Computer Science

Courses offered by the Department of Computer Science are listed under the subject code CS on the Stanford Bulletin's ExploreCourses (http://explorecourses.stanford.edu/CourseSearch/search?view=catalog&catalog=&page=0&q=CS&filter-catalognumber-CS=on&filter-coursestatus-Active=on) web site.

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

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

Mission of the Undergraduate Program in Computer Science

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

Learning Outcomes (Undergraduate)

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

Students are expected to be able to:

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

Learning Outcomes (Graduate)

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

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

Graduate Programs in Computer Science

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

Computer Science Course Catalog Numbering System

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

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

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

<table>
<thead>
<tr>
<th>Digit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00-09</td>
<td>Introductory, miscellaneous</td>
</tr>
<tr>
<td>10-19</td>
<td>Hardware and Software Systems</td>
</tr>
<tr>
<td>20-39</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>40-49</td>
<td>Software Systems</td>
</tr>
<tr>
<td>50-59</td>
<td>Mathematical Foundations of Computing</td>
</tr>
<tr>
<td>60-69</td>
<td>Analysis of Algorithms</td>
</tr>
<tr>
<td>70-79</td>
<td>Computational Biology and Interdisciplinary Topics</td>
</tr>
<tr>
<td>90-99</td>
<td>Independent Study and Practicum</td>
</tr>
</tbody>
</table>

Bachelor of Science in Computer Science

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

In addition to Computer Science itself, Stanford offers several interdisciplinary degrees with a substantial computer science component. The Symbolic Systems major (in the School of Humanities and Sciences) offers an opportunity to explore computer science and its relation to linguistics, philosophy, and psychology. The Mathematical and Computational Sciences major (also Humanities and Sciences) allows students to explore computer science along with more mathematics, statistics, and operations research.

**Computer Science (CS)**

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

**Mission of the Undergraduate Program in Computer Science**

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

**Requirements**

**Mathematics (26 units minimum)—**

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 103</td>
<td>Mathematical Foundations of Computing</td>
<td>5</td>
</tr>
<tr>
<td>CS 109</td>
<td>Introduction to Probability for Computer Scientists</td>
<td>5</td>
</tr>
<tr>
<td>MATH 41 &amp; MATH 42</td>
<td>Calculus and Calculus</td>
<td>10</td>
</tr>
</tbody>
</table>

Plus two electives

**Science (11 units minimum)—**

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHYSICS 41</td>
<td>Mechanics</td>
<td>4</td>
</tr>
<tr>
<td>PHYSICS 43</td>
<td>Electricity and Magnetism</td>
<td>4</td>
</tr>
</tbody>
</table>

Science elective

**Technology in Society (3-5 units)—**

One course; see Basic Requirement 4

**Engineering Fundamentals (13 units minimum; see Basic Requirement 3)—**

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 106B or CS 106X</td>
<td>Programming Abstractions (Accelerated)</td>
<td>5</td>
</tr>
<tr>
<td>ENGR 40 or ENGR 40A or ENGR 40M*</td>
<td>Introductory Electronics</td>
<td>5</td>
</tr>
</tbody>
</table>

Fundamentals Elective (may not be 70A, B, or X) 3-5

*Students who take ENGR 40A or 40M for fewer than 5 units are required to take 1-2 additional units of ENGR Fundamentals (13 units minimum), or 1-2 additional units of Depth (27 units minimum for track and elective courses).

**Writing in the Major—**

Select one of the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 181W</td>
<td>Computers, Ethics, and Public Policy</td>
<td></td>
</tr>
<tr>
<td>CS 191W</td>
<td>Writing Intensive Senior Project</td>
<td></td>
</tr>
<tr>
<td>CS 194W</td>
<td>Software Project</td>
<td></td>
</tr>
<tr>
<td>CS 210B</td>
<td>Software Project Experience with Corporate Partners</td>
<td></td>
</tr>
<tr>
<td>CS 294W</td>
<td>Writing Intensive Research Project in Computer Science</td>
<td></td>
</tr>
</tbody>
</table>

**Computer Science Core (15 units)—**

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 107 or CS 107E</td>
<td>Computer Organization and Systems</td>
<td>5</td>
</tr>
<tr>
<td>CS 110</td>
<td>Principles of Computer Systems</td>
<td>5</td>
</tr>
<tr>
<td>CS 161</td>
<td>Design and Analysis of Algorithms</td>
<td>5</td>
</tr>
</tbody>
</table>

**Computer Science Depth B.S.**

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

**Artificial Intelligence Track**

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
<td>4</td>
</tr>
</tbody>
</table>

Select two of the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td></td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td></td>
</tr>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td></td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td></td>
</tr>
<tr>
<td>CS 131</td>
<td>Computer Vision: Foundations and Applications</td>
<td></td>
</tr>
<tr>
<td>or CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
<td></td>
</tr>
</tbody>
</table>

One additional course from the list above or the following: 3-4

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 124</td>
<td>From Languages to Information</td>
<td></td>
</tr>
<tr>
<td>CS 222</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 224U</td>
<td>Natural Language Understanding</td>
<td></td>
</tr>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td></td>
</tr>
<tr>
<td>CS 225A</td>
<td>Experimental Robotics</td>
<td></td>
</tr>
<tr>
<td>CS 225B</td>
<td>Robot Programming Laboratory</td>
<td></td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td></td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition (If not taken for track requirement B)</td>
<td></td>
</tr>
<tr>
<td>CS 231B</td>
<td>The Cutting Edge of Computer Vision</td>
<td></td>
</tr>
<tr>
<td>CS 231M</td>
<td>Mobile Computer Vision</td>
<td></td>
</tr>
<tr>
<td>CS 231N</td>
<td>Convolutional Neural Networks for Visual Recognition</td>
<td></td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td></td>
</tr>
<tr>
<td>CS 276</td>
<td>Information Retrieval and Web Search</td>
<td></td>
</tr>
<tr>
<td>CS 277</td>
<td>Experimental Haptics</td>
<td></td>
</tr>
<tr>
<td>CS 279</td>
<td>Computational Biology: Structure and Organization of Biomolecules and Cells</td>
<td></td>
</tr>
</tbody>
</table>
### Biocomputation Track—

The Mathematics, Science, and Engineering Fundamentals requirements are non-standard for this track. See Handbook for Undergraduate Engineering Programs for details.

**Select one of the following:** 3-4
- CS 121 (Not given this year)
- CS 221 Artificial Intelligence: Principles and Techniques
- CS 228 Probabilistic Graphical Models: Principles and Techniques
- CS 229 Machine Learning
- CS 231A Computer Vision: From 3D Reconstruction to Recognition

**Select one of the following:**
- CS 173 A Computational Tour of the Human Genome
- CS 273A A Computational Tour of the Human Genome
- CS 262 Computational Genomics
- CS 270 Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
- CS 274 Representations and Algorithms for Computational Molecular Biology

### Computer Engineering Track—

**Units**

<table>
<thead>
<tr>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 275 Translational Bioinformatics</td>
</tr>
<tr>
<td>CS 279 Computational Biology: Structure and Organization of Biomolecules and Cells</td>
</tr>
</tbody>
</table>

One additional course from the lists above or the following: 3-4
- CS 124 From Languages to Information
- CS 145 Introduction to Databases
- CS 147 Introduction to Human-Computer Interaction Design
- CS 148 Introduction to Computer Graphics and Imaging
- CS 248 Interactive Computer Graphics

One course selected from either the Biomedical Computation (BMC) Informatics electives list (go to http://bmc.stanford.edu and select 'Informatics' electives list, BIOE 101, or from the general CS electives list 5

One course from the BMC Informatics elective list (go to http://bmc.stanford.edu) 3-4
- CS 140 Operating Systems and Systems Programming
- EE 109 Digital Systems Design Lab
- EE 271 Introduction to VLSI Systems

Satisfy the requirements of one of the following concentrations:

#### 1) Digital Systems Concentration

<table>
<thead>
<tr>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140 Operating Systems and Systems Programming</td>
</tr>
<tr>
<td>CS 143 Compilers</td>
</tr>
<tr>
<td>EE 109 Digital Systems Design Lab</td>
</tr>
<tr>
<td>EE 271 Introduction to VLSI Systems</td>
</tr>
</tbody>
</table>

Plus two of the following (6-8 units):
- CS 140 Operating Systems and Systems Programming (if not counted above)
- CS 143 Compilers
- CS 144 Introduction to Computer Networking
- CS 149 Parallel Computing
- CS 240E
- CS 244 Advanced Topics in Networking
- EE 273 Digital Systems Engineering
- EE 282 Computer Systems Architecture

**Units**

<table>
<thead>
<tr>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE 108 Digital System Design</td>
</tr>
<tr>
<td>EE 180 and Digital Systems Architecture</td>
</tr>
</tbody>
</table>

Select two of the following: 8
- EE 101A Circuits I
- EE 101B Circuits II
- EE 102A Signal Processing and Linear Systems I
- EE 102B Signal Processing and Linear Systems II

Plus two of the following (6-8 units):
- CS 140 Operating Systems and Systems Programming (if not counted above)
- CS 143 Compilers
- CS 144 Introduction to Computer Networking
- CS 149 Parallel Computing
- CS 240E
- CS 244 Advanced Topics in Networking
- EE 273 Digital Systems Engineering
- EE 282 Computer Systems Architecture

#### 2) Robotics and Mechatronics Concentration

<table>
<thead>
<tr>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 223A Introduction to Robotics</td>
</tr>
<tr>
<td>ME 210 Introduction to Mechatronics</td>
</tr>
<tr>
<td>ENGR 105 Feedback Control Design</td>
</tr>
</tbody>
</table>

Plus one of the following (3-4 units):
- CS 225A Experimental Robotics
- CS 225B Robot Programming Laboratory
- CS 231A Computer Vision: From 3D Reconstruction to Recognition
## Graphics Track—

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CME 104</td>
<td>Linear Algebra and Partial Differential Equations for Engineers</td>
<td></td>
</tr>
<tr>
<td>CME 108</td>
<td>Introduction to Scientific Computing</td>
<td></td>
</tr>
<tr>
<td>MATH 52</td>
<td>Integral Calculus of Several Variables</td>
<td></td>
</tr>
<tr>
<td>MATH 113</td>
<td>Linear Algebra and Matrix Theory</td>
<td></td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
<td></td>
</tr>
<tr>
<td>or CS 131</td>
<td>Computer Vision: Foundations and Applications</td>
<td></td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
<td></td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 348A</td>
<td>Computer Graphics: Geometric Modeling</td>
<td></td>
</tr>
<tr>
<td>CS 348B</td>
<td>Computer Graphics: Image Synthesis Techniques</td>
<td></td>
</tr>
<tr>
<td>CS 348V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 448</td>
<td>Topics in Computer Graphics</td>
<td></td>
</tr>
</tbody>
</table>

Select two of the following: 6-8 units

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARTSTUDI 160</td>
<td>Intro to Digital / Physical Design</td>
</tr>
<tr>
<td>ARTSTUDI 170</td>
<td>Introduction to Photography</td>
</tr>
<tr>
<td>ARTSTUDI 179</td>
<td>Digital Art I</td>
</tr>
<tr>
<td>CME 302</td>
<td>Numerical Linear Algebra</td>
</tr>
<tr>
<td>CME 306</td>
<td>Numerical Solution of Partial Differential Equations</td>
</tr>
<tr>
<td>EE 262</td>
<td>Two-Dimensional Imaging</td>
</tr>
</tbody>
</table>

Select one of the following: 3-5 units

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE 264</td>
<td>Digital Signal Processing</td>
</tr>
<tr>
<td>EE 278</td>
<td>Introduction to Statistical Signal Processing</td>
</tr>
<tr>
<td>EE 368</td>
<td>Digital Image Processing</td>
</tr>
<tr>
<td>ME 101</td>
<td>Visual Thinking</td>
</tr>
<tr>
<td>PSYCH 30</td>
<td>Introduction to Perception</td>
</tr>
<tr>
<td>PSYCH 221</td>
<td>Applied Vision and Image Systems</td>
</tr>
</tbody>
</table>

Human-Computer Interaction Track—

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 147</td>
<td>Introduction to Human-Computer Interaction Design</td>
<td>4</td>
</tr>
<tr>
<td>CS 247</td>
<td>Human-Computer Interaction Design Studio</td>
<td>4</td>
</tr>
</tbody>
</table>

Select two of the following: 6-8 units

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 142</td>
<td>Web Applications</td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
</tr>
<tr>
<td>CS 194H</td>
<td>User Interface Design Project</td>
</tr>
<tr>
<td>CS 210A</td>
<td>Software Project Experience with Corporate Partners</td>
</tr>
<tr>
<td>CS 376</td>
<td>Human-Computer Interaction Research</td>
</tr>
<tr>
<td>Any CS 377A/B/C/ Topics in HCT of three or more units</td>
<td></td>
</tr>
<tr>
<td>CS 448B</td>
<td>Data Visualization</td>
</tr>
<tr>
<td>ME 216M</td>
<td></td>
</tr>
</tbody>
</table>

At least two additional courses from above list, the general CS electives 3-6 list, or the following: 5 units

Any d.school class of 3+ units; any class of 3+ units at hci.stanford.edu under the 'courses' link.

