Computer Science

Chair: Janet Davis
William Bares
Cary Gray
John Stratton

Students of computer science gain insight into a technology on which we increasingly rely, while learning new ways of thinking and tools to solve problems in many domains. Central to computer science is the concept of an algorithm—a precise, repeatable procedure for solving a well-defined problem. Computer scientists discover, define and characterize computational problems; they design, implement, and evaluate algorithmic solutions. Studying computer science in the context of a liberal arts education enables graduates to approach problems from multiple perspectives and communicate effectively with diverse colleagues and stakeholders.

Computer Science 167 is suitable for both potential majors and non-majors who have no prior computer science experience. Students with prior experience should discuss their placement with a computer science faculty member.

Learning Goals: Upon graduation, a student majoring in Computer Science will be able to:

- Understand and apply fundamental algorithms and data structures;
- Understand the abstractions supporting modern software systems, and how the construction of those mechanisms affects the supported systems;
- Apply mathematical techniques to justify computational solutions and explore the limitations of computers;
- Communicate computational ideas through speech, writing, diagrams, and programs;
- Work with a team to design and implement a substantial, integrative project;
- Propose and compare multiple solutions to computational challenges, with consideration for the context and impact of each solution on the creators, maintainers, and users of that solution.

Distribution: Some courses completed in Computer Science apply to the quantitative analysis distribution area. See General Studies Program for lists of courses that apply.

Total credit requirements for a Computer Science major: A student who enters Whitman College with no prior experience in computer science will need to complete 33 credits.

The Computer Science major:

- 29 credits (36 credits with neither prior credit for Mathematics 125 nor placement above Computer Science 167)
- Required Courses
  - 310, 320, 327, 370, 495, and 496
  - 12 additional credits of 200 level or higher
    - A student will typically take Computer Science 167, 210, 220, and 270 as prerequisites to the explicitly required courses and three additional elective credits at the 200 level or higher.
- Other notes
  - Mathematics 125 (3 credits) is a prerequisite to Computer Science 220.
    - If no AP credit in Mathematics and Statistics
  - Students with a 4 or 5 on the AP computer science (A) test are considered to have completed the equivalent of Computer Science 167 and receive four credits in computer science.
  - No more than 10 credits earned in domestic or foreign study programs, transfer credits, and/or AP or IB credits may be used to satisfy the course and credit requirements for the major.
    - If considering graduate study, CS majors are encouraged to take additional courses in Mathematics and Statistics, particularly Mathematics 240 (Linear Algebra) or 247 (Statistics with Applications)

    - No PDF after declaration
- Senior Requirements
    - 495 and 496
    - One-hour oral examination
        - Topics from Data Structures and the 300-level Computer Science core
        - The capstone project will be considered as context for some questions
    - Satisfactory performance on the written Major Field Test

- Honors
    - Students do not apply for admission to candidacy for honors
    - Students must submit a proposal for their thesis or project
        - Must be submitted within the first six weeks of the two-semester period in which student is eligible
    - Accumulated at least 87 credits
    - Completed two semesters of residency at Whitman.
    - Cumulative GPA of at least 3.300 on all credits earned at Whitman College
    - Major GPA of at least 3.500
    - Complete a written thesis or research project prepared exclusively for the satisfaction of this program
    - Earn a grade of at least A- on the honors thesis or project and the honors thesis course
    - Pass the senior assessment with distinction
    - Chair of the department will notify the Registrar of students attaining Honors no later than the beginning of week 12 of the semester.
    - An acceptable digital copy of the Honors Thesis must be submitted to Penrose Library no later than Reading Day

The Computer Science minor:

- 15 Credits in courses numbered 200 and above
- Other notes
    - No independent studies may be used
    - No PDF after declaration

The Data Science minor: The requirements are fully described in the *Mathematic and Statistics* listing of the catalog.

The Geology-Computer Science combined major: The requirements are fully described in the *Geology* listing of the catalog.

100-104 Special Topics in Introductory Computer Science
1-4 credits

A course which examines special topics in computer science at the introductory level. *Prerequisite:* Computer Science 167. Any current offerings follow.

