

Computer Science

*Departmental Counselor: Sharon Salveter, Ry 161B, 834-2773,
salveter@cs.uchicago.edu*

*Student Services Representative: Margaret Jaffey, Ry 161A,
702-6011, margaret@cs.uchicago.edu*

Web: www.cs.uchicago.edu

Program of Study

The computer science concentration program prepares students for either graduate work or employment in computer science by offering both the B.A. and B.S. degrees. Students receiving the B.A. will have sufficient breadth and depth for either graduate study or immediate employment in computer science. Recipients of the B.S. will, in addition, have substantial depth and breadth in a field outside of computer science through the completion of an approved minor program.

A concentration in mathematics with a specialization in computer science continues to meet the needs of mathematics concentrators who also have a strong interest in computing. For a description of that program, see the Mathematics section of this catalog.

Program Requirements

Both the B.A. and B.S. in Computer Science require fulfillment of the College's general education requirements. The mathematical sciences requirement in general education must be satisfied 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 B.A. and B.S. students take at least eleven computer science courses chosen from an approved program.

B.A. students also take three approved elective courses, either in computer science or in a related area. B.S. students take two approved elective courses, either in computer science or a related area, one course in linear algebra, and a three-course minor in an approved related field outside computer science.

Students taking a bachelor's degree in computer science should note that by judicious choice of courses from another field for extra-departmental requirements or for electives, a minor field can be developed that is often in itself a solid basis for graduate or professional work in that field. Some disciplines where this collateral minor benefit applies include biology, biophysics, chemistry, education, geophysical sciences, history, linguistics, mathematics, philosophy, political science, psychology, physics, sociology, statistics, and theoretical economics.

Advanced Placement. Computer science concentrators may not use AP credit for computer science to meet concentration requirements. Students with AP scores of 4 or 5 on Computer Science AB receive two quarters of elective credit. NOTE: Students must forego AP elective credit if they

register for CMSC 10500, 10600, 15100, or 15200. Students who enroll in CMSC 16100 and 16200 may retain AP elective credit.

Computer science students may use AP credit for chemistry or physics to meet their physical sciences requirement in general education or physical science components of the concentration. 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 concentration requirements.

Approved Programs. The notion of “approval” in the concentration program requirements allows timely response to change in the course offerings of the various departments. The computer science department counselor is responsible for approval of specific courses and sequences. An initial list of approved course sequences follows, but additional courses may be approved. See the departmental counselor for details on specific courses you are considering taking to meet the requirements.

Approved Computer Science Concentration Program

For the authoritative version of the Department of Computer Science requirements and course descriptions, see *www.cs.uchicago.edu*.

There is a single approved program comprising required courses in four topical areas plus the minor. This is a general program in computer science and is used for either the B.A. or the B.S. 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.

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, 23500, 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 25000-25100
 - b. Advanced Systems Sequence (two courses required):
 CMSC 22100, 22200, 22610, 22620, 23000, 23300, 23500 or 23700
 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, or 28520, or 28530.

Approved Course Sequences from Outside Computer Science

Students in the B.A. and B.S. programs may choose from among the following courses for approved courses taken in a related area outside computer sciences. Other courses are acceptable as approved by the departmental counselor.

ASTR 21300-24200

BIOS 20191-20192, 20300, 21000, 21100, 21200, 21300, 21400, 21500, 21600, 21700, 21800, 22000, 22100, 22200, 22600, 22800, 22900, 23000, 23100, 23600, 24700, 25600, 25800, 25900

CHEM 11101-11102/11201-11202 or equivalent (if chemistry is not used to meet the physical sciences requirement)

CHEM 20100, 20200, 22000, 22100/23100

ECON 20000, 20100, 20200, 20300, 21000, 21100, 22500, 23100, 25000, 27000, 27100, 28100

GEOS 23100-23400, 23500-23600, 23900

LING 20100-21000, 21700

MATH 20300-21100, 25400-27800

MUSI 26300-26400

PHIL 23500-28500

PHYS 12100-12200, 13100-13200, 14100-14200 (if physics is not used to meet the physical sciences requirement)

PHYS 22500-22700, 23400-23500, 23600-23700

PSYC 20300, 20400, 20700, 21100, 22500, 23200, 24000, 27000, 27500, 27600, 28300

STAT 22000-25100

For students in the B.S. program, approved linear algebra courses include MATH 25000, 25500, and 25800. Three-course minor programs must be approved by the departmental counselor. Students are urged to see the departmental counselor as soon as possible to discuss their choice.

