

Computer Science

The major and minor in computer science are administered by the Department of Computer Science: Professor Dobosh (*chair*); Associate Professors Ballesteros, Lerner; Assistant Professor St. John.

Contact Persons

Wendy Queiros, *senior administrative assistant*

Paul Dobosh, *chair*

Computer science is an exciting field with applications across many other disciplines, including biology, chemistry, physics, mathematics and economics. The main role of a computer scientist is that of a problem solver. A degree in the field signifies formal training in computational and analytical approaches to problem solving as well as the skills necessary to develop software to tackle new challenges. These computational approaches can be applied to a wide spectrum of problems, including locomotion by robots, protein folding and flexibility, remotely controlled surgery, video games, graphics arts, and publishing. In truth, it is difficult to think of a scenario in which the tools acquired in computer science do not provide a powerful advantage.

Requirements for the Major

Credits

- A minimum of 40 credits

Courses

- Computer science (36 credits):
 - 101, Problem Solving and Structured Programming
 - 201, Advanced Object-Oriented Programming
 - 211, Data Structures
 - 221, Introduction to Computer Systems
 - 312, Algorithms
 - 322, Operating Systems

- Three additional computer science courses (12 credits) with two of these at the 300-level (8 credits)
- Mathematics (4 credits):
 - 232, Discrete Mathematics

The skills and abstract reasoning of mathematics are especially important in computer science. It is strongly recommended that students take additional mathematics courses (at least through Mathematics 101 and 202). Mathematics 211, Linear Algebra, is very useful for computer graphics. Computer science majors who elect a mathematics or statistics minor may not count Mathematics 232 for credit in both mathematics or statistics and computer science.

Students planning to pursue an advanced degree in computer science should take additional computer science courses and include independent research leading to a thesis in their plans.

Requirements for the Minor

Credits

- A minimum of 20 credits

Courses

- Computer science:
 - 101, Problem Solving and Structured Programming
 - 201, Advanced Object-Oriented Programming
 - 211, Data Structures
 - Two additional computer science courses (8 credits) with one at the 300-level.

Please note that certain 300-level courses also require CS 221. Students interested in a computer science minor should consult with a member of the computer science faculty.

Getting Started

The recommended way to begin a study of computer science is with CS 101. This course

is an introduction to the use of computers as a problem-solving tool. Students with some experience in JAVA may consider beginning with 201. Any member of the computer science faculty can advise students who have questions about their course of study. CS101, 201 are offered both semesters.

Honors

To graduate with honors in computer science, a student must complete a project and write an accompanying thesis. This is often a full year commitment, during which the student works closely with a faculty member to explore a topic in depth by reading research papers, writing programs, and experimenting with ideas. Preliminary research usually begins in the summer following her junior year, with the student submitting and defending a thesis proposal early in the fall of her senior year. Upon department approval of this proposal, she will complete the research during her senior year, writing and defending her thesis in the spring. Some honors students attend conferences and/or coauthor papers with their mentors.

Programming-Intensive Courses

Courses designed to offer students significant software design and programming experience are labeled as "Programming intensive."

Course Offerings

100fs An Introduction to Computer Science

An introduction to basic computer science concepts and issues with a focus on HTML and JavaScript programming. Additional topics will be chosen from: origins of computers, computer architecture, assemblers and compilers, digital logic, the Internet, and artificial intelligence. Laboratory assignments will offer some experience with programming and the use of application software.

Meets Science and Math II-A requirement
The department, P. Dobosh
students should NOT take this course after
Computer Science 101; 4 credits

101fs Problem Solving and Object-Oriented Programming

Computers are used every day for an enormous variety of tasks, from playing games and chatting with friends to transferring billions of dollars, delivering radiation treatments, and controlling the electrical grid. Computer programs are an essential ingredient in allowing for this great diversity of applications. In this course, you will learn to create your own programs, based on core programming concepts and analytical problem solving approaches. You will develop dynamic programs by first using Adobe Flash CS4 and AS3 (ActionScript 3), the technology behind many Web applications. The last portion of the course will teach you Java, a very popular modern programming language. We assume no prior study of computer science. Programming intensive.

