Computer Science

Department Website: https://www.cs.uchicago.edu

Program of Study

The computer science program offers BA and BS degrees. Students who earn the BA degree study a breadth of foundational computer science topics, while students who earn the BS degree study these foundations in greater depth. Both degrees prepare students for various careers, including graduate study in computer science or a career in industry. The department also offers a minor, as well as combined BA/MS and BS/MS degrees.

In addition to the information below, please see the Undergraduate Program Overview (https://cs.uchicago.edu/academics/undergraduate/program-overview/) page as well as course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu/) for the most up-to-date information about courses and course scheduling, as well as the learning goals that motivate the major requirements in our undergraduate program.

Where to Start

Computer Science offers an introductory sequence for students interested in further study in computer science:

- CMSC 14100 Introduction to Computer Science I
- CMSC 14200 Introduction to Computer Science II
- CMSC 14300 Systems Programming I
- CMSC 14400 Systems Programming II

Students with no prior experience in computer science should plan to start the sequence at the beginning in CMSC 14100. Students with prior experience should plan to take the placement exam(s) (described below) to identify the appropriate place to start the sequence.

Note that, although these four courses form a sequence, not all CMSC 200-level courses require all four courses as prerequisites.

Students who are interested in data science should consider starting with DATA 11800 Introduction to Data Science I.

Students who are interested in the visual arts or design should consider CMSC 11111 Creative Coding.

Placement

Programming Placement Exams

Students with prior experience may place out of one or more of the introductory courses by successfully completing placement exam(s). The College and the Department of Computer Science offer three programming placement exams to help determine the correct starting point:

- The “CS 1 exam” which tests material covered in CMSC 14100
- The “CS 2 exam” which tests material covered in CMSC 14200
- The “CS 3 exam” which tests material covered in CMSC 14300

The placement exams will be offered in Summer Quarter, possibly with additional offerings in Autumn Quarter. Each of these exams will be conducted online. Students may take a particular exam at most once in an academic year. See the Program Overview (https://cs.uchicago.edu/academics/undergraduate/program-overview/) page for more information about the schedule for placement exams.

Solely based on the CS 1 exam, students may be placed into CMSC 14100 or CMSC 14200.

Students who place into CMSC 14200 will be invited to take the CS 2 exam. Solely based on the CS 2 exam, students may be placed into CMSC 14300.

Students who place into CMSC 14300 will be invited to take the CS 3 exam. Solely based on the CS 3 exam, students may be placed into CMSC 14400.

Exam Credit

Students who place into CMSC 14200 will receive credit for CMSC 14100 upon successfully completing CMSC 14200.

Students who place into CMSC 14300 will receive credit for CMSC 14100 and CMSC 14200 upon successfully completing CMSC 14300.

Students who place into CMSC 14400 will receive credit for CMSC 14100 and CMSC 14200 (but not CMSC 14300) upon successfully completing CMSC 14400.

Summary of Requirements for Computer Science Majors

General Education Requirement in the Mathematical Sciences

Both the BA and BS in computer science require the general education requirement in the mathematical sciences to be fulfilled by an approved two-quarter Calculus Sequence (http://collegecatalog.uchicago.edu/thecollege/mathematicalsciencescore/#calculussequences).

General Education Requirement in the Physical Sciences

To earn a BA in computer science, any sequence or pair of courses approved by the Physical Sciences Collegiate Division (http://collegecatalog.uchicago.edu/thecollege/physicalsciences/) may be used to complete the general education requirement in the physical sciences.

To earn a BS in computer science, the general education requirement in the physical sciences must be satisfied by completing a two-quarter sequence chosen from the General Education Sequences for Science Majors (http://collegecatalog.uchicago.edu/thecollege/physicalsciences/#generaleducationsequencesforsciencemajors). Students are encouraged, but not required, to fulfill this requirement with a physics sequence.

BA Requirements

BA students take at least 14 computer science courses as described below.

GENERAL EDUCATION	
Any two-quarter calculus sequence that fulfills the mathematical sciences requirement [*]	200
Any sequence or pair of courses that fulfills the physical sciences requirement	200
Total Units	400

MAJOR	
Introductory Sequence (see below)	400
Theory Requirement (two courses from the list below)	200
Eight electives, spanning at least five curriculum areas, numbered CMSC 20000 or above [§]	800
Total Units	1400

(see footnotes below)

BS Requirements

BA students take at least 15 computer science courses as described below.