Summary of Requirements

General Education	CHEM 11101-11201/11102-11202 or equivalent†, or PHYS 12100-12200 or higher† MATH 13100-13200 or higher†
Concentration	11 courses in computer science from the approved program <i>plus the following requirements:</i>

B.A.		B.S.	
3	approved elective courses, either in computer science or a related field	1	course in linear algebra
		2	approved elective courses, either in computer science or a related field
<hr/> 14		3	courses in an approved minor program in a related field
		<hr/> 17	

† *Credit may be granted by examination.*

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

Nonconcentrators may take courses for either quality grades or, subject to College regulations and with consent of instructor, for *P/F* grades. 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 those 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. Students must make arrangements in advance with instructors and obtain their written consent to receive Incompletes.

Honors. Students may earn a B.A. or B.S. degree with honors by attaining a grade of *B* or higher in all courses in the concentration and a grade of *B* or higher in a three-course sequence (taken as a minor or as electives) consisting of graduate computer science courses (30000-level and above).

Students may also earn a B.A. or B.S. degree with honors by attaining the same minimum *B* grade in all courses in the concentration 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 10600-10700 (Fundamentals of Computer Programming II, III). 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 25000 and 25100 (Introduction to Artificial Intelligence). Graduate-level AI courses are open to College students. These courses are numbered CMSC 35000 to 35400. See course listings for details.

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) 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 Language and Automata) and CMSC 28100 (Introduction to Complexity Theory). Once students have completed CMSC 27100, 27200 and 28000 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), CMSC 28520 (High-Performance Scientific Software), and CMSC 28530 (Scientific Systems Programming).

The department also offers a number of special-interest courses that are detailed in the course descriptions. New courses are added on a regular basis. See the departmental counselor and www.cs.uchicago.edu.

Preparation for Graduate Study in Computer Science. Students interested in continuing their studies beyond the undergraduate level should concentrate 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, 27100, 27200, 28000, and 28100. For more information about options for graduate study, consult the departmental counselor and the director of graduate studies.

Faculty

Y. Amit, L. Babai, D. Beazley, T. Dupont, R. Findler, I. Foster, L. Irakliotis, R. Kirby, S. Kurtz, G.-A. Levow, D. MacQueen, K. Mulmuley, S. Nestorov, P. Niyogi, M. O'Donnell, J. Reppy, A. Rogers, S. Salveter, L. R. Scott, J. Simon, R. Soare, R. Stevens, E. Vigoda

Courses: Computer Science (CMSC)

Graduate courses and seminars offered by the Department of Computer Science are open to College students with the consent of the instructor and the departmental counselor. See the departmental Student Services Representative for more information.

Undergraduate Courses

10000. Web Design: Aesthetics and Languages. (=HUMA 25100, ISHU 29600) As a complement to courses in criticism, aesthetics, cultural studies, or Web programming, this course explores Web design as a liberal art of technology. Good multimedia design is based on our sensory intelligences. Yet, on the Web it requires syntheses of natural languages and artificial languages; of grammar, rhetoric, and logic; and, of course, mastery of the subject matter. What design principles communicate information, narratives, and explanations? We examine and create design environments in print and electronic media, with a focus on the Internet. *M. Browning. Winter.*

10100. Introduction to Programming for the World Wide Web (HTML, CGI, and Java) I. 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 computer programming, programming Web sites, 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. *Winter, Summer.*

10200. Introduction to Programming for the World Wide Web (HTML, CGI, and Java) II. *PQ: Knowledge of HTML.* This course meets the general education requirement in the mathematical sciences. Topics include dynamic hypertext markup languages (dHTML), Common Gateway Interface (CGI) scripts (using PERL), and Java. Students learn how to set up a Web server, write Java applets, and use communication components such as ActiveX and Javabeans. *R. Kirby. Spring.*

