Computer Science (CPSC)

CPSC S100a / CPSC 100a, Introduction to Computing and Programming  Cody Murphey and Jay Lim
In-person Course. 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. 1 Credit. Tuition: $4,650.
Session A: May 30 - July 1.  QR

* 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.  QR

CPSC 100a / CPSC S100a, Introduction to Computing and Programming  Cody Murphey and Jay Lim
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  o Course cr

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

CPSC 123b / PLSC 351b / S&DS 123b / S&DS 523b, YData: An Introduction to Data Science  Ethan Meyers
Computational, programming, and statistical skills are no longer optional in our increasingly data-driven world; these skills are essential
for opening doors to manifold research and career opportunities. This course aims to dramatically enhance knowledge and capabilities in
fundamental ideas and skills in data science, especially computational and programming skills along with inferential thinking. YData is an
introduction to Data Science that emphasizes the development of these skills while providing opportunities for hands-on experience and
practice. YData is accessible to students with little or no background in computing, programming, or statistics, but is also engaging for
more technically oriented students through extensive use of examples and hands-on data analysis. Python 3, a popular and widely used
computing language, is the language used in this course. The computing materials will be hosted on a special purpose web server.  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 184b, Intellectual Property in the Digital Age  Cecillia Xie
The seminar focuses on the evolving and oftentimes vexing intellectual property regime of the new digital age. Topics include 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 200a, 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.  QR

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

**CPSC 202a or b, Mathematical Tools for Computer Science**  
Staff  
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

**CPSC 262b / AMTH 262b / S&D 262b, Computational Tools for Data Science**  
Roy Lederman  
Introduction to the core ideas and principles that arise in modern data analysis, bridging statistics and computer science and providing students the tools to grow and adapt as methods and techniques change. Topics include principal component analysis, independent component analysis, dictionary learning, neural networks and optimization, as well as scalable computing for large datasets. Assignments include implementation, data analysis and theory. Students require background in linear algebra, multivariable calculus, probability and programming. Prerequisites: after or concurrently with MATH 222, 225, or 231; after or concurrently with MATH 120, 230, or ENAS 151; after or concurrently with CPSC 100, 112, or ENAS 130; after S&D 100-108 or S&D 230 or S&D 241 or S&D 242. Enrollment is limited; requires permission of the instructor. QR

**CPSC 276b, Introduction to Applications of Computer and Data Science for the Digital Humanities**  
Holly Rushmeier  
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 Yale humanities departments, the Digital Humanities Lab, the Institute for the Preservation of Cultural Heritage, and/or the Computer Science department. Meets with CPSC 376/HSAR 567/CLSS 840. 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, HU

* **CPSC 280a 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 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.

**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/PLSC 369 or for CPSC 310; not for both. Prerequisite: CPSC 223 or the equivalent. QR, SO 0 Course cr

**CPSC 323a or b, Introduction to Systems Programming and Computer Organization**  
Staff  
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 327a or b, Object-Oriented Programming**  
Timothy Barron  
Object-oriented programming as a means to designing and writing efficient, reliable, modular, and reusable code. Covers core concepts and features of object-oriented languages (classes, inheritance, composition, encapsulation, polymorphism, and exceptions) as well as the use of object-oriented design patterns (iterator, decorator, strategy, adapter, observer, etc.). This course was previously number CPSC 427. After CPSC 223. QR

**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. Q R P

**CPSC 338b / EENG 348b, Digital Systems**  Rajit Manohar
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 microcontroller programming. Recommended preparation: EENG 201. Q R

**CPSC 365b / ECON 365b, Algorithms**  Staff
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. Q R

**CPSC 376b, Advanced Computer and Data Science Applications for the Digital Humanities**  Holly Rushmeier
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 Yale humanities faculty, the Digital Humanities Lab, the Institute for the Preservation of Cultural Heritage, and the Computer Science department. Meets with CPSC 376/ CPSC 276/HSAR 567/CLSS 840. Students may earn credit for CPSC 276 or 376; not both. Prerequisite: CPSC 223 or equivalent, or permission of the instructor. Q R H U

**CPSC 413a, Computer System Security**  Timothy Barron
Overview of the principles and practice behind analyzing, designing, and implementing secure computer systems. Covers problems that have continued to plague computer systems for years as well as recent events and research in this rapidly evolving field of computer science. Learn to think from the perspective of an adversary; to understand systems well enough to see how their flaws could be exploited, and to consequently defend against such exploitation. Offers opportunities for hands-on exploration of attacks and defenses in the contexts of web applications, networks, and system level software. Also discusses ethical considerations and responsibilities associated with security research and practice. After CPSC 323.

* **CPSC 414a, Computing Then and Now: How Digital Technology Evolves**  Michael Fischer
The goal of this course is to provide the historical perspective needed to think critically about today's emerging computing technologies such as AI, self-driving cars, autonomous drones, quantum computers, and blockchains. This course traces the evolution of selected examples of digital technology from their intellectual bases through ubiquitous deployment. Examples are drawn from computer hardware and software systems, networking, algorithms, and applications. Prerequisite: CPSC 223 and junior or senior level standing in the major.

