COMPUTER SCIENCE

Director of undergraduate studies: Y. Richard Yang (yang.r.yang@yale.edu), AKW 208A, 432-6400 [F]; James Aspnes (james.aspnes@yale.edu), AKW 401, 432-1232 [Sp]; cpsc.yale.edu

The Department of Computer Science offers both B.S. and B.A. degree programs, as well as four combined majors in cooperation with other departments: Electrical Engineering and Computer Science, Computer Science and Economics, Computer Science and Mathematics, and Computer Science and Psychology. Each major program not only provides a solid technical education but also allows students either to take a broad range of courses in other disciplines or to complete the requirements of a second major.

The Computer Science and combined major programs share a common core of five computer science courses. The first is CPSC 201, a survey that demonstrates the breadth and depth of the field to students who have taken the equivalent of an introductory programming course. The remaining core courses cover discrete mathematics, data structures, systems programming and computer architecture, and algorithm analysis and design. Together these courses include the material that every major should know.

The core courses are supplemented by electives (and, for the combined majors, core courses in the other discipline) that offer great flexibility in tailoring a program to each student’s interests. The capstone is the senior project, through which students experience the challenges and rewards of original research under the guidance of a faculty mentor.

Prospective majors are encouraged to discuss their programs with the director of undergraduate studies (DUS) as early as possible.

INTRODUCTORY COURSES
The department offers a broad range of introductory courses to meet the needs of students with varying backgrounds and interests. Except for CPSC 200 and CPSC 201, none assumes previous knowledge of computers.

1. CPSC 100, taught jointly with Harvard University, teaches students majoring in any subject area how to program a computer and solve problems. No prior experience is required.
2. CPSC 112 teaches students majoring in any subject area how to program a computer and solve problems using the language Java. Students with previous programming experience should consider taking CPSC 201 instead.
3. CPSC 134 provides an introduction to computer music, including musical representations for computing, automated music analysis and composition, interactive systems, and virtual instrument design.
4. CPSC 150 explores how some of the key ideas in computer science have affected philosophy of mind, cognitivism, connectionism, and related areas. This humanities-style course requires a significant amount of reading and writing a paper, and satisfies the writing and the humanities and arts distributional requirements.
5. CPSC 151 studies the history of the graphical user interface in an attempt to guess its future. This course also satisfies the writing distributional requirement.
6. CPSC 183 explores the myriad ways that law and technology intersect, with a special focus on the role of cyberspace. This course satisfies the social science distributional requirement.
7. CPSC 200, intended as a survey course for non-majors, focuses on practical applications of computing technology while examining topics including computer hardware, computer software, and related issues such as security and software engineering.
8. CPSC 201 surveys the field of computer science, including systems (computers and their languages) and theory (algorithms, complexity, and computability). Students with sufficient programming experience may elect CPSC 201 without taking CPSC 112. (These courses meet at the same time so that students are easily able to change levels if necessary.)
9. CPSC 202 presents the formal methods of reasoning and the concepts of discrete mathematics and linear algebra used in computer science and related disciplines.

REQUIREMENTS OF THE MAJOR
The B.S. and the B.A. degree programs have the same required five core courses: CPSC 201; CPSC 202 or MATH 244; CPSC 223; CPSC 323; and CPSC 365 or 366. CPSC 480 and 490 may not be counted toward these core courses.

B.S. degree program The B.S. degree program requires a total of twelve courses, six additional intermediate or advanced courses in Computer Science and the senior requirement.

B.A. degree program The B.A. degree program requires a total of ten courses, four additional intermediate or advanced course in Computer Science and the senior requirement.

Combined B.S./M.S. degree Exceptionally able and well-prepared students may complete a course of study leading to the simultaneous award of the B.S. and M.S. degrees after eight terms of enrollment. General eligibility requirements are described in the Academic Regulations, section K, Special Arrangements, "Simultaneous Award of the Bachelor’s and Master’s Degrees." Specific requirements for the combined degree in Computer Science are as follows:
1. Candidates must satisfy the Yale College requirements for the B.S. degree in Computer Science.