10500. Fundamentals of Computer Programming (Scheme) I. *PQ: MATH 10600, or placement into 13100 or equivalent; or consent of departmental counselor. This course meets the general education requirement in the mathematical sciences. CMSC 10500 and 10600 may be taken in sequence or individually.* This course is an introduction to computer programming using the functional programming language Scheme. We emphasize design, algorithm construction, and procedural/functional/data abstraction. Both CMSC 10500 and CMSC 15100 are general introductions to computer programming. CMSC 10500 assumes no previous computer experience and less advanced mathematical knowledge. *S. Salveter, Staff. Autumn.*

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, Staff. Winter.*

10700. Fundamentals of Computer Programming III. *PQ: Knowledge of C++. This course meets the general education requirement in the mathematical sciences.* This course is an introduction to data structures. Topics include linked and sequential storage allocation, linked lists, stacks, queues, trees, and graphs. *S. Salveter. Spring.*

11000-11100. Multimedia Programming as an Interdisciplinary Art I, II. (=ISHU 23500-23600) *PQ: MATH 10600, or placement into MATH 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 core 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. *W. Sterner. Winter, Spring.*

11200. Introduction to Interactive Logic. (=ISHU 23700) *PQ: MATH 10600, or placement into 13100, or equivalent. Some experience with computers helpful.* 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. 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. W. Sterner. *Spring*.

15100-15200. Introduction to Computer Science I, II. *PQ: Placement into MATH 15100 or equivalent, or consent of departmental counselor. Nonconcentrators may use either course in this sequence to meet the general education requirement in the mathematical sciences; concentrators must use either CMSC 15100-15200 or 16100-16200 to meet concentration requirements.* An introduction to 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. This sequence is recommended for all students planning to take more advanced courses in computer science. S. Kurtz, R. Findler, R. Kirby, S. Nestorov. *Autumn, Winter*.

15300. Foundations of Software. *PQ: CMSC 15200. Required of concentrators.* 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. S. Kurtz, J. Reppy. *Spring*.

15400. Introduction to Computer Systems. *PQ: CMSC 15200 or 11600. Required of concentrators.* This course covers the basics of computer systems from a programmer's perspective. Topics include data representation, machine language programming, exceptions, code optimization, performance measurement, memory systems, and system-level I/O. Extensive programming required. A. Rogers. *Autumn*.

16100-16200. Honors Introduction to Computer Science I, II. *PQ: Placement into MATH 16100 or equivalent, or consent of departmental counselor. Students who have taken CMSC 15100 may take CMSC 16200 with consent of instructor. Nonconcentrators may use either course in this sequence to meet the general education requirement in the mathematical sciences; concentrators must use either CMSC 15100-15200 or 16100-16200 to meet concentration requirements.* This sequence is a honors version of CMSC 15100-15200. Topics are covered more quickly with greater depth and breadth. We emphasize studying programs as formal objects. M. O'Donnell, T. Dupont. *Autumn, Winter*.

Undergraduate and Graduate Courses

Other 20000-level courses may be offered. Please see www.cs.uchicago.edu and the quarterly Time Schedules for the most up-to-date information.

21500. Logic and Logic Programming. (=MATH 27900) *PQ: MATH 25400 or CMSC 27700, or consent of instructor. Students are encouraged to take both CMSC 21500 and 27700. Programming knowledge not required.* Predicate logic is a precise logical system developed to formally express mathematical reasoning. Prolog is a computer language intended to

implement a portion of predicate logic. This course covers both predicate logic and Prolog, which are presented to complement each other and to illustrate the principles of logic programming and automated theorem proving. Topics include syntax and semantics of propositional and predicate logic, tableaux proofs, resolution, Skolemization, Herbrand's theorem, unification, and refining resolution. It includes weekly classes and programming assignments in Prolog (e.g., searching, backtracking, cut). *R. Soare. Winter.*

22001. Software Construction. *PQ: CMSC 15400 or 11700.* Beyond specific domain skills, building software is a craft that requires careful design. This course teaches key software design principles in a studio setting. Each week, students present their programs to the class for a design review. Together, the class evaluates the programs for their correctness and, more importantly, their clarity and design. Students learn how to build reliable, maintainable, extensible software and how to evaluate other code for those same properties. *R. Findler. Spring.*