Communication-

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMM 121</td>
<td>Behavior and Social Media</td>
</tr>
<tr>
<td>COMM 124</td>
<td>Digital Deception</td>
</tr>
<tr>
<td>or COMM 224</td>
<td>Digital Deception</td>
</tr>
<tr>
<td>COMM 140</td>
<td>Digital Media Entrepreneurship</td>
</tr>
<tr>
<td>or COMM 240</td>
<td>Digital Media Entrepreneurship</td>
</tr>
<tr>
<td>COMM 166</td>
<td>Virtual People</td>
</tr>
<tr>
<td>COMM 169</td>
<td>Computers and Interfaces</td>
</tr>
<tr>
<td>or COMM 269</td>
<td>Computers and Interfaces</td>
</tr>
<tr>
<td>COMM 172</td>
<td>Media Psychology</td>
</tr>
<tr>
<td>or COMM 272</td>
<td>Media Psychology</td>
</tr>
<tr>
<td>COMM 182</td>
<td></td>
</tr>
<tr>
<td>COMM 324</td>
<td>Language and Technology</td>
</tr>
</tbody>
</table>

Art Studio-

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARTSTUDI 160</td>
<td>Intro to Digital / Physical Design</td>
</tr>
<tr>
<td>ARTSTUDI 162</td>
<td>Embodied Interfaces</td>
</tr>
<tr>
<td>ARTSTUDI 163</td>
<td>Drawing with Code</td>
</tr>
<tr>
<td>ARTSTUDI 164</td>
<td>DESIGN IN PUBLIC SPACES</td>
</tr>
<tr>
<td>ARTSTUDI 165</td>
<td>Social Media and Performatives</td>
</tr>
<tr>
<td>ARTSTUDI 168</td>
<td>Data as Material</td>
</tr>
<tr>
<td>ARTSTUDI 264</td>
<td>Advanced Interaction Design</td>
</tr>
<tr>
<td>ARTSTUDI 266</td>
<td>Sulptural Screens / Malleable Media</td>
</tr>
</tbody>
</table>
Information Track—

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 124</td>
<td>From Languages to Information</td>
<td>4</td>
</tr>
<tr>
<td>CS 145</td>
<td>Introduction to Databases</td>
<td>4</td>
</tr>
<tr>
<td>Two courses, from different areas:</td>
<td></td>
<td>6-9</td>
</tr>
<tr>
<td>1) Information-based AI applications</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td></td>
</tr>
<tr>
<td>CS 229A</td>
<td>(Not given this year)</td>
<td></td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
<td></td>
</tr>
</tbody>
</table>

2) Database and Information Systems

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>4</td>
</tr>
<tr>
<td>CS 142</td>
<td>Web Applications</td>
<td></td>
</tr>
<tr>
<td>CS 245</td>
<td>Database Systems Principles</td>
<td></td>
</tr>
<tr>
<td>CS 246</td>
<td>Mining Massive Data Sets</td>
<td></td>
</tr>
<tr>
<td>CS 341</td>
<td>Project in Mining Massive Data Sets</td>
<td></td>
</tr>
<tr>
<td>CS 345</td>
<td>(Offered occasionally)</td>
<td></td>
</tr>
<tr>
<td>CS 346</td>
<td>Database System Implementation</td>
<td></td>
</tr>
<tr>
<td>CS 347</td>
<td>Parallel and Distributed Data Management</td>
<td></td>
</tr>
</tbody>
</table>

3) Information Systems in Biology

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td></td>
</tr>
<tr>
<td>CS 270</td>
<td>Modeling Biomedical Systems: Ontology, Terminology, Problem Solving</td>
<td></td>
</tr>
<tr>
<td>CS 274</td>
<td>Representations and Algorithms for Computational Molecular Biology</td>
<td></td>
</tr>
</tbody>
</table>

4) Information Systems on the Web

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td></td>
</tr>
<tr>
<td>CS 276</td>
<td>Information Retrieval and Web Search</td>
<td></td>
</tr>
<tr>
<td>CS 364B</td>
<td>(Not given this year)</td>
<td></td>
</tr>
</tbody>
</table>

At least three additional courses from the above areas or the general CS electives list.

Systems Track—

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>4</td>
</tr>
</tbody>
</table>

Select one of the following: 3-4

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 143</td>
<td>Compilers</td>
<td></td>
</tr>
<tr>
<td>EE 180</td>
<td>Digital Systems Architecture</td>
<td></td>
</tr>
</tbody>
</table>

Two additional courses from the list above or the following: 6-8

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
<td></td>
</tr>
<tr>
<td>CS 145</td>
<td>Introduction to Databases</td>
<td></td>
</tr>
<tr>
<td>CS 149</td>
<td>Parallel Computing</td>
<td></td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td></td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td></td>
</tr>
<tr>
<td>CS 242</td>
<td>Programming Languages</td>
<td></td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
<td></td>
</tr>
<tr>
<td>CS 244</td>
<td>Advanced Topics in Networking</td>
<td></td>
</tr>
<tr>
<td>CS 245</td>
<td>Database Systems Principles</td>
<td></td>
</tr>
<tr>
<td>EE 271</td>
<td>Introduction to VLSI Systems</td>
<td></td>
</tr>
<tr>
<td>EE 282</td>
<td>Computer Systems Architecture</td>
<td></td>
</tr>
</tbody>
</table>

Track Electives: at least three additional courses selected from the list above, the general CS electives list, or the following: 9-12

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 240E</td>
<td>Embedded Systems Workshop</td>
<td></td>
</tr>
<tr>
<td>CS 241</td>
<td>Embedded Systems Workshop</td>
<td></td>
</tr>
<tr>
<td>CS 244C</td>
<td>Readings and Projects in Distributed Systems</td>
<td></td>
</tr>
<tr>
<td>CS 244E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 315A</td>
<td>Parallel Computer Architecture and Programming</td>
<td></td>
</tr>
<tr>
<td>CS 316</td>
<td>Advanced Multi-Core Systems</td>
<td></td>
</tr>
<tr>
<td>CS 341</td>
<td>Project in Mining Massive Data Sets</td>
<td></td>
</tr>
<tr>
<td>CS 343</td>
<td>(Not given this year)</td>
<td></td>
</tr>
<tr>
<td>CS 344</td>
<td>Topics in Computer Networks</td>
<td></td>
</tr>
<tr>
<td>CS 345</td>
<td>(Offered occasionally)</td>
<td></td>
</tr>
<tr>
<td>CS 346</td>
<td>Database System Implementation</td>
<td></td>
</tr>
<tr>
<td>CS 347</td>
<td>Parallel and Distributed Data Management</td>
<td></td>
</tr>
<tr>
<td>CS 349</td>
<td>Topics in Programming Systems (with permission of undergraduate advisor)</td>
<td></td>
</tr>
</tbody>
</table>
**Theory Track**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
</tr>
<tr>
<td>CS 167</td>
<td>Readings in Algorithms (Not given this year)</td>
</tr>
<tr>
<td>CS 168</td>
<td>The Modern Algorithmic Toolbox</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
</tr>
<tr>
<td>CS 261</td>
<td>Optimization and Algorithmic Paradigms</td>
</tr>
<tr>
<td>CS 264</td>
<td></td>
</tr>
<tr>
<td>CS 265</td>
<td>Randomized Algorithms and Probabilistic Analysis</td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
</tr>
<tr>
<td>CS 361A</td>
<td></td>
</tr>
<tr>
<td>CS 361B</td>
<td></td>
</tr>
</tbody>
</table>

Two additional courses from the list above or the following: 6-8

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 143</td>
<td>Compilers</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
</tr>
<tr>
<td>CS 157</td>
<td>Logic and Automated Reasoning</td>
</tr>
<tr>
<td>PHIL 151</td>
<td>Metalogic</td>
</tr>
<tr>
<td>CS 166</td>
<td>Data Structures</td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
</tr>
<tr>
<td>CS 242</td>
<td>Programming Languages</td>
</tr>
<tr>
<td>CS 254</td>
<td></td>
</tr>
<tr>
<td>CS 259</td>
<td>((With adviser consent); Not given this year)</td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
</tr>
<tr>
<td>CS 263</td>
<td>Algorithms for Modern Data Models</td>
</tr>
<tr>
<td>CS 266</td>
<td></td>
</tr>
<tr>
<td>CS 267</td>
<td>Graph Algorithms</td>
</tr>
<tr>
<td>CS 354</td>
<td>Topics in Circuit Complexity (Not given this year)</td>
</tr>
<tr>
<td>CS 355</td>
<td>(Not given this year)</td>
</tr>
<tr>
<td>CS 357</td>
<td>Advanced Topics in Formal Methods (Not given this year)</td>
</tr>
<tr>
<td>CS 358</td>
<td>Topics in Programming Language Theory</td>
</tr>
<tr>
<td>CS 359</td>
<td>Topics in the Theory of Computation (with adviser consent)</td>
</tr>
<tr>
<td>CS 364A</td>
<td>Algorithmic Game Theory</td>
</tr>
<tr>
<td>CS 364B</td>
<td>(Not given this year)</td>
</tr>
<tr>
<td>CS 366</td>
<td>(Not given this year)</td>
</tr>
<tr>
<td>CS 367</td>
<td>Algebraic Graph Algorithms (Not given this year)</td>
</tr>
<tr>
<td>CS 369</td>
<td>Topics in Analysis of Algorithms (with adviser consent)</td>
</tr>
<tr>
<td>CS 374</td>
<td>Algorithms in Biology</td>
</tr>
<tr>
<td>MSE 310</td>
<td>Linear Programming</td>
</tr>
</tbody>
</table>

**Unspecialized Track**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
</tr>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
</tr>
<tr>
<td>CS 143</td>
<td>Compilers</td>
</tr>
</tbody>
</table>

Select one of the following: 4

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
</tr>
</tbody>
</table>

Select one of the following: 3-4

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 242</td>
<td>Introduction to Databases</td>
</tr>
<tr>
<td>CS 243</td>
<td>Introduction to Human-Computer Interaction Design</td>
</tr>
<tr>
<td>CS 248</td>
<td>Interactive Computer Graphics</td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
</tr>
</tbody>
</table>

At least two courses from the general CS electives list 5

**Individually Designed Track**

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

**Senior Capstone Project (3 units minimum)**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 191</td>
<td>Senior Project</td>
</tr>
<tr>
<td>CS 191W</td>
<td>Writing Intensive Senior Project</td>
</tr>
<tr>
<td>CS 194</td>
<td>Software Project</td>
</tr>
<tr>
<td>CS 194H</td>
<td>User Interface Design Project</td>
</tr>
<tr>
<td>CS 210B</td>
<td>Software Project Experience with Corporate Partners</td>
</tr>
<tr>
<td>CS 294W</td>
<td>Writing Intensive Research Project in Computer Science</td>
</tr>
</tbody>
</table>

For additional information and sample programs see the Handbook for Undergraduate Engineering Programs (UGHB) (http://ughb.stanford.edu)
1 MATH 19, MATH 20, and MATH 21 may be taken instead of MATH 41 and MATH 42 as long as at least 26 MATH units are taken. AP Calculus must be approved by the School of Engineering.

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

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

4 Students who take ENGR 40A (3 units) are required to take two additional units of ENGR Fundamentals (13 units minimum), or 2 additional units of Depth (27 units minimum for track and elective courses).


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

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

Honors Program

The Department of Computer Science (CS) offers an honors program for undergraduates whose academic records and personal initiative indicate that they have the necessary skills to undertake high-quality research in computer science. Admission to the program is by application only. To apply for the honors program, students must be majoring in Computer Science, have a grade point average (GPA) of at least 3.6 in courses that count toward the major, and achieve senior standing (135 or more units) by the end of the academic year in which they apply. C-terminal master’s students are eligible to apply as long as they have not already received their undergraduate degree. Beyond these requirements, students who apply for the honors program must find a Computer Science faculty member who agrees to serve as the thesis adviser for the project. Thesis advisers must be members of Stanford’s Academic Council.

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

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

1. Complete at least 9 units of CS 191 or CS 191W under the direction of their project sponsor.
2. Attend a weekly honors seminar Winter and Spring quarters.
3. Complete an honors thesis deemed acceptable by the thesis adviser and at least one additional faculty member.
4. Present the thesis at a public colloquium sponsored by the department.
5. Maintain the 3.6 GPA required for admission to the honors program.

Guide to Choosing Introductory Courses

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

For students whose principal interest is an exposure to the fundamental ideas behind computer science and programming, CS 101 or CS 105 are the most appropriate courses. They are intended for students in nontechnical disciplines who expect to make some use of computers, but who do not expect to go on to more advanced courses. CS 101 and CS 105 meet the new Ways of Thinking Ways of Doing breadth requirements in Formal Reasoning and include an introduction to programming and the use of modern Internet-based technologies. Students interested in learning to use the computer should consider CS 1C, Introduction to Computing at Stanford.

Students who intend to pursue a serious course of study in computer science may enter the program at a variety of levels, depending on their background. Students with little prior experience or those who wish to take more time to study the fundamentals of programming should take CS 106A followed by CS 106B. Students in CS 106A need not have prior programming experience. Students with significant prior exposure to programming or those who want an intensive introduction to the field should take CS 106X or may start directly in CS 106B. CS 106A uses Java as its programming language; CS 106B and X use C++. No prior knowledge of these languages is assumed, and the prior programming experience required for CS 106B or X may be in any language. In all cases, students are encouraged to discuss their background with the instructors responsible for these courses.

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

In summary:

For exposure:

<table>
<thead>
<tr>
<th>Course</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 1C</td>
<td>Introduction to Computing at Stanford</td>
</tr>
</tbody>
</table>

For nontechnical use:

<table>
<thead>
<tr>
<th>Course</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 101</td>
<td>Introduction to Computing Principles</td>
</tr>
<tr>
<td>or CS 105</td>
<td>Introduction to Computers</td>
</tr>
</tbody>
</table>

For scientific use:

<table>
<thead>
<tr>
<th>Course</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 106A</td>
<td>Programming Methodology</td>
</tr>
</tbody>
</table>

For a technical introduction:
Overseas Studies Courses in Computer Science

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

Joint Major Program: Computer Science and a Humanities Major

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

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

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

Mission

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

Computer Science Major Requirements in the Joint Major Program

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

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

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

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

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

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

Computer Engineering Track:
• EE 108A and 108B
• One of the following: EE 101A, 101B, 102A, 102B
• Satisfy the requirements of one of the following concentrations:
  1. Digital Systems Concentration: CS 140 or 143; EE 109, 271; plus one of CS 140 or 143 (if not counted above), 144, 149, 240E, 244; EE 273, 282
  2. Robotics and Mechatronics Concentration: CS 205A, 223A; ME 210; ENGR 105
  3. Networking Concentration: CS 140, 144; plus two of the following, CS 240, 240E, 244, 244B, 244E, 249A, 249B, EE 179, EE 276

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

HCI Track:
No Interdisciplinary HCI Electives required
Information Track:
One Track Elective (rather than three)

Systems Track:
One Track Elective (rather than three)

Theory Track:
One Track Elective (rather than three)

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

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

Declaring a Joint Major Program

To declare the joint major, students must first declare each major through Axess, and then submit the Declaration or Change of Undergraduate Major,
Minor, Honors, or Degree Program. (http://studentaffairs.stanford.edu/sites/default/files/registrar/files/change_UG_program.pdf) The Major-Minor and Multiple Major Course Approval Form (http://studentaffairs.stanford.edu/sites/default/files/registrar/files/MajMin_MultMaj.pdf) is required for graduation for students with a joint major.

**Dropping a Joint Major Program**

To drop the joint major, students must submit the Declaration or Change of Undergraduate Major, Minor, Honors, or Degree Program. (http://studentaffairs.stanford.edu/sites/default/files/registrar/files/change_UG_program.pdf). Students may also consult the Student Services Center (http://studentaffairs.stanford.edu/student-services-center) with questions concerning dropping the joint major.

