COMPUTER SCIENCE

Chair, Associate Professor Stephanie Taylor, Associate Chair, Assistant Professor Ying Li
Professors Dale Skrien and Bruce Maxwell; Associate Professor Stephanie Taylor; Assistant Professors Eric Aaron, Naser Al Madi, Stacy Doore, Oliver Layton, Ying Li, and Hannah Wolfe

Computer science studies the design of computational processes, computing systems, and virtual objects. Our goal is to provide students with a strong background in computer science, including the integration of knowledge from other disciplines. Our graduates have the ability and experience to enable and to produce innovative discoveries.

Students with a variety of interests may want to explore computer science, as it affects and interacts with virtually every discipline. Many advances in the natural and social sciences, engineering, and the humanities would not have been possible without the exponential growth in computing power and the corresponding design of advanced algorithms by computer scientists. Students who become majors or minors, or take just a few courses, will extend their potential by knowing more about how to effectively use computers and computation.

Students in computer science courses learn primarily through programming projects that provide them with experience in design, the application of computational thinking, and problem solving. Computational thinking is the ability to deconstruct a problem or process and describe it at the level of computable operations. Computational thinking integrates abstraction, hierarchical design, information management, and an understanding of complexity. The projects students undertake increase in scope and complexity both within a single course and as students progress through the major.

The computer science major prepares students for graduate work in computer science and related areas and for a wide variety of careers. The computer science minor provides students with the ability to effectively apply computational thinking to other disciplines. The interdisciplinary CS+X majors in biology, environmental studies, psychology, music, or theater and dance give students the opportunity to integrate computer science with a focus discipline. Students interested in any of these programs should enroll in one of the Computer Science 15X courses in their first year.

Requirements for the Major in Computer Science

Computer Science 15X, 231, 232, 251 or 252, 333, and 375, 376, or 378; one elective numbered 200 or above; two electives numbered 300 or above; one elective numbered 400 or above; and one 200-level mathematics or statistics course. Students may count only Computer Science 15X, 231, and 251/2 toward both the computer science major or minor and any interdisciplinary computation major.

Requirements for the Honors Program in Computer Science

An honors program is available for students who wish to pursue a topic more deeply. Students must have a grade point average of at least 3.6 in all computer science courses numbered 200 or higher and complete a yearlong, preapproved honors project (Computer Science 483 and 484) of at least seven credits, culminating in both a written paper and a colloquium presentation. Four credits of the honors project satisfy an elective in the major requirements. Students who successfully complete the requirements and receive the recommendation of the department will graduate with "Honors in Computer Science."

Requirements for the Minor in Computer Science

One Computer Science course numbered 150 or above, 231, 251 or 252, one course numbered 200 or above, one course numbered 300 or above, and a capstone experience. The capstone experience can be either (a) a course numbered 400 or above, (b) a four- (or more) credit independent study with a significant computing component in the student’s major department, or (c) two 300-level courses. Option (b) must be preapproved by a computer science advisor.

The point scale for retention of the major/minor applies to all courses in the major/minor. No requirement for the major/minor may be taken satisfactory/unsatisfactory.

A minor in data science is described in the “Data Science” section of the catalogue.

Requirements for the Interdisciplinary CX+X Majors

Listed under “Biology” (as Computational Biology), “Environmental Studies (as Environmental Computation),” “Psychology” (as Computational Psychology), “Music,” and “Theater and Dance.”

Course Offerings

CS125f  Introduction to R  An introduction to the programming language R and how it can be used for statistical analysis and visualization of data. Students will learn how to write basic R programs that can read, write, and manipulate data. They will make use of R functions for executing common statistical analysis and learn how to display the results using graphs and charts. Through a series of projects, students will get experience with writing their own functions, learn how to make use of R documentation and how to extend their
own knowledge of the language.  

CS151fs  Computational Thinking: Visual Media  An introduction to computational thinking: how we can describe and solve problems using a computer. Using the Python language, students will learn how to write algorithms, manipulate information, and design programs. They will learn about abstraction, how to divide and organize a process into appropriate components, how to describe processes in a computer language, and how to analyze and understand the behavior of their programs. The projects will focus on manipulating images or generating complex and interesting scenes and animations each week through writing well-constructed programs. Students may not receive credit for both this course and Computer Science 152 or 153.  

