COMPUTER SCIENCE (CPSC)

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 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 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. QR 0 Course cr

CPSC 110a or b, 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. QR 0 Course cr

CPSC 112b, Introduction to Programming Timothy Barron and Cody Murphey
Development on the computer of programming skills, problem-solving methods, and selected applications. No previous experience with computers necessary. QR 0 Course cr

* **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 200, 201 or equivalent. QR

* **CPSC 280a or b, Directed Reading** Y. Richard Yang
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** Y. Richard Yang
Individual research. Requires a faculty supervisor and the permission of the director of undergraduate studies. May be taken more than once for credit.

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

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. 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 364a, Introduction to Blockchains, Cryptocurrencies, Smart Contracts, and Decentralized Applications Fan Zhang
This course offers an introduction to blockchain technology and its practical applications. The objective is to provide students with a comprehensive overview of the fundamental concepts and hands-on experience in building on actual blockchains. The course covers the technological foundation of the blockchain stack (consensus layer, ordering layer, execution layer, etc.), the design of representative applications (cryptocurrencies, smart contracts, Decentralized Finance, etc.), and the principles for writing secure smart contracts, and ends with an overview of the latest research directions. To provide a hands-on building experience, the course hosts a Catch-the-Flag (CTF) competition where students are asked to hack buggy smart contracts within a controlled environment. Prerequisite: CPSC 201. QR

CPSC 365a or b / ECON 365a or b, 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. Only one of CPSC 365, CPSC 366, or CPSC 368 may be taken for credit. Prerequisites: CPSC 202 and 223. QR

*** CPSC 366b / AMTH 366b / ECON 366b, Intensive Algorithms** Anna Gilbert
Mathematically sophisticated treatment of the design and analysis of algorithms and the theory of NP completeness. Algorithmic paradigms including greedy algorithms, divide and conquer, dynamic programming, network flow, approximation algorithms, and randomized algorithms. Problems drawn from the social sciences, Data Science, Computer Science, and engineering. For students with a flair for proofs and problem solving. Only one of CPSC 365, CPSC 366, or CPSC 368 may be taken for credit. Prerequisites: MATH 244 and CPSC 223. QR

CPSC 370a, Artificial Intelligence Tesca Fitzgerald
How can we enable computers to make rational, intelligent decisions? This course explores fundamental techniques for Artificial Intelligence (AI), covering topics such as search, planning, learning, and reasoning under uncertainty. Through hands-on programming projects, students learn conceptual, algorithmic, and practical considerations for implementing foundational AI algorithms. By the end of this class, students have an understanding of the history and breadth of AI problems and topics, and are prepared to undertake more advanced courses in robotics, computer vision, natural language processing, and machine learning. Prerequisites: CPSC 202 and CPSC 223. Students should also be familiar with basic object-oriented programming concepts in Python.

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 414b, 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. NOTE: This course meets during Reading Period: the week between the last week of classes and finals week. Prerequisite: CPSC 223 and junior or senior level standing in the major.

CPSC 419a or b, 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 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 421a, 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 422b, Design and Implementation of Operating Systems Anurag Khandelwal
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 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 428b, Language-Based Security Zhong Shao
Basic design and implementation of language-based approaches for increasing the security and reliability of systems software. Topics include proof-carrying code, certifying compilation, typed assembly languages, runtime checking and monitoring, high-confidence embedded systems and drivers, and language support for verification of safety and liveness properties. After CPSC 202, 323, and MATH 222, or equivalents. QR

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

CPSC 433b, Computer Networks Y. Richard 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 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 435b, 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 or b, Introduction to Database Systems Avi Silberschatz
Introduction to database systems. Data modeling. The relational model and the SQL query language. Relational database design, integrity constraints, functional dependencies, and normal forms. Object-oriented databases. Database data structures: files, B-trees, hash indexes. After CPSC 223. QR

* **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 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 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. QR