GENERAL EDUCATION	
Any two-quarter calculus sequence that fulfills the mathematical sciences requirement [*]	200
Any two-quarter sequence that fulfills the physical sciences requirement for science majors	200
Total Units	400

MAJOR	
Introductory Sequence (see below)	400
Systems Requirement (one course from the list below)	100
Theory Requirement (three courses from the list below)	300
Machine Learning Requirement (one course from the list below)	100
Specialization Requirement (three courses from a designated curriculum area below)	300
Three electives numbered CMSC 20000 or above [§]	300
Total Units	1500

[*] Credit may be granted by examination.

[§] While a student may enroll in CMSC 29700 or CMSC 29900 for multiple quarters, only one instance of each may be counted toward the major. Courses numbered at the CMSC 300-level must be designated as PhD Core or PhD Elective in order to satisfy the electives requirement for the major. Because these courses can change on a quarterly basis, they are not listed in the catalog. Please see the PhD Course Designation section of course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu/). A small number of CMSC 200-level courses may be used as College electives but not as major electives. Courses that fall into this category will be marked as such.

Introductory Sequence

Students who major in computer science (either BA or BS) must complete the introductory sequence:

CMSC 14100	Introduction to Computer Science I	100
CMSC 14200	Introduction to Computer Science II	100

CMSC 14300	Systems Programming I	100
CMSC 14400	Systems Programming II	100

Students who place out of CMSC 14300 based on the CS 3 exam are required to take an additional approved elective, increasing the total number of electives required by one (from three to four for the BS, or from eight to nine for the BA).

Students who entered the College prior to Autumn Quarter 2022 and have already completed the recently retired introductory sequence (CMSC 12100 Computer Science with Applications I, CMSC 15100 Introduction to Computer Science I, CMSC 15200 Introduction to Computer Science II, and/or CMSC 16100 Honors Introduction to Computer Science I) should plan to follow the requirements in the Academic Year 2021-2022 Catalog (http://collegecatalog.uchicago.edu/archives/2021-2022/thecollege/). Students who were unable to complete the retired introductory sequence before it was retired should contact the Director of Undergraduate Studies for Computer Science for guidance.

Systems Requirement

BS students are required to take one of the following systems programming courses.

One of the following:		100
CMSC 22200	Computer Architecture *	
CMSC 22240	Computer Architecture for Scientists *	
CMSC 22600	Compilers for Computer Languages	
CMSC 23000	Operating Systems	
CMSC 23320	Foundations of Computer Networks	
CMSC 23500	Introduction to Database Systems	

* Students may take either CMSC 22200 or CMSC 22240 for major or minor requirements, but not both.

Theory Requirement

BA students must choose two courses from the following (one course each from areas A and B). BS students must choose three courses from the following (one course each from areas A, B, and C).

Area A		100
CMSC 27100	Discrete Mathematics	
CMSC 27130	Honors Discrete Mathematics	
Area B		100
CMSC 27200	Theory of Algorithms	
CMSC 27230	Honors Theory of Algorithms	
Area C		0-100
CMSC 27410	Honors Combinatorics	
CMSC 27500	Graph Theory	
CMSC 27502	Advanced Algorithms	
CMSC 27530	Honors Graph Theory	
CMSC 28000	Introduction to Formal Languages	
CMSC 28100	Introduction to Complexity Theory	
CMSC 28130	Honors Introduction to Complexity Theory	
CMSC 28400	Introduction to Cryptography	
MATH 27700	Mathematical Logic I	
MATH 27800	Mathematical Logic II	

Students can petition to count graduate theory courses for the Theory C requirement. We strongly encourage all computer science majors to complete their theory courses by the end of their third year.

Discrete Mathematics Placement Exam

Students who have prior knowledge of the topics covered in CMSC 27100 Discrete Mathematics may optionally take a Discrete Mathematics Placement Exam, which is offered a small number of times each year.

Students who pass the Discrete Mathematics Placement Exam have the option to replace the required Theory A course with an *additional* Theory C course. For example, a BA student can satisfy the Theory AB requirement by passing the Discrete Mathematics Placement Exam and taking one Theory B course and one Theory C course. We will refer to this as the "Theory BC" option. Similarly, a BS student can satisfy the Theory ABC requirement by passing the Discrete Mathematics Placement Exam and taking one Theory B course and two Theory C courses. We refer to this as the "Theory BCC" option.

In addition, students who pass the Discrete Mathematics Placement Exam may be given priority for registration in CMSC 27130 Honors Discrete Mathematics, as an alternative to pursuing the Theory BCC option.