22100. Programming Languages. *PQ: CMSC 15300 or 11700.* 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. *D. MacQueen. Autumn.*

22200. Computer Architecture. *PQ: CMSC 15400 or 11700.* 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. The emphasis is on major component subsystems of high-performance computers: pipelining, instruction-level parallelism, memory hierarchies, input/output, and network-oriented interconnections. We may cover topics such as portable computers, high-performance parallel computers, graphics computers, and performance modeling. *A. Rogers. Spring.*

22610. Implementation of Computer Languages I. *PQ: CMSC 15300 and 15400 required; CMSC 22100 strongly recommended. Prior experience with ML programming not required.* This course covers principles and techniques for implementing computer languages, such as programming languages, query languages, specification languages and 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.*

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.*

22800. Free Software Practicum. *PQ: Unix programming experience and consent of instructor.* Students who are already proficient in programming are provided with the experience of working on real software and the opportunity to collaborate with distributed teams of developers. The course work consists entirely of one or more programming tasks, which must produce freely distributed code. Course work may be chosen from the bug lists and to-do lists of well-known free software projects (i.e., Gnome, KDE, Hurd, Mozilla). Students may work individually or in groups. To earn credit, the work of the student must be incorporated into the chosen project's distribution. *M. O'Donnell. Autumn, Winter, Spring.*

23000/33000. Operating Systems. *PQ: CMSC 15400 or 11700.* This course covers basic concepts of operating systems. Topics discussed 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. The course is taught on a Sun workstation using UNIX. *D. Beazley. Winter.*

23300/33300. Networks and Distributed Systems. *PQ: CMSC 23000 or consent of instructor. Basic knowledge of C, C++, and Java, as well as operating system concepts such as processes and threads.* 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 (CORBA and DCOM), and commonly used network protocols including TCP/IP, UDP, FTP, and 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. *Spring.*

23500. Introduction to Database Systems. *PQ: CMSC 15300 or 11700 required; CMSC 27100 recommended.* An introduction to 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. Students design and implement e-commerce database applications using a commercial RDBMS. *S. Nestorov. Autumn.*

23700. Introduction to Computer Graphics. *PQ: CMSC 15400 or 11700.* An introduction to the basic concepts and techniques used in 3D 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. The course involves both written assignments and programming projects using OpenGL. *J. Reppy. Autumn.*

23710/33710. Scientific Visualization. *PQ: CMSC 15200 and 23700, or consent of instructor.* Introduces the concepts and technologies for scientific visualization and advanced display environments. Topics include data models, algorithms for 2D and 3D data and information visualization, programming environments and toolkits, and interaction techniques and advanced displays such as stereoscopic displays, tiled displays and virtual reality. Requirements include written assignments, programming assignments, and a final course project. The course is aimed at those with needs for visualization of domain data as well as those interested in scientific visualization as a research project. *R. Stevens. Winter.*

24000. Information Theory and Coding. (=PSYC 28800/38800) *PQ: Consent of instructor.* An introduction to the mathematical theory of information with emphasis on coding, especially the development of efficient codes. Topics include an introduction to coding, quantification of information and its properties, Huffman codes, arithmetic codes, L to Z, and other adaptive coding techniques, and specific applications. *A. Bookstein. Winter.*

25000-25100. Introduction to Artificial Intelligence and LISP I, II. *PQ: CMSC 15300 or 11700.* An introduction to the theoretical, technical, and philosophical issues of AI. 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, search and reasoning. LISP and LISP programming are introduced. The second course looks at natural language processing, planning, problem solving, diagnostic systems, naive physics, and game playing. *G. Levow, P. Niyogi. Winter, Spring.*

27100. Discrete Mathematics. *PQ: Placement into MATH 15100 or equivalent, or consent of departmental counselor. This is a directed course in mathematical topics and techniques needed by students taking Algorithms (CMSC 27200). It is also a prerequisite to several other courses, including Honors Combinatorics and Probability (CMSC 27400/MATH 28400).* This course emphasizes 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. *L. Babai. Autumn.*

