

# Computer Science

*Departmental Counselor:*

*Sharon Salveter, Ry 150, 834.2773, salveter@cs.uchicago.edu*

*Student Services Representative:*

*Margaret Jaffey, Ry 156, 702.6011, margaret@cs.uchicago.edu*

*Web: [www.cs.uchicago.edu](http://www.cs.uchicago.edu)*

## Program of Study

The computer science program prepares students for either graduate work or employment in computer science by offering both the BA and BS degrees. Students receiving the BA will have sufficient breadth and depth for either graduate study or immediate employment in computer science. Recipients of the BS will also have substantial depth and breadth in a field outside of computer science through the completion of an approved related area.

Students in other fields of study may complete a minor in Computer Science. Information follows the description of the major.

## Program Requirements

Both the BA and BS in Computer Science require fulfillment of the mathematical sciences requirement in general education by completing an approved two-quarter calculus sequence. The physical sciences requirement in general education must be satisfied by completing an approved two-quarter sequence in either chemistry or physics. Both BA and BS students take at least fourteen computer science courses chosen from an approved program. BS students also take three courses in an approved related field outside computer science.

Students pursuing a bachelor's degree in computer science should note that by judicious choice of courses from another field a supplementary field can be developed that is often in itself a solid basis for graduate or professional work in that field. Some examples are biology, biophysics, chemistry, geophysical sciences, history, linguistics, mathematics, philosophy, political science, psychology, physics, sociology, statistics, and economics.

**Advanced Placement.** Students who are majoring in computer science may not use AP credit for computer science to meet requirements in the major. Students with AP scores of 4 or 5 on Computer Science AB receive two quarters of elective credit. NOTE: Students must forgo AP elective credit if they register for CMSC 10500, 10600, 15100, or 15200. Students who enroll in CMSC 12100, 12200, 16100, and 16200 may retain AP elective credit.

Computer science majors may use AP credit for chemistry or physics to meet their physical sciences requirement in general education or physical science components of the major. However, no credit designated simply as "physical

science” (from either AP or the College’s physical sciences examinations) may be used to meet general education or requirements in the computer science majors.

**Approved Programs.** The notion of “approval” in the program requirements allows timely response to change in the course offerings of the various departments. The computer science departmental counselor is responsible for approval of specific courses and sequences. Students should consult the departmental counselor for details on specific courses they are considering taking to meet the requirements.

## Approved Computer Science Program

For the authoritative version of the Department of Computer Science requirements and course descriptions, visit [www.cs.uchicago.edu](http://www.cs.uchicago.edu).

There is a single approved program comprising required courses in four topical areas, plus three elective computer science courses. This is a general program in computer science and is used for either the BA or the BS degree. Upper-level or graduate courses in similar topics may be substituted for those on the list that follows, with the approval of the departmental counselor. *Students considering a computer science major are strongly advised to register for an introductory sequence in their first year.*

1. Introductory Sequence (four courses required):  
 CMSC 15100 or 16100, and  
 CMSC 15200 or 16200, and  
 CMSC 15300, and  
 CMSC 15400
2. Programming Languages and Systems Sequence (two courses required):  
 Two courses from CMSC 22100, 22200, 22610, 23000,  
 23300, 23500, or 23700
3. Algorithms and Theory Sequence (three courses required):  
 CMSC 27100, and  
 CMSC 27200, and  
 CMSC 28000 or CMSC 28100
4. Other Sequences (one two-course sequence required):
  - a. Artificial Intelligence Sequence (two courses required):  
 CMSC 25010 and one of 25020, 25030, 25040, or 25050
  - b. Advanced Systems Sequence (two courses required):  
 CMSC 22100, 22200, 22610, 22620, 23000, 23300, 23500, 23700,  
 or 23800, depending upon what courses the student has taken in the  
 Programming Languages and Systems Sequence  
 (courses may not be used to meet both requirements)
  - c. Scientific Computing Sequence (two courses required):  
 CMSC 28510 and an additional approved course

5. Electives (three courses required):  
 Three additional elective Computer Science courses numbered 20000 or above. A BS student with a double major in a related area may petition to have some or all of the electives be courses in the other major.

### Summary of Requirements

<i>General Education</i>		CHEM 10100-10200 or 11100-11200 or equivalent*, or PHYS 12100-12200 or higher* MATH 13100-13200 or higher*
<i>Major</i>	4	Introductory Sequence (CMSC 15100 or 16100, and 15200 or 16200, and 15300, and 15400)
	2	Programming Languages and Systems Sequence (CMSC 22100, 22200, 22610, 23000, 23300, 23500, or 23700)
	3	Algorithms and Theory Sequence (CMSC 27100, 27200, and 28000 or 28100)
	2	courses from an approved sequence
	3	electives numbered CMSC 20000 or above

*plus the following requirements:*

	BA	BS
$\frac{0}{14}$	no other courses required	3 courses in an approved program in a related field
		$\frac{3}{17}$

\* Credit may be granted by examination.

**Grading.** Computer science majors must take courses in the major for quality grades. A grade of C- or higher must be received in each course in the major. Any 20000-level computer science course taken as an elective beyond requirements for the major may, with consent of instructor, be taken for P/F grading.

Nonmajors may take courses for either quality grades or, subject to College regulations and with consent of instructor, for P/F grading. A Pass grade is given only for work of C- quality or higher. Courses taken to meet general education requirements must be taken for quality grades.

Incompletes are typically given in the Department of Computer Science only to students who have done at least 60 percent of the course's work of a passing quality and who are unable to complete all course work by the end of the quarter. Other restrictions on Incompletes are the province of individual instructors, many of whom do not permit Incompletes. To receive an Incomplete, students must make arrangements in advance with the instructor; a consent form to be signed by the instructor is available from the College adviser.