**Transcript and Diploma**

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

**Computer Science (CS) Minor**

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

### Introductory Programming (AP Credit may be used to fulfill this requirement):

- **CS 106B** Programming Abstractions 5
- **or CS 106X** Programming Abstractions (Accelerated)

### Core:

- **CS 103** Mathematical Foundations of Computing 5
- **CS 107** Computer Organization and Systems 5
- **or CS 107E** Computer Systems from the Ground Up
- **CS 109** Introduction to Probability for Computer Scientists 5

Electives (choose two courses from different areas):

**Artificial Intelligence—**

- **CS 124** From Languages to Information 4
- **CS 221** Artificial Intelligence: Principles and Techniques 4
- **CS 229** Machine Learning 3-4

**Human-Computer Interaction—**

- **CS 147** Introduction to Human-Computer Interaction Design 4

**Software—**

- **CS 108** Object-Oriented Systems Design 4
- **CS 110** Principles of Computer Systems 5

**Systems—**

- **CS 140** Operating Systems and Systems Programming 4
- **CS 143** Compilers 4
- **CS 144** Introduction to Computer Networking 4
- **CS 145** Introduction to Databases 4
- **CS 148** Introduction to Computer Graphics and Imaging 4

**Theory—**

- **CS 154** Introduction to Automata and Complexity Theory 4
- **CS 157** Logic and Automated Reasoning 3
- **CS 161** Design and Analysis of Algorithms 5

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

---

**Master of Science in Computer Science**

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

**Admission**

Applications to the M.S. program and all supporting documents must be submitted and received online by the published deadline. Information on admission requirements and deadlines is available at http://cs.stanford.edu/admissions/. Exceptions are made for applicants who are already students at Stanford and are applying to the coterminal program. See http://cs/content/coterminal-program-deadline.

**University Coterminal Requirements**

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

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

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

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

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

**Requirements**

A candidate is required to complete a program of 45 units. At least 36 of these must be graded units, passed with a grade point average (GPA) of 3.0 (B) or better. The 45 units may include no more than 10 units of courses from those listed below in Requirement 1. Thus, students needing to take more than two of the courses listed in Requirement 1 actually complete more than 45 units of course work in the program. Only well-prepared students may expect to finish the program in one year; most students complete the program in six quarters. Students hoping to complete the program with 45 units should already have a substantial background in computer science, including course work or experience equivalent to all of Requirement 1 and some prior course work related to their specialization area.
Requirement 1: Foundations—
Students must complete the following courses, or waive out of them by providing evidence to their advisers that similar or more advanced courses have been taken, either at Stanford or another institution (total units used to satisfy foundations requirement may not exceed 10):

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 103</td>
<td>Mathematical Foundations of Computing Probability</td>
</tr>
</tbody>
</table>

Select one of the following:

- CS 109 Introduction to Probability for Computer Scientists
- STATS 116 Theory of Probability
- MSE 220 Probabilistic Analysis
- CME 106 Introduction to Probability and Statistics for Engineers

Requirement 2: Significant Software Implementation—
Students must complete at least one course designated as having a significant software implementation component. The list of such courses includes:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
</tr>
<tr>
<td>CS 143</td>
<td>Compilers</td>
</tr>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
</tr>
<tr>
<td>CS 145</td>
<td>Introduction to Databases</td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
</tr>
<tr>
<td>CS 210B</td>
<td>Software Project Experience with Corporate Partners</td>
</tr>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
</tr>
<tr>
<td>CS 248</td>
<td>Interactive Computer Graphics</td>
</tr>
<tr>
<td>CS 341</td>
<td>Project in Mining Massive Data Sets</td>
</tr>
<tr>
<td>CS 346</td>
<td>Database System Implementation (no longer offered)</td>
</tr>
</tbody>
</table>

Requirement 3: Specialization—
Students may choose to satisfy this requirement through one of two options, Single Depth or Dual Depth, outlined following. All courses taken for this requirement must be taken for letter grades.

Option 1—Single Depth
- A program of 27 units in a single area of specialization must be completed. A maximum of 9 units of independent study (CS 393, CS 395, CS 399) may be counted toward the specialization.
- Additionally, students must complete three breadth courses from the list of approved breadth courses associated with their chosen specialization. Individual specializations explicitly have different breadth requirements; see the individual specialization sheets at http://cs.stanford.edu/degrees/mscs/programsheets for details.
- Breadth courses may not be waived, must be taken for at least 3 units each, and must be completed for a letter grade.

Option 2—Dual Depth
- Students select distinct primary and secondary areas.
- A program of 39 units in the primary and secondary area of specialization must be completed. A maximum of 9 units of independent study (CS 393, CS 395, CS 399) may be counted toward the primary specialization.
- Students must also complete a program of five courses satisfying the requirements for their secondary area of specialization.
- Breadth courses are not required.

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

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

1. Artificial Intelligence—
   A.
   - CS 221 Artificial Intelligence: Principles and Techniques **
   B. Select at least four of the following:
     - CS 223A Introduction to Robotics
     - CS 224M Multi-Agent Systems
     - CS 224N Natural Language Processing
     - CS 224S Spoken Language Processing
     - CS 224U Natural Language Understanding
     - CS 224W Social Information and Network Analysis
     - CS 228 Probabilistic Graphical Models: Principles and Techniques
     - CS 229 Machine Learning
     - CS 231A Computer Vision: From 3D Reconstruction to Recognition
   C. Sufficient depth units from category (b) and the following:
     - CS 173 A Computational Tour of the Human Genome or CS 273A A Computational Tour of the Human Genome
     - CS 225A Experimental Robotics
     - CS 225B Robot Programming Laboratory
     - CS 227B General Game Playing
     - CS 229A (Not given this year)
     - CS 229T Statistical Learning Theory
     - CS 231B The Cutting Edge of Computer Vision
     - CS 231M Mobile Computer Vision
     - CS 231N Convolutional Neural Networks for Visual Recognition
     - CS 232 Digital Image Processing
     - CS 233 The Shape of Data: Geometric and Topological Data Analysis
     - CS 238 Decision Making under Uncertainty
     - CS 239 Advanced Topics in Sequential Decision Making
     - CS 246 Mining Massive Data Sets
     - CS 262 Computational Genomics
     - CS 270 Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
     - CS 274 Representations and Algorithms for Computational Molecular Biology
Those students who have waived out of CS 221 may take an additional course in either area (b) or (c).

Students with a secondary area of specialization (per Option 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), or (c) requirements above.

Those students who have waived out of CS 221 may take an additional course in either area (b) or (c).

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

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 143</td>
<td>Compilers</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 145</td>
<td>Introduction to Databases</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 147</td>
<td>Introduction to Human-Computer Interaction Design</td>
<td>3-5</td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 149</td>
<td>Parallel Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td>3</td>
</tr>
<tr>
<td>CS 157</td>
<td>Logic and Automated Reasoning</td>
<td>3</td>
</tr>
<tr>
<td>CS 166</td>
<td>Data Structures</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 168</td>
<td>The Modern Algorithmic Toolbox</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 240E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 249A</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td>3</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td>3</td>
</tr>
<tr>
<td>CS 261</td>
<td>Optimization and Algorithmic Paradigms</td>
<td>3</td>
</tr>
<tr>
<td>CS 264</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 265</td>
<td>Randomized Algorithms and Probabilistic Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 266</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 267</td>
<td>Graph Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CME 108</td>
<td>Introduction to Scientific Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CME 302</td>
<td>Numerical Linear Algebra</td>
<td>3</td>
</tr>
<tr>
<td>EE 180</td>
<td>Digital Systems Architecture</td>
<td>3-4</td>
</tr>
<tr>
<td>EE 282</td>
<td>Computer Systems Architecture</td>
<td>3</td>
</tr>
</tbody>
</table>

2. Biocomputation—

A. Select at least four of the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 173</td>
<td>A Computational Tour of the Human Genome</td>
<td></td>
</tr>
<tr>
<td>CS 173A</td>
<td>A Computational Tour of the Human Genome</td>
<td></td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td></td>
</tr>
<tr>
<td>CS 270</td>
<td>Modeling Biomedical Systems: Ontology, Terminology, Problem Solving</td>
<td></td>
</tr>
<tr>
<td>CS 272</td>
<td>Introduction to Biomedical Informatics Research Methodology</td>
<td></td>
</tr>
<tr>
<td>CS 274</td>
<td>Representations and Algorithms for Computational Molecular Biology</td>
<td></td>
</tr>
<tr>
<td>CS 279</td>
<td>Computational Biology: Structure and Organization of Biomolecules and Cells</td>
<td></td>
</tr>
</tbody>
</table>

B. Sufficient depth units from category (a) and the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td></td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td></td>
</tr>
<tr>
<td>CS 229A</td>
<td>(Not given this year)</td>
<td></td>
</tr>
</tbody>
</table>
Biocomputation Breadth Courses

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

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td>3</td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td>2-4</td>
</tr>
<tr>
<td>CS 224U</td>
<td>Natural Language Understanding</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td>3</td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
<td>3</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 240E</td>
<td>Computer Systems Architecture</td>
<td>3</td>
</tr>
<tr>
<td>CS 249A</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td>3</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td>3</td>
</tr>
<tr>
<td>CS 276</td>
<td>Information Retrieval and Web Search</td>
<td>3</td>
</tr>
<tr>
<td>CME 108</td>
<td>Introduction to Scientific Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CME 302</td>
<td>Numerical Linear Algebra</td>
<td>3</td>
</tr>
<tr>
<td>EE 180</td>
<td>Digital Systems Architecture</td>
<td>3-4</td>
</tr>
<tr>
<td>EE 282</td>
<td>Computer Systems Architecture</td>
<td>3</td>
</tr>
</tbody>
</table>

3. Computer and Network Security—

A. Select at least three of the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>**</td>
</tr>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
<td>**</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td>3</td>
</tr>
<tr>
<td>CS 244</td>
<td>Advanced Topics in Networking</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td>3</td>
</tr>
</tbody>
</table>

B. Select at least three of the following:

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 142</td>
<td>Web Applications</td>
<td>3</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 244B</td>
<td>Distributed Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 244C</td>
<td>Readings and Projects in Distributed Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 261</td>
<td>Optimization and Algorithmic Paradigms</td>
<td>3</td>
</tr>
<tr>
<td>CS 265</td>
<td>Randomized Algorithms and Probabilistic Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 340</td>
<td>Topics in Computer Networks</td>
<td>3</td>
</tr>
<tr>
<td>CS 344</td>
<td>Topics in Computer Networks</td>
<td>3</td>
</tr>
<tr>
<td>CS 355</td>
<td>(Not given this year)</td>
<td>3</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 240E</td>
<td>Embedded Systems Workshop</td>
<td>3</td>
</tr>
<tr>
<td>CS 244E</td>
<td>Computer Systems Architecture</td>
<td>3</td>
</tr>
<tr>
<td>CS 245</td>
<td>Database Systems Principles</td>
<td>3</td>
</tr>
<tr>
<td>CS 251</td>
<td>Bitcoin and Crypto Currencies</td>
<td>3</td>
</tr>
<tr>
<td>CS 264</td>
<td>(Offered occasionally)</td>
<td>3</td>
</tr>
<tr>
<td>CS 294S</td>
<td>Research Project in Software Systems and Security</td>
<td>(Not given this year)</td>
</tr>
<tr>
<td>CS 341</td>
<td>Project in Mining Massive Data Sets</td>
<td>3</td>
</tr>
<tr>
<td>CS 345</td>
<td>(Offered occasionally)</td>
<td>3</td>
</tr>
<tr>
<td>CS 347</td>
<td>Parallel and Distributed Data Management</td>
<td>3</td>
</tr>
<tr>
<td>CS 361A</td>
<td>Computational Structural Biology</td>
<td>3-4</td>
</tr>
</tbody>
</table>

- Students with a secondary area of specialization (per Option 2 above) must come from area (a) and the remaining two courses may come from either area (a) or (b).
- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) and (b) requirements above.
Computer and Network Security Breadth Courses

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 268</td>
<td>Computer and Network Security Breadth</td>
<td></td>
</tr>
<tr>
<td>CS 269</td>
<td>Internet Routing Protocols and Standards</td>
<td></td>
</tr>
<tr>
<td>EE 384A</td>
<td>Wireless Local and Wide Area Networks</td>
<td></td>
</tr>
<tr>
<td>EE 384S</td>
<td>Performance Engineering of Computer Systems &amp; Networks</td>
<td></td>
</tr>
<tr>
<td>EE 384X</td>
<td>Packet Switch Architectures</td>
<td></td>
</tr>
</tbody>
</table>

• Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.

4. Human-Computer Interaction—

A.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 147</td>
<td>Introduction to Human-Computer Interaction Design</td>
<td></td>
</tr>
<tr>
<td>CS 247</td>
<td>Human-Computer Interaction Design Studio</td>
<td></td>
</tr>
</tbody>
</table>

Students with equivalent course work may waive with adviser approval

B. Select any three of the following:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 142</td>
<td>Web Applications</td>
<td></td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
<td></td>
</tr>
<tr>
<td>CS 194H</td>
<td>User Interface Design Project</td>
<td></td>
</tr>
<tr>
<td>CS 210A</td>
<td>Software Project Experience with Corporate Partners</td>
<td></td>
</tr>
<tr>
<td>CS 248</td>
<td>Interactive Computer Graphics</td>
<td></td>
</tr>
<tr>
<td>CS 376</td>
<td>Human-Computer Interaction Research</td>
<td></td>
</tr>
<tr>
<td>CS 377</td>
<td>Topics in Human-Computer Interaction (Any CS 377 A/B/C suffix)</td>
<td></td>
</tr>
<tr>
<td>CS 448B</td>
<td>Data Visualization</td>
<td></td>
</tr>
<tr>
<td>ME 216M</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

a. Broader CS

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
<td></td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td></td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td></td>
</tr>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td></td>
</tr>
<tr>
<td>CS 224U</td>
<td>Natural Language Understanding</td>
<td></td>
</tr>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td></td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td></td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td></td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td></td>
</tr>
<tr>
<td>CS 229A</td>
<td>(Not given this year)</td>
<td></td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
<td></td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
<td></td>
</tr>
<tr>
<td>CS 242</td>
<td>Programming Languages</td>
<td></td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
<td></td>
</tr>
<tr>
<td>CS 246</td>
<td>Mining Massive Data Sets</td>
<td></td>
</tr>
<tr>
<td>CS 249A</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td></td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td></td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b. Art Studio

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARTSTUDI 160</td>
<td>Intro to Digital / Physical Design</td>
<td></td>
</tr>
<tr>
<td>ARTSTUDI 162</td>
<td>Embodied Interfaces</td>
<td></td>
</tr>
<tr>
<td>ARTSTUDI 163</td>
<td>Drawing with Code</td>
<td></td>
</tr>
</tbody>
</table>
### Computer Science