2. At the end of their fifth term of enrollment candidates must have earned at least nine of their Computer Science required course credits, which together with three additional Computer Science required course credits, satisfy the requirements for the B.S. in Computer Science. Candidates must also have achieved A grades in at least three quarters of these courses.

3. Candidates must also complete eight graduate courses from the approved list, up to two of which may, with the permission of the director of undergraduate studies (DUS) and the director of graduate studies, also be applied toward completion of the B.S. degree. At most, one of these eight courses may be CPSC 690, 691, or 692. All eight graduate courses must be completed in the final four terms of enrollment, and at least six of them must be completed in the final three terms of enrollment.

**Credit/D/Fail** Courses taken Credit/D/Fail may not be counted toward the major. All courses in the major must be taken for a letter grade.

**Roadmap** See visual roadmap of the requirements.

**SENIOR REQUIREMENT**

In the senior year students must take CPSC 490, an independent project course in which students select an adviser to guide them in research in a subfield of computer science. With permission of the DUS, students may enroll in 490 more than once or before their senior year.

**ADVISING**

All Computer Science majors in the sophomore, junior, and senior years should have their programs approved by the DUS. Students majoring in Computer Science are advised to complete CPSC 201 and 223 by the end of the sophomore year.

**Electives** The Computer Science department encourages interdisciplinary study in which computer science plays a major role. Advanced courses in other departments that involve concepts from computer science and are relevant to an individual program may, with permission of the DUS, be counted toward the requirements.

Students considering graduate study in computer science are advised to take CPSC 421 and 422, as well as courses covering the breadth of computer science, including programming languages and systems, artificial intelligence, scientific computing, and theoretical computer science.

Students interested in using computers to solve scientific and engineering problems are advised to take CPSC 440 as well as computational courses offered in Applied Mathematics and in Engineering and Applied Science.

The core mathematical background necessary to complete the Computer Science major is provided in CPSC 202. However, many advanced courses in graphics, computer vision, neural networks, and numerical analysis assume additional knowledge of linear algebra and calculus. Students who plan to take such courses as electives and who are unsure whether they have the appropriate mathematical background are encouraged to take MATH 222 or 225, and MATH 120.

**Typical programs** For students who already know how to program, typical B.S. programs starting in the first and sophomore years are indicated below. For typical B.A. programs, two of the electives would be omitted.

<table>
<thead>
<tr>
<th>First-Year</th>
<th>Sophomore</th>
<th>Junior</th>
<th>Senior</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSC 201</td>
<td>CPSC 202</td>
<td>Two electives</td>
<td>CPSC 490</td>
</tr>
<tr>
<td>CPSC 323</td>
<td>CPSC 365 or 366</td>
<td>Two electives</td>
<td>One elective</td>
</tr>
<tr>
<td>CPSC 223</td>
<td>One elective</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sophomore</th>
<th>Junior</th>
<th>Senior</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSC 223</td>
<td>CPSC 365 or 366</td>
<td>Two electives</td>
</tr>
<tr>
<td>CPSC 201</td>
<td>CPSC 323</td>
<td>CPSC 490</td>
</tr>
<tr>
<td>CPSC 202</td>
<td>One elective</td>
<td>Two electives</td>
</tr>
</tbody>
</table>

**REQUIREMENTS OF THE MAJOR**

**Prerequisites** None

**Number of courses** B.S. — 12 term courses taken for letter grades (incl senior project); B.A. — 10 term courses taken for letter grades (incl senior project)

**Specific courses required** B.S. and B.A. — CPSC 201; CPSC 202 or MATH 244; CPSC 223; CPSC 323; and CPSC 365 or 366

**Distribution of courses** B.S. — 6 addtl intermediate or advanced Comp Sci courses; B.A. — four addtl intermediate or advanced Comp Sci courses

