COMPUTER SCIENCE

Director of undergraduate studies: Zhong Shao, 314 AKW, 432-6828, zhong.shao@yale.edu [F]; James Aspnes, 401 AKW, 432-1232, james.aspnes@yale.edu [Sp]; cpsc.yale.edu

FACULTY OF THE DEPARTMENT OF COMPUTER SCIENCE

Professors Dana Angluin, James Aspnes, † Dirk Bergemann, † Ronald Coifman, Julie Dorsey, Stanley Eisenstat, Joan Feigenbaum (Chair), Michael Fischer, David Gelernter, † Mark Gerstein, Drew McDermott, Vladimir Rokhlin, Holly Rushmeier, Brian Scassellati, Martin Schultz (Emeritus), Zhong Shao, Avi Silberschatz, Daniel Spielman, Y. Richard Yang, Steven Zucker

Associate Professors Daniel Abadi, Mahesh Balakrishnan

Assistant Professors † Wenjun Hu, † Amin Karbasi, Eric Koskinen (Visiting), † Sahand Negahban, Ruzica Piskac, Mariana Raykova, † Frederick Shic, † Jakub Szefer

Senior Lecturer Stephen Slade

Lecturers Jason Hirschhorn, † Kyle Jensen, Donya Quick, Brad Rosen, Ewa Syta, Xiyin Tang, Katherine Tsui

† A joint appointment with primary affiliation in another department or school.

The Department of Computer Science offers both B.S. and B.A. degree programs, as well as three combined majors in cooperation with other departments: Electrical Engineering and Computer Science (http://catalog.yale.edu/ycps/subjects-of-instruction/electrical-engineering-computer-science), Computer Science and Mathematics (http://catalog.yale.edu/ycps/subjects-of-instruction/computer-science-mathematics), and Computer Science and Psychology (http://catalog.yale.edu/ycps/subjects-of-instruction/computer-science-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 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. With the exception of 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 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 has significant readings and a paper, and satisfies the writing or the humanities and arts distributional requirement.
4. 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.
5. 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.
6. 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.)
7. 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 core courses: CPSC 201; CPSC 202 or MATH 244; CPSC 223, 323, 365, and 490. The B.S. degree program requires six additional intermediate or advanced courses in Computer Science, for a total of twelve; the B.A. degree program, four, for a total of ten. CPSC 480 and 490 may not be counted toward these electives. All courses in the major must be taken for a letter grade.
Students majoring in Computer Science are advised to complete CPSC 201 and 223 by the end of the sophomore year.

For students who already know how to program, typical B.S. programs starting in the freshman and sophomore years are:

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

and

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

For typical B.A. programs, two of the electives would be omitted.

**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 director of undergraduate studies, 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 (http://catalog.yale.edu/ycps/subjects-of-instruction/applied-mathematics) and in Engineering and Applied Science (http://catalog.yale.edu/ycps/subjects-of-instruction/engineering-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.

**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 director of undergraduate studies, students may enroll in 490 more than once or before their senior year.

**Schedule approval** All Computer Science majors in the sophomore, junior, and senior years should have their programs approved by the director of undergraduate studies.

**Combined B.S./M.S. degree program in Computer Science** 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. Eligibility requirements are described under “Simultaneous Award of the Bachelor’s and Master’s Degrees” in Section K, Special Arrangements (http://catalog.yale.edu/ycps/academic-regulations/special-arrangements), in the Academic Regulations. 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. In fulfilling these requirements, students must complete eight graduate courses from the approved list, up to two of which may, with the permission of the director of undergraduate studies 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.
3. At the end of their fifth term of enrollment students must have achieved at least three-fourths A or A– grades in all of their course credits directly relating to the major.

