COMPUTER SCIENCE

Department Website:

https://www.haverford.edu/computer-science

Computer science is the representation and manipulation of information; it is the study of the theory, analysis, design, and implementation of the data structures that represent information and the algorithms that transform them. Computer science is interdisciplinary, with roots in mathematics, physics, and engineering, and with applications in virtually every academic discipline and professional enterprise.

Computer science at Haverford College covers these fundamental concepts, with emphasis on depth of thought, clarity of expression and attention to ethical impact. This approach is consistent with the principles of scientific education in the liberal arts. Our aim is to provide students with a base of skills and capabilities that support a wide variety of postgraduation goals, rather than to follow short-term fashions and fluctuations in computer hardware and software.

Learning Goals

Each student in computer science will be able to:

- Realize their full ability to think deeply. This involves mastering discipline-specific concepts such as abstraction, correctness, and complexity, and recognizing their broad and deep applications, both theoretically and practically, in new contexts.
 - Identify the role of abstraction in a computational problem situation; for example, distinguish a general problem from a specific problem instance, or understand the mapping between an abstract data type (ADT) and a given representation of that ADT.
 - Develop original, correct solutions demonstrating an appropriate level of abstraction, using two or more design techniques specific to the field.
 - Express a general solution in an appropriate programming language.
 - Analyze and compare the efficiency of alternative solutions, both quantitatively and qualitatively.
 - Increase confidence in a solution through a variety of approaches, including code review, testing, and mathematical reasoning.
- Communicate their thinking clearly and effectively. This involves taking a discovered or developed solution (or a given problem definition, etc.) and sharing that solution with

peers, managers, clients, and other professionals, in a complete and persuasive manner, and with appropriate use of vocabulary and other tools (e.g., charts, proofs, demonstrations).

 Identify, interpret and evaluate the theoretical, practical, and ethical implications of their work in the field.
This work is most easily identified as software, but other results might be papers written and published, projects chosen over others ignored, and even questions raised during the software development process.

Haverford's Institutional Learning Goals are available on the President's website, at http://hav.to/ learninggoals.

Curriculum

Computer science offers:

- a major.
- a concentration for mathematics majors.
- a minor.

NB: The concentration will not be available to students admitted for the fall of 2019 and beyond; for prior classes, these programs will remain available on an "if space permits" basis.

Computer science also contributes substantially to the Concentration in Scientific Computing. More information on this concentration can be found on the program's website (*https://www.haverford.edu/ scientific-computing*) or catalog entry.

The major in computer science is designed for students who wish to explore fundamental questions about computation and the role of computation in society. As part of this exploration, we provide many opportunities for students to design, implement, and analyze algorithms and data structures, and develop a larger-scale hardware/software system over the course of multiple semesters. These opportunities include both individual projects and group work, and provide experience with a variety of programming languages and with computer hardware. The senior experience, and the final projects in many classes, provide opportunities for students to explore their own interests in computer science.

Major Requirements

The major program covers the foundations of the discipline and provides a range of elective opportunities. While the computer science major is inspired by guidance from existing professional societies in computing, it is uniquely "Haverfordian" in its emphasis on a collaborative approach to a rigorous field of inquiry.

Requirements are:

- Introduction: CMSC H105 and CMSC H106, or CMSC H107, or Bryn Mawr equivalents
- A 200-level + 300-level sequence in each of the three tracks, which would include the following courses:
 - theory: CMSC H231 followed by either CMSC H340 or CMSC H345
 - systems: (CMSC H240 or CMSC H251) + CMSC H356, or (CMSC B223 or CMSC H251) + CMSC H350, or (CMSC B223 or CMSC H251) + CMSC B355
 - applications: CMSC H260 followed by either CMSC H325 or CMSC H360 or CMSC H364
- Electives (two courses): one CMSC course at the 200+ level, one CMSC course at the 300+ level
- Thesis (one course): CMSC H399

A maximum of two courses for the major can be transferred from outside the Quaker Consortium; the introductory sequence must be taken at the Bi-Co, the 200 level core and two of the three 300 level requirements, must be taken within the Quaker Consortium.