**ARTSTUDI 164**
Design in Public Spaces

**ARTSTUDI 165**
Social Media and Performative Practices

**ARTSTUDI 168**
Data as Material

**ARTSTUDI 264**
Advanced Interaction Design

**ARTSTUDI 266**
Sculptural Screens / Malleable Media

**ARTSTUDI 267**
Emerging Technology Studio

c. Communication

**COMM 224**
Digital Deception

**COMM 240**
Digital Media Entrepreneurship

**COMM 266**
Virtual People

**COMM 269**
Computers and Interfaces

**COMM 272**
Media Psychology

Comm 282

**COMM 324**
Language and Technology
d. Empirical Methods

**COMM 314**
Qualitative Social Science Research Methods

**EDUC 200B**
Introduction to Qualitative Research Methods

**EDUC 291X**
Introduction to Applied Statistics

**PSYCH 252**
Statistical Methods for Behavioral and Social Sciences

**PSYCH 254**
Lab in Experimental Methods

**STATS 203**
Introduction to Regression Models and Analysis of Variance
e. Learning Design & Tech

**EDUC 239X**

**EDUC 281X**

**EDUC 337X**
f. Man Sci & Engr

**MSE 185**
Global Work

**MSE 331**
The Structure of Social Data

g. Mechanical Engr

**ME 203**
Design and Manufacturing

**ME 210**
Introduction to Mechatronics

**ME 216A**
Advanced Product Design: Needfinding

h. Music

**MUSIC 220A**
Fundamentals of Computer-Generated Sound

**MUSIC 220B**
Compositional Algorithms, Psychoacoustics, and Computational Music

**MUSIC 220C**
Research Seminar in Computer-Generated Music

**MUSIC 250A**
Physical Interaction Design for Music

i. Psych

**PSYCH 204**
Computation and cognition: the probabilistic approach

**PSYCH 209**
Neural network and deep learning models for cognition and cognitive neuroscience

j. Sym Sys

**SYMSYS 245**
Cognition in Interaction Design

Any d.school class listed at http://dschool.stanford.edu, or any HCI class listed at http://hci.stanford.edu/courses/; such courses must be numbered 100 or above and be taken for at least 3 units to count for this requirement

- Or any d.school class listed at http://dschool.stanford.edu, or any HCI class listed at http://hci.stanford.edu/courses. Such courses must be numbered 100 or above and be taken for at least 3 units to count for this requirement.
- Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a) through (c) requirements above.
- Students with a secondary area of specialization (per Option 2 above) in Human-Computer Interaction must take five courses satisfying the areas (a) through (c).

### Human-Computer Interaction Breadth Courses

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

<table>
<thead>
<tr>
<th>Course</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 143</td>
<td>Compilers</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
<td>3-4</td>
</tr>
<tr>
<td>or EE 284</td>
<td>Introduction to Computer Networks</td>
<td></td>
</tr>
<tr>
<td>CS 145</td>
<td>Introduction to Databases</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 149</td>
<td>Parallel Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td>3</td>
</tr>
<tr>
<td>CS 157</td>
<td>Logic and Automated Reasoning</td>
<td>3</td>
</tr>
<tr>
<td>CS 166</td>
<td>Data Structures</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 168</td>
<td>The Modern Algorithmic Toolbox</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 173</td>
<td>A Computational Tour of the Human Genome</td>
<td>3</td>
</tr>
<tr>
<td>or CS 273A</td>
<td>A Computational Tour of the Human Genome</td>
<td></td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td>3</td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td>2-4</td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td>3</td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 240H</td>
<td>Functional Systems in Haskell</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 244</td>
<td>Advanced Topics in Networking</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 244B</td>
<td>Distributed Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 244E</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td>3</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td>3</td>
</tr>
<tr>
<td>CS 261</td>
<td>Optimization and Algorithmic Paradigms</td>
<td>3</td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td>3</td>
</tr>
<tr>
<td>CS 264</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 265</td>
<td>Randomized Algorithms and Probabilistic Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 266</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 267</td>
<td>Graph Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
### 5. Information Management and Analytics

A. CS 145 **Introduction to Databases** 3-4

B. Select at least four of the following:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 245</td>
<td>Database Systems Principles</td>
<td>3</td>
</tr>
<tr>
<td>CS 246</td>
<td>Mining Massive Data Sets</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 276</td>
<td>Information Retrieval and Web Search</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 345</td>
<td>(Offered occasionally)</td>
<td>3</td>
</tr>
<tr>
<td>CS 346</td>
<td>Database System Implementation (no longer offered)</td>
<td>3</td>
</tr>
<tr>
<td>CS 347</td>
<td>Parallel and Distributed Data Management</td>
<td>3-4</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking</td>
<td>3</td>
</tr>
<tr>
<td>CS 173</td>
<td>A Computational Tour of the Human Genome</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td>3</td>
</tr>
<tr>
<td>CS 224U</td>
<td>Natural Language Understanding</td>
<td>3</td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td>3</td>
</tr>
<tr>
<td>CS 229A</td>
<td>(Not given this year)</td>
<td>3</td>
</tr>
<tr>
<td>CS 229T</td>
<td>Statistical Learning Theory</td>
<td>3</td>
</tr>
<tr>
<td>CS 231A</td>
<td>Computer Vision: From 3D Reconstruction to Recognition</td>
<td>3</td>
</tr>
<tr>
<td>CS 231N</td>
<td>Convolutional Neural Networks for Visual Recognition</td>
<td>3</td>
</tr>
<tr>
<td>CS 233</td>
<td>The Shape of Data: Geometric and Topological Data Analysis</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 242</td>
<td>Programming Languages</td>
<td>3</td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 244</td>
<td>Advanced Topics in Networking</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 244B</td>
<td>Distributed Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 244C</td>
<td>Readings and Projects in Distributed Systems</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 249A</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 249B</td>
<td>Large-scale Software Development</td>
<td>3</td>
</tr>
<tr>
<td>CS 251</td>
<td>Bitcoin and Crypto Currencies</td>
<td>3</td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td>3</td>
</tr>
<tr>
<td>CS 270</td>
<td>Modeling Biomedical Systems: Ontology, Terminology, Problem Solving</td>
<td>3-4</td>
</tr>
</tbody>
</table>

### Information Management and Analytics Breadth Courses

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 124</td>
<td>From Languages to Information</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 147</td>
<td>Introduction to Human-Computer Interaction Design</td>
<td>3-5</td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 149</td>
<td>Parallel Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td>3</td>
</tr>
<tr>
<td>CS 157</td>
<td>Logic and Automated Reasoning</td>
<td>3</td>
</tr>
<tr>
<td>CS 166</td>
<td>Data Structures</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 168</td>
<td>The Modern Algorithmic Toolbox</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td>3</td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td>3</td>
</tr>
<tr>
<td>CS 240E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 244E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 261</td>
<td>Optimization and Algorithmic Paradigms</td>
<td>3</td>
</tr>
<tr>
<td>CS 264</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 265</td>
<td>Randomized Algorithms and Probabilistic Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 266</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>CS 267</td>
<td>Graph Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CS 268</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Students in the single depth specialization must complete three of the following:

A. Select two of the following:
   - CS 140 Operating Systems and Systems Programming
   - CS 144 Introduction to Computer Networking
   - CS 244 Advanced Topics in Networking

B. Select one of the following:
   - CS 142 Web Applications
   - CS 147 Introduction to Human-Computer Interaction Design
   - CS 247 Human-Computer Interaction Design Studio

C. Select one of the following:
   - CS 155 Computer and Network Security
   - CS 255 Introduction to Cryptography

D. CS 294S Research Project in Software Systems and Security

E. Sufficient depth units from categories (a) through (d) and the following:
   - CS 224W Social Information and Network Analysis
   - CS 241 Embedded Systems Workshop
   - CS 244E
   - CS 246 Mining Massive Data Sets
   - CS 251 Bitcoin and Crypto Currencies
   - CS 344 Topics in Computer Networks
   - CS 344E Advanced Wireless Networks
   - CS 364A Algorithmic Game Theory
   - CS 376 Human-Computer Interaction Research
   - CS 393 Computer Laboratory
   - CS 395 Independent Database Project
   - CS 399 Independent Project
   - EE 359 Wireless Communications
   - EE 384A Internet Routing Protocols and Standards
   - EE 384B Multimedia Communication over the Internet (not given this year)
   - EE 384C Wireless Local and Wide Area Networks
   - EE 384E Networked Wireless Systems
   - EE 384S Performance Engineering of Computer Systems & Networks
   - COMM 268
   - PSYCH 252 Statistical Methods for Behavioral and Social Sciences

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

Mobile and Internet Computing Breadth Courses

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

- CS 124 From Languages to Information
- CS 143 Compilers
- CS 145 Introduction to Databases
- CS 148 Introduction to Computer Graphics and Imaging
- CS 149 Parallel Computing
- CS 154 Introduction to Automata and Complexity Theory
- CS 157 Logic and Automated Reasoning
- CS 166 Data Structures
- CS 168 The Modern Algorithmic Toolbox
- CS 173 A Computational Tour of the Human Genome
- CS 221 Artificial Intelligence: Principles and Techniques
- CS 223A Introduction to Robotics
- CS 224M Multi-Agent Systems
- CS 224N Natural Language Processing
- CS 224S Spoken Language Processing
- CS 224U Natural Language Understanding
- CS 227B General Game Playing
- CS 228 Probabilistic Graphical Models: Principles and Techniques
- CS 229 Machine Learning
- CS 229A (Not given this year)
- CS 231A Computer Vision: From 3D Reconstruction to Recognition
- CS 233 The Shape of Data: Geometric and Topological Data Analysis
- CS 240 Advanced Topics in Operating Systems
- CS 240E (no longer offered)
- CS 240H Functional Systems in Haskell
- CS 242 Programming Languages
- CS 243 Program Analysis and Optimizations
- CS 244B Distributed Systems
- CS 249A Object-Oriented Programming from a Modeling and Simulation Perspective
- CS 261 Optimization and Algorithmic Paradigms
- CS 262 Computational Genomics
- CS 264
- CS 265 Randomized Algorithms and Probabilistic Analysis
- CS 266
- CS 267 Graph Algorithms
- CS 268
- CS 270 Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
- CS 274 Representations and Algorithms for Computational Molecular Biology
- CS 276 Information Retrieval and Web Search
- CS 279 Computational Biology: Structure and Organization of Biomolecules and Cells
- CME 108 Introduction to Scientific Computing
- CME 302 Numerical Linear Algebra
- EE 180 Digital Systems Architecture
- EE 282 Computer Systems Architecture

7. Real-World Computing—

A. Select at least three of the following:
   - CS 148 Introduction to Computer Graphics and Imaging
   - CS 223A Introduction to Robotics
CS 231A  Computer Vision: From 3D Reconstruction to Recognition
CS 248  Interactive Computer Graphics

B. Select at least three of the following:

CS 205B  Mathematical Methods for Fluids, Solids, and Interfaces
CS 233  The Shape of Data: Geometric and Topological Data Analysis
CS 249A  Object-Oriented Programming from a Modeling and Simulation Perspective
CS 249B  Large-scale Software Development
CS 262  Computational Genomics
CS 268
CS 277  Experimental Haptics
CS 348A  Computer Graphics: Geometric Modeling
CS 348B  Computer Graphics: Image Synthesis Techniques
CS 374  Algorithms in Biology
CME 302  Numerical Linear Algebra
CME 306  Numerical Solution of Partial Differential Equations

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

CS 173  A Computational Tour of the Human Genome
or CS 273A  A Computational Tour of the Human Genome
CS 225A  Experimental Robotics
CS 225B  Robot Programming Laboratory
CS 228  Probabilistic Graphical Models: Principles and Techniques
CS 229  Machine Learning
CS 229A  (Not given this year)
CS 231B  The Cutting Edge of Computer Vision
CS 231M  Mobile Computer Vision
CS 232  Digital Image Processing
or EE 368  Digital Image Processing
CS 247  Human-Computer Interaction Design Studio
CS 270  Modeling Biomedical Systems: Ontology, Terminology, Problem Solving
CS 272  Introduction to Biomedical Informatics Research Methodology
CS 274  Representations and Algorithms for Computational Molecular Biology
CS 294A  Research Project in Artificial Intelligence *
CS 327A  Advanced Robotic Manipulation (Not given this year)
CS 328  Topics in Computer Vision
CS 331A  Advanced Reading in Computer Vision
CS 331B  3D Representation and Recognition
CS 393  Computer Laboratory *
CS 395  Independent Database Project *
CS 399  Independent Project *
CS 448  Topics in Computer Graphics
EE 267  Virtual Reality

• Students with a 27- or 21-unit depth option (Option 1 or 2 above) must take 27 or 21 units respectively subject to satisfying the area (a), (b), and (c) requirements above.

**Real-World Computing Breadth Courses**

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

CS 124  From Languages to Information 3-4
CS 140  Operating Systems and Systems Programming 3-4
CS 143  Compilers 3-4
CS 144  Introduction to Computer Networking 3-4
or EE 284  Introduction to Computer Networks 3-4
CS 145  Introduction to Databases 3-4
CS 147  Introduction to Human-Computer Interaction Design 3-5
CS 149  Parallel Computing 3-4
CS 154  Introduction to Automata and Complexity Theory 3-4
CS 155  Computer and Network Security 3
CS 157  Logic and Automated Reasoning 3
CS 166  Data Structures 3-4
CS 168  The Modern Algorithmic Toolbox 3-4
CS 221  Artificial Intelligence: Principles and Techniques 3-4
CS 224M  Multi-Agent Systems 3
CS 224N  Natural Language Processing 3-4
CS 224S  Spoken Language Processing 2-4
CS 224U  Natural Language Understanding 3-4
CS 224W  Social Information and Network Analysis 3
CS 227B  General Game Playing 3
CS 240  Advanced Topics in Operating Systems 3
CS 240E  (no longer offered) 3-4
CS 240H  Functional Systems in Haskell 3-4
CS 242  Programming Languages 3
CS 243  Program Analysis and Optimizations 3-4
CS 244  Advanced Topics in Networking 3-4
CS 244B  Distributed Systems 3
CS 244E
CS 246  Mining Massive Data Sets 3
CS 255  Introduction to Cryptography 3
CS 261  Optimization and Algorithmic Paradigms 3
CS 264
CS 265  Randomized Algorithms and Probabilistic Analysis 3
CS 266
CS 267  Graph Algorithms 3
CS 276  Information Retrieval and Web Search 3
CS 279  Computational Biology: Structure and Organization of Biomolecules and Cells 3
CME 108  Introduction to Scientific Computing 3-4
EE 180  Digital Systems Architecture 3-4
EE 282  Computer Systems Architecture 3

**8. Software Theory—**

A. CS 243  Program Analysis and Optimizations

B. Select at least one of the following:

CS 244  Advanced Topics in Networking
CS 245  Database Systems Principles
CS 341  Project in Mining Massive Data Sets
CS 229A (Not given this year) 3-4
CS 231A Computer Vision: From 3D Reconstruction to Recognition 3
CS 233 The Shape of Data: Geometric and Topological Data Analysis 3
CS 240 Advanced Topics in Operating Systems 3
CS 240E (no longer offered) 2
CS 240H Functional Systems in Haskell 3-4
CS 244B Distributed Systems 3
CS 244E 3
CS 246 Mining Massive Data Sets 3-4
CS 249A Object-Oriented Programming from a Modeling and Simulation Perspective 3
CS 262 Computational Genomics 3
CS 270 Modeling Biomedical Systems: Ontology, Terminology, Problem Solving 3
CS 274 Representations and Algorithms for Computational Molecular Biology 3-4
CS 276 Information Retrieval and Web Search 3
CS 279 Computational Biology: Structure and Organization of Biomolecules and Cells 3
CME 108 Introduction to Scientific Computing 3-4
CME 302 Numerical Linear Algebra 3
EE 180 Digital Systems Architecture 3-4
EE 282 Computer Systems Architecture 3

9. Systems—

A.  

