COMPUTER SCIENCE (CPSC)

*** CPSC 0350b / MUSI 0035b, 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. QR

CPSC 1001a or b, Introduction to Programming Staff

Development on the computer of programming skills, problem-solving methods, and selected applications. No previous experience with computers necessary. QR o Course cr

CPSC 1100a, 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, but other majors are also welcome. QR o Course cr

CPSC 1230a or b / PLSC 3508a or b / S&DS 1230a or b / S&DS 5230a or b, YData: An Introduction to Data Science Staff

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 1500a, 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 1710a, Introduction to AI Applications Xiuye (Sue) Chen

'Introduction to AI Applications' demystifies the core principles and practical applications of artificial intelligence for students with introductory programming experience. Covering essential topics like machine learning models, data handling,

ethical AI use, and real-world problem-solving with AI, this course incorporates hands-on projects to foster an intuitive understanding of AI's capabilities and limitations. Students will emerge from the class with foundational AI knowledge, ready to apply AI solutions across various domains and explore more specialized AI disciplines. CPSC 100 or equivalent introductory programming experience. QR

CPSC 1830a, 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 1840b, 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 1850b, 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 2000a or b, 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 2010a or b, Introduction to Computer Science Staff
Introduction to the concepts, techniques, and applications of computer science. Topics include computer systems (the design of computers and their languages); theoretical foundations of computing (computability, complexity, algorithm design); and artificial intelligence (the organization of knowledge and its representation for efficient search). Examples stress the importance of different problem-solving methods. After CPSC 100, CPSC 112 or equivalent. QR

CPSC 2020a 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 0 Course cr

*** CPSC 2155b / AMST 2255b, Artificial Intelligence, Ethics, and Society** Julian Posada
This seminar examines the development and implementation of artificial intelligence technologies across a broad array of social contexts, incorporating historical, cultural, economic, legal, and political perspectives. The course provides an in-depth study of contemporary AI, from its historical development and varied definitions to

current issues, emphasizing the relationship between power dynamics and ethical considerations. After establishing a foundation in theories and the study of ethics and power, the course delves into diverse aspects of AI, including the implications of human labor and material infrastructures in the development of the technology, concerns related to bias and discrimination, and its impacts on the environment. The concluding module applies these discussions to real-world scenarios, exploring how to address ethical and societal issues through legal and human rights frameworks, governance and regulation, and grassroots initiatives. This course is ideal for both computer science and engineering students seeking a socio-humanistic perspective on artificial intelligence, and humanities and social sciences students interested in the societal implications of AI. HU, SO

CPSC 2230a 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 2000 or 2010 (formerly CPSC 200 or 201). QR 0 Course cr

* **CPSC 2265a / AMST 2265a, Topics in Critical Computing** Julian Posada and Theodore Kim
This course introduces the social, cultural, and political contexts shaping the contemporary development and use of computing and information technology. Through structured discussions, lectures, and collaborative activities, participants will explore computing's historical evolution, ethical and societal implications, and tangible impacts, including its reliance on transnational infrastructures and environmental effects. Emphasis will be placed on analyzing computer-related social issues through theoretical and critical approaches, empirical research, and governance frameworks, as well as both technical and social strategies for addressing key challenges. The course is designed for students from diverse academic backgrounds across all divisions, aiming to develop a nuanced understanding of computation's intersection with broader social systems and to equip them with tools to engage with critical issues in the rapidly shifting digital landscape. HU, SC

* **CPSC 2800a, Directed Reading** Theodore Kim
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 2900a, Directed Research** Theodore Kim
Individual research. Requires a faculty supervisor and the permission of the director of undergraduate studies. May be taken more than once for credit.

