and probable when it comes to media innovation. nnSpeakers from multiple disciplines and industry will discuss a range of topics in the context of evolving media with a focus on the technical trends, opportunities and challenges surfacing in the unfolding media ecosystem. Speakers will underscore the need to innovate to survive in the media and information industries. Open to both undergraduates and graduate students.

CS 7. Personal Finance for Engineers. 1 Unit.
Introduction to the fundamentals and analysis specifically needed by engineers to make informed and intelligent financial decisions. Course will focus on actual industry-based financial information from technology companies and realistic financial issues. Topics include: behavioral finance, budgeting, debt, compensation, stock options, investing and real estate. No prior finance or economics experience required.

CS 801. TGR Project. 0 Units.
.

CS 802. TGR Dissertation. 0 Units.
.

CS 80Q. Race and Gender in Silicon Valley. 3 Units.
This course interrogates the social challenges of Silicon Valley, a place of privilege, privation, and precarity, and encourages students to perform their own ethnographical studies through writing, coding, engagement, digital culture, and social practice. We will learn about the importance of technology in shaping our critical understanding of social conditions in our community and the global economy.

CS 83. Playback Theater For Research. 3 Units.
Playback combines elements of theater, community work and storytelling. In a playback show, a group of actors and musicians create an improvised performance based on the audience's personal stories. A playback show brings about a powerful listening and sharing experience. During the course, we will tell, listen, play together, and train in playback techniques. We will write diaries to process our experience in the context of education and research. The course is aimed to strengthen listening abilities, creativity and the collaborative spirit, all integral parts of doing great science. In playback, as in research, we are always moving together, from the known, to the unknown, and back. There is limited enrollment for this class. Application is required.

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.