**Substitution permitted** Advanced courses in other depts, with DUS permission

**Senior requirement** Senior project (CPSC 490)
INTRODUCTORY COURSES

* CPSC 035b / MUSI 035b, Twenty-First Century Electronic and Computer Music Techniques  Scott Petersen
  Exploration of twenty-first century electronic and computer music through the diverse subjects and issues at the intersection of technology and new music. How computers have changed and challenged the analysis, composition, production, and appreciation of music over the last fifty years. Knowledge of basic music theory and the ability to read Western musical notation is assumed. Enrollment limited to first-year students. Preregistration required; see under First-Year Seminar Program.

CPSC 100a / CPSC S100, Introduction to Computing and Programming  Benedict Brown
  Introduction to the intellectual enterprises of computer science and to the art of programming. Students learn how to think algorithmically and solve problems efficiently. Topics include abstraction, algorithms, data structures, encapsulation, resource management, security, software engineering, and web development. Languages include C, Python, SQL, and JavaScript, plus CSS and HTML. Problem sets inspired by real-world domains of biology, cryptography, finance, forensics, and gaming. See CS50’s website, https://cs50.yale.edu, for additional information. No previous programming experience required. Open to students of all levels and majors. QR

CPSC 112b, Introduction to Programming  Benedict Brown
  Development on the computer of programming skills, problem-solving methods, and selected applications. No previous experience with computers necessary. QR

* CPSC 150a, Computer Science and the Modern Intellectual Agenda  David Gelernter
  Introduction to the basic ideas of computer science (computability, algorithm, virtual machine, symbol processing system), and of several ongoing relationships between computer science and other fields, particularly philosophy of mind. No previous experience with computers necessary. Enrollment limited to 25. WR, HU

CPSC 183a, Law, Technology, and Culture  Brad Rosen
  An exploration of the myriad ways in which law and technology intersect, with a special focus on the role of cyberspace. Topics include digital copyright, free speech, privacy and anonymity, information security, innovation, online communities, the impact of technology on society, and emerging trends. No previous experience with computers or law necessary. SO

* CPSC 185b, Control, Privacy, and Technology  Brad Rosen
  The evolution of various legal doctrines with and around technological development. Topics include criminal law, privacy, search and seizure, digital rights, and the implications of technologically permitted methods of control on the law. Special attention to case law and policy. After CPSC 183. WR, SO

CPSC 200b, Introduction to Information Systems  Staff
  The real-world artifacts and implementations that comprise the vital computational organisms that populate our world. Hardware and software and the related issues of security, privacy, regulation, and software engineering. Examples stress practical applications of technology, as well as limitations and societal issues. After CPSC 100 or 112 or equivalent. QR

CPSC 201a or b, Introduction to Computer Science  Staff
  Introduction to the concepts, techniques, and applications of computer science. Topics include computer systems (the design of computers and their languages); theoretical foundations of computing (computability, complexity, algorithm design); and artificial intelligence (the organization of knowledge and its representation for efficient search). Examples stress the importance of different problem-solving methods. After CPSC 112 or equivalent. QR
CPSC 202a, Mathematical Tools for Computer Science  Dana Angluin
Introduction to formal methods for reasoning and to mathematical techniques basic to computer science. Topics include propositional logic, discrete mathematics, and linear algebra. Emphasis on applications to computer science: recurrences, sorting, graph traversal, Gaussian elimination.  QR

CPSC 223a or b, Data Structures and Programming Techniques  Staff
Topics include programming in C; data structures (arrays, stacks, queues, lists, trees, heaps, graphs); sorting and searching; storage allocation and management; data abstraction; programming style; testing and debugging; writing efficient programs. After CPSC 201 or equivalent.  QR  RP