**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, 323, 365

**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 079b, Digital Photorealism**  Julie Dorsey  
Basic methods used to define shapes, materials, and lighting when creating computer-generated images. Mathematical models for shape, texture models, and lighting techniques. Principles are applied through the use of modeling/rendering/animation software. Proficiency in high school–level mathematics is assumed. No previous programming experience necessary. Enrollment limited to freshmen. Preregistration required; see under Freshman Seminar Program.  QR

**CPSC 100a, Introduction to Computing and Programming**  Brian Scassellati  
Introduction to the intellectual enterprises of computer science and to the art of programming, with attention to algorithmic thinking and efficient problem solving. Topics include abstraction, algorithms, data structures, encapsulation, resource management, security, software engineering, and Web development. Programming in the languages C, PHP, and JavaScript, as well as SQL, CSS, and HTML. Applications in biology, cryptography, finance, forensics, and gaming. Students view most course lectures on line; all sections, office hours, and related events are held locally. No previous programming experience required. Open to students of all levels and majors.  QR

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

**CPSC 113b, Programming and Entrepreneurship**  Kyle Jensen  
Techniques for advanced software development, management, and entrepreneurship that are used to build software start-up companies. Prerequisite: CPSC 100 or 112 or equivalent programming experience.

**CPSC 134a / MUSI 372a, Programming Musical Applications**  Donya Quick  
Topics in computer music, including musical representations for computing, automated music analysis and composition, interactive systems, and virtual instrument design. Use of domain-specific programming languages and libraries to explore how the principles of computer science can be applied to music to create new interfaces, instruments, and tools. Recommended preparation: the ability to read music or play an instrument.  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 184a or b, Intellectual Property in the Digital Age**  Xiyin Tang  
The evolving and oftentimes vexing intellectual property regime of the new digital age. Focus on copyright, fair use, remix culture, access to knowledge, technological innovations, the increasing relevance of trademarks in the new information society, the tension between creativity/creating and the intellectual property rules which either foster or inhibit it, and the new information culture of the digital age. Prerequisite: CPSC 183 or permission of instructor.  HU, 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**  Stephen Slade  
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.

**CPSC 201a or b, Introduction to Computer Science**  Stephen Slade  
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

Math: Stat/Applied Math

**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 223b, Data Structures and Programming Techniques  Stanley Eisenstat
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 257b, Information Security in the Real World  Ewa Syta
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 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.

MATH 244a / AMTH 244a, Discrete Mathematics  Asaf Ferber
Basic concepts and results in discrete mathematics: graphs, trees, connectivity, Ramsey theorem, enumeration, binomial coefficients, Stirling numbers. Properties of finite set systems. Recommended preparation: MATH 115 or equivalent. QR
Math: Stat/Applied Math
Math: Algebra/Number Theory

Intermediate Courses

CPSC 323a, Introduction to Systems Programming and Computer Organization  Stanley Eisenstat
Machine architecture and computer organization, systems programming in a high-level language, issues in operating systems, software engineering, prototyping in scripting languages. After CPSC 223. QR RP

CPSC 365b, Design and Analysis of Algorithms  Daniel Spielman
Paradigms for problem solving: divide and conquer, recursion, greedy algorithms, dynamic programming, randomized and probabilistic algorithms. Techniques for analyzing the efficiency of algorithms and designing efficient algorithms and data structures. Algorithms for graph theoretic problems, network flows, and numerical linear algebra. Provides algorithmic background essential to further study of computer science. After CPSC 202 and 223. QR
Math: Stat/Applied Math

Advanced Courses

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.

CPSC 424b, Parallel Programming Techniques  Andrew Sherman
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. After CPSC 223 and MATH 222 or 225, or equivalents. QR

[ CPSC 426, Building Decentralized Systems ]

CPSC 427b, Object-Oriented Programming  Michael Fischer
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 430, Formal Semantics ]

CPSC 432b, Computer Music: Sound Representation and Synthesis  Donya Quick
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 433b, Computer Networks  Yang Yang
An introduction to the design, implementation, analysis, and evaluation of computer networks and their protocols. Topics include layered network architectures, applications, transport, congestion, routing, data link protocols, local area networks, performance analysis, multimedia networking, network security, and network management. Emphasis on protocols used in the Internet. After CPSC 323. QR
[ CPSC 434, Mobile Computing and Wireless Networking ]
[ CPSC 435, Internet-Scale Applications ]
[ CPSC 436, Networked Embedded Systems and Sensor Networks ]