27200. Theory of Algorithms. *PQ: CMSC 27100 or consent of instructor.* 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. *L. Babai. Winter.*

27400. Honors Combinatorics and Probability. (=MATH 28400) *PQ: MATH 25000 or 25400, or CMSC 27100, or consent of instructor; and 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. Among other topics are 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.*

27502/37502. Advanced Algorithms. *PQ: CMSC 27200.* This course covers advanced topics in algorithms. *E. Vigoda. Spring.*

27700. Mathematical Logic I. (=MATH 27700) *PQ: MATH 25400.* This course provides an introduction to mathematical logic. Topics include propositional and predicate logic and the syntactic notion of proof versus the semantic notion of truth, including soundness and completeness. We also discuss the Goedel completeness theorem, the compactness theorem, and applications of compactness to algebraic problems. *Autumn.*

27800. Mathematical Logic II. (=MATH 27800) *PQ: MATH 27700 or equivalent.* Some of the topics examined are 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.*

28000. Introduction to Formal Languages. (=MATH 28000) *PQ: MATH 25000 or 25500, or CMSC 27100; and experience with mathematical proofs.* Topics include automata theory, regular languages, CFLs, and Turing machines. This course is a basic introduction to computability theory and formal languages. *J. Simon. Autumn.*

28100. Introduction to Complexity Theory. (=MATH 28100) *PQ: MATH 25000 or 25500, or CMSC 27100; 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 nondeterminism, NP-completeness, and the P versus NP question are investigated. *Spring.*

28510. Introduction to Scientific Computing. *PQ: A year of calculus (MATH 15300 or higher), a quarter of linear algebra (MATH 25000 or higher), and CMSC 10600 or higher; or consent of instructor.* Numerical computation is used in science primarily to model the behavior of continuous systems, things described by real numbers as opposed to integers,

and it is this aspect of computation that is the focus of this course and the sequence that it introduces. Basic processes of numerical computation are examined from both an experimental and theoretical point of view. The 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. The course concentrates on a few widely used methods in each area covered. *T. Dupont. Autumn.*

28520. High-Performance Scientific Software. *PQ: CMSC 28510 or consent of instructor.* This course revisits many numerical algorithms studied in CMSC 28500 from a software performance perspective. We examine issues of single-processor performance and code reusability, including object orientation and interfacing with standard packages such as BLAS and LAPACK. The linear algebraic tools discussed early in the class are used to address problems in interpolation, integration, nonlinear equations, and differential equations. *R. Kirby. Winter.*

28530. Scientific Systems Programming. *PQ: CMSC 28520 or consent of instructor. Basic knowledge of elementary numerical methods, programming in C or C++, and Unix required; prior experience with operating systems not required.* Computational scientists are increasingly required to build, maintain, and utilize software that relies upon software components, scripting languages, databases, and parallel processing. Although it is relatively easy to understand these concepts at a high level, the underlying foundations that make such systems work is rarely discussed. Therefore, the goal of this course is to introduce topics from operating systems, networks, and software architecture that are directly applicable to research in computational science and scientific computing. Topics include programming with threads, interprocess communication and message passing, software components, I/O systems, and networking. *D. Beazley. Spring. Not offered 2003-04.*

29500. Digital Sound Modeling. *PQ: Knowledge of computer programming or consent of instructor.* This course studies 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, such as music or 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 to both concentrators and nonconcentrators. Students are required to submit the College Reading and Research Course Form.* 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 the departmental counselor. Not all 30000-level courses listed here will be offered each year, and other 30000-level courses may be offered that are not listed. For details, see www.cs.uchicago.edu.