CPSC 235b / EENG 245b, Self-Driving Cars: Theory and Practice  Man-Ki Yoon
This course explores the theory and practice of building self-driving cars using advanced computing technologies. Topics include embedded system programming, sensor fusion, control theory, and introductory planning and navigation techniques using machine learning and computer vision. Students work in small teams to design and build miniaturized self-driving cars that autonomously navigate an indoor track that resembles real road environments. The final project involves driving competitions and project report/presentation of their work. Prerequisite: CPSC 112, 201, 223, or equivalent. Instructor’s permission is required to waive the prerequisites. Enrollment limited to 18.  QR

CPSC 257a, Information Security in the Real World  Stephen Slade
Introduction to information security, the practice of protecting information from unauthorized actions, in the context of computer systems. Topics include current security-related issues, basic adversarial models and threats to computer systems, potential defenses, security tools, and common security breaches and their wider impacts. Prerequisite: CPSC 100, 112, or equivalent programming experience, or with permission of instructor.  QR

CPSC 276a, Introduction to Web Application for the Digital Humanities  Benedict Brown
Introduction to applications of computer and data science in the humanities, including web technologies, visualization, and database design. Students work in teams to develop a variety of applications proposed by faculty and staff from the Digital Humanities Lab, the Institute for the Preservation of Cultural Heritage, and the Computer Science department. Meets with CPSC 376. Students may earn credit for CPSC 276 or 376; not both. Prerequisite: CPSC 110, CPSC 112, equivalent programming experience, or permission of the instructor.  QR

* CPSC 290a or b, Directed Research  Staff
Individual research. Requires a faculty supervisor and the permission of the director of undergraduate studies. May be taken more than once for credit.

Intermediate Courses

CPSC 310b, Technology, Power, and Security: Political Challenges of the Computer Age  Joan Feigenbaum
Twenty-first century societies are faced with both threats and opportunities that combine sophisticated computation with politics and international relations in critical ways. Examples include cyber warfare; cyber espionage; cyber crime; the role of social media in democratic self-governance, authoritarian control, and election "hacking"; cryptocurrencies; and mass surveillance. This course examines the political challenges wrought by massive increases in the power of computational and communication technologies and the potential for citizens and governments to harness those technologies to solve problems. It is co-taught by one faculty member in computer science and one in political science. Programming experience and some knowledge of basic computer science is required. Meets with CPSC 210/PLSC 369. Students may earn credit for CPSC 210 or 369; not both. Prerequisite: CPSC 223 or the equivalent.  QR, SO

CPSC 334a, Creative Embedded Systems  Scott Petersen
Ubiquitous computing is creating new canvases and opportunities for creative ideas. This class explores the use of microprocessors, distributed sensor networks, IoT, and intermedia systems for the purposes of creative expression. The course is delivered in a mixed lecture and lab format that introduces the fundamental concepts and theory behind embedded systems as well as issues particular to their creative employment. The key objective of the course is for students to conceive of and implement creative uses of computation. To this end, skills to be obtained during the course are as follows: (1) appreciate the current efforts and motivation to push the limitations of computation for creative expression, both in new application and new foundational research; (2) weigh factors such as cost, power, processing, memory, I/O capabilities, and networking capabilities when choosing a set of embedded devices and sensors; (3) contextualize unfamiliar hardware and languages through examples, documentation, and familiar design pattern; and (4) manage communication between multiple languages, devices, and protocols. Additionally, at the end of the course students will have a portfolio of their work in the form of writing, code, video, audio, and physical artifacts. Prerequisite: CPSC 223 or equivalent or by permission of instructor.  QR  RP
CPSC 338b / EENG 348b, Digital Systems  Staff
Development of engineering skills through the design and analysis of digital logic components and circuits. Introduction to gate-level circuit design, beginning with single gates and building up to complex systems. Hands-on experience with circuit design using computer-aided design tools and programmable logic devices. Recommended preparation: EENG 201b. RP

CPSC 365b / ECON 366b, Algorithms  James Glenn
Paradigms for algorithmic problem solving: greedy algorithms, divide and conquer, dynamic programming, and network flow. NP completeness and approximation algorithms for NP-complete problems. Algorithms for problems from economics, scheduling, network design and navigation, geometry, biology, and optimization. Provides algorithmic background essential to further study of computer science. Either CPSC 365 or CPSC 366 may be taken for credit. Prerequisites: CPSC 202 and 223. QR