**CPSC 437a, Introduction to Databases**  Abraham Silberschatz

**CPSC 438b, Database System Implementation and Architectures**  Daniel Abadi
A study of systems programming techniques, with a focus on database systems. In the first half of the term, students analyze the design of a traditional DBMS and build components of a DBMS prototype, e.g., a catalog-manager, a buffer-manager, and a query execution engine. In the second half, students examine nontraditional architectures such as parallel databases, data warehouses, stream databases, and Web databases. After or concurrently with CPSC 202 and 223.  QR

**CPSC 439b, Software Engineering**  Ruzica Piskac
Introduction to fundamental concepts in software engineering and to the development and maintenance of large, robust software systems. The process of collecting requirements and writing specifications; project planning and system design; methods for increasing software reliability, including delta debugging and automatic test-case generation; type systems, static analysis, and model checking. Students build software in teams. After CPSC 323.  QR, RP

**CPSC 440b, Numerical Computation**  Vladimir Rokhlin
Algorithms for numerical problems in the physical, biological, and social sciences: solution of linear and nonlinear systems of equations, interpolation and approximation of functions, numerical differentiation and integration, optimization. After CPSC 112 or an equivalent introductory programming course; MATH 120; and MATH 222 or 225 or CPSC 202.  QR

Math: Stat/Applied Math

**CPSC 445a, Introduction to Data Mining**  Vladimir Rokhlin
A study of algorithms and systems that allow computers to find patterns and regularities in databases, to perform prediction and forecasting, and to improve their performance generally through interaction with data. After CPSC 202, 223, and MATH 222, or equivalents.  QR

* **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.  QR, SC

**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.  RP

[ CPSC 455, Economics and Computation ]

* **CPSC 457b, Sensitive Information in a Connected World**  Ewa Syta
Issues of ownership, control, privacy, and accuracy of the huge amount of sensitive information about people and organizations that is collected, stored, and used by today's ubiquitous information systems. Readings consist of research papers that explore both the power and the limitations of existing privacy-enhancing technologies such as encryption and "trusted platforms." After or concurrently with CPSC 365 and 467.  QR

**CPSC 458a, Automated Decision Systems**  Stephen Slade
The spectrum of automated decision models and tools, with a focus on their costs and effectiveness. Examples from a variety of fields, including finance, risk management, robotics, medicine, and politics. After CPSC 201 and 223 or equivalents.  QR

[ CPSC 462, Graphs and Networks ]

**CPSC 465b, Theory of Distributed Systems**  James Aspnes
Models of asynchronous distributed computing systems. Fundamental concepts of concurrency and synchronization, communication, reliability, topological and geometric constraints, time and space complexity, and distributed algorithms. After CPSC 323 and 365.  QR

**CPSC 467a, Cryptography and Computer Security**  Michael Fischer
A survey of such private and public key cryptographic techniques as DES, RSA, and zero-knowledge proofs, and their application to problems of maintaining privacy and security in computer networks. Focus on technology, with consideration of such societal issues as balancing individual privacy concerns against the needs of law enforcement, vulnerability of societal institutions to electronic attack, export regulations and international competitiveness, and development of secure information systems. Some programming may be required. After CPSC 202 and 223.  QR
CPSC 468b, Computational Complexity  Joan Feigenbaum
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 with permission of instructor. QR

[ CPSC 469, Randomized Algorithms ]

CPSC 470a, Artificial Intelligence  Drew McDermott
Introduction to artificial intelligence research, focusing on reasoning and perception. Topics include knowledge representation, predicate calculus, temporal reasoning, vision, robotics, planning, and learning. After CPSC 201 and 202. QR

[ CPSC 471, Advanced Topics in Artificial Intelligence ]

CPSC 472a, Intelligent Robotics  Katherine Tsui
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.

* 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.

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. After CPSC 112 and MATH 120, or with permission of instructor. QR, SC, RP

CPSC 478a, Computer Graphics  Holly Rushmeier
Introduction to the basic concepts of two- and three-dimensional computer graphics. Topics include affine and projective transformations, clipping and windowing, visual perception, scene modeling and animation, algorithms for visible surface determination, reflection models, illumination algorithms, and color theory. After CPSC 202 and 223. QR

* CPSC 480b, Directed Reading  James Aspnes
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, Special Projects  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.