Meets Science and Math II-A requirement
A. St.John, A. StJohn
4 credits

105f The Computer Science Revolution Is Here

(First-year seminar) Social software such as Facebook and Skype and mobile devices like smartphones and iPads have revolutionized personal computing in recent years. In this course we will look at issues surrounding this revolution in personal computing, such as privacy and free speech. We will also look at how computing is revolutionizing other disciplines, e.g., medicine, assistive technology for the disabled, and scientific research. Topics will include understanding the potential of Watson to build smarter search engines, the role of computing in drug discovery, and how countries can censor or even turn off the Internet.

Meets Science and Math II-A requirement
B. Lerner
Prereq. fj; 4 credits

106s Introduction to Scientific Computing

The matrix-based programming language Matlab will be used to introduce students to programming and fundamental computing methodologies such as top-down design and objects. Course work will involve vectors, matrices, numerical integration and differentiation, curve-fitting and graphics. The course is programming intensive.

Does not meet a distribution requirement

P. Dobosh

Prereq. Mathematics 101; 2 credits

201fs Advanced Object-Oriented Programming

This course builds on the basic programming concepts learned in Computer Science 101. Emphasis is on developing the skills needed to write more sophisticated programs. This includes strategies to aid in assuring the correctness of programs through the use of assertions and unit testing as well as advanced Java features such as inheritance, polymorphism, and network programming. We will also introduce some widely used data structures such as vectors and linked lists. This course is programming-intensive.

Meets Science and Math II-A requirement

B. Lerner

Prereq. Computer Science 101; 4 credits

211fs Data Structures

Using Java. Solving problems with computers is accomplished by writing programs that operate on data to produce a desired result. The way data is organized and presented to the program can significantly affect its efficiency and simplicity and can sometimes determine whether or not a program can be written to solve the problem at all. This course presents ways of organizing data into “data structures” and analyzes how structuring the data can improve program performance. **This course is programming intensive.**

Meets Science and Math II-A requirement

The department

Prereq. Computer Science 101, 102 or 201; 4 credits

215f Software Design

Building large software systems introduces new challenges to software development. Appropriate design decisions early in the development of large software can make a major difference in developing software that is correct and maintainable. In this course, students will learn techniques and tools to help them address these problems and develop larger software projects, improving their skills in designing, writing, debugging, and testing software. Topics include design patterns, UML, designing for maintainability, software architecture, and designing concur-

rent and fault tolerant systems. Programming intensive.

Meets Science and Math II-A requirement

B. Lerner

Prereq. Computer Science 102 or 201; 4 credits

221s Introduction to Computing Systems

This course looks at the inner workings of a computer and computer systems. It is an introduction to computer architecture. Specific topics include assembly language programming, memory, and I/O issues. **This course is programming intensive.**

Meets Science and Math II-A requirement

The department

Prereq. Computer Science 101, 102 (aka 201); 4 credits

295fs Independent Study

Does not meet a distribution requirement

The department

Prereq. soph, permission of instructor; 1-4 credits

311f Theory of Computation

Are there any limits to what computers can do? Does the answer to this question depend on whether you use a PC or a Mac? Is C more powerful than PASCAL? This seminar explores these questions by investigating several models of computation, illustrating the power and limitations of each of these models, and relating them to computational problems and applications. Topics include finite state automata, pushdown automata, grammars, Turing machines, the Universal Turing Machine, and computability.

Meets Science and Math II-A requirement

A. StJohn

Prereq. Computer Science 101,102(aka 201) and Mathematics 232; 4 credits

312s Algorithms

How does Mapquest find the best route between two locations? How do computers help to decode the human genome? At the heart of these and other complex computer applications are nontrivial algorithms. While algorithms must be specialized to an application, there are some standard ways of approaching algorithmic problems that tend to be useful in many applications. Among other topics, we will explore graph algorithms, greedy algorithms, divide-and-conquer, dynamic pro-

gramming, and network flow. We will learn to recognize when to apply each of these strategies as well as to evaluate the expected runtime costs of the algorithms we design.