CS152  Computational Thinking: Science  An introduction to computational thinking: how we can describe and solve problems using a computer. Using the Python language, students will learn how to write algorithms, manipulate information, and design programs. They will learn about abstraction, how to divide and organize a process into appropriate components, how to describe processes in a computer language, and how to analyze and understand the behavior of their programs. The projects focus on: (1) reading, writing, managing, and analyzing data; (2) modeling ecological systems using population and agent-based approaches; and (3) simulating physical systems with gravity and collisions. Students may not receive credit for both this course and Computer Science 151 or 153.  

CS153  Computational Thinking: Smart Systems  An introduction to computational thinking: how we can describe and solve problems using a computer. Students will learn how to write algorithms, manipulate information, and design programs. They will learn about abstraction, how to divide and organize a process into appropriate components, how to describe processes in a computer language, and how to analyze and understand the behavior of their programs. The projects will make use of Arduino embedded devices connected to sensors and motors to create smart systems that can sense and react to their environment. Students may not receive credit for both this course and Computer Science 151 or 152.  

CS231fs  Data Structures and Algorithms  Focuses on the common structures used to store data and the standard algorithms for manipulating them. Standard data structures include lists, stacks, queues, trees, heaps, hash tables, and graphs. Standard algorithms include searching, sorting, and traversals. Along with implementation details, students will learn to analyze the time and space efficiency of algorithms and how to select appropriate data structures and algorithms for a specific application. In homework, labs, and programming projects, students will implement their own data structures and make use of existing libraries to solve a variety of computational problems.  

CS232f  Computer Organization  Computer organization focuses on how computers work. Students learn the fundamental hardware components, including storage (RAM, hard disks), input/output mechanisms, and the central processing unit (CPU). They learn how components are designed and built on several levels, including the design of electrical components, machine language, and assembly language. They also learn to program in assembly language for one or more simple processors. Students learn primarily through projects where they design digital circuits, design components of a CPU, or write programs in assembly language.  

CS251s  Data Analysis and Visualization  Prepares students to apply computational data analysis and visualization approaches to real information from a variety of disciplines and applications. Data visualization is the interactive visual exploration of 2-D and 3-D graphic information using techniques that highlight patterns and relationships. Data analysis incorporates data management, data transformations, statistical analysis, data mining, and machine learning. Through programming projects, students will gain hands-on experience with the fundamentals of data analysis and visualization using data from active research projects at Colby and other institutions. Students may not receive credit for both this course and Computer Science 252.  

CS252s  Mathematical Data Analysis and Visualization  An introduction to the analysis and visualization of scientific data. Topics include data management, basic statistical analysis, data mining techniques, and fundamental concepts of machine learning. Emphasis is placed on the mathematical basis of algorithms, which are then applied to real datasets. As time allows, additional techniques involving linear algebra and calculus are covered. Students also learn how to visualize different types of data, focusing on dis-covering patterns and relationships. Through programming projects, students gain hands-on experience analyzing and selecting appropriate visualizations for real datasets. Students may not receive credit for both this course and Computer Science 251.  

CS267  Interactive Digital Media  Digital forms of text, sound, images, and video enable rapid communication and manipulation of large amounts of information. Digital sensors provide easy access to information about the environment. Connecting sensors with digital media enables the creation of artistic installations with dynamic narratives that respond to a user's actions. Students will learn to manipulate digital media and collect sensor data using both applications and their own computer programs. By combining the two, they will create their own artistic installation and demonstrate it at the end of the term.  

Generated February 12, 2021, on colby.edu  

Colby College 2020-2021 Catalogue
A.

[CS269] **Computer Game Design** Focuses on the process of taking stories, situations, puzzles, or challenges and defining them so a computer can execute the construct within which the game resides. Students will design a 2-D computer game using a standard game engine. Topics include game design, artistic concepts, image manipulation, game scripting, and artificial intelligence. Students will work in groups to design and develop a 2-D game to be distributed at the end of the term. Each group will make weekly presentations to the class, demonstrating their progress in game design. *Prerequisite:* Computer Science 151, 152, 153, or 231.  
**Three credit hours.**

CS297 **Data Driven Decision-Making** Develops the students' foundational competencies related to data management that allows them to perform effectively in a broad range of knowledge professions. All students need the ability to critically analyze complex problems using a variety of data sources correctly and effectively present their ideas to others. We will be using data searching techniques, basic statistics, SQL queries and Tableau visualizations to develop a cohesive and persuasive story through the use of data. *Prerequisite:* Computer Science 151, 152, or 153.  
**Three credit hours.**