**Honors.** Students may earn a BA or BS degree with honors by attaining a grade of *B* or higher in all courses in the major and a grade of *B* or higher in three approved graduate computer science courses (30000-level and above). These courses may be courses taken for the major or as electives.

Students may also earn a BA or BS degree with honors by attaining the same minimum *B* grade in all courses in the major and by writing a successful bachelor's thesis as part of CMSC 29900. This thesis must be based on an approved research project that is directed by a faculty member and approved by the departmental counselor.

## Recommended Sequences in Computer Science

**Introductory Sequences.** The kinds of computer science courses appropriate for undergraduates will vary according to each student's interests. Students interested in a *general programming background* are encouraged to take CMSC 15100 and 15200 (Introduction to Computer Science I, II). Students in the humanities (or others with a humanistic background) and social sciences may consider CMSC 11000-11100 (Multimedia Programming as an Interdisciplinary Art I, II). Students with a strong mathematics background should consider the CMSC 16100-16200 (Honors Introduction to Computer Science I, II). Students interested in a two-quarter introduction to the discipline should consider CMSC 10500-10600 (Fundamentals of Computer Programming I, II), or CMSC 12100-12200 (Computer Science with Applications). CMSC 15100 or 16100 is recommended for students interested in further programming study. Students who are interested in *web page design and implementation* should take CMSC 10100-10200.

**Courses in Specific Areas of Computer Science.** Students interested in *artificial intelligence* (AI) should take CMSC 25010 (Artificial Intelligence) followed by any of the elective AI courses numbered 25020-25050.

Students interested in *advanced programming* and systems should take CMSC 22100 (Programming Languages), and CMSC 22200 (Computer Architecture). Time permitting, they should also take CMSC 22610-22620 (Implementation of Computer Languages), CMSC 23000 (Operating Systems), CMSC 23300 (Networks and Distributed Systems), CMSC 23500 (Databases), CMSC 23700 (Graphics), CMSC 23800 (Game Construction), and such courses in advanced programming topics that may be offered.

Students interested in *theoretical computer science* should take CMSC 27100 (Discrete Mathematics), CMSC 27200 (Theory of Algorithms), CMSC 28000 (Formal Languages) and CMSC 28100 (Introduction to Complexity Theory). Once students have completed CMSC 27100, 27200, 28000, and/or 28100, they will be qualified for most of the advanced topics courses offered at the 30000-level and above.

Students interested in *numerical and scientific computing* should take CMSC 28510 (Introduction to Scientific Computing).

The department also offers a number of special-interest courses that are detailed in the course descriptions. For information on new courses that are added on a regular basis, consult the departmental counselor and visit [www.cs.uchicago.edu](http://www.cs.uchicago.edu).

**Preparation for Graduate Study in Computer Science.** Students interested in continuing their studies beyond the undergraduate level should major in computer science and take as many computer science courses as possible. The most important courses are CMSC 15100, 15200, 15300, 15400, 22100, 22200, 22610, 22620, 23000, 23300, 23500, 23700, 25010, 27100, 27200, 28000, and 28100. For more information about options for graduate study, consult the departmental counselor and the director of graduate studies.

### Minor Program in Computer Science

The minor in Computer Science requires seven or eight courses, depending on which introductory computer science courses are taken. The introductory sequence of computer science courses is followed by three approved upper-level courses chosen to complement a student's major or personal interest. Courses in the minor must be taken for quality grades, with a grade of C- or higher in each course. Students may not use AP credit for computer science to meet requirements for the minor.

No courses in the minor can be double counted with the student's major(s) or with other minors; nor can they be counted toward general education requirements. More than half of the requirements for the minor must be met by registering for courses bearing University of Chicago course numbers. By the end of Spring Quarter of their third year, students must submit approval of their minor program from the departmental counselor on a form obtained from their College adviser.

Other programs may be approved in consultation with the departmental counselor.

**Introductory Courses.** Students must choose four to five courses from the following:

*One course:*

- CMSC 10500. Fundamentals of Computer Programming I
- CMSC 12100. Computer Science with Applications I
- CMSC 15100. Introduction to Computer Science I
- CMSC 16100. Honors Introduction to Computer Science I

*One to two courses:*

- CMSC 10600-10700. Fundamentals of Computer Programming II, III
- CMSC 12200. Computer Science with Applications II
- CMSC 15200. Introduction to Computer Science II
- CMSC 16200. Honors Introduction to Computer Science II

CMSC 15300. Foundations of Software

CMSC 15400. Introduction to Computer Systems

NOTE: Students completing the minor program who have registered for CMSC 10500, 10600, and 10700 will have met the prerequisite for CMSC 15300 and 15400.

**Upper-Level Courses.** Three 20000-level or above computer science courses must be approved by the departmental counselor or selected from a pre-approved three-course group below. A 20000-level course must replace each 10000-level course in the list above that was used to meet general education requirements.