Requests for exception must be pre-approved by the Chair of the Department.

Senior Thesis

The senior thesis in computer science is a capstone experience under the guidance of a faculty member.

Students complete a thorough literature review in the initial term, and can continue with a research project into the subsequent term. Oral, poster and written presentations are required. This experience can include original work, but it must demonstrate deep thinking and an original exposition of an advanced topic.

Students are required to enroll in a one-credit senior seminar course in the Fall term to ensure that they successfully complete this graduation requirement. There is a series of class activities and deadlines to help keep students on track for completing their thesis. In the fall semester, these include: the advisor selection process; submitting the topic proposal; completing the literature review; and the public poster presentation. In the optional spring semester, these include: implementing their project proposed in the previous term; completing a rough draft of their thesis; rehearsing their oral presentations; submitting the final thesis document; and giving their oral presentation. A second reader provides feedback periodically to the student and their advisor as to whether progress is satisfactory.

A detailed schedule is provided to all students in the seminar at the beginning of the year.

Senior Project Learning Goals

The thesis work culminates in the writing and oral presentation of a paper. The student must also demonstrate the research skills required to produce this paper, in accordance with departmental deadlines.

An undergraduate senior paper may or may not include original research, but must present an indepth exploration of a topic in computer science (with particular focus on understanding and evaluating some element of the computer science literature). The paper should demonstrate the student's ability to apply, in a new context, the fundamental themes and objectives that connect all computer science classes, such as:

- separating a problem definition from its solution.
- describing clearly a proposed solution (typically with examples).
- understanding the correctness and applicability of a proposed solution.
- comparing several proposed solutions in terms of clarity, resource requirements, etc.

It is common for the thesis to center on a particular algorithm or computing system, and present the correctness and/or computational complexity thereof. However, this is not required. Students have successfully pursued other topics, such as humancomputer interaction. The one core requirement is that the student demonstrates the ability to think deeply and communicate clearly about a computer science topic beyond the depth covered in classes.

The written thesis often resembles a review article, which explores in depth a collection of primary source articles from a single research group, or a survey article, which compares primary source articles from different origins.

The oral presentation is given after the thesis has been completed, though preliminary presentations are often also given as practice (and for formative assessment) during the year. The presentation is not graded, although all students are required to give one.

The learning goals for the research that goes into the thesis experience are as follows:

Aspirational (for the best students):

A substantial written contribution that demonstrates original thinking and/or insight about a research area

inside computer science, under the supervision of a faculty member. This should include a full literature review, appropriate replication of existing work, and either:

- a clear hypothesis (model), validation (proof/ experiments), and analysis; or
- original expository work, including the extension of a proof, or a new proof of an existing theorem.
- Since such theses include original material, they may constitute part of a publication (typically a joint publication with the advisor). However, publication is not required.

Achievable (for most students):

A confirmation and reiteration of existing work with an incremental contribution. Specifically, this includes a full literature review and either:

- a good and complete confirmation of an existing experiment on new data, including a good analysis; or
- an exposition of non-trivial graduate-level published work, including an existing proof or deep explanation of its extension/applicability (or its lack of extension) to other related concepts.

Required (of all students):

A non-trivial literature review/exposition of existing graduate-level published work, specifically:

The introductory material must be:

- readable by someone who has understood only the core computer science undergraduate material (e.g., programming languages, hardware, theory, algorithms, and at least one intensive systems course such as compilers or operating systems).
- detailed enough to be clear to someone within the field.

The discussion of related work should:

- include all the important related/foundational work.
- clearly identify what problem is being addressed by each work (possibly one statement of this for many/all the works).
- clearly state the basic approach being taken.
- explain how each paper supports/evaluates its own results (proof/empirical-study/ad-hoc argument).
- make clear how this work relates to the thesis itself.
- in at least one case, really address the details of how the approach works (possibly several such

discussions will be needed to address the point above).