CS298s **Creating Future Worlds: Computing Ethics and Society** Explores ethical and moral narratives associated with advances in computing and technology. Students will investigate issues in computing ethics highlighting the responsibility of computing professionals to examine the impact of their work on individuals and society. Topics will include: algorithmic bias, digital disparities, autonomous systems, intellectual property law, cybercrime, information privacy and security, tech corporate culture, and professional conduct in a diverse workplace. Students will critically evaluate and apply literature on responsible technology development and professional codes of ethics with evolving technologies. *Prerequisite:* Computer Science 151 or 231 and sophomore or higher standing.  
**Four credit hours.**

[CS325] **Web Programming** The art and science of building dynamic (interactive) websites. Students will learn the fundamentals of the Internet and its HTTP/TCP/IP protocols, HTML and CSS, and how to use them to create well-designed web pages that follow industry standards. They will learn to program in JavaScript to create client-side dynamic web pages, in PHP or another language to create server-side dynamic web pages, and in SQL to create, access, and modify a relational database. Finally, they will learn about XML, DOM, and AJAX, and how to use them to add Web 2.0 features to web pages. *Prerequisite:* Computer Science 231.  
**Three credit hours.**

CS331s **Computer Networks** An introduction to fundamental concepts of computer networks and widely used networking technologies. Topics include application protocol design; principles of congestion and error control protocols; network routing; local, wireless, and access networks; network security; and networking programming. The Internet suite of protocols will be discussed in depth. Students will learn about conventional and state-of-the-art computer networks through problem sets and programming projects. *Prerequisite:* Computer Science 231 and 232.  
**Four credit hours.**

CS333fs **Programming Languages** A survey of programming languages and paradigms focusing on the design of programming languages and comparing and contrasting different language families, including imperative, object-oriented, functional, and logic paradigms. Topics include syntax, context-free grammars, parsing, semantics, abstract representations of programming processes and structures, memory management, and exceptions. Students will undertake small programming projects in various languages and more extensive projects in two languages of their choice, presenting the characteristics of their chosen languages to their peers at the end of the term. *Prerequisite:* Computer Science 231.  
**Four credit hours.**

[CS336] **Parallel and Distributed Processing** An introduction to the principles and applications of parallel and distributed computing, with an emphasis on parallel computing. Within the context of (1) multi-threaded programming with POSIX threads and (2) MPI programming, we study dead-lock avoidance, load-balancing with appropriate data distribution schemes, basic parallel abstractions such as scan and reduce, parallel sorting algorithms, and performance analysis. Students will learn through a series of programming projects and problem sets. The final project is a significant report analyzing the performance of three sorting algorithms. *Prerequisite:* Computer Science 231 and 232.  
**Four credit hours.**

CS341 **Systems Biology I** An introduction to the field of molecular systems biology, which aims to understand the mechanisms underlying complex biological processes. Key to this endeavor is the process of formulating and analyzing mathematical models. Students will learn how to develop, simulate, and analyze ordinary differential equation models of biological systems as well as to read and understand relevant journal articles and perform in-depth analysis of model dynamics. *Prerequisite:* Computer Science 231, and Mathematics 122 or equivalent, and one of the following: Biology 163 or 164; Statistics 212, Mathematics 253, or a 300-level course; or any 300-level computer science course.  
**Four credit hours.**

[CS342] **Embedded Systems** Introduction to the design and use of embedded systems. Students will engage in hands-on exploration of the real-time control of microcontroller platforms and sensorimotor devices by designing and implementing their own electromechanical prototypes. Students will use C++ and basic circuit theory to interface with off-board sensors and actuators, giving their prototypes the ability to sense and respond to the physical world. Prototyping projects will be accompanied by student-generated technical documentation that supports the end user and ensures reproducibility. Students will become true makers and masters of any device with a data sheet.
CS346s  Computational Modeling Simulation I: Interdisciplinary Science  A programming-oriented introduction to techniques in computational modeling and simulation, motivated by applications to the natural and social sciences. Topics may include: dynamical system simulation; finite difference equations; numerical error in simulation; numerical methods for integration; Monte Carlo simulation; cellular automata; and agent-based modeling. Students complete projects in multiple application domains to develop interdisciplinary breadth; to understand explanatory models and methods underlying computational science; and to develop programming style and skills that support easily extended and maintained code. Prerequisite: Computer Science 251 and Mathematics 121 or 161; 122 or 162 are recommended.  Four credit hours.  LAYTON