*Programming Languages and Systems (choose any three):*

CMSC 22100. Programming Languages

CMSC 22200. Computer Architecture

CMSC 22610. Implementation of Computer Languages I

CMSC 23000. Operating Systems

CMSC 23300. Networks and Distributed Systems

CMSC 23500. Introduction to Database Systems

CMSC 23700. Introduction to Computer Graphics

CMSC 23800. Game Construction

*Algorithms and Theory:*

CMSC 27100. Discrete Mathematics

CMSC 27200. Algorithms

CMSC 28000. Introduction to Formal Languages; *or*

CMSC 28100. Introduction to Complexity Theory

*Scientific Computing:*

CMSC 28510. Introduction to Scientific Computing

Two approved elective courses

*Artificial Intelligence (choose any three):*

CMSC 25010. Artificial Intelligence

CMSC 25020. Computational Linguistics

CMSC 25030. Computational Models of Speech

CMSC 25040. Introduction to Computer Vision

CMSC 25050. Computer Vision

## Faculty

Y. Amit, L. Babai, T. Dupont, P. Felzenszwalb, I. Foster, J. Goldsmith, N. Hinrichs, G. Kindlmann, C. Klivans, S. Kurtz, D. MacQueen, K. Mulmuley, P. Niyogi, M. O'Donnell, A. Razborov, J. Reppy, A. Rogers, S. Salveter, L. R. Scott, J. Simon, R. Soare, R. Stevens

## Courses: Computer Science (CMSC)

*Graduate courses and seminars offered by the Department of Computer Science are open to College students with consent of instructor and departmental counselor. For more information, consult the departmental counselor.*

### *Undergraduate Courses*

**10100. Introduction to Programming for the World Wide Web I: HTML, Java, and CGI.** *This course **does not** meet the general education requirement in the mathematical sciences.* This course teaches the basics of building and maintaining a site on the World Wide Web. We discuss Internet terminology and how the Internet and its associated technologies work. Topics include programming websites, hypertext markup language (HTML), Cascading Style Sheets (CSS), and Common Gateway Interface (CGI) scripts (using PERL). Students also learn how to use JavaScript to add client-side functionality. *W. Sterner. Winter.*

**10200. Introduction to Programming for the World Wide Web II: HTML, CGI, and Java.** *PQ: MATH 10600, or placement into 13100 or equivalent; and knowledge of HTML.* This course meets the general education requirement in the mathematical sciences. This course introduces computer programming in Java. We cover the basics of object-oriented programming in Java with applications to the World Wide Web. We also discuss applets, basic graphics, and graphical user interfaces. As time permits, more advanced topics (e.g., threads, animation, Java Server Pages) are presented. *Spring.*

**10500. Fundamentals of Computer Programming I: Scheme.** *PQ: MATH 10600, or placement into 13100 or equivalent; or consent of departmental counselor required; previous computer experience and advanced mathematical knowledge not required.* CMSC 10500 and 10600 may be taken in sequence or individually. This course meets the general education requirement in the mathematical sciences. This course introduces computer programming using the functional programming language Scheme. We emphasize design, algorithm construction, and procedural/functional/data abstraction. *Autumn. Not offered 2009–10; will be offered 2010–11.*

**10600. Fundamentals of Computer Programming (C++) II.** *PQ: MATH 10600, or placement into 13100 or equivalent; or consent of departmental counselor.* CMSC 10500 and 10600 may be taken in sequence or individually. This course meets the general education requirement in the mathematical sciences. This course is an introduction to computer programming using the object-oriented programming language C++. We emphasize design and algorithm construction. Topics include complex types, iteration, recursion, procedural/functional/data abstraction, classes, methods, inheritance, and polymorphism. *S. Salveter. Winter.*

**11000-11100. Multimedia Programming as an Interdisciplinary Art I, II.** (=ISHU 23500-23600) *PQ: MATH 10600, or placement into 13100 or equivalent; or consent of instructor.* Either course in this sequence meets the general

*education requirement in the mathematical sciences.* Like other classic Chicago general education courses, this sequence provides students with both practical programming skills and core ideas in computer science in interdisciplinary applications. Our ideas of the arts, the character of “images” and “texts,” and the ways we form communities are being transformed by the conjunction of media and computing (e.g., QuickTime, scripting). Students program on an Apple Macintosh using an advanced programming language. This course presents techniques of problem solving, program coding, algorithm construction, and debugging using a high-level prototyping environment. We treat programs as genres of argument. *This course is offered in alternate years. W. Sterner. Winter, Spring.*

**11200. Introduction to Interactive Logic.** *PQ: MATH 10600, or placement into 13100 or equivalent. Some experience with computers helpful. This course does not meet the general education requirement in the mathematical sciences.* No programming skills are assumed, but those with some programming background do projects with HyperCard, a Computer Assisted Design package, Prolog, or other software. The course continues in the same spirit as CMSC 11000-11100, but they are not prerequisites. This hands-on course presents logic as a concrete discipline that is used for understanding and creating human-computer technology in the context of science, technology, and society. We look at computer science, logic, philosophy, aesthetics, design, and the study of technology, as well as at the software packages of Tarski’s World and possibly HyperProof. *This course is offered in alternate years. W. Sterner. Spring. Not offered 2009–10; will be offered 2010–11.*

**12100-12200. Computer Science with Applications I, II.** *PQ: Placement into MATH 15200 or higher, or consent of instructor. Either course in this sequence meets the general education requirement in the mathematical sciences.* This two-quarter sequence teaches computational thinking and skills to students in the sciences, mathematics, economics, etc. Lectures cover topics in (1) Programming, such as recursion, abstract data types, and processing data; (2) Computer Science, such as machine architecture, distributed systems (client/server applications), and theory of computation; and (3) Numerical Computation, such as approximating functions and their derivatives and integrals, solving systems of linear equations, and simple Monte Carlo techniques. Robust computing techniques are emphasized throughout the courses. Applications from a wide variety of fields serve both as examples in lectures and as the basis for programming assignments. Students learn Java, C, and at least one scripting language, such as Python or Matlab. *A. Rogers. Autumn, Winter.*

**15100-15200. Introduction to Computer Science I, II.** *PQ: Placement into MATH 15100 or equivalent, or consent of departmental counselor. Nonmajors may use either course in this sequence to meet the general education requirement in the mathematical sciences; students who are majoring in Computer Science must use either CMSC 15100-15200 or 16100-16200 to meet requirements for the major.* This course, which is recommended for all students planning to take more advanced courses in computer science, introduces computer science using both



functional (Scheme) and object-oriented (C++) programming languages. Topics include control and data abstraction, self-reference, time and space analysis, and data structures. *Autumn, Winter, Summer.*