Meets Science and Math II-A requirement

B. Lerner

Prereq. Computer Science 211 and Mathematics 232; 4 credits

*316s Software Practicum

Tired of writing programs that nobody ever uses? Then, this is the course for you. Software enables enterprises to carry out previously tedious or impossible tasks, but many organizations lack the resources to develop needed software. You will apply your programming skills to develop and deliver software to meet the requirements of a client from the community. You will learn critical communication skills required to work with a client, work as a team with classmates, and experience the software lifecycle from requirements elicitation through delivery. You will synthesize many topics learned in courses as well as new technologies required to complete the project. Programming intensive.

Meets Science and Math II-A requirement

B. Lerner

Prereq. Computer Science 215; 4 credits

322f Operating Systems

An introduction to the issues involved in orchestrating the use of computer resources. Topics include operating system evolution, file-handling systems, memory management, virtual memory, resource scheduling, multi-programming, deadlocks, concurrent processes, protection, and design principles. Course emphasis: understanding the effects of operating system design on computer system performance. **This course is programming intensive.**

Meets Science and Math II-A requirement

The department

Prereq. Computer Science 221; 4 credits

*331s Computer Graphics

The creation of pictorial images using a computer. Topics include drawing of two- and three-dimensional scenes using OpenGL and other graphical environments; transformations of objects (translations, scalings, rotations, shearings) using homogeneous

coordinates; creating perspective in three-dimensional drawing; algorithms for enhancing realism and visual effect; and the mathematical underpinnings of graphic design. Students will complete a number of graphics projects based on readings and class discussion. *This course is programming intensive.*

Meets Science and Math II-A requirement

The department

Prereq. Computer Science 101, 211 (may be

taken concurrently), and at least one of

Mathematics 203, Mathematics 211,

Mathematics 232, or permission of instructor;

4 credits

*334f Artificial Intelligence

An introduction to AI research issues in fields such as Natural Language Processing and Machine Learning. **This course is programming intensive.**

Meets Science and Math II-A requirement

L. Ballesteros

Prereq. Computer Science 211 and

Mathematics 232; 4 credits

*336s Intelligent Information Retrieval

Introduces the basic concepts, methodologies, and research findings in information retrieval. Special topics include Web searching, cross-language retrieval, data mining, and data extraction. Completion of this course will provide the necessary foundation to work in today's business environment where competitive advantage is obtained by retrieving needed information.

Meets Science and Math II-A requirement

L. Ballesteros

Prereq. Computer Science 211; 4 credits

341s Topics

Spring 2012

*341s(1) Robotics and HCI

This is a seminar-based course, drawing on a combination of traditional lectures, student presentations and projects. We will cover core robotics and HCI (Human Computer Interaction) concepts, with additional topics directed by student interest.

Meets Science and Math II-A requirement

A. St. John

Prereq. Computer Science 211 Data Structures,

Math 232 Discrete - Recommended: CS 312 Algorithms, Math 211 Linear Algebra; 4 credits

341s(2) Computational Geometry in Video Games

Developing video games is an exciting and challenging domain, involving many areas of computer science, such as graphics, artificial intelligence and robotics. In this course, we focus on the geometric problems that arise in video game programming. Due to the expensive computations often demanded by current video game technology, efficient algorithms are required that not only satisfy speed requirements, but result in realistic user experiences. Topics will include standard problems from computational geometry, such as triangulation of 2-D and 3-D objects, point detection and visibility.

*Meets Science and Math II-A requirement
A. St. John*

Prereq. Computer Science 211 Data Structures, Math 232 Discrete - Recommended: CS 312 Algorithms, Math 211 Linear Algebra; 4 credits

395fs Independent Study

*Does not meet a distribution requirement
The department*

Prereq. jr, sr, permission of instructor; 1-8 credits