* CPSC 366b / ECON 366b, Intensive Algorithms  Yang Cai
Mathematically sophisticated treatment of the design and analysis of algorithms and the theory of NP completeness. Algorithmic paradigms including greedy algorithms, divide and conquer, dynamic programming, network flow, approximation algorithms, and randomized algorithms. Problems drawn from the social sciences, Data Science, Computer Science, and engineering. For students with a flair for proofs and problem solving. Either CPSC 365 or CPSC 366 may be taken for credit. Prerequisites: MATH 244 and CPSC 223. QR

CPSC 376a, Advanced Web Application Development in the Digital Humanities  Benedict Brown
Advanced applications of computer and data science in the humanities, including web technologies, visualization, and database design. Students work in teams to develop a variety of applications proposed by faculty and staff from the Digital Humanities Lab, the Institute for the Preservation of Cultural Heritage, and the Computer Science department. Meets with CPSC 376. Students may earn credit for CPSC 276 or 376; not both. Prerequisite: CPSC 223 or equivalent, or permission of the instructor. QR

Advanced Courses

* CPSC 421b, Compilers and Interpreters  Staff
Compiler organization and implementation: lexical analysis, formal syntax specification, parsing techniques, execution environment, storage management, code generation and optimization, procedure linkage and address binding. The effect of language-design decisions on compiler construction. After CPSC 323. QR

CPSC 422a, Design and Implementation of Operating Systems  Zhong Shao
The design and implementation of operating systems. Topics include synchronization, deadlock, process management, storage management, file systems, security, protection, and networking. After CPSC 323. QR

CPSC 423b, Principles of Operating Systems  Abraham Silberschatz
A survey of the underlying principles of modern operating systems. Topics include process management, memory management, storage management, protection and security, distributed systems, and virtual machines. Emphasis on fundamental concepts rather than implementation. After CPSC 323. QR

CPSC 424b, Parallel Programming Techniques  Staff
Practical introduction to parallel programming, emphasizing techniques and algorithms suitable for scientific and engineering computations. Aspects of processor and machine architecture. Techniques such as multithreading, message passing, and data parallel computing using graphics processing units. Performance measurement, tuning, and debugging of parallel programs. Parallel file systems and I/O. Prerequisite: CPSC 323, or CPSC 223 and significant experience with C/C++ programming in another science, social science or engineering discipline, or permission of instructor. QR RP

[ CPSC 426, Building Distributed Systems ]

CPSC 427a or b, Object-Oriented Programming  Staff
Object-oriented programming as a means to efficient, reliable, modular, reusable code. Use of classes, derivation, templates, name-hiding, exceptions, polymorphic functions, and other features of C++. After CPSC 223. QR

CPSC 428b, Language-Based Security  Zhong Shao
Basic design and implementation of language-based approaches for increasing the security and reliability of systems software. Topics include proof-carrying code, certifying compilation, typed assembly languages, runtime checking and monitoring, high-confidence embedded systems and drivers, and language support for verification of safety and liveness properties. After CPSC 202, 323, and MATH 222, or equivalents. QR

[ CPSC 430, Formal Semantics ]

CPSC 431a / MUSI 428a, Computer Music: Algorithmic and Heuristic Composition  Scott Petersen
Study of the theoretical and practical fundamentals of computer-generated music, with a focus on high-level representations of music, algorithmic and heuristic composition, and programming languages for computer music generation. Theoretical concepts are supplemented with pragmatic issues expressed in a high-level programming language. Ability to read music is assumed. After CPSC 202 and 223. QR