**15300. Foundations of Software.** *PQ: CMSC 15200 or 12200. Required of students who are majoring in Computer Science.* This course is concerned with the mathematical foundations of computer software. We introduce a number of mathematical areas used in the modeling of programming languages and software, including propositional and predicate logic, basic set theory, relations, and automata theory. The connection between mathematics and software is made via examples and small programming assignments. *N. Hinrichs. Autumn.*

**15400. Introduction to Computer Systems.** *PQ: CMSC 15200 or 12200. Required of students who are majoring in Computer Science.* This course covers the basics of computer systems from a programmer's perspective. Topics include data re-presentation, machine language programming, exceptions, code optimization, performance measurement, memory systems, and system-level I/O. Extensive programming required. *G. Kindlmann. Spring.*

**16100-16200. Honors Introduction to Computer Science I, II.** *PQ: Placement into MATH 15100 or equivalent and programming experience, or consent of departmental counselor. Students who have taken CMSC 15100 may take 16200 with consent of instructor. Nonmajors may use either course in this sequence to meet the general education requirement in the mathematical sciences; students who are majoring in Computer Science must use either CMSC 15100-15200 or 16100-16200 to meet requirements for the major.* This sequence is an honors version of CMSC 15100-15200. Topics are covered more quickly with greater depth and breadth. *S. Kurtz. Autumn, Winter.*

### *Undergraduate and Graduate Courses*

*Other 20000-level courses may be offered. For the most up-to-date information, visit [www.cs.uchicago.edu](http://www.cs.uchicago.edu) and [timeschedules.uchicago.edu](http://timeschedules.uchicago.edu).*

**22100/32100. Programming Languages.** *PQ: CMSC 15300.* Programming language design aims at the closest possible correspondence between the structures of a program and the task it performs. This course studies some of the structural concepts affecting programming languages: iterative and recursive control flow, data types and type checking, procedural versus functional programming, modularity and encapsulation, fundamentals of interpreting and compiling, and formal descriptions of syntax and semantics. Students write short programs in radically different languages to illuminate the variety of possible designs. *Autumn.*

**22200/32200. Computer Architecture.** *PQ: CMSC 15400.* This course is a survey of contemporary computer organization covering CPU design, instruction sets, control, processors, busses, ALU, memory, pipelined computers, multiprocessors, networking, and case studies. We focus on the techniques of quantitative analysis and evaluation of modern computing systems, such as the selection of

appropriate benchmarks to reveal and compare the performance of alternative design choices in system design. We emphasize major component subsystems of high-performance computers: pipelining, instruction-level parallelism, memory hierarchies, input/output, and network-oriented interconnections. *Autumn*.

**22300. Functional Programming.** *PQ: CMSC 15300.* This course presents the functional programming paradigm and related program design concepts and techniques for systematically developing and reasoning about programs. The Haskell programming language is used as the vehicle for illustrating functional programming; it is a pure, lazy functional language that represents the functional paradigm in its most undiluted form. By providing better support for modularity and abstraction, functional programming languages make it easier to produce correct and well-structured software. We look at a range of programming examples covering lists, trees, and symbolic and textual structures. *D. MacQueen. Spring.*

**22610. Implementation of Computer Languages I.** *PQ: CMSC 15300 and 15400 required; CMSC 22100 recommended. Prior experience with ML programming not required.* This course covers principles and techniques for implementing computer languages (e.g., programming languages, query languages, specification languages, domain-specific languages). Topics include lexical analysis, parsing, tree representations of programs (both parse trees and abstract syntax trees), types and type checking, interpreters, abstract machines, and run-time systems. This is a project-based course involving the implementation of a small language using Standard ML. *J. Reppy. Winter. Not offered 2009–10; will be offered 2010–11.*

**22620/32620. Implementation of Computer Languages II.** *PQ: CMSC 22610 required; CMSC 22100 strongly recommended.* This course is a continuation of CMSC 22610, covering compilers for general-purpose languages. Topics include compiler-immediate representations, continuation-passing style, runtime representations, code generation, code optimization, register allocation, instruction scheduling, and garbage collection. This is a project-based course in which students construct a complete, working compiler for a small language using Standard ML. *Spring. Not offered 2009–10; will be offered 2010–11.*

**22630/32630. Advanced Implementation of Computer Languages.** *PQ: CMSC 22100 and 22620, or equivalent.* This course explores advanced topics in the implementation of high-level programming languages that vary each year (e.g., control-flow analysis algorithms, abstract interpretation, partial evaluation, advanced optimizations, runtime system representations, garbage collection algorithms, foreign-function interfaces). Students are expected to develop both a foundational and applied understanding of these topics. *J. Reppy. Autumn. Not offered 2009–10; will be offered 2010–11.*

**23000. Operating Systems.** *PQ: CMSC 15400.* This course covers basic concepts of operating systems. Topics include the notion of a process, interprocess communication and synchronization, main memory allocation, segmentation, paging, linking and loading, scheduling, file systems, and security and privacy.