Senior Thesis Assessment

The grade is approximately 75% based on the work done under the supervision of the faculty advisor and about 25% based on meeting the deadlines of and participating in the senior seminar, including the fall poster and spring presentation.

The senior paper is primarily assessed by the student's advisor. Usually one or more other members of the department also read the paper and provide feedback for the student and advisor. If the student has a separate subject-matter advisor at another institution, that advisor is consulted during the grading of the paper if at all possible. All faculty involved in the thesis (and many students) are typically in attendance for the oral presentation.

The grade for the senior experience is assigned by the advisor, based on the quality of the student's written paper (judged in terms of illustrating mastery of the learning objectives relevant to the chosen topic), on participation in the oral presentation, and on the work habits illustrated during the year's work.

After thorough discussion by the Department, a student's grade on the thesis will reflect how closely they have met the qualitative goals stated above. Specifically:

- 4.0: meets aspirational goals stated above.
- 3.0: meets achievable goals stated above.
- 2.0: meets required goals stated above.

All students should reach at least a 2.0 level of work on the material they submit by the end of the fall semester, and the faculty will certify students as having achieved this level (or not) in January.

In addition to submitting the written thesis document, students must also complete the assigned *presentation* elements, which typically include a December poster presentation of the thesis topic and scope, and the final oral presentation of the thesis. These presentations are graded on evidence of preparation and on participation (i.e. showing up on time for one's own presentation, attending the rehearsals of a few others, and providing feedback and/or asking questions). Faculty will provide informal feedback to the presenters on speaking style, professionalism, diction/grammar, poise, etc., but these elements are not included in the grade.

The Computer Science minor requirements follow the same philosophy and structure as the major:

- a. the introductory sequence
- b. breadth: a 200-level course in *each* element of the field (theory, systems, and applications)
- c. depth: one year-long sequence (200-level into 300-level) in *either* theory, systems, or applications

Minor Requirements

- CMSC H105 Introduction to Computer Science or CMSC H107 or Bryn Mawr CMSC B113.
- CMSC H106 Introduction to Data Structures or CMSC H107 or Bryn Mawr CMSC B151.
- CMSC H231 Discrete Mathematics
 - Students with strong backgrounds in mathematics and prior knowledge of the topics covered in CMSC H231 may wish to seek instructor permission to place into CMSC H340 /CMSC H345 without prior completion of CMSC H231—in this case, the student may complete the requirements for the minor with another course covering discrete mathematics, from the following list: MATH H210 (Linear Optimization), MATH H394 (Logic), MATH H394 (Cryptography), MATH H395 (Combinatorics), or STAT H203, STAT H218, STAT H286, or STAT H396.
- CMSC H251 Principles of Computing Systems
 - Students wishing to continue to CMSC B355 may substitute CMSC B223 Systems Programming
 - Students not taking a 35X course may substitute CMSC H240 Principles of Computer Organization or CMSC H245 Principles of Programming Languages
- CMSC H260 Foundations of Data Science
- One 300-level core course from the following list
 - CMSC H340 Analysis of Algorithms
 - CMSC H345 Theory of Computation
 - CMSC H350 Compiler Design
 - CMSC B355 Operating Systems
 - CMSC H356 Concurrency and Co-Design in Operating Systems
 - CMSC H325 Computation Linguistics
 - CMSC H360 Machine Learning

Concentration Requirements

The Computer Science Department supports the Concentration in Scientific Computing, available to a variety of majors (*https://www.haverford.edu/scientific-computing*), and provides a computer science concentration specific to mathematics majors.

Computer Science Concentration for Mathematics Majors Requirements

NB: This concentration will not be available to students admitted for the fall of 2019 and beyond; for prior classes, these programs will remain available on an "if space permits" basis.

- CMSC H105 (Introduction to Computer Science) and CMSC H106 (Introduction to Data Structures), or CMSC H107.
- *Either* CMSC H240 (Principles of Computer Organization) *or* CMSC H245 (Principles of Programming Languages).
- *Either* CMSC H340 (Analysis of Algorithms) *or* CMSC H345 (Theory of Computation).
- One cross-listed MATH/CMSC course (Note that CMSC H231 meets this requirement and is the prerequisite for CMSC H340 and CMSC H345.)
- One additional 300-level computer science course.