100 ST: Web Design
Spring **Bares** **3 credits**

Students will learn how to analyze, design, and build responsive Web pages by applying principles of accessibility, visual perception, graphical design, and interactive prototyping. We will apply the basics of writing Web pages in HTML, CSS, and JavaScript. Course topics and goals include the following: Understand that Web sites are socio-technical systems that enable far-reaching communication via the Web with potential societal impacts such as equity or trustworthiness of information. We will also learn how to

evaluate a given Web site and to improve its usability for people having different abilities, and to apply visual prototyping methods to design the appearance and navigation between forms and pages. No previous programming or visual design experience is required. Distribution area: none.

167 Introduction to Computational Problem Solving

Fall, Spring **J. Davis, Gray** **4 credits**

Students will learn to design, document, implement, test, and debug algorithmic solutions to computational problems in a high-level, object-oriented programming language. We introduce core concepts: algorithms, data structures, and abstraction. We apply foundational constructs common to all programming languages: data types, variables, conditional execution, iteration, and subroutines. Students will gain experience with exploratory and structured approaches to problem solving through collaborative in-class exercises. Frequent programming projects will address applications of computing to problems arising from other disciplines.

200-204 Special Topics in Introductory Computer Science

1-4 credits

A course which examines special topics in computer science at the introductory level. *Prerequisite:* Computer Science 167. Any current offerings follow.

210 Computer Systems Fundamentals

Spring **Gray** **3 credits**

This course integrates key ideas from digital logic, computer architecture, compilers, and operating systems, in one unified framework. This will be done constructively, by building a general-purpose computer system from ground up: from the low-level details of switching circuits to the high level abstractions of modern programming languages. In the process, we will explore software engineering and algorithmic techniques used in the design of modern hardware and software systems. We will discuss fundamental trade-offs and future trends. *Prerequisite:* Computer Science 167.

215 Introduction to Data Science

Fall **Ptukhina** **4 credits**

An introduction to the approaches and tools of exploratory data analysis and visualization. Through a series of projects, we explore large data sets through methods like cleaning, filtering, sorting, boolean selections and merging. As large amounts of data typically are stored in lists, we use algorithmic thinking to transform raw data into usable form. We develop hypotheses and supporting visualizations to tell the story of the data. We learn and practice technical communication in both oral and written form. Through a series of readings and discussions, we learn best practices for the ethical use of data and how to identify problematic uses of data in society. May be elected as Mathematics 215. *Prerequisites*: Mathematics 125 and Computer Science 167.

220 Discrete Mathematics & Functional Programming

Fall **J. Davis** **3 credits**

Students will practice formal reasoning over discrete structures through two parallel modes: mathematical proofs and computer programs. We will introduce sets and lists, Boolean logic, and proof techniques. We will explore recursive algorithms and data types alongside mathematical and structural induction. We consider relations and functions as mathematical objects built on set theory and develop idioms of higher-order programming. May be elected as Mathematics 220. *Prerequisites:* Computer Science 167 and Mathematics 125.

255 Computer Simulation Methods

Spring **Stratton** **3 credits**

From the earliest days of electronic computers, some of the most pressing applications involved not just organization and processing of data, but digital replication of scenarios in the physical world to improve our understanding or make decisions. Computer simulation allows us to conduct virtual experiments that would be impossible, expensive, or unethical to conduct in the physical world, but answer critical questions that could arise from almost any academic discipline. This course will examine a range of simulation methods, such as N-body, finite difference,

discrete event, and actor-based modeling. Students who have already completed Computer Science 270 are strongly encouraged to concurrently enroll in Computer Science 355. *Prerequisite*: Computer Science 167.

267 Human-Computer Interaction

Not offered 2021-22 **4 credits**

How do people interact with computers? And how can we design computer systems that make people's lives better? Students will learn to critique user interfaces using principles based on psychological theories of perception, memory, attention, planning, and learning. Through a semester-long team project, students will practice iterative design including stages of contextual inquiry, task analysis, ideation, prototyping, and evaluation. We will also explore current research on new application areas, design techniques, or interaction paradigms, as well as social implications of computing.

270 Data Structures

Fall, Spring **Bares** **4 credits**

This course addresses the representation, storage, access, and manipulation of data. We discuss appropriate choices of data structures for diverse problem contexts. We consider abstract data types such as stacks, queues, maps, and graphs, as well as implementations using files, arrays, linked lists, tree structures, heaps, and hash tables. We analyze and implement methods of updating, sorting, and searching for data in these structures. We develop object-oriented programming concepts such as inheritance, polymorphism, and encapsulation. We consider implementation issues including dynamic memory management, as well as tools for programming in the large. *Prerequisite:* Computer Science 167.