CPSC 3230a 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 3270a 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 3340a, 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 3380b / ECE 3481b, Digital Systems Staff
Development of engineering skills through the design and analysis of digital logic components and circuits. Introduction to gate-level circuit design, beginning with single gates and building up to complex systems. Hands-on experience with circuit design using computer-aided design tools and microcontroller programming. Recommended preparation: EENG 201. QR

CPSC 3650a or b / ECON 3365a or b, Algorithms James Glenn
Paradigms for algorithmic problem solving: greedy algorithms, divide and conquer, dynamic programming, and network flow. NP completeness and approximation algorithms for NP-complete problems. Algorithms for problems from economics, scheduling, network design and navigation, geometry, biology, and optimization. Provides algorithmic background essential to further study of computer science. Only one of CPSC 365 or CPSC 366 may be taken for credit. Prerequisites: CPSC 202 or MATH 244, CPSC 223. QR

*** CPSC 3660b / AMTH 3660b / ECON 3366b, 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 3700a, 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 4130b, 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 4150b, Law and Large Language Models Ruzica Piskac and Scott Shapiro
This course is intended for computer science and law students interested in how artificial intelligence can be applied to legal reasoning. It combines basic AI theory with practical project work, focusing on using tools like large language models (LLMs) and other AI technologies for tasks common in legal practice. Students learn how to automate case summarization, draft legal memos and briefs, simulate oral arguments for better argumentation skills, and assist in the preparation of pro-se motions for self-represented litigants. The course emphasizes hands-on experience, helping students build real-world skills in applying AI in legal settings. Our goal is to bring together students from computer science and from law and match them together in the teams. Each team works on a project that automates a specific aspect of the legal process or legal reasoning, focusing on practical, real-world applications. Prerequisites: Basic coding skills, including knowledge of Python; interest in Large Language Models (LLMs); basic understanding of linear algebra and calculus; familiarity with basic probability (e.g., likelihood, averages) and simple statistical concepts (like mean and variance) SO

CPSC 4160a, Lattices and Post-Quantum Cryptography Katerina Sotiraki
This course explores the role of lattices in modern cryptography. In the last decades, novel computational problems, whose hardness is related to lattices, have been instrumental in cryptography by offering: (a) a basis for "post-quantum" cryptography, (b) cryptographic constructions based on worst-case hard problems, (c) numerous celebrated cryptographic protocols unattainable from other cryptographic assumptions. This course covers the foundations of lattice-based cryptography from fundamental definitions to advanced cryptographic constructions. More precisely, we introduce the Learning with Error (LWE) and the Short Integer Solutions (SIS) problems and study their unique properties, such as the fact that their average-case hardness is based

on the worst-case hardness of lattice problems. Next, we cover lattice constructions of advanced cryptographic primitives, such as fully homomorphic encryption and signature schemes. Finally, we introduce some notions of quantum cryptography and explore the role of lattices in this area.#Overall, this course offers insights into the foundations and recent advancements in lattice-based cryptography. Prerequisite: CPSC 467/567 or equivalent and linear algebra

CPSC 4190a 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 4200b / ECE 4201b, 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 4210a, Compilers and Interpreters** Zhong Shao
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 4230b, 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 4240b, Parallel Programming Techniques Quanquan Liu
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 4261a, Building AI Infra Systems** Y. Richard Yang
This course covers the design, deployment, operations, and optimization of infrastructure systems that power large-scale modern AI systems such as Large-Language Models (LLMs). It takes an all-resources view, considering AI infrastructure components spanning compute, memory, storage, network, data, and energy resources. Focusing on core AI infrastructure design goals including efficiency, scalability, and stability, the course studies not only the basic mechanisms but also the complete systems to realize the goals. Labs and a capstone project form the core of the course. Prerequisite: CPSC 323, or an equivalent course in systems programming. Background in AI is highly recommended.

CPSC 4270a, C++ Programming for Stability, Security, and Speed Michael Fischer
Computer programming involves both abstraction and practice. Lower-level programming courses focus on learning how to correctly implement algorithms for carrying out a task. This course treats a computer program as an artifact with additional attributes of practical importance including execution efficiency, clarity and readability, redundancy, safety in the face of unexpected or malicious environments, and longevity - the ability to evolve over time as bugs are discovered and requirements change. This course is taught using modern C++. Prerequisite CPSC 223.