Classes are taught on a Sun workstation using UNIX. *Spring. Not offered 2009–10; will be offered 2010–11.*

**23300/33300. Networks and Distributed Systems.** *PQ: CMSC 15400.* This course focuses on the principles and techniques used in the development of networked and distributed software. Topics include programming with sockets; remote procedure calls (RPC); interprocess communication (IPC); distributed objects (i.e., CORBA, DCOM); and commonly used network protocols (e.g., TCP/IP, UDP, FTP, HTTP). In addition, data encoding, encryption, and compression algorithms are presented. This is a project-oriented course in which students are required to develop software in the UNIX programming environment. *A. Rogers. Spring.*

**23340/33340. Grid Computing.** *PQ: Substantial programming experience.* The new Open Grid Services Architecture (OGSA) defines interfaces and protocols that promise to make it far easier to construct decentralized, dynamic, large-scale systems. We explore and evaluate this technology by using it to develop a range of scalable distributed services. We use the Globus Toolkit, an open source implementation of key OGSA standards, to design and build services. We then evaluate our implementations from the perspectives of performance and programmability. *I. Foster. Winter. Not offered 2009–10; will be offered 2010–11.*

**23500. Introduction to Database Systems.** *PQ: CMSC 15300 and 15400.* This course introduces database design and programming. Topics include entity-relationship and relational models, relational algebra, functional dependencies, and normal forms. We extensively cover SQL and take a brief look at data warehousing and data mining. *This course is offered in alternate years. S. Salveter. Spring.*

**23700. Introduction to Computer Graphics.** *PQ: CMSC 15400.* This course introduces the basic concepts and techniques used in three-dimensional computer graphics. The focus is on real-time rendering techniques, such as those found in computer games. These include coordinate systems and transformations, the graphics pipeline, basic geometric algorithms, texture mapping, level-of detail optimizations, and shadows. Students are required to complete both written assignments and programming projects using OpenGL. *This course is offered in alternate years. J. Reppy. Winter.*

**23710/33710. Scientific Visualization.** *PQ: Strong programming skills and basic knowledge of linear algebra and calculus.* Scientific visualization combines the image synthesis methods of computer graphics, numerical methods of scientific computing, and mathematical models of the physical world to create a visual framework for discovering, understanding, and solving scientific problems. This course describes the context, methods, and application of modern scientific visualization, giving students the skills required to evaluate, design, and create effective visualizations. This course, which uses mainly Python software and packages that have convenient Python interfaces, is also intended for nonmajors

with scientific data visualization needs. *This course is offered in alternate years. G. Kindlmann. Spring. Not offered 2009–10; will be offered 2010–11.*

**23800. Game Construction.** *PQ: CMSC 15400; 23700 or 25000; and one of 23000, 23300, or 23500. Strong background in programming and expertise in at least two technical areas underlying computer games (e.g., graphics, scientific computing, networking).* Computer games are one of the most exciting applications of computer technology. They also are large software systems that embody cutting-edge graphics, as well as techniques from scientific simulation, networking, AI, and databases. This course introduces issues of computer game construction. Students work in teams to design and create games using existing libraries for graphics, physics simulation, and so forth. *This course is offered in alternate years. J. Reppy. Spring.*

**25010. Artificial Intelligence.** *PQ: CMSC 15300.* This course introduces the theoretical, technical, and philosophical issues of AI. We emphasize computational and mathematical modes of inquiry into the structure and function of intelligent systems. Topics include learning and inference, speech and language, vision and robotics, and search and reasoning. *Winter.*

**25020/35050. Computational Linguistics.** *PQ: CMSC 15200 or 12200, or knowledge of C++.* This course introduces the problems of computational linguistics and the techniques used to deal with them. Topics are drawn primarily from phonology, morphology, and syntax. Special topics include automatic learning of grammatical structure and the treatment of languages other than English. *J. Goldsmith. Autumn.*

**25030. Computational Models of Speech.** *PQ: CMSC 25010.* This course covers computational models of speech, including acoustic-phonetic recognition, speech processing, and Fourier analysis. *P. Niyogi. Spring. Not offered 2009–10; will be offered 2010–11.*

**25040/35040. Introduction to Computer Vision.** *PQ: CMSC 25010.* This course introduces some of the fundamental concepts in computer vision. We cover selected topics in low-level image analysis, perceptual grouping, three-dimensional reconstruction, motion estimation, and object recognition. Our approach is to focus on mathematical models and efficient algorithms. *Spring.*

**25050/35500. Computer Vision.** *PQ: CMSC 25010.* This course covers methods and models for computer vision. Topics include segmentation and feature detection. *Y. Amit. Winter.*

**27100. Discrete Mathematics.** *PQ: CMSC 15300, or placement into MATH 15100 or equivalent. This is a directed course in mathematical topics and techniques that is a prerequisite for courses such as CMSC 2720 and 27400.* We emphasize mathematical discovery and rigorous proof, which are illustrated on a refreshing variety of accessible and useful topics. Basic counting is a recurring theme and provides the most important source for sequences, which is another recurring theme. Further topics include proof by induction; recurrences and Fibonacci

numbers; graph theory and trees; number theory, congruences, and Fermat's little theorem; counting, factorials, and binomial coefficients; combinatorial probability; random variables, expected value, and variance; and limits of sequences, asymptotic equality, and rates of growth. *Autumn*.

**27200. Theory of Algorithms.** *PQ: CMSC 27100 or consent of instructor.* This course covers design and analysis of efficient algorithms, with emphasis on ideas rather than on implementation. Algorithmic questions include sorting and searching, discrete optimization, algorithmic graph theory, algorithmic number theory, and cryptography. Design techniques include "divide-and-conquer" methods, dynamic programming, greedy algorithms, and graph search, as well as the design of efficient data structures. Methods of algorithm analysis include asymptotic notation, evaluation of recurrent inequalities, the concepts of polynomial-time algorithms, and NP-completeness. *Winter*.