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming **</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 144</td>
<td>Introduction to Computer Networking **</td>
<td>3</td>
</tr>
<tr>
<td>CS 240</td>
<td>Advanced Topics in Operating Systems</td>
<td></td>
</tr>
</tbody>
</table>

B. Select at least four of the following:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 242</td>
<td>Programming Languages</td>
<td></td>
</tr>
<tr>
<td>CS 243</td>
<td>Program Analysis and Optimizations</td>
<td></td>
</tr>
<tr>
<td>CS 244</td>
<td>Advanced Topics in Networking</td>
<td></td>
</tr>
<tr>
<td>CS 245</td>
<td>Database Systems Principles</td>
<td></td>
</tr>
<tr>
<td>CS 248</td>
<td>Interactive Computer Graphics</td>
<td></td>
</tr>
<tr>
<td>CS 348B</td>
<td>Computer Graphics: Image Synthesis Techniques</td>
<td></td>
</tr>
<tr>
<td>EE 271</td>
<td>Introduction to VLSI Systems</td>
<td></td>
</tr>
<tr>
<td>EE 282</td>
<td>Computer Systems Architecture</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 240E</td>
<td>(no longer offered)</td>
<td>2</td>
</tr>
<tr>
<td>CS 240H</td>
<td>Functional Systems in Haskell</td>
<td></td>
</tr>
<tr>
<td>CS 241</td>
<td>Embedded Systems Workshop</td>
<td></td>
</tr>
<tr>
<td>CS 244B</td>
<td>Distributed Systems</td>
<td></td>
</tr>
<tr>
<td>CS 244C</td>
<td>Readings and Projects in Distributed Systems</td>
<td></td>
</tr>
<tr>
<td>CS 244E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 246</td>
<td>Mining Massive Data Sets</td>
<td></td>
</tr>
<tr>
<td>CS 249A</td>
<td>Object-Oriented Programming from a Modeling and Simulation Perspective</td>
<td></td>
</tr>
<tr>
<td>CS 249B</td>
<td>Large-scale Software Development</td>
<td></td>
</tr>
<tr>
<td>CS 251</td>
<td>Bitcoin and Crypto Currencies</td>
<td></td>
</tr>
<tr>
<td>CS 255</td>
<td>Introduction to Cryptography</td>
<td></td>
</tr>
<tr>
<td>CS 262</td>
<td>Computational Genomics</td>
<td></td>
</tr>
<tr>
<td>CS 270</td>
<td>Modeling Biomedical Systems: Ontology, Terminology, Problem Solving</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 229</td>
<td>Machine Learning</td>
<td>3-4</td>
</tr>
</tbody>
</table>

Software Theory Breadth Courses

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

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 124</td>
<td>From Languages to Information</td>
<td>3</td>
</tr>
<tr>
<td>CS 140</td>
<td>Operating Systems and Systems Programming</td>
<td>3</td>
</tr>
<tr>
<td>CS 147</td>
<td>Introduction to Human-Computer Interaction Design</td>
<td>3-5</td>
</tr>
<tr>
<td>CS 148</td>
<td>Introduction to Computer Graphics and Imaging</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 149</td>
<td>Parallel Computing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 154</td>
<td>Introduction to Automata and Complexity Theory</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 155</td>
<td>Computer and Network Security</td>
<td>3</td>
</tr>
<tr>
<td>CS 157</td>
<td>Logic and Automated Reasoning</td>
<td>3</td>
</tr>
<tr>
<td>CS 173</td>
<td>A Computational Tour of the Human Genome</td>
<td>3</td>
</tr>
<tr>
<td>CS 221</td>
<td>Artificial Intelligence: Principles and Techniques</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 223A</td>
<td>Introduction to Robotics</td>
<td>3</td>
</tr>
<tr>
<td>CS 224M</td>
<td>Multi-Agent Systems</td>
<td>3</td>
</tr>
<tr>
<td>CS 224N</td>
<td>Natural Language Processing</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224S</td>
<td>Spoken Language Processing</td>
<td>2-4</td>
</tr>
<tr>
<td>CS 224U</td>
<td>Natural Language Understanding</td>
<td>3-4</td>
</tr>
<tr>
<td>CS 224W</td>
<td>Social Information and Network Analysis</td>
<td>3</td>
</tr>
<tr>
<td>CS 227B</td>
<td>General Game Playing</td>
<td>3</td>
</tr>
<tr>
<td>CS 228</td>
<td>Probabilistic Graphical Models: Principles and Techniques</td>
<td>3-4</td>
</tr>
</tbody>
</table>

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

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

- CS 124 From Languages to Information 3-4
- CS 147 Introduction to Human-Computer Interaction Design 3-5
- CS 154 Introduction to Automata and Complexity Theory 3-4
- CS 155 Computer and Network Security 3
- CS 157 Logic and Automated Reasoning 3
- CS 166 Data Structures 3-4
- CS 168 The Modern Algorithmic Toolbox 3-4
- CS 173 A Computational Tour of the Human Genome 3
- CS 175 A Computational Tour of the Human Genome 3
- CS 221 Artificial Intelligence: Principles and Techniques 3-4
- CS 222A Introduction to Robotics 3
- CS 233 The Shape of Data: Geometric and Topological Data Analysis 3
- CS 244M Multi-Agent Systems 3
- CS 244N Natural Language Processing 3-4
- CS 244S Spoken Language Processing 2-4
- CS 244U Natural Language Understanding 3-4
- CS 244W Social Information and Network Analysis 3
- CS 247 General Game Playing 3
- CS 228 Probabilistic Graphical Models: Principles and Techniques 3-4
- CS 229 Machine Learning 3-4
- CS 229A (Not given this year) 3-4
- CS 231A Computer Vision: From 3D Reconstruction to Recognition 3
- CS 261 Optimization and Algorithmic Paradigms 3
- CS 264 3
- CS 265 Randomized Algorithms and Probabilistic Analysis 3
- CS 266 3
- CS 267 Graph Algorithms 3
- CS 268 3
- CS 274 Representations and Algorithms for Computational Molecular Biology 3-4
- CS 279 Computational Biology: Structure and Organization of Biomolecules and Cells 3
- CME 108 Introduction to Scientific Computing 3
- CME 302 Numerical Linear Algebra 3

10. Theoretical Computer Science—

A. CS 261 Optimization and Algorithmic Paradigms ***

B. Sufficient additional units chosen from:
- CS 166 Data Structures
- CS 168 The Modern Algorithmic Toolbox
- CS 228 Probabilistic Graphical Models: Principles and Techniques
- CS 233 The Shape of Data: Geometric and Topological Data Analysis
- CS 246 Mining Massive Data Sets
- CS 251 Bitcoin and Crypto Currencies
- CS 254
- CS 255 Introduction to Cryptography
- CS 262 Computational Genomics
- CS 263 Algorithms for Modern Data Models
- CS 264
- CS 265 Randomized Algorithms and Probabilistic Analysis
- CS 266
- CS 267 Graph Algorithms
- CS 268
- CS 334A Convex Optimization I
- CS 341 Project in Mining Massive Data Sets
- CS 345 (Offered occasionally)
- CS 354 Topics in Circuit Complexity (Not given this year)
- CS 355 (Not given this year)
- CS 357 Advanced Topics in Formal Methods (Not given this year)
CS 358  Topics in Programming Language Theory
CS 359  Topics in the Theory of Computation*
CS 361B
CS 362  (Not given this year)
CS 364A  Algorithmic Game Theory
CS 364B  (Not given this year)
CS 366  (Not given this year)
CS 367  Algebraic Graph Algorithms (Not given this year)
CS 369  Topics in Analysis of Algorithms*
CS 374  Algorithms in Biology (not given this year)
CS 393  Computer Laboratory*
CS 395  Independent Database Project*
CS 399  Independent Project*
CS 468  *

MSE 310  Linear Programming

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

**Theoretical Computer Science Breadth Courses**

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

CS 124  From Languages to Information 3-4
CS 140  Operating Systems and Systems Programming 3-4
CS 143  Compilers 3-4
CS 144  Introduction to Computer Networking or EE 284 3-4
Introduction to Computer Networks
CS 145  Introduction to Databases 3-4
CS 147  Introduction to Human-Computer Interaction Design 3-5
CS 148  Introduction to Computer Graphics and Imaging 3-4
CS 149  Parallel Computing 3-4
CS 154  Introduction to Automata and Complexity Theory 3-4
CS 155  Computer and Network Security 3
CS 157  Logic and Automated Reasoning 3
CS 173  A Computational Tour of the Human Genome or CS 273A 3
A Computational Tour of the Human Genome
CS 211  Artificial Intelligence: Principles and Techniques 3-4
CS 223A  Introduction to Robotics 3
CS 224M  Multi-Agent Systems 3
CS 224N  Natural Language Processing 3-4
CS 224S  Spoken Language Processing 2-4
CS 224U  Natural Language Understanding 3-4
CS 224W  Social Information and Network Analysis 3
CS 227B  General Game Playing 3
CS 229  Machine Learning 3-4
CS 229A  (Not given this year) 3-4

CS 231A  Computer Vision: From 3D Reconstruction to Recognition 3
CS 240  Advanced Topics in Operating Systems 3
CS 240E
CS 240H  Functional Systems in Haskell 3-4
CS 242  Programming Languages 3
CS 243  Program Analysis and Optimizations 3-4
CS 244  Advanced Topics in Networking 3-4
CS 244B  Distributed Systems 3
CS 244E
CS 249A  Object-Oriented Programming from a Modeling and Simulation Perspective 3
CS 270  Modeling Biomedical Systems: Ontology, Terminology, Problem Solving 3
CS 274  Representations and Algorithms for Computational Molecular Biology 3-4
CS 276  Information Retrieval and Web Search 3
CS 279  Computational Biology: Structure and Organization of Biomolecules and Cells 3
CME 108  Introduction to Scientific Computing 3-4
CME 302  Numerical Linear Algebra 3
EE 180  Digital Systems Architecture 3-4
EE 282  Computer Systems Architecture 3

* With consent of faculty adviser.
** Students with equivalent course work may waive with approval of their adviser.
*** CS 361B may be used as substitute for CS 261.

**Requirement 4**

Additional elective units must be technical courses (numbered 100 or
above) related to the degree program and approved by the adviser. All CS
courses numbered above 110 (with the exception of CS 196 and 198) taken
for 3 or more units are pre-approved as elective courses. Additionally, up
to a maximum of 3 units of 500-level CS seminars, CS 300, EE 380, EE
385A, or other 1-2 unit seminars offered in the School of Engineering may
be counted as electives. Elective courses may be taken on a satisfactory/no
credit basis provided that a minimum of 36 graded units is presented within
the 45-unit program.

**Master of Science with Distinction in Research**

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

1. **Research Experience**—The program must include significant research
experience at the level of a half-time commitment over the course of
three academic quarters. In any given quarter, the half-time research
commitment may be satisfied by a 50 percent appointment to a
departmentally supported research assistantship, 6 units of independent
study (CS 393, CS 395, or CS 399), or a prorated combination of
the two (such as a 25 percent research assistantship supplemented by 3
units of independent study). This research must be carried out under the
direction of the primary or secondary adviser.
2. **Supervised Writing and Research**—In addition to the research experience outlined in the previous requirement, students must enroll in at least 3 units of independent research (CS 393, CS 395, or CS 399) under the direction of their primary or secondary adviser. These units should be closely related to the research described in the first requirement, but focused more directly on the preparation of the research report described in the next section. The writing and research units described in parts (1) and (2) may be counted toward the 45 units required for the degree.

3. All independent study units (CS 393, CS 395, CS 399) must be taken for letter grades and a GPA of 3.0 (B) or better must be maintained.

4. **Research Report**—Students must complete a significant report describing their research and its conclusions. The research report represents work that is publishable in a journal or at a high-quality conference, although it is presumably longer and more expansive in scope than a typical conference paper. A copy of the research report must be submitted to the student services office in the department three weeks before the beginning of the examination period in the student’s final quarter. Both the primary and secondary adviser must approve the research report before the distinction-in-research designation can be conferred.

**Joint M.S. and MBA Degree**

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

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

Additional information on the MS in Computer Science/MBA Joint Degree Program and its requirements is available on the web at: [http://cs.stanford.edu/education/masters](http://cs.stanford.edu/education/masters)

**Joint M.S. and Law Degree**

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

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

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

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

**Teaching and Research Assistantships in Computer Science**

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

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

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

**Doctor of Philosophy in Computer Science**

The University’s basic requirements for the Ph.D. degree are outlined in the “Graduate Degrees” section of this bulletin. Department requirements are stated below.

**Requirements**

Applications to the Ph.D. program and all supporting documents must be submitted and received online by the published deadline. Please see [http://www-cs.stanford.edu/admissions](http://www-cs.stanford.edu/admissions) for admissions requirements and the application deadline. Changes or updates to the admission process are posted in September.

The following are general department requirements. Contact the Computer Science Ph.D. administrator for details.

1. A student should plan and complete a coherent program of study covering the basic areas of computer science and related disciplines. The student’s adviser has primary responsibility for the adequacy of the program, which is subject to review by the Student Services Office.

2. The first year of the Ph.D. program is spent working with 1-3 different professors on a rotating basis. The intent is to allow the first-year Ph.D. student to work with a variety of professors before aligning with a permanent program adviser. Students who don’t need the full year to find a professor to align with will have the option of aligning within the first or second quarter.

3. The CS 300 Departmental Lecture Series seminar gives faculty the opportunity to explain their research to first year CS Ph.D. students.
First year CS Ph.D. students are required to attend 2/3 of the classes to receive credit.