**CPSC 415a, Law, Security, and Logic**  Ruzica Piskac and Scott Shapiro
Martin Davis famously described computers as "Engines of Logic." Thanks to advances in computer science in the area of automated reasoning, these engines of logic are efficient machines capable of performing complex and laborious tasks in little time. This course teaches the methods of computer automated reasoning and examines their use in two contexts: cybersecurity and legal reasoning. Basic familiarity with predicate and propositional logic; basic knowledge and understanding of computer programming is required. H U R P

**CPSC 419a, Full Stack Web Programming**  Jay Lim
This course introduces students to a variety of advanced software engineering and programming techniques in the context of full-stack web programming. The focus of the course includes both client- and server-side programming (and database programming), client/server communication, user interface programming, and parallel programming. This course is designed for students who have taken CPSC 223 (but do not need CPSC 233 or higher-level computer science systems courses) and wish to learn the complete programming framework of Web programming. For a systematic treatment of core software engineering techniques, using Web programming as a running example framework, consider taking CPSC 439, which targets students with more extensive programming experiences (after CPSC 323). Prerequisite: CPSC 223

**CPSC 420b / EENG 420b, Computer Architecture**  Abhishek Bhattacharjee
This course offers a treatment of computer architectures for high-performance and power/energy-efficient computer systems. Topics include the foundations of general-purpose computing, including instruction set architectures, pipelines, superscalar and out-of-order execution, speculation, support for precise exceptions, and simultaneous multi-threading. We also cover domain-specific hardware (e.g., graphics processing units), and ongoing industry efforts to elevate them to the status of first-class computing units. In tandem, we cover topics relevant to both general-purpose and domain-specific computing, including memory hierarchies, address translation and virtual memory, on-chip networks, machine learning techniques for resource management, and coherence techniques. If time permits, we will study the basics of emerging non-classical computing paradigms like neuromorphic computing. Overall, this course offers insights on how the computing industry is combating the waning of traditional technology scaling via acceleration and heterogeneity. Prerequisites: CPSC 323, 223, and 202. This is a programming-intensive course, so comfort with large programming projects is essential.
* CPSC 421b, Compilers and Interpreters  Jay Lim
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 322. QR

CPSC 424a, 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. 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 426b, Building Distributed Systems  Y. Richard Yang
Ubiquitous services such as Google, Facebook, and Amazon run on the back of massive distributed systems. This course covers the fundamental principles, abstractions, and mechanisms that inform the design of such systems, as well as the practical details of real-world implementations. Technical topics covered include properties such as consistency, availability, durability, isolation, and failure atomicity; as well as protocols such as RPC, consensus, consistent hashing, and distributed transactions. The final project involves implementing a real-world distributed service. After CPSC 323. QR

CPSC 429a, Principles of Computer System Design  Lin Zhong
Humans are stupid; computers are limited. Yet a collaboration of humans and computers has led to ever more powerful and complex computer systems. This course examines the limitations of humans and computers in this endeavor and how they shape the design, implementation, and evaluation of computer systems. It surveys the empirical knowledge reported by scholars and practitioners that overcome such limitations. The lectures, reading assignments, and classroom discussions travel through psychology and philosophy and revisit important results from theoretical computer science, with a goal of elucidating the rationales behind the best practices in computer systems research and development. Prerequisite: CPSC 323 or equivalent. Students should have the ability to write significant system programs in at least one systems programming language (e.g., C, C++ and Rust).

CPSC 431a / MUSI 427a, 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 428b, 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 435a, Building an Internet Router  Robert Soule
Over the course of the semester, students build a fully functioning Internet router. Students design the control plane in Python on a Linux host and design the data plane in the new P4 language on the bmv2 software switch. To provide context and background for the design of their router, students read a selection of papers to get both a historical perspective and exposure to current research in networking. Prerequisite: CPSC 433.

CPSC 437a, Introduction to Database Systems  Avi Silberschatz

CPSC 439a or b, Software Engineering  Timos Antonopoulos
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 446a, 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.
CPSC 447a, Introduction to Quantum Computing  Yongshan Ding
This course introduces the fundamental concepts in the theory and practice of quantum computation. Topics include information processing, quantum programming, quantum compilation, quantum algorithms, and error correction. The objective of the course is to engage students in applying fresh thinking to what computers can do – we establish an understanding of how quantum computers store and process data, and discover how they differ from conventional digital computers. We anticipate this course will be of interest to students working in computer science, electrical engineering, physics, or mathematics. Prerequisites: CPSC 201 and CPSC 202. Basic familiarity with discrete probability and linear algebra is recommended. Prior experience in quantum computing is useful but not required. SC

CPSC 448a / EENG 426a / ENAS 876a, Silicon Compilation  Rajit Manohar
An upper-level course on compiling computations into digital circuits using asynchronous design techniques. Emphasis is placed on the synthesis of circuits that are robust to uncertainties in gate and wire delays by the process of program transformations. Topics include circuits as concurrent programs, delay-insensitive design techniques, synthesis of circuits from programs, timing analysis and performance optimization, pipelining, and case studies of complex asynchronous designs. Prerequisite: EENG 201 and introductory programming, or permission of instructor.