Related Concentration

Concentration in Scientific Computing

Computation is the *object of study* for the computer science major and minor; computation is also an important *tool* with which to study many other disciplines. The Concentration in Scientific Computing focuses on the application of computational techniques in other natural and social sciences.

For more information about the concentration, please see the program's catalog entry or website.

Affiliated Program

Engineering

Computer science majors may pursue various engineering disciplines via our partnerships with the University of Pennsylvania and CalTech. More information on this partnership can be found on the Engineering website.

Study Away

A maximum of two courses for the major can be transferred from outside the Bi-Co, and the introductory sequence, CMSC H240, CMSC H245, and either CMSC H340 or CMSC H345, must be taken at the Bi-Co.

Requests for exception to this policy must be preapproved by the Chair of the Department.

Facilities

Information on all hardware and software resources for the programs in computer science may be found on the departmental website.

Affiliated Faculty

Jane Chandlee

Associate Professor of Linguistics (TriCo)

John Dougherty Associate Professor and Chair of Computer Science

Rebecca Everett Associate Professor of Mathematics and Statistics

Sorelle Friedler The Shibulal Family Computer Science Professor; Professor of Computer Science

Alvin Grissom Associate Professor of Computer Science

Dakotah Lambert Visiting Assistant Professor of Computer Science

Suzanne Lindell Amanuensis

Steven Lindell Professor of Computer Science

David Lippel

Visiting Assistant Professor of Mathematics and Statistics

Robert Manning

Professor of Mathematics and Statistics; William H. and Johanna A. Harris Chair of Computational Science; Chair of Mathematics and Statistics

Sara Mathieson

Associate Professor of Computer Science; Coordinator of Scientific Computing

Xerxes Minocher

Mellon Post-Doctoral Fellow in the John B. Hurford '60 Center for the Arts and Humanities and Visiting Assistant Professor of Peace, Justice and Human Rights

Thao Nguyen

Assistant Professor of Computer Science

David Wonnacott

Professor of Computer Science

Yuxin Zhou Visiting Assistant Professor of Computer Science

Faculty at Bryn Mawr

Elizabeth Dinella Assistant Professor of Computer Science

Deepak Kumar Professor of Computer Science

Aline Normoyle Assistant Professor of Computer Science

Adam Poliak Assistant Professor of Computer Science

Dianna Xu Chair and Professor of Computer Science

Yuxin Zu Visiting Assistant Professor of Computer Science

Courses

NB: Bryn Mawr courses are described at https:// www.brynmawr.edu/cs/courses

CMSC H105 INTRODUCTION TO COMPUTER SCIENCE (1.0 Credit)

Alvin Grissom, Suzanne Lindell Division: Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes Introduction to the intellectual and software tools used to create and study algorithms: formal and informal problem specification; problem solving and algorithm design techniques; reliability, formal verification, testing, and peer code review techniques; program clarity, complexity and efficiency; functional and imperative paradigms; associated programming skills. Students must attend a one-hour weekly lab. Labs will be sectioned by course professor. Prerequisite(s): May not be taken by students who have taken any one of HC: CMSC 104, CMSC 107; BMC: CMSC 110, except by instructor consent (**Offered:** Fall 2024)

CMSC H106 INTRODUCTION TO DATA STRUCTURES (1.0 Credit)