CPSC 441a, Zero-Knowledge Proofs Benjamin Fisch
This is a course in cryptographic proof systems. In the digital world today, we trust services to perform many kinds of computations on our data, from managing financial ledgers and databases to complex analytics. We trust these services not only to operate correctly, but also to keep our information private. Proof systems allow us to remove this trust. A succinct proof system is a system that enables a service to attach a small certificate on the correctness of its computation, and the certificate can be verified by small devices, even if the original computation needs substantial computation to compute this result. Beyond correctness, a zero-knowledge proof system enables us to prove knowledge of secret information, including hidden inputs to a computation that achieves a certain output. Both types of proof systems have incredible applications to privacy and verifiability in a decentralized web. Prerequisites: CPSC 201 and 202 (or equivalent, e.g. MATH 244). Recommended: CPSC 467 (Cryptography), MATH 225 (Linear Algebra).

CPSC 442a, Theory of Computation Dylan McKay
This course first introduces core, traditional ideas from the theory of computation with more modern ideas used in the process, including basic ideas of languages and automata. Building on the core ideas, the course then covers a breadth of topics in modular units, where each unit examines a new model and potentially a new perspective on computation. Topics may include: basic notions of Complexity Theory, provability and logic, circuits and non-uniform computation, randomized computation, quantum computation, query-based computation, notions of machine learning, compression, algebraic models of computation. Additional topics might be introduced in lectures or student projects, according to student interests, including mechanism design, voting schemes, cryptography, biological computation, distributed computation, and pseudorandomness. Prerequisite: one of CPSC 365, 366, or 368 is required. This course is a proof-based theory course and mathematical maturity is expected.

CPSC 443a, Optimal Transport: Theory, Algorithms and Applications to Data Science Smita Krishnaswamy
Optimal transport started with Gaspart Monge in the 1700s when he stated the problem of moving a large pile of sand (whose shape is a probability distribution) to a target pile with minimal effort. The optimal transport plan not only gives a coupling between distributions but also a metric between such probability measures, which has found use in everything from modern neural networks to economic resource allocation problems, to shape matching in computer vision. This class covers the theoretical foundations as well as computational aspects of optimal transport starting with the original formulations as maps between discrete measures, and extending to general measures as well as the key Kantorovich relaxation as a coupling between measures, and its metric properties. We also cover algorithmic foundations of optimal transport using linear programs that have recently been sped-up via entropic regularizations.

In addition to the primal form we cover the dual form and relaxations which lead to integral probability metrics. We vary the ground space of optimal transport from Euclidean, to arbitrary metrics, to graphs. We move from static to dynamic formulations of optimal transport, which can provide paths of flow for dynamics that are energy-constrained. Finally, we cover important extensions such as unbalanced optimal transport which allows for transport between generic measures (without the same volume), and for Gromov-Wasserstein distances between measures on different metric spaces. Prerequisites: MATH 241, CPSC 202, CPSC 223, and CPSC 365. Knowledge of python programming is also required.

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. Prerequiste: 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 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 454a, Software Analysis and Verification Ruzica Piskac
Introduction to concepts, tools, and techniques used in the formal verification of software. State-of-the art tools used for program verification; detailed insights into algorithms and paradigms on which those tools are based, including model checking,

abstract interpretation, decision procedures, and SMT solvers. After CPSC 202 and 323 or equivalents. QR RP

CPSC 455a / ECON 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. QR

* **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 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 463b, Algorithms for Convex Optimization Nisheeth Vishnoi
Convex optimization has played a major role in the recent development of fast algorithms for problems arising in areas such as theoretical computer science, discrete optimization, and machine learning. The goal of this course is to design state-of-the-art algorithms for various classical discrete problems through the use of continuous optimization/sampling. The approach is to first formulate the problem as a continuous (convex) optimization problem, even though the problem is over a discrete domain, adapt or develop deterministic or randomized continuous-time dynamical systems to solve it, and then design algorithms for the problem via appropriate discretizations. The algorithmic applications include maximum flow in graphs, maximum matching in bipartite graphs, linear programming, submodular function minimization, and counting problems involving discrete objects such as matroids. We present approaches gradient descent, mirror descent, interior-point methods, and cutting plane methods.