CS334f  Neural Networks  An introduction to neural networks from biological and machine learning perspectives. Focuses on neural networks for classification and regression involving large image and text datasets. Topics include fundamental design principles; supervised and unsupervised learning; fully connected and convolutional networks; transfer learning. Students obtain hands-on experience implementing and analyzing the neural networks covered each week in regular projects that explore different application areas. Prerequisite: Computer Science 251 and Mathematics 121 or 161; 122 or 162 are recommended.  Four credit hours.  LAYTON

CS351  Computer Graphics  An introduction to computer graphics covering 2-D graphic primitives, clipping graphic objects to boundaries, linear transformations, creating and representing 3-D objects, converting 3-D models into 2-D images, and rendering complex 3-D scenes made of thousands of polygons. Students will build a comprehensive 3-D rendering engine in sequential weekly projects for which they generate images and develop portfolios of their own work. Prerequisite: Computer Science 251.  Four credit hours.  WOLFE

CS353f  Interactive Systems I: Interactive Agents, Environments, and Systems  A hands-on introduction on designing and building interactive systems. Topics include physical computing, multimodal interfaces, rapid prototyping, multimedia systems, design methodologies, and more. The course incorporates examples of early and contemporary work in human computer interaction and new media art. Prerequisite: Computer Science 251 and 333.  Four credit hours.  WOLFE

CS361  Object-Oriented Design  Object-oriented design focuses on the art and science of designing programs so that they are reusable, readable, maintainable, extensible, and robust. Students will learn object-oriented design techniques for producing such software, focusing on learning good programming style, object-oriented design principles, and design patterns. Students will also examine case studies of moderately large programs, will learn to use tools such as CRC cards and the UML, and will undertake significant programming projects. Prerequisite: Computer Science 231.  Four credit hours.

CS365  Computer Vision  Investigates designing computer programs that extract information from digital images. Major topics include image formation and acquisition, gray-scale and color image processing, image filters, feature detection, texture, object segmentation, classification, recognition, and motion estimation. Students are introduced to classic and contemporary vision techniques with examples for homework and programming assignments drawn from biological and medical imaging, robotics, augmented reality, and digital photography. They will develop a medium-scale vision system using data from active research projects at Colby. Prerequisite: Computer Science 231 and 251 (may be taken concurrently).  Four credit hours.

CS375s  Analysis of Algorithms  Focuses on classical algorithms in computer science and the analysis of the space and time efficiency of such algorithms as those that sort arrays and lists and search various data structures, including lists, trees, graphs, and strings. All major categories of algorithms are discussed, including iteration, divide and conquer, brute force, exhaustive search, greedy, dynamic programming, and approximation. Unsolvable and intractable problems are also covered, as is the role of NP-completeness. If time permits, some parallel and distributed algorithms will be discussed. Students will learn through problem sets and short programming projects. Prerequisite: Computer Science 231 and one 200-level mathematics or statistics course.  Four credit hours.  AARON

CS376f  Algorithm Design and Analysis  Focuses on rigorous algorithm design and analysis techniques, including asymptotic complexity bounds, NP-Completeness, and correctness proofs for iterative and recursive algorithms. Motivating examples include classic sorting, searching, and optimization algorithms, for problem solving with data structures including arrays, lists, trees, and graphs. Emphasis is placed on concepts underlying general techniques, with applications to the design and analysis of correct and efficient algorithms in the iterative, divide-and-conquer, dynamic programming, and greedy paradigms. Students may not receive credit for both this course and Computer Science 375. Prerequisite: Computer Science 231 and Mathematics 274 (or equivalent).  Four credit hours.  AARON

CS378s  Introduction to the Theory of Computation  Focuses on formal languages, automata, computability, complexity classes, and undecidability. Languages discussed include regular languages, context-free languages, and recursively enumerable languages. Both deterministic and non-deterministic forms of the corresponding machines (finite automata, push-down automata, and Turing machines) are
also discussed. Unsolvable and intractable problems are addressed, as is the role of NP-completeness. Students will learn through problem sets, possibly augmented by short programming projects. **Prerequisite:** Computer Science 231 and either Mathematics 274 or 275.  **Four credit hours.**  AARON

[CS431] Advanced Computer Networks Application of the principles of computer networks learned in CS331 to computing-relevant projects and the study of system performance. Students will work in groups to complete a research project that has some measurable and assessable networking features with the option of using mobile test-beds. Students will learn through project design and implementation, experiment development and data analysis, weekly oral presentations and written summaries, a public presentation, and an IEEE-format final report. **Prerequisite:** Computer Science 331.  **Four credit hours.**