31000. Foundations of Computer Science. *PQ: Consent of instructor.* An upper-level survey of computability and complexity theory. *J. Simon. Autumn.*

31900. Lambda Calculus. *PQ: Knowledge of algebra or equivalent level of math.* The lambda calculus is a formal system for studying the definitions of functions, independently of the values on which they operate. It provides a common foundation for logical formalism and meaning, number theory, computability, meaning and compiling in programming languages, nondeterministic and parallel computing. This course covers the crucial properties of the lambda calculus and its variant the combinator calculus, emphasizing 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. Autumn. Not offered 2003-04.*

32900-33900. High-Performance Computing on the Internet I, II. *PQ: Consent of instructor.* This course teaches students how to create high-performance computations that span computers connected by local and wide-area networks. Examples of such computations are “smart instruments” that use remote computers to enhance instrument data, “networked parallel computers” that link distributed resources to solve hard computational problems, and “networked virtual spaces” that link distributed computers and graphics resources. High-performance Internet computing introduces demanding performance requirements in addition to the usual concerns that arise in distributed systems. *I. Foster. Winter, Spring.*

33100. Advanced Operating Systems. *PQ: CMSC 23000/33000 and consent of instructor.* This course covers advanced topics in operating systems and systems research. Possible topics include, but are not limited to, parallel computing and multiprocessing, threads, message passing, networking, distributed systems, linkers, loaders, dynamic loading, debuggers, garbage collection, software components, file systems, and security. *D. Beazley. Autumn. Not offered 2003-04.*

33501. Topics in Databases. *PQ: Consent of instructor.* A selection of advanced topic in database systems. *S. Nestorov. Autumn.*

33510. Data Mining. *PQ: CMSC 15200 or 11700, and 27100; or consent of instructor.* Data mining, an emerging field at the intersection of machine learning, statistics, and databases, is broadly defined as finding novel and interesting patterns in large amounts of data. In this research-oriented course, we survey data-mining techniques and applications, emphasizing the database perspective. Major themes include association rules, web search and mining, clustering, and sequence and stream mining. The course involves an independent data-mining project. *S. Nestorov. Spring.*

34000. Scientific Parallel Computing. *PQ: Consent of instructor required; experience in scientific computing helpful.* The use of multiple processors cooperating to solve a common task. We study issues related to this general problem in the areas of computer architecture, performance analysis, prediction and measurement, programming languages, and algorithms for large-scale computation. The course involves programming at least one parallel computer. Possibilities include one of the clusters of workstations connected by high-speed networks currently at the University of Chicago. We focus on the state of the art in parallel algorithms for scientific computing. Specific topics are chosen based on student interest. General principles of parallel computing are emphasized. *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.*

34700. Scalable Internet Services. *PQ: Consent of instructor.* The demands and opportunities of the World Wide Web present challenges for operating and distributed systems research in wide-area, Internet-scale systems. This class surveys current research in this area, including work in wide-area caching, prefetching, replication, naming, distributed computation, scalable servers, security, and communication protocols. A primary goal is to provide the background necessary for doing research on these topics. We read and evaluate research papers selected from the literature. In addition to lectures, students are asked to evaluate the papers as a basis for discussion. Students enrolled for full credit also do a class project in small groups. *I. Foster. Autumn. Not offered 2003-04.*

34701. Advanced Internet Services. *PQ: Consent of instructor. I. Foster. Spring.*

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. *G. Levow. Spring.*

35400. Machine Learning. *PQ: CMSC 25000/35000 or consent of instructor.* This course is an introduction to the theory and practice of machine learning. The course emphasizes statistical approaches to the

problem. Topics covered 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; 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. Autumn. Not offered 2003-04.*

36500. Algorithms in Finite Groups. (=MATH 37500) *PQ: Linear algebra, finite fields, and a first course in group theory (Jordan-Holder and Sylow theorems). No prior knowledge of algorithms is required, and there will be no programming problems.* We consider the asymptotic complexity of some of the basic problems of computational group theory, but even students who don't care about computation will enjoy the course. The course will demonstrate the relevance of a delightful mix of mathematical techniques, ranging from lovely 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. *L. Babai. Spring.*

37000. Algorithms. *PQ: CMSC 27200 or consent of instructor.* 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.* For many optimization problems, it is impossible (or NP-hard) to design an algorithm that finds an optimal solution fast. It is interesting and important to study approximation algorithms that work faster, at the cost that we find only a good solution, not necessarily an optimal one. Sometimes we are also restricted in our access to the input, namely we have to react to partial input (typically first few requests of a request sequence) without knowledge of the complete input; thus we are building a solution step by step. Such algorithms are called on-line. The subject of the course is a theoretical study of these two classes of algorithms, illustrated on a number of optimization and combinatorial problems. The course includes recent results and thus provides an introduction to current research problems. *Autumn.*