Sara Mathieson, Suzanne Lindell **Division:** Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes An introduction to the fundamental data structures of computer science: strings, lists, stacks, queues, trees, BSTs, graphs, sets and their accompanying algorithms. Principles of algorithmic analysis and object reasoning and design will be introduced using mathematical techniques for the notions of both complexity and correctness. More practical issues, such as memory management and hashing, will also be covered. The programming language used to illustrate and implement these concepts will be able to support functional, imperative and objectoriented approaches. Emphasis will be placed on recursive thinking and its connection to iteration. Students must attend a one-hour weekly lab. Labs will be sectioned by course professor. Prerequisite(s): CMSC 105 (or 110 or 113 at Bryn Mawr) or instructor consent; may not be taken by students who have taken any one of HC: CMSC 104, CMSC 107; BMC: CMSC 206, CMSC 151, except by instructor consent (**Offered:** Fall 2024, Spring 2025)

CMSC H107 INTRODUCTION TO COMPUTER SCIENCE AND DATA STRUCTURES (1.0 Credit) Suzanne Lindell

Division: Natural Science; Quantitative Domain(s): C: Physical and Natural Processes An accelerated treatment of CMSC 105/106 for students with significant programming experience. Reviews programming paradigms, while focusing on techniques for reasoning about about software: methodical testing, formal verification, code reviews, other topics as time permits. Includes lab work. Prerequisite(s): CMSC104 or instructor consent, or placement by CS faculty, based on CS placement test. If you are interested in CMSC 107, you should preregister for the CMSC 105 section at the same time and take the placement test by the deadline, typically Wednesday before classes start; may not be taken by students who have taken any one of HC: CMSC 105, CMSC 106; BMC: CMSC 206, except by instructor consent (Offered: Fall 2024)

CMSC H208 SPEECH SYNTHESIS AND RECOGNITION (1.0 Credit)

Jane Chandlee

Division: Natural Science; Symbolic Reasoning **Domain(s):** C: Physical and Natural Processes An introduction to the methodologies used in the automated recognition and synthesis of human speech, focusing on Hidden Markov Models in recognition and unit selection in synthesis. Students will get hands-on experience with implementing the various components of these systems to better understand the techniques, challenges, and open areas of research. Crosslisted: Computer Science, Linguistics Prerequisite(s): LING 204, CS105 and 106 OR CS107 OR BMC 110 and 206 OR instructor consent

CMSC H222 SCIENTIFIC COMPUTING: CONTINUOUS SYSTEMS (1.0 Credit) Rebecca Everett

Division: Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes A survey of major algorithms in modern scientific computing, with a focus on continuous problems. Topics include numerical differentiation and integration, numerical linear algebra, rootfinding, optimization, Monte Carlo methods, and discretization of differential equations. Basic ideas of error analysis are presented. Regular computer work in class introduces students to the software package Matlab, in which the algorithms are implemented and applied to various problems in the natural and social sciences. Crosslisted: Mathematics, Computer Science Prerequisite(s): Math 121

CMSC H231 DISCRETE MATHEMATICS (1.0 Credit)

Steven Lindell

Division: Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes An introduction to discrete mathematics with strong applications to computer science. Topics include set theory, functions and relations, propositional logic, proof techniques, difference equations, graphs, and trees. Co-requisite(s): CMSC 105, 107, or B110 or B113 or instructor consent (**Offered:** Fall 2024, Spring 2025)

CMSC H245 PRINCIPLES OF PROGRAMMING LANGUAGES (1.0 Credit)

Division: Natural Science **Domain(s):** C: Physical and Natural Processes Study of the design and implementation of modern programming languages: lexical and syntactic analysis; scoping mechanisms; runtime environments; implementation of structured, functional, object-oriented, and concurrent programming languages. Lectures cover theoretical foundations of language design and implementation; labs provide opportunities to both use and implement language features. Prerequisite(s): CMSC 106, or 107 or 206, and CMSC/Math 231 (or instructor consent)

CMSC H251 PRINCIPLES OF COMPUTING SYSTEMS (1.0 Credit)

Staff

Division: Natural Science

Domain(s): C: Physical and Natural Processes What actually happens when you hit "run", after writing your program? This course introduces the elements of hardware and language/O.S. software that execute a program, serving as a foundation for later work in these areas, and providing insights into computing efficiency that may be important to a wide range of programmers. Includes weekly lab exercises, on principles covered in lecture, and details from lecture and self-teaching (according to resource-use principles presented in the course). Pre-requisite(s): Both introductory CS (CMSC H106, H107, or B151) and CMSC 231, with the latter allowed as co-requisite (Note that CMSC 223 and 251 cover substantially the same material, and thus students may not take both); (**Offered:** Fall 2024, Spring 2025)