4. A student must complete 135 course units for graduation. Computer Science Ph.D. students take 8-10 units per quarter. Credit for coursework done elsewhere (up to the maximum of 45 course units) may be applied to graduation requirements. Students must also take at least three units of coursework from four different faculty members. There are NO courses specifically required by the CS Ph.D. program except for the 1-unit CS 300 Departmental Lecture Series and CS 499 Advanced Reading and Research or its equivalent.

5. Each student, to remain in the Ph.D. program, must satisfy the breadth requirement covering introductory-level graduate material in major areas of computer science. A student must fulfill two breadth-area requirements in each of three general areas by the end of the second year in the program. If students have fulfilled the six breadth-area requirements, and taken courses from at least four different faculty members, they are eligible to apply for candidacy prior to the second year in the program. An up-to-date list of courses that satisfy the breadth requirements can be found at http://cs.stanford.edu/education/phil. The student must completely satisfy the breadth requirement by the end of the second year in the program and must pass a qualifying exam in the general area of their expected dissertation by the end of the third year in the program.

6. University policy requires that all doctoral students declare candidacy by the end of the sixth quarter in residence, excluding summers. However, after aligning with a permanent adviser, passing six breadth requirements, and taking classes with four different faculty, a student is eligible to file for candidacy prior to the sixth quarter. The candidacy form serves as a "contract" between the department and the student. The department acknowledges that the student is a bona fide candidate for the Ph.D. and agrees that the program submitted by the student is sufficient to warrant granting the Ph.D. upon completion. The student may petition the department for modification of his or her program. Candidacy expires five years from the date of submission of the candidacy form, rounded to the end of the quarter. In special cases, the department may extend a student's candidacy, but is under no obligation to do so.

7. Each student is required to pass a qualifying exam in their area by the end of their third year in the program. A student may only take the qualifying exam twice. If the student fails the qualifying exam a second time, the Ph.D. Program Committee is convened to discuss the student's lack of a reasonable academic progress. Failing the exam a second time is cause for dismissal from the Computer Science Ph.D. program and the committee will meet to discuss the final outcome for the student.

8. As part of the training for the Ph.D., the student is also required to complete at least four units (a unit is ten hours per week for one quarter) as a course assistant or instructor for courses in Computer Science numbered 100 or above.

9. The Reading Committee form and Oral Thesis Proposal must be submitted within one year of passing the qualifying exam.

10. The most important requirement is the dissertation. After passing the required qualifying examination, each student must secure the agreement of a member of the department faculty to act as the dissertation adviser. The dissertation adviser is often the student's program adviser.

11. The student must pass a University oral examination in the form of a defense of the dissertation. This is typically held after all or a substantial portion of the dissertation research has been completed.

12. The student is expected to demonstrate the ability to present scholarly material orally in the dissertation defense.

13. The dissertation must be accepted by a reading committee composed of the principal dissertation adviser, a second member from within the department, and a third member chosen from within or outside of the University. The department requires at least two committee members to be affiliated with the Computer Science department. The principal adviser and at least one of the other committee members must be Academic Council members.

Guidelines for Reasonable Progress

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

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

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

Within one year of passing the Qualifying Examination, a student should submit a signed Reading Committee Form (http://studentaffairs.stanford.edu/sites/default/files/registrar/files/doc_diss_rdg_ctte.pdf) and Thesis Proposal.

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

Ph.D. Minor in Computer Science

For a minor in Computer Science, a candidate must complete 20 units of Computer Science coursework numbered 200 or above, except for the 100-level courses listed on the Ph.D. Minor Worksheet found at http://cs.stanford.edu/content/phd-minor. At least three of the courses must be master’s core courses to provide breadth and one course numbered 300 or above to provide depth. One of the courses taken must include a significant programming project to demonstrate programming efficiency. Courses must be taken for a letter grade and passed with a grade of 'B' or better. Applications for a minor in Computer Science are submitted at the same time as admission to candidacy.


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

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

CS 12SC. Computational Decision Making, 2 Units.
Although we make decisions every day, many people base their decisions on initial reactions or "gut" feelings. There are, however, powerful frameworks for making decisions more effectively based on computationally analyzing the choices available and their possible outcomes. In this course we give an introduction to some of these frameworks, including utility theory, decision analysis, game theory, and Markov decision processes. We also discuss why people sometimes make seemingly reasonable, yet irrational, decisions. We begin the class by presenting some of the basics of probability theory, which serves as the main mathematical foundation for the decision making frameworks we will subsequently present. Although we provide a mathematical/computational basis for the decision making frameworks we examine, we also seek to give intuitive (and sometimes counterintuitive) explanations for actual decision making behavior through in-class demonstrations. No prior experience with probability theory is needed (we'll cover what you need to know in class), but students should be comfortable with mathematical manipulation at the level of Math 20 or Math 41. Sophomore College Course: Application required, due noon, April 7, 2015. Apply at http://soco.stanford.edu.

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

CS 29N. Computational Decision Making, 3 Units.
Although we make decisions every day, many people base their decisions on initial reactions or "gut" feelings. There are, however, powerful frameworks for making decisions more effectively based on computationally analyzing the choices available and their possible outcomes. In this course we give an introduction to some of these frameworks, including utility theory, decision analysis, and game theory. We also discuss why people sometimes make seemingly reasonable, yet irrational, decisions. We begin the class by presenting some of the basics of probability theory, which serves as the main mathematical foundation for the decision making frameworks we will subsequently present. Although we provide a mathematical/computational basis for the decision making frameworks we examine, we also seek to give intuitive (and sometimes counterintuitive) explanations for actual decision making behavior through in-class demonstrations. No prior experience with probability theory is needed (we'll cover what you need to know in class), but students should be comfortable with mathematical manipulation at the level of Math 20 or Math 41.

CS 44N. Computational Thinking and Systems in the Real-World. 3 Units.
Computing in the real-world is too often viewed as working away concocting some computer incantations hidden inside some high technology company. However, computing and computer communication has infiltrated and in many cases revolutionized several iquest;systems; help systems; and the real world, including financial systems, inventory management, advertising systems, supply chain management, transportation systems, defense systems and so on. Moreover, the discipline of thinking that has developed to build these systems, computational thinking, has powerful applicability to real-world problems and situations outside of computer programming. This course provides an introduction and exposure to some of these dramatic trends, opportunities and risks. Also included is an introduction to some basic ideas in iquest;computational thinking; and the course will include guest speakers. No programming competence is assumed but exposure to programming would be useful. Interest in the real world and interest is not being run-over by this trend is essential.

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 46N. Big Data, Big Discoveries, Big Fallacies. 3 Units.
A sea change has occurred in science, technology, medicine, politics, and society as a whole: many of the world's biggest discoveries and decisions are now being made on the basis of analyzing massive data sets, referred to as "big data". Everyday examples include social-network friend recommendations, and weather predictions far more accurate than a decade ago. Both use vast collections of data to model the past and predict the future. But it is surprisingly easy to come to false conclusions from data analysis alone. For example, we might conclude that acne medicine prevents heart attacks and strokes, if we forget to factor in the age of the patients. Privacy is a major concern: Target stores analyzed buying patterns to predict with remarkable accuracy which of their shoppers had just become pregnant, but trouble arose when they sent baby ads to the homes of pregnant teens whose parents weren't yet in the know. We will start by surveying the history of data-driven activities, leading up to the recent Big Data explosion. A variety of data analysis techniques will be covered, leading students to appreciate that even simple techniques can go a long way when the data set is large enough. Common stumbling blocks leading to false conclusions will be discussed, and students will be asked to debate the many issues surrounding privacy. In one project, students will see whose analysis techniques can best predict user movie ratings based on past rating behavior. A second project will be individually designed in an area of the student's choosing. The seminar will include a mix of assigned readings, small-scale investigations and assignments, classroom discussions, and two projects. No computer programming or special math skills are required; students will learn the basic techniques and tools they need to complete the data analysis assignments and projects.

CS 54N. Great Ideas in Computer Science. 3 Units.
Stanford Introductory Seminar. Preference to freshmen. Covers the intellectual tradition of computer science emphasizing ideas that reflect the most important milestones in the history of the discipline. No prior experience with programming is assumed. Topics include programming and problem solving; implementing computation in hardware; algorithmic efficiency; the theoretical limits of computation; cryptography and security; and the philosophy behind artificial intelligence.
CS 74N. Digital Dilemmas. 3 Units.
Preparation to freshmen. Issues where policy decision making requires understanding computer and communications technology. Technology basics taught in non-technology terms. Topics include consumer privacy, government surveillance, file sharing and intellectual property, and electronic voting.

CS 75N. Cell Phones, Sensors, and You. 3 Units.
Focuses on the role of cell phones as the first prevalent wearable sensors that gather information about you that can be both useful and potentially harmful. Topics include the state of technology, sociological and privacy implications, potential governmental regulation, etc. Addresses omniscient "big brother" technology including radar guns and the recording devices that led to the Watergate scandal. Students will gather and compile information on topics and come to class ready to discuss and debate with formulated opinions.

CS 90SI. CS + Social Good: Using Web Technologies to Change the World. 2 Units.
Learn web technologies by working on real world projects focused on creating positive social impact. The class will cover basic topics related to web development and provide resources for more advanced learning. Students will work on small teams to implement high-impact projects for partner organizations. The aim of the class is to empower students to leverage technology for social good by inspiring action, facilitating collaboration, and forging pathways toward change. No web application experience required. Prerequisite: 106B. Application required; apply online at http://bit.ly/90siApp. Applications accepted until midnight on September 14th.

CS 92SI. Happy Coder: The Python Programming Language. 2 Units.
The fundamentals and contemporary usage of the Python programming language. Primary focus on developing best practices in writing Python and exploring the extensible and unique parts of Python that make it such a powerful scripting language. Topics include: data structures (e.g. lists and dictionaries), and characteristic pythonic conventions like anonymous functions, iterables, and powerful built-ins (e.g. map, filter, zip). Time permitting, we will also cover object-oriented design, modules, (e.g. request, itertools), and modern Python-based web frameworks. Prerequisite: 106A. Application required.

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 as they are not mysterious or intimidating. Course features 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 103. Mathematical Foundations of Computing. 3-5 Units.
Mathematical foundations required for computer science, including propositional predicate logic, induction, sets, functions, and relations. Formal language theory, including regular expressions, grammars, finite automata, Turing machines, and NP-completeness. Mathematical rigor, proof techniques, and applications. Prerequisite: 106A or equivalent.

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

CS 105. Introduction to Computers. 3-5 Units.
For non-technical majors. What computers are and how they work. Practical experience in programming. Construction of computer programs and basic design techniques. A survey of Internet technology and the basics of computer hardware. Students in technical fields and students looking to acquire programming skills should take 106A or 106X. Students with prior computer science experience at the level of 106 or above require consent of instructor. Prerequisite: minimal math skills.

CS 106A. Programming Methodology. 3-5 Units.
Introduction to the engineering of computer applications emphasizing modern software engineering principles: object-oriented design, decomposition, encapsulation, abstraction, and testing. Uses the Java programming language. Emphasis is on good programming style and the built-in facilities of the Java language. No prior programming experience required. Summer quarter enrollment is limited. Priority given to Stanford students. Same as: ENGR 70A

CS 106B. Programming Abstractions. 3-5 Units.
Abstraction and its relation to programming. Software engineering principles of data abstraction and modularity. Object-oriented programming, fundamental data structures (such as stacks, queues, sets) and data-directed design. Recursion and recursive data structures (linked lists, trees, graphs). Introduction to time and space complexity analysis. Uses the programming language C++ covering its basic facilities. Prerequisite: 106A or equivalent. Summer quarter enrollment is limited. Priority given to Stanford students. Same as: ENGR 70B

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

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

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

CS 108. Object-Oriented Systems Design. 3-4 Units.
Software design and construction in the context of large OOP libraries. Taught in Java. Topics: OOP design, design patterns, testing, graphical user interface (GUI) OOP libraries, software engineering strategies, approaches to programming in teams. Prerequisite: 107.
CS 109. Introduction to Probability for Computer Scientists. 3-5 Units.
Topics include: counting and combinatorics, random variables, conditional probability, independence, distributions, expectation, point estimation, and limit theorems. Applications of probability in computer science including machine learning and the use of probability in the analysis of algorithms. Prerequisites: 103, 106B or X, multivariate calculus at the level of MATH 51 or CME 100 or equivalent.

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

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

CS 123. Programming Your Personal Robot. 3 Units.
An introduction to the programming of a sensor-rich personal robot. This course extends programming from the virtual environment into the physical world, which presents unique challenges. Focus is on three areas of intellectual discourse that are fundamental to the programming of physical devices: communication with the devices; programming of event driven behaviors; and reasoning with uncertainty. The concepts introduced will be put into practical use through a series of class projects centered around programming your personal robot. This course also serves as a good introduction to Experimental Robotics by exposing students to basic concepts and techniques that are relevant for real world robot programming. Prerequisite: Basic knowledge of computer programming (as covered in CS 106). Knowledge of Python is recommended.

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

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

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

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

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.

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

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

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 techniques for writing parallel software. Topics: varieties of parallelism in current hardware (e.g., fast networks, multicore, accelerators such as GPUs, vector instruction sets), importance of locality, implicit vs. explicit parallelism, shared vs. non-shared memory, synchronization mechanisms (locking, atomicity, transactions, barriers), and parallel programming models (threads, data parallel/streaming, futures, SPMD, message passing, SIMT, transactions, and nested parallelism). Significant parallel programming assignments will be given as homework. The course is open to students who have completed the introductory CS course sequence through 110 and have taken CS 143.

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

CS 155. Computer and Network Security. 3 Units.
For seniors and first-year graduate students. Principles of computer systems security. Attack techniques and how to defend against them. Topics include: network attacks and defenses, operating system security, application security (web, apps, databases), malware, privacy, and security for mobile devices. Course projects focus on building reliable code. Prerequisite: 110. Recommended: basic Unix.

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

CS 161. Design and Analysis of Algorithms. 3-5 Units.

CS 166. Data Structures. 3-4 Units.
Techniques in the design, analysis, and implementation of data structures. Isomteries 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 167. Readings in Algorithms. 3 Units.
Recent research in the design and analysis of algorithms. Readings cover both classical and emerging topics, such as: computational models for massive data sets; data privacy; dimensionality reduction; exact and approximate algorithms for NP-hard problems; graph algorithms; hashing; online learning; search trees; streaming and sketching. Students are expected to respond to research papers, deliver an oral presentation, and complete a reading or programming project. Limited enrollment; preference given to undergraduates. Prerequisites: 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. Same as: MUSIC 128

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

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

CS 181W. Computers, Ethics, and Public Policy. 4 Units.
Writing-intensive version of CS181. Satisfies the WIM requirement for Computer Science, Engineering Physics, STS, and Math/Comp Sci undergraduates. Same as: WIM
CS 183C. Technology-enabled Blitzscaling. 2 Units.
We are all familiar with the power of technological innovation to reshape markets and daily lives. But what many overlook is how technology enables a far more rapid scaling of organizations and businesses. This rapid scaling, or ‘blitzscaling’, confers massive competitive advantage, but requires massive adjustments at every stage of growth. "Technology-enabled Blitzscaling" examines how technology enables this hyper growth and how technology can help entrepreneurs and organizations manage that growth.