**27400. Honors Combinatorics and Probability.** (=MATH 28400) *PQ: MATH 25000 or 25400, or CMSC 27100, or consent of instructor. Experience with mathematical proofs.* Methods of enumeration, construction, and proof of existence of discrete structures are discussed in conjunction with the basic concepts of probability theory over a finite sample space. Enumeration techniques are applied to the calculation of probabilities, and, conversely, probabilistic arguments are used in the analysis of combinatorial structures. Other topics include basic counting, linear recurrences, generating functions, Latin squares, finite projective planes, graph theory, Ramsey theory, coloring graphs and set systems, random variables, independence, expected value, standard deviation, and Chebyshev's and Chernoff's inequalities. *L. Babai. Spring*.

**27500. Graph Theory.** *PQ: CMSC 15300.* This course covers the basics of the theory of finite graphs. Topics include shortest paths, spanning trees, counting techniques, matchings, Hamiltonian cycles, chromatic number, extremal graph theory, Turan's theorem, planarity, Menger's theorem, the max-flow/min-cut theorem, Ramsey theory, directed graphs, strongly connected components, directed acyclic graphs, and tournaments. Techniques studied include the probabilistic method. *Spring*.

**27600. Computational Biology.** *PQ: Familiarity with basic discrete mathematics/statistics/algorithms and biology is helpful but not required.* This course serves as a general introduction to the basic algorithms used to understand current problems in biology. Topics may include sequence alignment algorithms to study DNA and protein sequences, algorithms and experiments for protein structure prediction, dynamics, and folding, clustering and machine learning methods for gene expression analysis, computational models of RNA structure, and DNA computing and self-assembly. *N. Hinrichs. Spring. Not offered 2009–10; will be offered 2010–11.*

**27700. Mathematical Logic I.** (=MATH 27700) *PQ: MATH 25400 or 25700.* This course introduces mathematical logic. Topics include propositional and

predicate logic and the syntactic notion of proof versus the semantic notion of truth (e.g., soundness, completeness). We also discuss the Gödel completeness theorem, the compactness theorem, and applications of compactness to algebraic problems. *Autumn*.

**27800. Mathematical Logic II.** (=MATH 27800) *PQ: MATH 27700 or equivalent.* Topics include number theory, Peano arithmetic, Turing compatibility, unsolvable problems, Gödel's incompleteness theorem, undecidable theories (e.g., the theory of groups), quantifier elimination, and decidable theories (e.g., the theory of algebraically closed fields). *Winter*.

**27900. Chaos, Complexity, and Computers.** (=MATH 29200, PHYS 25100) *PQ: One year of calculus, two quarters of physics at any level, and PHYS 25000 or prior programming experience.* For course description, see Physics. *Winter. L.*

**28000. Introduction to Formal Languages.** (=MATH 28000) *PQ: CMSC 15300, or MATH 25000 or 25500.* This course is a basic introduction to computability theory and formal languages. Topics include automata theory, regular languages, context-free languages, and Turing machines. *Autumn*.

**28100. Introduction to Complexity Theory.** (=MATH 28100) *PQ: CMSC 27100, or MATH 25000 or 25500; and experience with mathematical proofs.* Computability topics are discussed (e.g., the s-m-n theorem and the recursion theorem, resource-bounded computation). This course introduces complexity theory. Relationships between space and time, determinism and non-determinism, NP-completeness, and the P versus NP question are investigated. *Spring*.

**28501. Topics in Scientific Computing.** *PQ: Consent of instructor.* This course covers current topics in scientific computing. *Autumn, Winter, Spring*.

**28510. Introduction to Scientific Computing.** *PQ: A year of calculus (MATH 15300 or higher), a quarter of linear algebra (MATH 19620 or higher), and CMSC 10600 or higher; or consent of instructor.* Basic processes of numerical computation are examined from both an experimental and theoretical point of view. This course deals with numerical linear algebra, approximation of functions, approximate integration and differentiation, Fourier transformation, solution of nonlinear equations, and the approximate solution of initial value problems for ordinary differential equations. We concentrate on a few widely used methods in each area covered. *T. Dupont. Autumn. Not offered 2009–10; will be offered 2010–11.*

**29500. Digital Sound Modeling.** *PQ: Knowledge of computer programming or consent of instructor.* Students taking this course study how the basic structure of sound perception affects the useful ways of representing sound in digital computations. We focus on basic synthesis techniques, rather than on signal analysis or on special applications of synthesis (e.g., music, speech). Course work is divided among intuitive mathematical studies, listening exercises, and a cooperative project using synthesis software to simulate an orchestral instrument. *M. O'Donnell. Spring*.

**29700. Reading and Research in Computer Science.** *PQ: Consent of instructor and approval of departmental counselor. Open both to students who are majoring in Computer Science and to nonmajors. Students are required to submit the College Reading and Research Course Form.* Students do reading and research in an area of computer science under the guidance of a faculty member. A written report is typically required. *Summer, Autumn, Winter, Spring.*

**29900. Bachelor's Thesis.** *PQ: Open to fourth-year students who are candidates for honors in Computer Science. Consent of instructor and departmental counselor. Students are required to submit the College Reading and Research Course Form.* *Summer, Autumn, Winter, Spring.*

### *Graduate Courses*

*College students may register for graduate courses with consent of departmental counselor. For details, visit [www.cs.uchicago.edu](http://www.cs.uchicago.edu).*

**31100. Big Ideas in Computer Science.** *PQ: Consent of instructor.* This course introduces many of the important concepts in the broad area of computer science. Each week a different professor gives a three-lecture sequence on a big idea in their field of specialty. Previous ideas have included undecidability, randomness, cryptography, stability of numerical algorithms, structural operational semantics, software engineering, and the Internet. *Autumn.*