37101. Markov Chain Monte Carlo Methods. *PQ: Consent of instructor.*

This course covers the Markov chain Monte Carlo method for enumeration and sampling problems. Analytic tools for the asymptotic analysis of Markov chain, including probabilistic, combinatorial and geometric tools are also discussed. We study these techniques in the context of prominent results in the field: estimating the volume of a convex body and computing the permanent of a matrix. Other topics include Markov chains, complexity class #P, exact sampling techniques, path coupling, stopping times, and connections to statistical physics. *E. Vigoda. 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. *L. Babai. Spring.*

37300. Parallel Algorithms. *PQ: CMSC 27200 or consent of instructor.* This course discusses models of parallel computation: shared memory and networks. Topics include basic algorithmic techniques (e.g., parallel prefix computation, list ranking, tree-contraction routing problems, complexity classes, completeness results) and randomized parallel algorithms. *J. Simon. Winter.*

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, as against purely existential techniques based on counting. *K. Mulmuley. Spring.*

37701. Topics in Bioinformatics. *PQ: Consent of instructor.* Current topics in bioinformatics are covered. *R. Scott. Spring.*

37720. Computational Systems Biology. *PQ: Consent of instructor.* Introductory concepts of systems biology, computational methods for analysis, reconstruction, visualization, modeling and simulation of complex cellular networks including biochemical pathways for metabolism, regulation and signaling. Students have the opportunity to 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. A particular focus is furthering our understanding of the computer science challenges in the engineering of prokaryotic organisms. Requirements include written assignments, programming assignments, and a final course project. *R. Stevens. Autumn.*

37800. Matrix Computation. (=STAT 30700) *PQ: Elementary programming experience and consent of instructor.* This course covers fundamental algorithms for the solution of linear equations, the decomposition of matrices, least squares, and finite-dimensional eigenvalue problems. We also discuss optimization problems and basic principles of Fourier and wavelet transformation. We investigate the mathematical foundations of algorithms and implement them in C++. *Y. Amit. Winter.*

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, including Turing machines and Kleene schemata, enumeration and s-m-n theorems, the recursion theorem, classification of unsolvable problems, and 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. *R. Soare. Winter, Spring.*

38200. Distributed Algorithms. *PQ: CMSC 27100 or consent of instructor.* This course studies algorithmic problems in distributed systems. Topics include models of distributed systems, problems of contention and cooperation among processes, distributed consensus and agreement, and leader election and clock synchronization. Also discussed are static and dynamic algorithms, fault tolerance, and uses of randomization. *J. Simon. Spring.*

38300. Numerical Solutions to Partial Differential Equations. *PQ: Consent of instructor.* This course covers the basic mathematical theory behind numerical solution of partial differential equations. The course investigates the convergence properties of finite element, finite difference and other discretization methods for solving partial differential equations. A brief introduction to Sobolev spaces and polynomial approximation theory is given. Special emphasis on error estimators, adaptivity and optimal-order solvers for linear systems arising from PDEs. Special topics include (from time to time) 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, such as primitive recursive functions, (general) recursive functions, and Turing machines; and their equivalence, the Church-Turing Thesis, unsolvable problems, diagonalization, and properties of computably enumerable (c.e.) sets. Part two deals with Kolmogorov complexity (resource bounded complexity) that studies the quantity of information in individual objects and uses the book by Li and Vitanyi. The third part covers functions computable with special bounds on time and space of the Turing machine, such as polynomial time computability, the classes P and NP, nondeterministic Turing machines, NP-complete problems, polynomial time hierarchy, and P-space complete problems. *Autumn.*

38600. Complexity Theory A. *PQ: Consent of instructor.* Topics are covered in computational complexity theory with an emphasis on machine-based complexity classes. *This course is offered in alternate years. Spring.*

38700. Complexity Theory B. *PQ: Consent of instructor.* Topics are covered in computational complexity theory with an emphasis on combinatorial problems in complexity. *This course is offered in alternate years. Winter.*

39000. Computational Geometry. *PQ: Consent of instructor.* This is a

seminar on topics in computational geometry. *K. Mulmuley. Autumn.*

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