CS 190. Software Design Studio. 3 Units.
This course will teach the art of software design: how to compose large complex systems into classes that can be implemented and maintained easily. Topics include information hiding, thick classes, API design, managing complexity, and how to write in-code documentation. The class will involve significant system software implementation and will use an iterative approach consisting of implementation, review, and revision. The course will be taught in a studio format with in-class discussions and code reviews in addition to lectures. Prerequisites: CS 140.

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. 1 Unit.
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 ten 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.

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

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

CS 194. Software Project. 3 Units.
Design, specification, coding, and testing of a significant team programming project under faculty supervision. Documentation includes a detailed proposal. Public demonstration of the project at the end of the quarter. Preference given to seniors. May be repeat for credit. Prerequisites: CS 110 and CS 161.

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

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

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

CS 198. Teaching Computer Science. 3-4 Units.
Students lead a discussion section of 106A while learning how to teach a programming language at the introductory level. Focus is on teaching skills, techniques, and course specifics. Application and interview required; see http://cs198.stanford.edu.

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

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

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

CS 204. Legal Informatics. 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 729. Prerequisite: basic concepts of programming.

CS 205A. Mathematical Methods for Robotics, Vision, and Graphics. 3 Units.
Continuous mathematics background necessary for research in robotics, vision, and graphics. Possible topics: linear algebra; the conjugate gradient method; ordinary and partial differential equations; vector and tensor calculus. Prerequisites: 106B or X; MATH 51; or equivalents.
CS 205B. Mathematical Methods for Fluids, Solids, and Interfaces. 3 Units.
Numerical methods for simulation of problems involving solid mechanics and fluid dynamics. Focus is on practical tools needed for simulation, and continuous mathematics involving nonlinear hyperbolic partial differential equations. Possible topics: finite element method, highly deformable elastic bodies, plasticity, fracture, level set method, Burgers’ equation, compressible and incompressible Navier-Stokes equations, smoke, water, fire, and solid-fluid coupling. Prerequisite: 205A or equivalent.

CS 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 210L. Introducing Software through Video Stories. 1 Unit.
In this one-unit lab where coding meets film, software development teams from CS210 are paired with film students. This resulting cross-disciplinary group will create a short video that tells an engaging and creative story about the software developed by the team in CS210. The class will introduce students to principles of short form narrative storytelling and the visual language of film, as well as cover the technical principles of DSLR cinematography and non-linear editing. This course will offer students the experience of creating a film in partnership with a producing team.

CS 211. Content Creation in Virtual Reality. 3-4 Units.
Students are immersed in a cutting edge virtual reality development environment consisting of both hardware and software elements. Students will progress from configuring a comprehensive development environment to designing and implementing networked content in VR. The deep development focus is overlaid with a discussion series with leaders in the VR space to provide both breadth and depth to a student's understanding of the VR space. Prerequisites: CS 107 or equivalent. A strong software development background is required that includes comfort with C++. Design experience a plus.

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.

CS 224M. Multi-Agent Systems. 3 Units.
For advanced undergraduates, and M.S. and beginning Ph.D. students. The course serves primarily as an introduction to game theory, including computational aspects. Topics: basic game representations and solution concepts, social choice and mechanism design, multi-agent learning, communication. Applications discussed as appropriate; emphasis is on conceptual matters and theoretical foundations. Prerequisites: very basic probability theory and optimization.

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

CS 224S. Spoken Language Processing. 2-4 Units.
Introduction to spoken language technology with an emphasis on dialogue and conversational systems. Automatic speech recognition, extraction of affect and social meaning from speech, speech synthesis, dialogue management, and applications to digital assistants, search, and recommender systems. Prerequisites: CS 124, 221, 224N, or 229.

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 PHIL 150.

CS 224W. Natural Language Understanding. 3-4 Units.
For advanced undergraduates, and M.S. and beginning Ph.D. students. The course 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.

CS 225B. Mathematical Methods for Fluids, Solids, and Interfaces. 3 Units.
Numerical methods for simulation of problems involving solid mechanics and fluid dynamics. Focus is on practical tools needed for simulation, and continuous mathematics involving nonlinear hyperbolic partial differential equations. Possible topics: finite element method, highly deformable elastic bodies, plasticity, fracture, level set method, Burgers’ equation, compressible and incompressible Navier-Stokes equations, smoke, water, fire, and solid-fluid coupling. Prerequisite: 205A or equivalent.

CS 228. Spoken Language Processing. 2-4 Units.
Introduction to spoken language technology with an emphasis on dialogue and conversational systems. Automatic speech recognition, extraction of affect and social meaning from speech, speech synthesis, dialogue management, and applications to digital assistants, search, and recommender systems. Prerequisites: CS 124, 221, 224N, or 229.

CS 229. 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 PHIL 150.

Same as: LINGUIST 284
CS 224W. Social Information and Network Analysis. 3-4 Units.
(Formerly 322) How do diseases spread? Who are the influencers? How can we predict friends and enemies in a social network? How information flows and mutates as it is passed through networks? Behind each of these questions there is an intricate wiring diagram, a network, that defines the interactions between the components. And we will never understand these questions unless we understand the networks behind them. The course will cover recent research on the structure and analysis of such large social and information networks and on models and algorithms that abstract their basic properties. Class will explore how to practically analyze large-scale network data and how to reason about it through models for network structure and evolution. Topics include methods for link analysis and network community detection, diffusion and information propagation on the web, virus outbreak detection in networks, and connections with work in the social sciences and economics.

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

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

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

CS 228. Probabilistic Graphical Models: Principles and Techniques. 3-4 Units.
Probabilistic graphical modeling languages for representing complex domains, algorithms for reasoning using these representations, and learning these representations from data. Topics include: Bayesian and Markov networks, extensions to temporal modeling such as hidden Markov models and dynamic Bayesian networks, exact and approximate probabilistic inference algorithms, and methods for learning models from data. Also included are sample applications to various domains including speech recognition, biological modeling and discovery, medical diagnosis, message encoding, vision, and robot motion planning. Prerequisites: basic probability theory and algorithm design and analysis.

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

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

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

CS 231B. The Cutting Edge of Computer Vision. 3 Units.
(Formerly 223C) More than one-third of the brain is engaged in visual processing, the most sophisticated human sensory system. Yet visual recognition technology has fundamentally influenced our lives on the same scale and scope as text-based technology has, thanks to Google, Twitter, Facebook, etc. This course is designed for those students who are interested in cutting edge computer vision research, and/or are aspiring to be an entrepreneur using vision technology. Course will guide students through the design and implementation of three core vision technologies: segmentation, detection and classification on three highly practical, real-world problems. Course will focus on teaching the fundamental theory, detailed algorithms, practical engineering insights, and guide them to develop state-of-the-art systems evaluated based on the most modern and standard benchmark datasets. Prerequisites: CS2223B or equivalent and a good machine learning background (i.e. CS221, CS228, CS229). Fluency in Matlab and C/C++.

CS 231M. Mobile Computer Vision. 3-4 Units.
The course surveys recent developments in computer vision, graphics and image processing for mobile application. Topics of interest include: feature extraction, image enhancement and digital photography, 3D scene understanding and modeling, virtual augmentation, object recognition and categorization, human activity recognition. As part of this course, students will familiarize with a state-of-the-art mobile hardware and software development platform: an NVIDIA Tegra-based Android tablet, with relevant libraries such as OpenCV and FCam. Tablets will be available for each student team. Prerequisites: Knowledge of linear algebra, probability, as well as concepts introduced in either CS131A or CS231A and CS232 (or equivalent) are necessary for understanding the material covered in this class. C++ (or Java) programming experience is expected.
CS 213N. Convolutional Neural Networks for Visual Recognition. 3-4 Units.
Computer Vision has become ubiquitous in our society, with applications in
search, image understanding, apps, mapping, medicine, drones, and self-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, multisolution 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. The Shape of Data: 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 matching, and map computation between geometric data sets. Function.

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 240H. Functional Systems in Haskell. 3-4 Units.
Covers an array of practical issues and techniques that arise when building real-world systems in the Haskell programming language. Topics include the basics of Haskell, laziness, monads, parsers, testing and debugging, performance tuning, interfacing to native code, concurrency and I/O paradigms, language extensions, meta-programming, and applications to the web and security. Concepts will be reinforced through a few individual programming assignments followed by a larger team project. Prior familiarity with Haskell may be helpful but is not required. Prerequisites: CS106B or 106X.

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

CS 242. Programming Languages. 3 Units.
Central concepts in modern programming languages, impact on software development, language design trade-offs, and implementation considerations. Functional, imperative, and object-oriented paradigms. Formal semantic methods and program analysis. Modern type systems, decision processes, approximate dynamic programming, and reinforcement learning. New approaches for overcoming challenges in generalization from experience, exploration of the environment, and model representation. Students are expected to produce an original research paper on a relevant topic. Prerequisites: CS107 (or equivalent).

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.
Classical papers, new ideas, and research papers in networking. Architectural principles: naming, addressing, routing; congestion control, traffic management, QoS; wireless and mobility; overlay networks and virtualization; network security; switching and routing; content distribution; and proposals for future Internet structures. Prerequisite: 144 or equivalent.
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 and 
CS 249A.

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

CS 245. Database Systems Principles. 3 Units.
File organization and access, buffer management, performance analysis, 
and storage management. Database system architecture, query optimization, 
transaction management, recovery, concurrency control. Reliability, 
protection, and integrity. Design and management issues. Prerequisites: 
145, 161.

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

CS 247. Human-Computer Interaction Design Studio. 3-4 Units.
Project-based focus on interaction design process, especially early-stage 
design and rapid prototyping. Methods used in interaction design including 
needs analysis, user observation, sketching, concept generation, scenario 
building, and evaluation. Prerequisites: 147 or equivalent background 
in design thinking; 106B or equivalent background in programming. 
Recommended: CS 142 or equivalent background in web programming.

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

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

CS 249A. Object-Oriented Programming from a Modeling and 
Simulation Perspective. 3 Units.
Topics: large-scale software development approaches for complex 
applications, class libraries and frameworks; encapsulation, use of 
inheritance and dynamic dispatch, design of interfaces and interface/ 
implementation separation, exception handling, smart pointers and 
reference management, minimalizing dependencies and value-oriented 
programming. Inheritance: when and why multiple inheritance naming 
directories, manager, and disciplined use of design patterns including 
functors, event notification and iterators. Prerequisites: C, C++, and 
programming methodology as developed in 106B or X, and 107 (107 may 
be taken concurrently). Recommended: 193D.

CS 249B. Large-scale Software Development. 3 Units.
Software engineering of high quality large-scale complex software with 
a focus on evolvability, performance and cost. Software development 
processes, people and practice; audit: integrating invariant checks 
with production software; concurrency with modular object-oriented 
programming; collection implementation; generic programming and 
templates; design of value types; named descriptions for large value types; 
memory management; controlling placement, locality and consumption; 
run-time vs. static type checking and identification.

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

CS 251P. Bitcoin & Crypto Currencies Lab. 1 Unit.
For advanced undergraduates and for graduate students. Intensive version 
of CS251 for students interested in practice as well as theory. Includes lab 
section of course which includes hardware work with Bitcoin mining chips 
and software project involving the use of Bitcoin for micro payments. Co-
quisite: CS251 (enroll in both to take CS251P). Prerequisite: CS110. 
Recommended: CS255.

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 261. Optimization and Algorithmic Paradigms. 3 Units.
Algorithms for network optimization: max-flow, min-cost flow, matching, 
assignment, and min-cut problems. Introduction to linear programming. 
Use of LP duality for design and analysis of algorithms. Approximation 
algorithms for NP-complete problems such as Steiner Trees, Traveling 
Salesman, and scheduling problems. Randomized algorithms. Introduction 
to online algorithms. Prerequisite: 161 or equivalent.

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

Same as: BIOMEDIN 262

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: MSE 317
CS 265. Randomized Algorithms and Probabilistic Analysis. 3 Units.
Randomness pervades the natural processes around us, from the formation of networks, to genetic recombination, to quantum physics. Randomness is also a powerful tool that can be leveraged to create algorithms and data structures which, in many cases, are more efficient and simpler than their deterministic counterparts. This course covers the key tools of probabilistic analysis, and application of these tools to understand the behaviors of random processes and algorithms. Emphasis is on theoretical foundations, though we will apply this theory broadly, discussing applications in machine learning and data analysis, networking, and systems. Topics include tail bounds, the probabilistic method, Markov chains, and martingales, with applications to analyzing random graphs, metric embeddings, random walks, and a host of powerful and elegant randomized algorithms. Prerequisites: CS 161 and STAT 116, or equivalents and instructor consent.
Same as: CME 309

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

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

CS 272. Introduction to Biomedical Informatics Research Methodology. 3 Units.
Hands-on software building. Student teams conceive, design, specify, implement, evaluate, and report on a software project in the domain of biomedical computing. Creating written proposals, peer review, providing status reports, and preparing final reports. Guest lectures from professional biomedical informatics systems builders on issues related to the process of project management. Software engineering basics. Because the team projects start in the first week of class, attendance that week is strongly recommended. Prerequisites: BIOMEDIN 210 or 211 or 214 or 217 or consent of instructor.
Same as: BIOE 212, BIOMEDIN 212, GENE 212

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

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, Gibbs Sampling, basic structural computations on proteins, protein structure prediction, protein threading techniques, homology modeling, molecular dynamics and energy minimization, statistical analysis of 3D biological data, integration of data sources, knowledge representation and controlled terminologies for molecular biology, microarray analysis, machine learning (clustering and classification), and natural language text processing. Prerequisites: programming skills; consent of instructor for 3 units.
Same as: BIOE 214, BIOMEDIN 214, GENE 214

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

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

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

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

CS 277. Experimental Haptics. 3 Units.
Computer haptics is the discipline of synthesizing touch feedback in simulated or virtual environments. Course objective is to study and develop computational methods for generating force feedback through haptic interfaces. Theoretical topics: haptic rendering in 3-D virtual environments, simulation of haptic interaction with rigid and deformable objects, haptic interfaces, psychophysics of touch. Applied topics: CHAI3D haptic library, implementation of algorithms for haptic rendering, collision detection, and deformable body simulation. Guest speakers; Lab/programming exercises; open-ended final project. Enrollment limited to 20. Prerequisite: experience with C++. Recommended: 148 or 248, 223A.
CS 279. Computational Biology: Structure and Organization of Biomolecules and Cells. 3 Units.
Computational approaches to understanding the three-dimensional spatial organization of biological systems and how that organization evolves over time. The course will cover cutting-edge research in both physics-based simulations and computational analysis of experimental data, at scales ranging from individual molecules to multiple cells. Prerequisites: elementary programming background (106A or equivalent) and an introductory course in biology or biochemistry.
Same as: 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 implementation of a large project in HCI. State-of-the-art methods related to the problem domain. 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, 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.
Registered to Computer Science and Computer Systems Engineering undergraduates. Students enroll in the CS 294W section attached to the CS 294 project they have chosen.