**31900. Lambda Calculus.** *PQ: Knowledge of algebra or equivalent level of math.* This course covers the crucial properties of the lambda calculus and its variant, the combinator calculus. We emphasize the deep connections between various areas of logic and computation that arise from the ability to interpret a lambda term equally naturally as a program of type A to B and as a proof that A implies B. *M. O'Donnell. Winter. Not offered 2009–10; will be offered 2010–11.*

**32001. Topics in Programming Languages.** *PQ: Consent of instructor.* This course covers a selection of advanced topics in programming languages. *Autumn, Winter, Spring.*

**32201. Topics in Computer Architecture.** *PQ: Consent of instructor.* This course covers a selection of advanced topics in computer architecture. *Autumn, Winter, Spring.*

**33501. Topics in Databases.** *PQ: Consent of instructor.* This course covers a selection of advanced topics in database systems. *Autumn, Winter, Spring.*

**33600. Type Systems for Programming Languages.** *PQ: CMSC 22100 recommended.* This course covers the basic ideas of type systems, their formal properties, their role in programming language design, and their implementation. Exercises involving design and implementation explore the various options and issues. *D. MacQueen. Winter.*



**34000. Scientific Parallel Computing.** *PQ: Consent of instructor required; experience in scientific computing helpful.* We study the use of multiple processors cooperating to solve a common task, as well as related issues in computer architecture, performance analysis, prediction and measurement, programming languages, and algorithms for large-scale computation. Programming at least one parallel computer is required. Possibilities include one of the clusters of workstations connected by high-speed networks on campus. We focus on state-of-the-art parallel algorithms for scientific computing. Topics are based on interest. General principles of parallel computing are emphasized. *L. R. Scott. Autumn.*

**34200. Numerical Hydrodynamics.** *PQ: Ability to program; and familiarity with elementary numerical methods and modeling physical systems by systems of differential equations.* This course covers numerical methods for the solution of fluid flow problems. We also make a theoretical evaluation of the methods and experimental study based on the opinionated book *Fundamentals of Computational Fluid Dynamics* by Patrick J. Roache. *T. Dupont. Spring.*

**34710. Wireless Sensor Networks.** *PQ: Graduate-level understanding of Unix/Linux operating systems, networking, computer architecture, and programming.* This course introduces the concepts and technologies for building embedded systems and wireless sensors nets by focusing on four areas: low-power hardware, wireless networking, embedded operating systems, and sensors. Two assignments provide hands-on experience by deploying small wireless sensor motes running TinyOS to form an ad-hoc peer-to-peer network that can collect environmental data and forward it back to an 802.11b-equipped embedded Linux module. Students also read and summarize papers, participate in classroom discussions, and work on a team research project. *R. Stevens. Not offered 2009–10; will be offered 2010–11.*

**35000. Introduction to Artificial Intelligence.** *PQ: CMSC 25000.* This course is an introduction to the theoretical, technical, and philosophical aspects of Artificial Intelligence. The emphasis is on computational and mathematical modes of inquiry into the structure and function of intelligent systems. Topics include learning and inference, speech and language, vision and robotics, and reasoning and search. *P. Niyogi. Winter.*

**35100. Natural Language Processing.** *PQ: CMSC 25000/35000 or consent of instructor.* This course is an introduction to the theory and practice of natural language processing, with applications to both text and speech. Topics include regular expressions, finite state automata, morphology, part of speech tagging, context free grammars, parsing, semantics, discourse, and dialogue. Symbolic and probabilistic models are presented. Techniques for automatic acquisition of linguistic knowledge are emphasized. *Spring. Not offered 2009–10; will be offered 2010–11.*

**35400. Machine Learning.** *PQ: CMSC 25000/35000 or consent of instructor.* This course is an introduction to the theory and practice of machine learning that emphasizes statistical approaches to the problem. Topics include pattern recognition, empirical risk minimization and the Vapnik Chervonenkis theory,



neural networks, decision trees, genetic algorithms, unsupervised learning, and multiple classifiers. *P. Niyogi. Spring.*

**35500. Computer Vision.** (=STAT 37900) *PQ: Consent of instructor.* This course covers deformable models for detecting objects in images. Topics include one-dimensional models to identify object contours and boundaries; two-dimensional models for image matching; and sparse models for efficient detection of objects in complex scenes. Mathematical tools needed to define the models and associated algorithms are developed. Applications include detecting contours in medical images, matching brains, and detecting faces in images. Neural network implementations of some of the algorithms are presented, and connections to the functions of the biological visual system are discussed. *Y. Amit. Winter.*

**35900. Topics in Artificial Intelligence.** *PQ: Consent of instructor.* This course covers topics in artificial intelligence. *Autumn, Winter, Spring.*

**36500. Algorithms in Finite Groups.** (=MATH 37500) *PQ: Linear algebra, finite fields, and a first course in group theory (Jordan-Holder and Sylow theorems) required; prior knowledge of algorithms not required.* We consider the asymptotic complexity of some of the basic problems of computational group theory. The course demonstrates the relevance of a mix of mathematical techniques, ranging from combinatorial ideas, the elements of probability theory, and elementary group theory, to the theories of rapidly mixing Markov chains, applications of simply stated consequences of the Classification of Finite Simple Groups (CFSG), and, occasionally, detailed information about finite simple groups. No programming problems are assigned. *L. Babai. Spring. Not offered 2009–10; will be offered 2010–11.*

**37000. Algorithms.** *PQ: CMSC 27200 or consent of instructor.* The focus of this course is the analysis and design of efficient algorithms, with emphasis on ideas rather than on implementation. Algorithmic questions include sorting and searching, discrete optimization, algorithmic graph theory, algorithmic number theory, and cryptography. Design techniques include “divide-and-conquer” methods, dynamic programming, greedy algorithms, and graph search, as well as the design of efficient data structures. Methods of algorithm analysis include asymptotic notation, evaluation of recurrent inequalities, the concepts of polynomial-time algorithms, and NP-completeness. *L. Babai. Winter.*