300-304 Special Topics in Computer Science

1-4 credits

A course which examines special topics in computer science at the intermediate level. Any current offerings follow.

300 ST: Intelligent User Interfaces

Fall **Bares** **3 credits**

Intelligent User Interfaces apply practices of computer-human interaction and artificial intelligence to produce systems that can adapt their performance to past human interactions. The course will apply intelligent techniques to a variety of modalities including animation, sound, tangible, and immersive experiences. We will explore novel and emerging forms of input technologies including eye- and motion tracking. Topics in intelligent systems will include user modeling, natural language processing, intelligent agents, crowdsourcing, and machine learning. We will explore select topics from recent research publications in the field. Course activities will include lecture, hands-on tutorials, prototyping, designing and running evaluations of intelligent interface systems, coding, exams, and a team project. *Prerequisite*: Computer Science 270. Distribution area: none.

301 ST: Databases with Web Interfaces

Fall **J. Davis** **3 credits**

Study and implementation of the three-tier architecture commonly used for Web-based applications such as social media and e-commerce sites. Through in-class exercises, small programming assignments, and a semester-long team project, students will learn to design and create relational databases, write and analyze SQL queries, create a forms-based Web interface with HTML and CSS, implement a session-based Web server application to connect the interface with the database, and use AJAX for responsive interactions. Along the way, we will address common Web application vulnerabilities, consider alternative back-end technologies, and reflect on ethical and social problems arising from database applications. *Prerequisite*: Computer Science 270. Distribution area: none.

310 Computer Systems Programming

Fall **Gray** **4 credits**

How does data move from a hard drive to memory to a CPU? How does a computer deal with input from a mouse and keyboard? How does one computer communicate with another, or many others? This class examines how operating systems interact with computer hardware to provide higher-level programming abstractions. Students will use the C programming language to explore topics such as processes, virtual memory, concurrency, threads, and networking. *Prerequisites:* Computer Science 210 and 270.

317 Software Performance Optimization

Not offered 2021-22 **3 credits**

Computers do not execute programs with equal speed, even when theoretical analyses indicate that two programs perform approximately the same amount of work. At the same time, software power efficiency affects the size of mobile devices and the energy consumption of datacenters. This course examines current trends in computer system architecture and draws out insights for developing software that is fast and energy-efficient. Students will work problem sets, write programs, conduct experiments, read and analyze technical articles, and carry out a team project of their choice. Throughout the course, we shall consider how computer system designs affect program structure, and in particular the tensions between efficiency and principled software organization. *Prerequisites:* Computer Science 210 and 270.

320 Theory of Computation

Fall **Stratton** **3 credits**

Which problems can be solved computationally? Which cannot? Why? We can prove that computers can perform certain computations and not others. This course will investigate which ones, and why. Topics will include formal models of computation such as finite state automata, push-down automata, and Turing machines, as well as formal languages such as context-free grammars and regular expressions. May be elected as Mathematics 320. *Prerequisite:* Computer Science/Mathematics 220 or Mathematics 260.

327 Algorithm Design & Analysis

Spring **Stratton** **3 credits**

How can we be confident that an algorithm is correct before we implement it? How can we compare the efficiency of different algorithms? We present rigorous techniques for design and analysis of efficient algorithms. We consider problems such as sorting, searching, graph algorithms, and string processing. Students will learn design techniques such as linear programming, dynamic programming, and the greedy method, as well as asymptotic, worst-case, average-case and amortized runtime analyses. Data structures will be further developed and analyzed. We consider the limits of what can be efficiently computed. May be elected as Mathematics 327. *Prerequisites:* Computer Science 270; Computer Science/Mathematics 220 or Mathematics 260.

339 Operations Research

Spring **Hundley** **3 credits**

Operations research is a scientific approach to determining how best to operate a system, usually under conditions requiring the allocation of scarce resources. This course will consider deterministic models, including those in linear programming (optimization) and related subfields of operations research. May be elected as Mathematics 339. *Prerequisites:* Mathematics 240 and Computer Science 167.

