CPSC S100a / CPSC 100a, Introduction to Computing and Programming  Ozan Erat and Cody Murphey
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. Session A: May 29 – June 30. Tuition: $4850. QR

CPSC 100a / CPSC S100a, Introduction to Computing and Programming  Ozan Erat and Cody Murphey
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.

CPSC 110a, Python Programming for Humanities and Social Sciences  Sohee Park
Introduction to computer science and Python programming with domain-specific applications. Students learn how to think algorithmically and solve problems efficiently. Topics include abstraction, algorithms, data structures, web development, and statistical tools. Students learn to apply computing techniques in the fields of social sciences & humanities by analyzing data. No previous programming experience is required. This course is intended for students of social sciences & humanities majors.

* 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.

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, 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, Mathematical Tools for Computer Science**  Dylan McKay

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, 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 323a, Introduction to Systems Programming and Computer Organization**  James Glenn

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, 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. QR RP

**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. QR

**CPSC 365a / ECON 365a, Algorithms**  Andre Wibisono
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. Only one of CPSC 365, CPSC 366, or CPSC 368 may be taken for credit. Prerequisites: CPSC 202 and 223.

**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 419a, Full Stack Web Programming**  Alan Weide
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 323 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 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 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 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 434a, Topics in Networked Systems  Y. Richard 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 437a, Introduction to Database Systems  Avi Silberschatz

* CPSC 438a, Big Data Systems: Trends & Challenges  Anurag Khandelwal
Today’s internet scale applications and cloud services generate massive amounts of data. At the same time, the availability of inexpensive storage has made it possible for these services and applications to collect and store every piece of data they generate, in the hopes of improving their services by analyzing the collected data. This introduces interesting new opportunities and challenges designing systems for collecting, analyzing and serving the so called “big data”. This course looks at technology trends that have paved the way for big data applications, survey state of the art systems for storage and processing of big data, and future research directions driven by open research problems. Our discussions span topics such as cluster architecture, big data analytics stacks, scheduling and resource management, batch and stream analytics, graph processing, ML/AI frameworks, serverless platforms and disaggregated architectures. Prerequisite: CPSC 323.

CPSC 439a, 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.

CPSC 440a, Database Design and Implementation  Robert Soule
This course covers advanced topics in Database Systems, expanding on the material covered in CPSC 437. Topics include complex data types; application development; big data; data analytics; parallel and distributed storage; parallel and distributed query processing; advanced indexing techniques; advanced relational database design; and object-based databases.

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.

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.

CPSC 455a / ECON 425a, Economics and Computation  Staff
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.
* CPSC 457a, *Sensitive Information in a Connected World*  
Michael Fischer  
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." Junior or senior level standing with some background in computer science is required.

* 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 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.  

**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.  

**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.  

**CPSC 478a, Computer Graphics**  Theodore Kim  
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.  

**CPSC 480a, Introduction to Computer Vision**  Alex Wong  
This course focuses on fundamental topics in computer vision. We begin with the image formation process and discuss the role of camera models and intrinsic calibration in perspective projection. Basic image processing techniques (i.e. filtering) is introduced. After which, we discuss techniques to describe an image, from edges to feature descriptors and methods to establish correspondences between different images of the same scene. The course additionally covers topics in recognition (i.e. image classification, segmentation, detection, etc.) and reconstruction (i.e. stereo, structure-from-motion, optical flow). Machine learning and deep learning based methods in a subset of the topics covered are also introduced. Students get hands-on experience in implementing the techniques covered in the class and applying them to real world datasets and applications. Students taking this course must have successfully passed courses in data structures and object-oriented programming (e.g. CPSC 223a or equivalent courses), and foundational mathematical tools such as discrete math and linear algebra (e.g. CPSC 202 or equivalent courses). It is recommended that students have taken or successfully passed calculus (e.g. MATH 112, MATH 115, MATH 120, or equivalent courses), and linear algebra (e.g. MATH 225, or equivalent courses). A background in statistics, machine learning and deep learning is useful, but not required. Experience in programming with Python is preferable, as we use it for assignments and projects. Familiarity with Google Colab, and numerical and image processing packages (i.e. NumPy, SciPy, and Sci-kit Image) is helpful throughout the course.
**CPSC 483a, Deep Learning on Graph-Structured Data**  
Rex Ying  
Graph structure emerges in many important domain applications, including but not limited to computer vision, natural sciences, social networks, languages and knowledge graphs. This course offers an introduction to deep learning algorithms applied to such graph-structured data. The first part of the course is an introduction to representation learning for graphs, and covers common techniques in the field, including distributed node embeddings, graph neural networks, deep graph generative models and non-Euclidean embeddings. The first part also touches upon topics of real-world significance, including auto-ML and explainability for graph learning. The second part of the course covers important applications of graph machine learning. We learn ways to model data as graphs and apply graph learning techniques to problems in domains including online recommender systems, knowledge graphs, biological networks, physical simulations and graph mining. The course covers many deep techniques (graph neural networks, graph deep generative models) catered to graph structures. We will cover basic deep learning tutorials in this course. Prerequisites: CPSC 201, CPSC 223, and one of CPSC 365 or CPSC 366. Knowledge of graphs as a data structure, and understanding of basic graph algorithms are essential for applying machine learning to graph-structured data. Familiarity with Python and important libraries such as Numpy and Pandas are helpful. CPSC 452 and CPSC 453 are highly recommended prior because they cover the foundations of deep neural networks. Experience in machine Learning courses such as CPSC 481, and Graph Theory courses such as CPSC 462 are welcomed as well.  

**CPSC 485a, Applied Planning and Optimization**  
Daniel Rakita  
This course introduces students to concepts, algorithms, and programming techniques pertaining to planning and optimization. At a high level, the course teaches students how to break down a particular problem into a state-space or a state-action space, how to select an effective planning or optimization algorithm given the problem at hand, and how to ultimately apply the selected algorithm to achieve desired outputs. Concepts are solidified through grounded, real-world examples (particularly in robotics, but also including machine learning, graphics, biology, etc.). These examples come in the form of programming assignments, problem sets, and a final project. General topics include discrete planning, sampling-based path planning, optimization via matrix methods, linear programming, computational differentiation, non-linear optimization, and mixed integer programming. After the course, students are able to generalize their knowledge of planning and optimization to any problem domain. Knowledge of linear algebra and calculus is expected. Students should be familiar with matrix multiplication, derivatives, and gradients.  

* **CPSC 490a, Senior Project**  
Sohee Park  
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.