CPSC 432b / MUSI 427b, Computer Music: Sound Representation and Synthesis  Scott Petersen
Study of the theoretical and practical fundamentals of computer-generated music, with a focus on low-level sound representation, acoustics and sound synthesis, scales and tuning systems, and programming languages for computer music generation. Theoretical
concepts are supplemented with pragmatic issues expressed in a high-level programming language. Ability to read music is assumed. After CPSC 202 and 223. QR

[ CPSC 433, Computer Networks ]

* CPSC 434b, Topics in Networked Systems Yang Yang
Study of networked systems such as the Internet and mobile networks which provide the major infrastructure components of an information-based society. Topics include the design principles, implementation, and practical evaluation of such systems in new settings, including cloud computing, software-defined networking, 5G, Internet of things, and vehicular networking. Concurrently with or after CPSC 323. QR

[ CPSC 435, Internet-Scale Applications ]

CPSC 437a, Introduction to Database Systems Abraham Silberschatz

[ CPSC 438, Database System Implementation and Architectures ]

[ CPSC 439, Software Engineering ]

CPSC 446a or b, Data and Information Visualization Holly Rushmeier
Visualization is a powerful tool for understanding data and concepts. This course provides an introduction to the concepts needed to build new visualization systems, rather than to use existing visualization software. Major topics are abstracting visualization tasks, using visual channels, spatial arrangements of data, navigation in visualization systems, using multiple views, and filtering and aggregating data. Case studies to be considered include a wide range of visualization types and applications in humanities, engineering, science, and social science. Prerequisite: CPSC 223 or equivalent.

* CPSC 451b, The User Interface David Gelernter
The user interface (UI) in the context of modern design, where tech has been a strong and consistent influence from the Bauhaus and U.S. industrial design of the 1920s and 1930s through the IBM-Eames design project of the 1950s to 1970s. The UI in the context of the windows-menus-mouse desktop, as developed by Alan Kay and Xerox in the 1970s and refined by Apple in the early 1980s. Students develop a detailed design and simple implementation for a UI. Prerequisite: CPSC 223 or equivalent.

CPSC 453a, Machine Learning for Biology Smita Krishnaswamy
Applications of machine learning methods in the analysis of high-throughput biological data with focus on genomic and proteomic data. Topics include methods for denoising data; non-linear dimensionality reduction for visualization and progression analysis; unsupervised clustering; and information theoretic analysis of gene regulatory and signaling networks.

CPSC 454a, Software Analysis and Verification Ruzica Piskac
Introduction to concepts, tools, and techniques used in the formal verification of software. State-of-the-art tools used for program verification; detailed insights into algorithms and paradigms on which those tools are based, including model checking, abstract interpretation, decision procedures, and SMT solvers. After CPSC 202 and 323 or equivalents. QR RP

CPSC 455a / ECON 452a, Economics and Computation Yang Cai
A mathematically rigorous investigation of the interplay of economic theory and computer science, with an emphasis on the relationship of incentive-compatibility and algorithmic efficiency. Our main focus is on algorithmic tools in mechanism design, algorithms and complexity theory for learning and computing Nash and market equilibria, and the price of anarchy. Case studies in Web search auctions, wireless spectrum auctions, matching markets, and network routing, and social networks. Prerequisite: CPSC 365 or permission of the instructor. Familiarity with basic microeconomic theory is helpful but not required. QR

* CPSC 456b / EENG 451b, Wireless Technologies and the Internet of Things Wenjun Hu
Fundamental theory of wireless communications and its application explored against the backdrop of everyday wireless technologies such as WiFi and cellular networks. Channel fading, MIMO communication, space-time coding, opportunistic communication, OFDM and CDMA, and the evolution and improvement of technologies over time. Emphasis on the interplay between concepts and their implementation in real systems. Prerequisites: 1) Introductory courses in mathematics, engineering, or computer science covering basics of the following topics: Linux skills, Matlab programming, probability, linear algebra, and Fourier transform; 2) Or by permission of the instructor. The course material will be self-contained as much as possible. The labs and homework assignments require Linux and Matlab skills and simple statistical and matrix analysis (using built-in Matlab functions). There will be a couple of introductory labs to refresh Linux and matlab skills if needed.