**37100. Topics in Algorithms.** *PQ: CMSC 27200 or consent of instructor.* This course discusses current topics in algorithms. *Autumn, Winter, Spring.*

**37110. Discrete Mathematics.** *PQ: Consent of instructor.* This course emphasizes mathematical discovery and rigorous proof, illustrated on a variety of accessible and useful topics, including basic number theory, asymptotic growth of sequences, combinatorics and graph theory, discrete probability, and finite Markov chains. The course includes an introduction to linear algebra. *L. Babai. Autumn.*

**37200. Combinatorics.** *PQ: Linear algebra, basic combinatorics, or consent of instructor.* Methods of enumeration, construction, and proof of existence of discrete structures are discussed. The course emphasizes applications of linear algebra, number theory, and the probabilistic method to combinatorics. Applications to the theory of computing are indicated, and open problems are discussed. *This course is offered in alternate years. L. Babai. Spring.*

**37400. Constructive Combinatorics.** *PQ: Advanced knowledge of mathematics and consent of instructor.* This course covers constructive combinatorial techniques in areas such as enumerative combinatorics, invariant theory, and representation theory of symmetric groups. Constructive techniques refer to techniques that have algorithmic flavor, such as those that are against purely existential techniques based on counting. *K. Mulmuley. Spring. Not offered 2009–10; will be offered 2010–11.*

**37600. Computational Biology.** *N. Hinrichs. Spring. Not offered 2009–10; will be offered 2010–11.*

**37701. Topics in Bioinformatics.** *PQ: Consent of instructor.* Current topics in bioinformatics are covered in this course. *Autumn, Winter, Spring.*

**37720. Computational Systems Biology.** *PQ: Consent of instructor.* This course introduces concepts of systems biology. We also discuss computational methods for analysis, reconstruction, visualization, modeling, and simulation of complex cellular networks (e.g., biochemical pathways for metabolism, regulation, and signaling). Students explore systems of their own choosing and participate in developing algorithms and tools for comparative genomic analysis, metabolic pathway construction, stoichiometric analysis, flux analysis, metabolic modeling, and cell simulation. We also focus on understanding the computer science challenges in the engineering of prokaryotic organisms. *R. Stevens. Autumn. Not offered 2009–10; will be offered 2010–11.*

**37800. Numerical Computation.** (=STAT 30700) *PQ: STAT 34300 or consent of instructor.* This course covers topics in numerical methods and computation that are useful in statistical research (e.g., simulation, random number generation, Monte Carlo methods, quadrature, optimization, matrix methods). *Autumn.*

**38000-38100. Computability Theory I, II.** (=MATH 30200-30300) *PQ: MATH 25500 or consent of instructor.* CMSC 38000 is concerned with recursive (computable) functions and sets generated by an algorithm (recursively enumerable sets). Topics include various mathematical models for computations (e.g., Turing machines and Kleene schemata, enumeration and s-m-n theorems, the recursion theorem, classification of unsolvable problems, priority methods for the construction of recursively enumerable sets and degrees). CMSC 38100 treats classification of sets by the degree of information they encode, algebraic structure and degrees of recursively enumerable sets, advanced priority methods, and generalized recursion theory. *This course is taught in alternate years. R. Soare. Winter, Spring.*

**38300. Numerical Solutions to Partial Differential Equations.** (=MATH 38300) *PQ: Consent of instructor.* This course covers the basic mathematical theory behind numerical solution of partial differential equations. We investigate the convergence properties of finite element, finite difference and other discretization methods for solving partial differential equations, introducing Sobolev spaces and polynomial approximation theory. We emphasize error estimators, adaptivity, and optimal-order solvers for linear systems arising from PDEs. Special topics include PDEs of fluid mechanics, max-norm error estimates, and Banach-space operator-interpolation techniques. *T. Dupont. Spring.*

**38500. Computability and Complexity Theory.** (=MATH 30500) *PQ: Consent of instructor.* Part one consists of models for defining computable functions: primitive recursive functions, (general) recursive functions, and Turing machines; the Church-Turing Thesis; unsolvable problems; diagonalization; and properties of computably enumerable sets. Part two deals with Kolmogorov (resource bounded) complexity: the quantity of information in individual objects. Part three covers functions computable with time and space bounds of the Turing machine: polynomial time computability, the classes P and NP, NP-complete problems, polynomial time hierarchy, and P-space complete problems. *A. Razborov. Winter.*

**38512. Kolmogorov Complexity.** *PQ: Consent of instructor.* This course introduces the theory of Kolmogorov Complexity with an emphasis on its use in theoretical computer science, mostly in computational complexity. If time permits, we may briefly touch on its uses in statics, prediction, and learning. *J. Simon. Autumn.*

**38600. Complexity Theory A.** *PQ: Consent of instructor.* Topics are covered in computational complexity theory with an emphasis on machine-based complexity classes. *Autumn. Not offered 2009–10; will be offered 2010–11.*

**38700. Complexity Theory B.** *PQ: Consent of instructor.* Topics are covered in computational complexity theory with an emphasis on combinatorial problems in complexity. *A. Razborov. Spring.*

**39000. Computational Geometry.** *PQ: Consent of instructor.* This course is a seminar on topics in computational geometry. *K. Mulmuley. Autumn. Not offered 2009–10; will be offered 2010–11.*

**39600. Topics in Theoretical Computer Science.** *PQ: Consent of instructor.* This course is a seminar on current research in theoretical computer science. *Autumn, Winter, Spring.*