CPSC 4310a / MUSI 4228a, 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 4320b / MUSI 4227b, 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 4330b, 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 4350b, 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 437oa, 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 438oa, 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 439oa, 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

* **CPSC 439 1b, Advanced Software Engineering** Timos Antonopoulos
This course builds on CPSC 439 Software Engineering with a focus on a) building systems that scale well, and b) the technical infrastructure and approaches that would guide or inform entrepreneurship/business decisions. During the whole semester, teams work on a term-length software project of students' design, most often a continuation of the project they worked on during CPSC 4390 or CPSC 5390. After CPSC 4390 or similar. Students will have to have a working product they built during CPSC 4390 or similar course, to further develop during this course. QR

CPSC 441oa, Verifiable, Private, Decentralized Computing in the Age of AI Ben Fisch
You type your question into ChatGPT and get back a response – how do you trust its accuracy? Perhaps you have reviewed the latest published benchmark results for GPT-4, or trust that others have. But how do you know the response you are getting from OpenAI's servers is the true output of GPT-4? Perhaps due to a bug or system overload your question was handled by a weaker AI model. Or worse, perhaps the servers were hacked by someone maliciously giving you incorrect results. And, how do you trust that the sensitive questions you are sending to ChatGPT will not be leaked or used against you? 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, such as large-language model (LLM) inference. We trust these services not only to operate correctly, but also to keep our information private. Cryptographic systems allow us to remove this trust. A "succinct" cryptographic proof enables a service to attach a small certificate on the correctness of its computation which can be verified easily on a low-power device, even if the original computation was extremely complex. The proof of some ML computation that was run for hours on a GPU farm can fit in an email and take just milliseconds to verify on a mobile device! 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. For instance, OpenAI could prove that the response is the true output of GPT-4 (a proprietary model) without revealing sensitive details about the model itself. In industry, the market for cryptographic proofs is currently around $75 million and projected to reach $10 billion by 2030, according to some estimates. Cryptographic proofs have become the leading technology for scaling blockchains and achieving privacy in cryptocurrencies. Verifiable and zero-knowledge computing also create an important foundation for decentralizing AI services. Training and serving large models currently require vast resources, leading to centralization. Decentralized ML networks offer a compelling alternative—letting many independent operators contribute incremental work to the overall task using their own resources big or small, from a single GPU to a server cluster, and earn a share of the payments clients make to use the service. In such a setting it is critical to verify that operators contribute incremental work correctly. Or, they may use private data to jointly train an ML model. Succinct zero-knowledge proofs would enable these operators to prove correctness of their work without revealing sensitive data. We will cover some of the other challenges and directions in decentralized model training and inference, such as reducing the amount of data that needs to be communicated between physically distributed islands of hardware and the potential role of reinforcement learning. Prerequisites: CPSC 201 and 202 (or equivalent, e.g. MATH 244). Recommended: CPSC 467 (Cryptography), MATH 225 (Linear Algebra).

CPSC 4420a, 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 4440b, Real-World Cryptography** Fan Zhang
Cryptography provides strong security and privacy guarantees in well-defined mathematical models, but applying it to real-world systems is an art—one that

must account for performance, cost, evolving adversarial threats, and even user behavior. This course aims to impart the art of designing and applying cryptography in the real world, by examining select advanced cryptographic tools used in practice. Topics include secure channels, identity and credentials, anonymity, end-to-end encrypted messaging, and Trusted Execution Environments (TEEs). Students are expected to be familiar with concepts in computer security and cryptography (e.g., from CPSC 4130, CPSC 4670, or similar courses). To set the stage, we will go over the content of Katz and Lindell (https://www.cs.umd.edu/~jkatz/imc.html) in the first few lectures at a quick pace.

CPSC 4460a, 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 4470a, 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 4490b, Quantum Information Systems Yongshan Ding
Quantum information systems encompass the hardware, software, and networking systems that are designed to encode, store, process, and distribute quantum information. In this course, students get a complete view of such information systems and explore the current advancement associated with building practical quantum computers and networks. This course is structured as four modules: quantum information theory, quantum processor, quantum memory, and quantum network. Prerequisite: CPSC 447 or PHYS 345 or equivalent. This course is intended for advanced undergraduates who are familiar with basic quantum computation and information. We anticipate this course will be of interest to students working in computer science, electrical engineering, or physics. SC