CS441s Systems Biology II The application of principles learned in Systems Biology I to a particular biological system. Students will work in interdisciplinary teams to complete a project focusing on one biological system and one or more mathematical models of this system. Involves reading journal articles, designing and running numerical experiments, analyzing results, and presenting challenges and results. Culminates in both a poster presentation and a comprehensive journal article-styled report and oral presentation. **Prerequisite:** Computer Science 341.  **Four credit hours.**  N. TAYLOR

[CS442] Computational Physiology: Bioinspiration in Design and Control Application of principles learned in Computer Science 342 to biologically inspired design and control, and the use of electromechanical prototypes as testbeds for hypotheses in the biological sciences. Students will work in teams to complete a research project that tests a hypothesis from the physiology of locomotion or bioinspired robotics using an electromechanical prototype of their own design. Involves writing weekly summaries and critiques of published journal articles, designing and running physics-based simulations and electromechanical experiments, analyzing results, and presenting original research. Culminates in a poster presentation, an IEEE-format conference paper, and an oral presentation. **Prerequisite:** Computer Science 342, Mathematics 122 or equivalent, and Physics 141 or equivalent.  **Four credit hours.**

[CS443] Computational Neuroscience Focuses on the simulation of human perception and behavior by implementing neural networks that emulate different systems of the brain. Neural systems relating to autonomous navigation — how humans perceive and plan movements through their environments — are a core theme of the course, but a range of human capabilities is considered, such as learning, memory, and vision. Projects provide hands-on experience applying brain inspired algorithms to problems in machine learning, computer vision, and artificial intelligence, such as motion estimation from video, supervised and unsupervised learning of datasets, and agent navigation toward goals in environments with obstacles. Topics include spiking and rate based networks, lateral inhibition, recurrent networks, content addressable memory, plasticity, and neural competition. **Prerequisite:** Computer Science 343.  **Four credit hours.**

CS446f Computational Modeling Simulation II: Interdisciplinary Science A research-oriented continuation of Computational Modeling and Simulation I. Students work in teams to complete a semester-long project in an interdisciplinary area, applying computational modeling and simulation techniques to address a research question. This involves all of the following: reading peer-reviewed articles; understanding the context of a research question in a research community; designing and implementing experiments and analyzing their results. Each team will produce a journal-style article and an oral presentation describing their project. **Prerequisite:** Computer Science 346.  **Four credit hours.**  AARON

[CS451] Advanced Computer Graphics Focuses on advanced algorithms for rendering both hyper-photorealistic and non-photorealistic images of objects and scenes. Topics will include ray tracing, radiosity and other global illumination methods, animation, motion capture and mapping, modeling unique materials, modeling painting and drawing techniques, and other topics selected by students. Each week students will prepare written summaries and critiques of technical papers in computer graphics. Programming projects will include OpenGL, a photorealistic project, a non-photorealistic project, and a final project of the student's choice. **Prerequisite:** Computer Science 346.  **Four credit hours.**

CS453s Interactive Systems II: Special Topics in New Media and Interactive Systems Students in groups will work on a semester-long project in which they design, build and run a human computer interaction study or interactive media arts installation. Students will learn how to conduct a literature review, formulate a question, collect data, evaluate the results and present findings. The course would culminate in a presentation and a report in the style of a conference paper or art installation submission. **Prerequisite:** Computer Science 353.  **Four credit hours.**  WOLFE

[CS461] Object-Oriented Software Systems Students will learn how to design and implement a significant software project that is robust, maintainable, extensible, and modular, building on their experience in Computer Science 361. The focus will vary from year to year. For example, students may gather specifications and then analyze, design, and implement a business application or dynamic website following standard software engineering practices. Students will learn through creating, implementing, and refining their own software
designs in an iterative design process. Prerequisite: Computer Science 232 (may be taken concurrently) and 361. Four credit hours.

[CS481f, 482s] Minor Capstone Independent project and capstone experience for minors. Taken in tandem with a course in the student's major to develop a computing project in consultation with his or her computer science advisor that relates to or extends a topic from the related course. Two to four credit hours.

FACULTY


CS491f, 492s Independent Study Independent study in an area of computer science of particular interest to the student. Prerequisite: Permission of the instructor. One to four credit hours. FACULTY