Prerequisites: CPSC 365 or permission of the instructor. S&DS 430 and a solid background in calculus, linear algebra, probability, and algorithms is recommended.

*** 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 466b, Blockchain and Cryptocurrency Benjamin Fisch
This course is an introduction to blockchain systems, such as Bitcoin and Ethereum. We begin with a brief history of blockchains and an overview of how they are being used today before launching into foundational topics, including distributed consensus, smart contracts, cryptographic building blocks from signatures to authenticated datastructures, and the economics of blockchains. We then cover advanced topics including the scalability and interoperability of blockchain systems and applications such as "decentralized finance" (DeFi). The lectures and assignments engage students in both theoretical and applied aspects of blockchain systems. The course assumes background in various fundamental areas of CS, including discrete math, probability, algorithms, data structures, and networks. Required: CPSC 202 and 223 (or equivalent). Recommended: CPSC 467 (Cryptography). QR

CPSC 467b, Introduction to Cryptography Staff
This class introduces modern symmetric and public-key cryptography as well as their broad applications, both from a theoretical and practical perspective. There is an initial emphasis on fundamental cryptographic primitives (e.g., block ciphers, pseudorandom functions, pseudorandom generators, one-way functions), their concrete efficiency and

implementation as well as their security definitions and proofs. Ways of combining such primitives that lead to more complex objects used to secure today's internet (e.g., via TLS), such as key exchange, randomized encryption, message authentication codes and digital signatures are also studied. The last part of the class is devoted to modern and more advanced applications of cryptography (some of which are deployed at scale today), such as authenticated data structures, zero-knowledge proofs, oblivious RAM, private information retrieval, secret sharing, distributed consensus and cryptocurrencies (e.g, Bitcoin). Some programming may be required. After CPSC 202 or MATH 244, and CPSC 223. 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 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 474a or b, 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 476b / BENG 476b, Advanced Computational Vision Steven Zucker
Advanced view of vision from a mathematical, computational, and neurophysiological perspective. Emphasis on differential geometry, machine learning, visual psychophysics, and advanced neurophysiology. Topics include perceptual organization, shading, color and texture analysis, and shape description and representation. After CPSC 475. QR, SC

CPSC 477b, Natural Language Processing Arman Cohan
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 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. QR

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

CPSC 480a, 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 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 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. QR

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. SO

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. QR

CPSC 486b, Probabilistic Machine Learning Andre Wibisono
This course provides an overview of the probabilistic frameworks for machine learning applications. The course covers probabilistic generative models; learning and inference; algorithms for sampling; and a survey of generative diffusion models. This course studies the theoretical analysis of the problems and how to design algorithms to solve them. This course familiarizes students with techniques and results in literature, and prepares them for research in machine learning. Prerequisites: Introductory machine learning (CPSC 481 or S&DS 265); linear algebra (MATH 222); probability (S&DS 241); and calculus (MATH 120).

CPSC 488a, AI Foundation Models Arman Cohan
Foundation models are a recent class of AI models that are large-scale in terms of number of parameters and are trained on broad data (generally using self-supervision at scale). These models have demonstrated exceptional capabilities in natural language processing, computer vision, and other tasks. Examples of Foundation Models are GPT-4, ChatGPT, GPT-3, Dall-E, Stable Diffusion, etc. In this course, we discuss building blocks of foundation models, including transformers, self-supervised learning, transfer learning, learning from human feedback, power of scale, large language models, in-context learning, chain-of-thought prompting, parameter-efficient fine-tuning, vision transformers, diffusion models, generative modeling, safety, ethical and societal considerations, their impact, etc. While the course primarily focuses on advances on large language models, we also cover foundation models in computer vision, as well as multi-modal foundation models. Prerequisite: Either CPSC 477 or CPSC 480, or permission of the instructor.

* **CPSC 490a or b, 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.