CMSC H260 FOUNDATIONS OF DATA SCIENCE (1.0 Credit)

Sara Mathieson, Sorelle Friedler **Division:** Natural Science

Domain(s): C: Physical and Natural Processes This course will introduce students to the principles of learning from data, including basic modeling, applied linear algebra, probability, statistics, and visualization. The lab component will focus on implementation and analysis in Python. Prerequisite(s): MATH 105 or equivalent, CMSC H106/ CMSC B151 (Data Structures), corequisite CMSC H231 (Discrete Math), or permission of the instructor.

(Offered: Fall 2024, Spring 2025)

CMSC H265 CRITICAL STUDY OF DATA AND ALGORITHMS (1.0 Credit)

Xerxes Minocher

Division: Social Science

Domain(s): B: Analysis of the Social World This class focuses on the social impact of data and algorithms. Students will be introduced to 1) what are data and algorithms, 2) how data and algorithms intersect with problems of peace, justice, and human rights, especially in terms of social inequality, 3) how to critically assess and challenge data and algorithms, and 4) the role of individual and collective action in responding to relevant problems. No prior experience with programming, data, or algorithms required. Crosslisted: CMSC, PEAC Lottery Preference: PJHR concentrators

CMSC H340 ANALYSIS OF ALGORITHMS (1.0 Credit)

Steven Lindell

Division: Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes Qualitative and quantitative analysis of algorithms and their corresponding data structures from a precise mathematical point of view. Performance bounds, asymptotic and probabilistic analysis, worst case and average case behavior. Correctness and complexity. Particular classes of algorithms such as sorting searching will be studied in detail. Crosslisted: Computer Science, Mathematics Prerequisite(s): CMSC 106 or 107 or B206, and 231, or instructor consent (**Offered:** Fall 2024)

CMSC H345 THEORY OF COMPUTATION (1.0 Credit)

Staff

Division: Natural Science

Domain(s): C: Physical and Natural Processes Introduction to the mathematical foundations of computer science: finite state automata, formal languages and grammars, Turing machines, computability, unsolvability, and computational complexity. Attendance at the weekly discussion section is required. Crosslisted: Computer Science, Mathematics Prerequisite(s): (CMSC 106, 107, 151, or 206) and CMSC 231, and junior or senior standing, or instructor consent (**Offered:** Spring 2025)

CMSC H350 COMPILER DESIGN (1.0 Credit) David Wonnacott

Division: Natural Science

Domain(s): C: Physical and Natural Processes An introduction to compiler design, including the tools and software design techniques required for compiler construction. Students construct a working compiler using appropriate tools and techniques in a semester-long laboratory project. Lectures combine practical topics to support lab work with more abstract discussions of software design and advanced compilation techniques. Prerequisite(s): CMSC H251 or CMSC B223; concurrent enrollment in this and two other CMSC lab courses requires instructor consent

CMSC H356 CONCURRENCY AND CO-DESIGN IN OPERATING SYSTEMS (1.0 Credit)

John Dougherty

Division: Natural Science

Domain(s): C: Physical and Natural Processes A practical introduction to the principles of sharedmemory concurrent programming and of hardware/ software co-design, which together underlie modern operating systems; includes a substantial laboratory component, currently using Java's highlevel concurrency and the HERA architecture. Prerequisite(s): CMSC 251 or B223 or H240; concurrent enrollment in this and two other CMSC lab courses requires permission of the instructor (**Offered:** Spring 2025)