* CPSC 459b, 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 452b, Deep Learning Theory and Applications  Smita Krishnaswamy
Deep neural networks have gained immense popularity within the last decade due to their success in many important machine learning tasks such as image recognition, speech recognition, and natural language processing. This course provides a principled and hands-on approach to deep learning with neural networks. Students master the principles and practices underlying neural networks including modern methods of deep learning, and apply deep learning methods to real-world problems including image recognition, natural language processing, and biomedical applications. The course is based on homework, a final exam, and a final project (either group or individual, depending on the total number enrolled). The project includes both a written and oral (i.e. presentation) component. The course assumes basic prior knowledge in linear algebra and probability. Prerequisites: CPSC 202 and knowledge of Python Programming.

CPSC 454b, 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 425a, 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 458b, 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 223 or equivalents. QR

* CPSC 459a, Building Interactive Machines  Marynel Vazquez
This advanced course brings together methods from machine learning, computer vision, robotics, and human-computer interaction to enable interactive machines to perceive and act in a variety of environments. Part of the course examines approaches for perception with different sensing devices and algorithms; the other part focuses on methods for decision making and applied machine learning for control. Understanding of probability, differential calculus, linear algebra, and planning (in Artificial Intelligence) is expected for this course. Programming assignments require proficiency in Python and high-level familiarity with C++. Prerequisites: CPSC 201, CPSC 202, and CPSC 470 (or 570), or permission of the instructor. QR

* CPSC 464a, Algorithms and their Societal Implications  Nisheeth Vishnoi
Today’s society comprises humans living in an interconnected world that is intertwined with a variety of sensing, communicating, and computing devices. Human-generated data is being recorded at unprecedented rates and scales, and powerful AI and ML algorithms, which are capable of learning from such data, are increasingly controlling various aspects of modern society: from social interactions. These data-driven decision-making algorithms have a tremendous potential to change our lives for the better, but, via the ability to mimic and nudge human behavior, they also have the potential to be discriminatory, reinforce societal prejudices, violate privacy, polarize opinions, and influence democratic processes. Thus, designing effective tools to govern modern society which reinforce its cherished values such as equity, justice, democracy, health, privacy, etc. has become paramount and requires a foundational understanding of how humans, data, and algorithms interact. This course is for students who would like to understand and address some of the key challenges and emerging topics at the aforementioned interplay between computation and society. On the one hand, we study human decision-
making processes and view them through the lens of computation and on the other hand we study and address the limitations of artificial decision-making algorithms when deployed in various societal contexts. The focus is on developing solutions through a combination of foundational work such as coming up with the right definitions, modeling, algorithms, and empirical evaluation. The current focus is on bias and privacy, with additional topics including robustness, polarization, and democratic representation. Solid mathematical and programming background is necessary to enroll in this course. CPSC 365 and S&DS 251 are recommended.

**CPSC 465a, 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 365 or 366. QR

**CPSC 467b, Cryptography and Computer Security**  Charalampos Papamanthou
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 or MATH 244, and 223. QR

**CPSC 468a, Computational Complexity**  Staff
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 469b, 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 470b, Artificial Intelligence**  Staff
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. Q8

**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, Computational Intelligence for Games**  James Glenn
Introduction to techniques used for creating computer players for games, particularly board games. Topics include combinatorial and classical game theory, stochastic search methods, applications of neural networks, and procedural content generation. Prerequisites: CPSC 202 and CPSC 223. 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. QR, SC, RP

**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, or permission of instructor. QR

**CPSC 478b, Computer Graphics**  Julie Dorsey
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 479a, 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 482b, Current Topics in Applied Machine Learning  David van Dijk
We cover recent advances in machine learning that focus on real-world data. We discuss a wide range of methods and their applications to diverse domains, such as finance, health care, genomics, protein folding, drug discovery, neuroscience, and natural language processing. The seminar is based on a series of lectures by the instructor, guest lecturers, and student presentations. Student presentations are expected to be on recent publications from leading journals and conferences in the field, and are followed by discussions. A final project involves the application of a machine learning method to real-world data. Prerequisites: Basic programming knowledge (e.g., CPSC 112 or equivalent, in Python); mathematical background in Linear algebra (e.g., MATH 222/225 or equivalent); and Calculus (e.g., MATH 120 or equivalent); or instructor permission.

CPSC 484b, Introduction to Human-Computer Interaction  Marynel Vazquez
This course introduces students to the interdisciplinary field of Human-Computer Interaction (HCI), with particular focus on Human-Robot Interaction (HRI). The first part of the course covers principles and techniques in the design, development, and evaluation of interactive systems. It provides students with an introduction to UX Design and User-Centered Research. The second part focuses on the emergent field of HRI and several other non-traditional interfaces, e.g., AR/VR, tangibles, crowdsourcing. The course is organized as a series of lectures, presentations, a mid-term exam, and a semester-long group project on designing a new interactive system. After CPSC 201 and 202 or equivalents. Students who do not fit this profile may be allowed to enroll with the permission of the instructor.

* 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.