350 Mathematical Modeling and Numerical Methods

Not offered 2021-22 **3 credits**

This course explores the process of building, analyzing and interpreting mathematical descriptions of physical processes. This may include theoretical models using statistics and differential equations, simulation modeling, and empirical modeling (meaning model building from data). The course will involve some computer programming, so previous programming experience is helpful. May be elected as Mathematics 350. *Prerequisites:* Mathematics 240 and 244.

355 Optimizing Simulation Methods

Spring **Stratton** **1 credit**

This course extends Computer Science 255, Computer Simulation Methods, with a focus on algorithms and data structures for improving the efficiency of computer simulations. Techniques may include Barnes-Hut spatial data structures, Next-Reaction methods for discrete event simulations, and distributed computing. *Prerequisite*: Computer Science 270. *Co-requisite*: Computer Science 255.

360 Interactive Computer Graphics

Not offered 2021-22 **3 credits**

An introduction to computer graphics covering 2-D and 3-D rendering pipelines and diverse user interaction techniques for graphics applications. Topics will include coordinate systems, geometric shapes, transformations, projection, color, lighting, shading, data visualization, and animation. Interaction techniques may include Web interfaces, immersive displays, vision systems, spatial interaction, music/sound, gesture, and tangible interfaces. We will apply these topics through a combination of hands-on and written exercises and programming with a current computer graphics library. We will read and analyze a selection of research publications that apply computer graphics in different application domains. We will analyze the societal impacts of select applications. We will implement and present a project that applies computer graphics and interaction techniques in a selected domain. *Prerequisite:* Computer Science 270.

370 Software Design

Spring **J. Davis** **4 credits**

What makes code beautiful? We consider how to design programs that are understandable, maintainable, extensible, and robust. Through examination of moderately large programs, we will study concepts including object-oriented design principles, code quality metrics, and design patterns. Students will learn design techniques such as Class-Responsibility-Collaborator (CRC) cards and the Unified Modeling Language (UML), and gain experience with tools to support large-scale software development such as a version control system and a test framework. Students will apply these concepts, techniques, and tools in a semester-long, team software development project. Students enrolling in Computer Science 370 also will be required to enroll in an associated laboratory course (Computer Science 370L). Weekly laboratory sessions will include time for design critiques, code reviews, and supervised teamwork. *Corequisite:* Computer Science 370L. *Prerequisite:* Computer Science 270.

400-404 Special Topics in Computer Science

1-4 credits

A course which examines special topics in computer science at the advanced level. *Prerequisites:* Computer Science 167 and 270. Any current offerings follow.

467 Numerical Analysis

Not offered 2021-22 **3 credits**

An introduction to numerical approximation of algebraic and analytic processes. Topics include numerical methods of solution of equations, systems of equations and differential equations, and error analysis of approximations. May be elected as Mathematics 467. *Prerequisite:* Computer Science 167. *Pre- or corequisite:* Mathematics 240.

481, 482 Independent Study

Fall, Spring **Bares, J. Davis, Stratton** **1-4 credits**

Directed study or research in selected areas of computer science. A curriculum or project is designed by the student(s) with the advice and consent of an instructor in the department. Inquiry may emerge from prior course work or explore areas not covered in the curriculum. *Prerequisite:* consent of instructor.

495 Capstone Project I

Fall **Bares, Staff** **2 credits**

First semester of a team project integrating skills and concepts from across the computer science curriculum. Students will develop project management and communication skills. In writing and documenting software, students will consider their responsibilities to future users or developers. Open only to senior computer science majors. *Prerequisite:* a 300-level computer science course.

496 Capstone Project II

Fall, Spring **Bares** **1 credit**

Second semester of a team project integrating skills and concepts from across the computer science curriculum. Students will develop project management and communication skills, culminating in a public presentation. In writing and documenting software, students will consider their responsibilities to future users or developers. All course work will be completed by the second Friday in March. *Prerequisite:* Computer Science 495.

497 Advanced Project

Fall, Spring **Bares** **1 credit**

Students will individually design, implement, document, and present an extension of the team capstone project developed in Computer Science 495 and 496. *Prerequisite:* Computer Science 495. *Corequisite:* Computer Science 496.

498 Honors Project

Fall, Spring **Bares** **1 credit**

Students will individually design, implement, document, and present an extension of the team capstone project developed in Computer Science 495 and 496. Required of and limited to senior honors candidates in computer science. *Prerequisite:* Computer Science 495. *Corequisite:* Computer Science 496.