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

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

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

CS 309A. Cloud Computing. 1 Unit.
For science, engineering, business, medicine, and law students. Cloud computing is bringing information systems out of the back office and making it core to the entire economy. This class is intended for all students who want to begin to understand the implications of this shift in technology. Guest industry experts are public company CEOs who are delivering application, software development, operations management, compute, storage & data center, and network cloud services.

CS 315A. Parallel Computer Architecture and Programming. 3 Units.
The principles and tradeoffs in the design of parallel architectures. Emphasis is on naming, latency, bandwidth, and synchronization in parallel machines. Case studies on shared memory, message passing, data flow, and data parallel machines illustrate techniques. Architectural studies and lectures on techniques for programming parallel computers. Programming assignments on one or more commercial multiprocessors. Prerequisites: EE 282, and reasonable programming experience.

CS 315B. Parallel Computing Research Project. 3 Units.
Advanced topics and new paradigms in parallel computing including parallel algorithms, programming languages, runtime environments, library debugging/tuning tools, and scalable architectures. Research project. Prerequisite: consent of instructor.

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

CS 319. Topics in Digital Systems. 3 Units.
Advanced material is often taught for the first time as a topics course, perhaps by a faculty member visiting from another institution. May be repeated for credit.

CS 325. Topics in Computational Sustainability. 3 Units.
Computational Sustainability focuses on developing computational models, methods and tools for sustainable development. In this course, we will study recent computational approaches that have contributed to addressing sustainability topics related to biodiversity, climate, environment, urban design, transportation, buildings and others. Computational themes include machine learning, optimization, statistical modeling, and data mining.

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, humanoid and whole-body control. Additional topics 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 331A. Advanced Reading in Computer Vision. 3 Units.
(Formerly CS332) The field of computer vision has seen an explosive growth in past decade. Much of recent effort in vision research is towards developing algorithms that can perform high-level visual recognition tasks on real-world images and videos. With development of Internet, this task becomes particularly challenging and interesting given the heterogeneous data on the web. Course will focus on reading recent research papers that are focused on solving high-level visual recognition problems, such as object recognition and categorization, scene understanding, human motion understanding, etc. Project required. Prerequisite: some experience in research with one of the following fields: computer vision, image processing, computer graphics, machine learning.
CS 331B. 3D Representation and Recognition. 3 Units.
The course surveys recent developments in high level and 3D computer vision and will focus on reading recent research papers on topics related to 3D object recognition and representation, spatial inference, activity understanding, human vision and 3D perception. The course is inspired by a famous series of workshops (called 3d-RR) which have been offered during the International Conference in Computer Vision (ICCV) since 2007. Prerequisites - Some experience in research with one of the following fields: computer vision, image processing, computer graphics, machine learning.

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

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

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

CS 344. Topics in Computer Networks. 3 Units.
High-performance embedded system design. Student teams of two software engineers (C experience required) and one hardware engineer (Verilog experience required) build a fully functioning Internet router Work in teams of three. How router interoperates with others in class. Open-ended design challenge judged by panel of industry experts. Prerequisites: CS 144, 244, or network programming experience.

CS 344E. Advanced Wireless Networks. 3 Units.
Networking research in wireless systems. Topics include: multi-channel/ multi-radio systems, routing, coding, physical layer hints, low power, mesh networking, interference cancellation, technological trends, and protocol design. Students implement and test research ideas on SWAN, a WiFi testbed.

CS 344G. Network Application Studio. 3 Units.
Graduate project class on computer networking, emphasizing end-to-end applications and protocols. Students will propose and execute an original project in teams of 2-3, culminating in a final writeup and presentation/ demonstration. Each week, students will read, present, and lead a discussion about a seminal paper or system. Prerequisites: programming experience.

CS 345D. Advanced Topics in Database Systems. 3 Units.
The first part of the course will describe classical database systems topics, including join processing, concurrency control, recovery, query optimization, and database theory. On each topic, there will be an in-depth discussion of a few representative papers and recent results. The second part of the course will focus on additional topics that are relevant to database systems, including MapReduce-style processing, information extraction, and predictive analytics. The course readings will primarily consist of classical and recent research papers. Prerequisites: CS 145 or equivalent.

CS 346. Database System Implementation. 3-5 Units.
A major database system implementation project realizes the principles and techniques covered in earlier courses. Students independently build a complete database management system, from file structures through query processing, with a person's designed feature or extension. Lectures on project details and advanced techniques in database system implementation, focusing on query processing and optimization. Guest speakers from industry on commercial DBMS implementation techniques. Prerequisites: 145, 245, programming experience in C++.

CS 347. Parallel and Distributed Data Management. 3 Units.
The principles and system organization of distributed and parallel database systems, including features of data fragmentation and distribution, distributed database design, query processing and optimization, distributed concurrency control, reliability and commit protocols, and replicated data management. Data management in peer-to-peer systems. Data management in the "cloud" using map-reduce and other distributed parallelism techniques.

CS 348A. Computer Graphics: Geometric Modeling. 3-4 Units.

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

CS 354. Topics in Circuit Complexity. 3 Units.
An overview of circuit complexity, focusing on limitations of solving computational problems with circuits. Classical methods: diagonalization; the gate elimination method and circuit size lower bounds; the method of random restrictions and formula size lower bounds; approximating circuits with polynomials and depth restricted lower bounds. Connections between circuit-analysis algorithms and circuit complexity: learning circuits via queries; pseudorandomness and derandomization; satisfiability algorithms. Prerequisite: CS 254 or the equivalent mathematical maturity.

CS 357. Advanced Topics in Formal Methods. 3 Units.
Topics vary annually. Recent offerings have covered the foundations of static analysis, including decision procedures for important theories (SAT, linear integer constraints, SMT solvers), model checking, abstract interpretation, and constraint-based analysis. May be repeated for credit. Prerequisite: 256.
CS 358. Topics in Programming Language Theory. 3 Units.
Topics of current research interest in the mathematical analysis of
programming languages, structured operational semantics, domain theory,
semantics of concurrency, rich type disciplines, problems of representation
independence, and full abstraction. See Time Schedule or Axess for
current topics. May be repeated for credit. Prerequisites: 154, 157, 258, or
equivalents. (Staff).

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 364A. Algorithmic Game Theory. 3 Units.
Topics at the interface of computer science and game theory such as:
algorithmic mechanism design; combinatorial auctions; computation of
Nash equilibria and relevant complexity theory; congestion and potential
games; cost sharing; game theory and the Internet; matching markets;
network formation; online learning algorithms; price of anarchy; prior-free
auctions; selfish routing; sponsored search. Prerequisites: 154N and 161, or
equivalents.

CS 367. Algebraic Graph Algorithms. 3 Units.
Due to the surprisingly fast algorithms for the problem, matrix
multiplication is routinely used as a basic building block for algorithms
beating the brute-force approach. This course explores matrix multiplication
algorithms and a variety of problems, mostly within graph algorithms,
that can be solved faster using a fast matrix multiplication algorithm.
Topics include: Fast Matrix Multiplication, algebraic algorithms for Graph
Transitive Closure, All Pairs Shortest Paths and variants of the problem,
Perfect Matching and Minimum Cycle, and a variety of equivalences
between problems involving matrix multiplication. Prerequisites: CS154,
CS161, or the equivalent mathematical maturity.

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 369L. Theoretical Perspective on Machine Learning. 3 Units.
Many problems in machine learning are intractable in the worst case,
and pose 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. Prerequisites:
A strong background in algorithms, probability and linear algebra.

CS 371. Computational Biology in Four Dimensions. 3 Units.
Computational approaches to understanding the three-dimensional spatial
organization of biological systems and how that organization evolves over
time. The course will cover cutting-edge research in both physics-based
simulation and computational analysis of experimental data, at scales
ranging from individual molecules to entire cells. 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 374. Algorithms in Biology. 2-3 Units.
Algorithms and computational models applied to molecular biology and
genetics. Topics vary annually. Possible topics include biological sequence
comparison, annotation of genes and other functional elements, molecular
evolution, genome rearrangements, microarrays and gene regulation,
protein folding and classification, molecular docking. RNA secondary
structure, DNA computing, and self-assembly. May be repeated for credit.
Prerequisites: 161, 262 or 274, or BIOCHEM 218, or equivalents.
Same as: BIOMEDIN 374

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;
broader 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, 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.
Content changes 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.
Content 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 will 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 needfinding 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 interleave a weekly lecture with a weekly studio
session where students apply the techniques hands-on in a small-scale,
supportive environment.

CS 377W. HCI Issues in Wearable Computing. 3 Units.
With devices like Pebble and Google Glass moving from labs to consumer
use, Wearable Computing represents the forefront of HCI innovation. In
this course, students will engage with a broad range of issues around
the design and development of wearable devices and systems and develop
their own wearable interaction. The course begins with use, analysis,
and redesign of an existing wearable, followed by a larger group project
integrating concepts from the course to prototype a novel wearable
interaction. Students work in project teams, prototyping their wearable
concept and communicating their progress through demonstration, final
report, and presentation. Google Glass will be available for students
interested in experimenting with this platform. Prerequisites: One of the
following: CS 147 or CS 247.

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 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. 390 A, B, C may each be taken once.

CS 390P. Part-time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390Q. Part-Time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390R. Part-Time Curricular Practical Training. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 390S. Part-Time CPT. 1 Unit.
For qualified computer science PhD students only. Permission number required for enrollment; see the CS PhD program administrator in Gates room 196. May be taken just once; not repeatable. Educational opportunities in high technology research and development labs in the computing industry. Qualified computer science students engage in research and integrate that work into their academic program. Students register during the quarter they are employed and complete a research report outlining their work activity, problems investigated, results, and follow-on projects they expect to perform. Students on F1 visas should be aware that completing 12 or more months of full-time CPT will make them ineligible for Optional Practical Training (OPT).

CS 393. Computer Laboratory. 1-9 Unit.
For CS graduate students. A substantial computer program is designed and implemented; written report required. Recommended as a preparation for dissertation research. Register using the section number associated with the instructor. Prerequisite: consent of instructor.

CS 395. Independent Database Project. 1-6 Unit.
For graduate students in Computer Science. Use of database management or file systems for a substantial application or implementation of components of database management system. Written analysis and evaluation required. Register using the section number associated with the instructor. Prerequisite: consent of instructor.

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

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

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

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

CS 424M. Learning Analytics and Computational Modeling in Social Science. 3-4 Units.
Computational modeling and data-mining are dramatically changing the physical sciences, and more recently also the social and behavioral sciences. Traditional analysis techniques are insufficient to investigate complex dynamic social phenomena as social networks, online gaming, diffusion of innovation, opinion dynamics, classroom behavior, and other complex adaptive systems. In this course, we will learn about how modeling, network theory, and basic data-mining can support research in cognitive, and social sciences, in particular around issues of learning, cognitive development, and educational policy. Same as: EDUC 390
CS 431. High-Level Vision: Object Representation. 3 Units.
(Formerly CS423 High-Level Vision: Behaviors, Neurons, and Computational Models) 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, Computer Science and Applied Statistics. Emphasis on discussing recent empirical findings, methods and theoretical debates in the field. Topics include: theories of object perception, neural computations underlying invariant object perception, how visual exemplars and categories are represented in the brain, what information is present in distributed activations across neural populations and how it relates to object perception, what modern statistical and analytical tools there are for multi-variate analysis of brain activations. Same as: PSYCH 250

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

CS 448A. 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. 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 448J. Creative Design for Computer Music Software. 3-4 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 448K. Concepts and Algorithms of Scientific and Visual Computing. 3 Units.
This course covers a selection of fundamental concepts and algorithms for scientific and visual computing. Based on prior knowledge in basis calculus, linear algebra, numerical interpolation and optimization, this course introduces the concept of the phase space, variational principles, methods for ordinary and partial differential equations, Fourier analysis, and multi-scale modeling. The lecture is algorithmically oriented, aiming to enable the students to develop efficient solutions for practically relevant problems, based on solid theoretical foundations and mathematically precise modeling. It covers practical applications, like the simulation of rigid and deformable objects, fibers, fluids, molecular dynamics, signal/image analysis and processing, as well as wavelet-based modeling on different scales. Prerequisites: Basic knowledge such as taught in MATH 41, MATH 42, CS 103, or CS 205A.

CS 448Z. Physically Based Animation and Sound. 3-4 Units.
Intermediate level, emphasizing physically based simulation techniques for computer animation and synchronized sound synthesis. Topics vary from year to year, but include integrated approaches to visual and auditory simulation of rigid bodies, deformable solids, collision detection and contact resolution, fracture, fluids and gases, and virtual characters. Written assignments and programming projects. Prerequisite: None. Recommended: Computer graphics (CS 148 and CS 248), and/or scientific computing (CS 205).

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

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

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

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

CS 545. Information and Data Analytics Seminar. 1 Unit.
Seminar features leading industrial and academic experts on big data analytics, information management, data mining, machine learning, and large-scale data processing.

CS 546. Seminar on Liberation Technologies. 1 Unit.
This one-unit seminar will present speakers relevant in a variety of ways to how various forms of information technology are being used to defend human rights, improve governance, deepen democracy, empower the poor, promote economic development, protect the environment, enhance public health, and pursue a variety of other social goods.

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

CS 548. Internet and Distributed Systems Seminar. 1 Unit.
Guest speakers from academia and industry. 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.
Prerequisites: CS 147, 247, and a declared HCI track.
CS 571. Surgical Robotics Seminar. 1 Unit.
Surgical robots developed and implemented clinically on varying scales. Seminar goal is to expose students from engineering, medicine, and business to guest lecturers from academia and industry. Engineering and clinical aspects connected to design and use of surgical robots, varying in degree of complexity and procedural role. May be repeated for credit. Same as: ME 571

CS 801. TGR Project. 0 Units.

CS 802. TGR Dissertation. 0 Units.