* **CPSC 4510b, 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 452o b, 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 454o a, 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 455o a / ECON 442 5a, Algorithmic Game Theory Manolis Zampetakis
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 3650 (formerly CPSC 365) or permission of the instructor. Familiarity with basic microeconomic theory is helpful but not required. QR

CPSC 458o b, 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 459o a, 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 4626a, Scalable and Private Graph Algorithms** Quanquan Liu
What techniques can we use to deal with modern real-world data with billions of data points? How do we account for strong adversaries that violate the privacy of users providing this data? This course will provide you with the knowledge to tackle

research questions in these domains. We will propose answers and techniques to these broad questions from an algorithmic standpoint, presenting foundational topics such as:1) The parallel, distributed, and streaming models and algorithmic techniques commonly used within these models 2) Differential privacy and mechanisms for private data analysis 3) Implementation techniques, tools, and examples that demonstrate the practicality of these algorithms in real-world systems. This course focuses on advanced topics in practical graph algorithms with provable guarantees beyond the sequential model used in most introductory algorithms classes. Specific topics include local graph techniques for problems such as maximal matching, independent set, k-core decomposition, densest subgraphs, and coloring as well as global techniques for problems like connectivity, shortest paths, and spanners. Introductory lectures will also feature techniques used beyond graph algorithms. Students are asked to read and present influential recent research papers on these topics. Papers come from prominent CS theory conferences such as STOC, FOCS, SODA as well as database and data mining conferences like VLDB, PODS, and WWW. In addition to these presentations, students also work on a final project which may be theoretical or implementation-based. The course will also feature voluntary open problem sections where we discuss (known) practice problems and open-ended research questions related to the topics in this course in a collaborative group setting. Prerequisites: CPSC 2020 and CPSC 3650

CPSC 4630b, Algorithms via 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. A solid background in calculus, linear algebra, probability, and algorithms is recommended. It is intended for students who are comfortable with proofs, who would like to be prepared for graduate school, and who want to improve their problem-solving abilities.

*** CPSC 4640a, 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 4650b, 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 4660b, Web3, Blockchains, and Cryptocurrencies Ben 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 4670a, Introduction to Cryptography Charalampos Papamanthou
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 4671b, Advanced Topics in Cryptography: Cryptography and Computation** Charalampos Papamanthou
Traditional cryptography is mostly concerned with studying the foundations of securing communication via, for example, encryption and message authentication

codes. This class studies the applications of cryptography in securing *computation*. Topics include, but are not limited to, fundamental results and the most recent progress in oblivious computation and private information retrieval (PIR), zero-knowledge proofs, secure computation, consensus algorithms, searchable encryption, and lattice-based cryptography. The class focuses both on theory and applications. Prerequisite: CPSC 4670 or equivalent. This course assumes prior knowledge of fundamental notions in cryptography and mathematical maturity as well as comfort with programming.

CPSC 4680b, Computational Complexity Dylan McKay
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 4690a, 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 4710a, Trustworthy Deep Learning Rex Ying
In recent years, deep learning has seen applications in many fields, from science and technology, to finance, humanity, and businesses. However, real-world, high-impact machine learning applications demand more than just model performance. In particular, deep learning models are often required to be "trustworthy," so that domain experts can trust that the models consistently behave in a way that corresponds to their domain knowledge. For example, medical experts would expect a deep learning diagnosis model to be able to explicitly utilize medical domain knowledge in its prediction; an insurance company would expect a decision on insurance price to be explainable in terms of risk factors; a financial company would expect its fraud detection model to be robust to adversarial attacks; a physicist would expect models to provide consistency with the underlying laws. This course introduces various fields of trustworthy deep learning, including model robustness, defenses for adversarial attacks, interpretability, explainability, fairness, privacy, domain adaptation, rules, and constraints. The course covers some of these aspects in the context of graph neural networks but also covers many other ML models in general deep learning, natural language processing, and computer vision. Prerequisites: a course in linear algebra and multi-variable calculus. Familiarity with PyTorch and other common Python libraries such as Numpy, Sklearn. Deep learning courses such as CPSC 452 or 453 are recommended.