CMSC H360 MACHINE LEARNING (1.0 Credit) Alvin Grissom

Division: Natural Science; Quantitative **Domain(s):** C: Physical and Natural Processes To explore both classical and modern approaches, with an emphasis on theoretical understanding. There will be a significant math component (statistics and probability in particular), as well as a substantial implementation component (as opposed to using high-level libraries). However, during the last part of the course we will use a few modern libraries such as TensorFlow and Keras. By the end of this course, students should be able to form a hypothesis about a dataset of interest, use a variety of methods and approaches to test your hypothesis, and be able to interpret the results to form a meaningful conclusion. We will focus on real-world, publicly available datasets, not generating new data. Prerequisite(s): CMSC 260 or instructor consent (**Offered:** Spring 2025)

CMSC H364 COMPUTATIONAL BIOLOGY (1.0 Credit)

Sara Mathieson

Division: Natural Science

Domain(s): C: Physical and Natural Processes This course introduces foundational algorithms that have become essential for learning from biological data. With the genome sequencing revolution, it has become easier and cheaper to obtain genetic data, but often challenging to store, analyze, and make sense of this data. These questions have driven new algorithm development and repurposed existing algorithms for biology. We will study these algorithms from a variety of angles, including theory, implementation, application, biological interpretation, and communication of results. Prerequisite(s): CS260 "Foundations of Data Science Lottery Preference: 1. senior CMSC majors; 2. junior CMSC majors; 3. senior CMSC minors; 4. junior CMSC minors; 5. Scientific Computing concentrators; senior LING majors; 7. junior LING majors; 8. other seniors; 9. other juniors; 10. sophomores; 11. everyone else

(Offered: Fall 2024)

CMSC H394 ADVANCED TOPICS IN THEORETICAL COMPUTER SCIENCE: MATH FOUNDATIONS OF MACHINE LEARNING (1.0 Credit)

Division: Natural Science

Domain(s): C: Physical and Natural Processes A 300-level course on the mathematical foundations of computer science, with the particular topic(s) varying each time it is offered. Fall 2023: An introduction to the mathematical principles behind modern machine learning algorithms. Covers advanced topics in linear algebra, vector calculus, probability theory, and optimization, with a particular focus on their relevance to machine learning tasks. We will also discuss various practical applications. Crosslisted: Mathematics, Computer Science. Prerequisite(s): MATH 121 and 215, or instructor permission

CMSC H395 ADVANCED TOPICS IN COMPUTER SYSTEMS: TYPE-THEORETIC FOUNDATIONS FOR RELIABLE SOFTWARE DESIGN (1.0 Credit) Staff

Division: Natural Science

Domain(s): C: Physical and Natural Processes Software reliability is of paramount importance in critical applications, like smart grids, online banking and automated life-support systems. Errors in software that supports such applications can lead to damaging consequences, like loss of life and property. Type systems provide an effective way to reduce such errors, thereby making the software more reliable. In this course, we shall dive into the foundations of type systems and see how they help us design reliable software. In particular, we shall study a foundational type system called the Simply-Typed Lambda-Calculus and discuss how it can be extended to design software that can be used reliably in critical applications. Prerequisite(s): CMSC 106/151/107 and CMSC 231, or permission of instructor

CMSC H396 ADVANCED TOPICS IN MACHINE LEARNING: DEEP LEARNING FOR COMPUTER VISION (1.0 Credit)

Division: Natural Science

Domain(s): C: Physical and Natural Processes Content varies by semester; course may sometimes have a specific subtitle, but normally focuses on machine learning itself or on related content of importance to students who have completed the regular machine learning courses. Pre-requisite(s): CMSC H260 and either CMSC H360 or CMSC H325 Lottery Preference: Senior CMSC majors; other CMSC majors; others

CMSC H399 SENIOR THESIS (1.0 Credit) Steven Lindell

Division: Natural Science Fall seminar required for seniors writing theses, dealing with the oral and written exposition of advanced material. Lottery Preference(s): Senior standing (**Offered:** Fall 2024)

CMSC H480 INDEPENDENT STUDY (1.0 Credit) Division: Natural Science

Independent study, supervised by a member of the Computer Science department. Prerequisite(s): Instructor consent (**Offered:** Fall 2024)