[ CPSC 457, Sensitive Information in a Connected World ]

[ CPSC 465, Theory of Distributed Systems ]

CPSC 468b, Computational Complexity James Aspnes
Introduction to the theory of computational complexity. Basic complexity classes, including polynomial time, nondeterministic polynomial time, probabilistic polynomial time, polynomial space, logarithmic space, and nondeterministic logarithmic space. The roles
of reductions, completeness, randomness, and interaction in the formal study of computation. After CPSC 365 or 366, or with permission of instructor. QR

**CPSC 469a, Randomized Algorithms**  James Aspnes
A study of randomized algorithms from several areas: graph algorithms, algorithms in algebra, approximate counting, probabilistically checkable proofs, and matrix algorithms. Topics include an introduction to tools from probability theory, including some inequalities such as Chernoff bounds. After CPSC 365 or 366; a solid background in probability is desirable. QR

* **CPSC 471, Advanced Topics in Artificial Intelligence**

**CPSC 472a, Intelligent Robotics**  Brian Scassellati
Introduction to the construction of intelligent, autonomous systems. Sensory-motor coordination and task-based perception. Implementation techniques for behavior selection and arbitration, including behavior-based design, evolutionary design, dynamical systems, and hybrid deliberative-reactive systems. Situated learning and adaptive behavior. After CPSC 201 and 202 or equivalents. May not be taken after CPSC 473. QR

* **CPSC 473b, Intelligent Robotics Laboratory**  Brian Scassellati
Students work in small teams to construct novel research projects using one of a variety of robot architectures. Project topics may include human-robot interaction, adaptive intelligent behavior, active perception, humanoid robotics, and socially assistive robotics. Enrollment limited to 20. After CPSC 472. QR

**CPSC 474a, Advanced Topics in Artificial Intelligence**  James Aspnes
A study of randomized algorithms from several areas: graph algorithms, algorithms in algebra, approximate counting, probabilistically checkable proofs, and matrix algorithms. Topics include an introduction to tools from probability theory, including some inequalities such as Chernoff bounds. After CPSC 365 or 366; a solid background in probability is desirable. QR

**CPSC 475a / BENG 475a / EENG 475a, Computational Vision and Biological Perception**  Steven Zucker
An overview of computational vision with a biological emphasis. Suitable as an introduction to biological perception for computer science and engineering students, as well as an introduction to computational vision for mathematics, psychology, and physiology students. Prerequisite: CPSC 112 and MATH 120, or with permission of instructor. Q, S, RP

* **CPSC 476, Advanced Computational Vision**

**CPSC 477b, Natural Language Processing**  Dragomir Radev
Linguistic, mathematical, and computational fundamentals of natural language processing (NLP). Topics include part of speech tagging, Hidden Markov models, syntax and parsing, lexical semantics, compositional semantics, machine translation, text classification, discourse, and dialogue processing. Additional topics such as sentiment analysis, text generation, and deep learning for NLP. Prerequisites: CPSC 202 and CPSC 223. QR

* **CPSC 479b, Advanced Topics in Computer Graphics**  Julie Dorsey
An in-depth study of advanced algorithms and systems for rendering, modeling, and animation in computer graphics. Topics vary and may include reflectance modeling, global illumination, subdivision surfaces, NURBS, physically-based fluids systems, and character animation. After CPSC 202 and 223. QR

* **CPSC 480a or b, Directed Reading**  Staff
Individual study for qualified students who wish to investigate an area of computer science not covered in regular courses. A student must be sponsored by a faculty member who sets the requirements and meets regularly with the student. Requires a written plan of study approved by the faculty adviser and the director of undergraduate studies. May be taken more than once for credit.

* **CPSC 490a or b, Senior Project**  Staff
Individual research intended to fulfill the senior requirement. Requires a faculty supervisor and the permission of the director of undergraduate studies. The student must submit a written report about the results of the project.