CPSC 4740a, 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 4750a / BENG 4475a / ECE 4750, 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 4770b, Natural Language Processing Arman Cohan
This course provides a deep dive into modern Natural Language Processing (NLP), with a strong focus on Language Modeling. The curriculum spans both foundational concepts and cutting-edge developments in the field. The course begins with core neural network concepts in NLP, covering word embeddings, sequence modeling, and attention mechanisms. Building on these foundations, we explore transformer architectures and their evolution, including early transformer language models like BERT, GPT and T5. The course examines how these models enable sophisticated language understanding and generation through pre-training and transfer learning. The latter portion covers contemporary advances: Large Language Models (LLMs), multi-modal integration, parameter-efficient fine-tuning, evaluation, multi-agent systems, reasoning, and model compression. We'll analyze the capabilities and limitations of current systems while discussing emerging research directions. Prerequisites: CPSC 202 and CPSC 223, or permission of instructor. QR

CPSC 4780b, 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 4790b, 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 4791a, Building Game Engines Michael Shah
This course teaches the fundamentals of building a reusable software architecture by building games. This is a programming intensive course where the end product of this course is a data-driven game engine that students work in small teams to implement in a systems programming language (e.g. C, C++, D, etc.). Students apply data structures, algorithms, and systems programming skills in the domain of games. Discussion and implementation of the components of a game engine may include: resource management (allocators, resource managers, serialization), abstraction (design patterns, game objects, scripting, graphics layers), graphics management algorithms (scene graphs, level of detail), physics (linear algebra, collision detection and resolution algorithms), artificial intelligence (e.g. pathfinding,decision making), and performance (concurrency, parallelism, math). Students work on a final course project for their portfolio. Prerequisites: CPSC 2230 and CPSC 3230

CPSC 4792b, Real-Time 3D Computer Graphics Programming Michael Shah
This course teaches the fundamentals of real-time 3D computer graphics programming using a systems programming language (e.g. C, C++, D, etc.). Students interested in making 3D games, virtual reality applications, simulations, medical visualizations, and other interactive applications are the target audience. Throughout the course students will also learn about co-processors (e.g. GPUs) for hardware accelerated graphics, and program in a graphics API enabling hardware accelerated graphics. Students will apply a sampling of mathematics in the domain of geometry, trigonometry, linear algebra, and calculus in order to generate photo and non-photorealistic images in real-time. Additional topics may include: geometry processing, scene organization, texturing techniques, advanced lighting techniques, compute shaders, and topics in performance. Prerequisite: either CPSC 411 or CPSC 478.

CPSC 4800a, 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 4830b, 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 4840b, 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 4850a, 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 4870b, 3D Spatial Modeling and Computing Daniel Rakita
Several areas of computer science and related fields must model and compute how objects are situated in three-dimensional space over time, such as robotics, computer vision, computer graphics, computational physics, computational biology, aerospace engineering, and so on. This course teaches students how to computationally model the spatial configuration of and spatial relationships between objects over time. Topics include various methods for representing spatial configurations and transformations (such as transformation matrices, Euler angles, unit quaternions, dual quaternions, etc.), hierarchical chaining of spatial transformations, derivatives of spatial representations concerning time, computing intersections and penetration depths between objects in space, interpolating over spatial representations (such as using splines), signal processing over spatial transformations, optimizing over spatial representations, and more. After CPSC 202 and CPSC 223. All students should have

proficiency in programming with mathematical reasoning. Background in linear algebra and calculus is recommended but not required.

*** CPSC 489ob, Robot Learning** Tesca Fitzgerald
This class explores methods for grounding machine learning algorithms in embodied, interactive robots. We cover topics including learning from demonstration, active learning, inverse reinforcement learning, representations for modeling high-level and low-level task information, and human factors for designing learning interactions. Students are asked to read and present research papers on these topics from top publication venues in AI, machine learning, robotics, and human-robot interaction. Students also complete lab assignments in which they implement and evaluate state-of-the-art methods for interactive robot learning on a physical robot arm. Required: CPSC 370/470/570 or an equivalent AI course (requires approval by instructor). Recommended: An introductory Machine Learning course such as S&DS 265/365 or CPSC 381/481.

*** CPSC 490oa 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.