More information about the structure and schedule of this placement exam can be found on the Undergraduate Program Overview (https://cs.uchicago.edu/academics/undergraduate/program-overview/) page.

Machine Learning Requirement

BS students are required to take one of the following machine learning courses.

One of the following:		100
CMSC 25300	Mathematical Foundations of Machine Learning	
CMSC 25400	Machine Learning	
TTIC 31020	Introduction to Machine Learning	

Alternatively, this requirement may be fulfilled by taking a more advanced course from the Machine Learning curriculum area, described below.

Electives and Specializations

In addition to the Introductory and Theory AB requirements, BA students must complete eight additional elective computer science courses at the CMSC 200-level or above. These courses must span at least five curriculum areas as described below.

In addition to the Introductory, Systems, Theory ABC, and Machine Learning requirements, BS students must complete one three-course specialization as described below, plus three additional elective computer science courses numbered 20000 or above.

Students may enroll in CMSC 29700 Reading and Research in Computer Science and CMSC 29900 Bachelor's Thesis for multiple quarters, but only one of each may be counted as a major elective.

Courses numbered at the CMSC 300-level must be designated as PhD Core or PhD Elective in order to satisfy the electives requirement for the major. Because these courses can change on a quarterly basis, they are not listed in the catalog. Please see the PhD Course Designation section of course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu).

A small number of CMSC 200-level courses may be used as College electives but not as major electives. Courses that fall into this category will be marked as such.

Curriculum Areas

CMSC 200-level courses are organized into the following curriculum areas:

- Computer Security and Privacy *
- Computer Systems *
- Data Science *
- Human Computer Interaction *
- Machine Learning *
- Programming Languages *
- Quantum Computing
- Robotics
- Software Engineering
- Theory *
- Visual Computing
- Miscellaneous

Each course may be included in multiple areas. The areas marked with an asterisk (*) are available as specializations for BS students. Please refer to course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu/) for the most up-to-date list of courses and their associated curriculum areas. Students may petition to have alternate courses count towards their breadth or specialization requirements.

BA Breadth Requirement

The eight elective courses required for the BA must span at least five curriculum areas. For the purposes of the breadth requirement, a single course cannot be counted towards multiple curriculum areas. Please note that a course that is counted as an elective may not also be counted towards the Theory AB requirement. There is one exception to this restriction: If a student satisfies the Theory AB requirement via the Theory BC option, then the

Theory C course also counts toward the Theory curriculum area. In other words, the eight additional electives must span at least *four* areas other than Theory.

BS Specialization Requirement

BS students are required to complete one three-course specialization. A specialization requires choosing three courses from one of the following designated curriculum areas. Please note that a course that is counted towards a specialization may not also be counted towards the Systems, Theory ABC, or Machine Learning requirements. There is one exception to this restriction: If a student satisfies the Theory ABC requirement via the Theory BCC option, then one Theory C course can be double-counted towards the Theory ABC requirement and a Theory specialization. (This means that, in both the Theory ABC and Theory BCC options, a student specializing in Theory will take four Theory C courses.)

The following specializations are currently available. The graduate versions of the courses below can be substituted for their undergraduate counterparts.

- **Computer Security and Privacy**
  - CMSC 23200 Introduction to Computer Security *(required for this specialization)*
  - CMSC 23206 Security, Privacy, and Consumer Protection
  - CMSC 23210 Usable Security and Privacy
  - CMSC 23218 Surveillance Aesthetics: Provocations About Privacy and Security in the Digital Age
  - CMSC 23260 Internet Censorship and Online Speech
  - CMSC 25800 Adversarial Machine Learning
  - CMSC 25910 Engineering for Ethics, Privacy, and Fairness in Computer Systems
  - CMSC 28400 Introduction to Cryptography
  - CMSC 29900 Bachelor's Thesis as approved for this specialization
- **Computer Systems**
  - CMSC 22200 Computer Architecture
  - CMSC 22240 Computer Architecture for Scientists
  - CMSC 23000 Operating Systems
  - CMSC 23010 Parallel Computing
  - CMSC 23310 Advanced Distributed Systems
  - CMSC 23320 Foundations of Computer Networks
  - CMSC 23500 Introduction to Database Systems
  - CMSC 23530 Advanced Database Systems
  - CMSC 25422 Machine Learning for Computer Systems
  - CMSC 29900 Bachelor's Thesis as approved for this specialization
- **Data Science**
  - CMSC 21800 Data Science for Computer Scientists *(required for this specialization)*
  - CMSC 23900 Data Visualization
  - CMSC 25300 Mathematical Foundations of Machine Learning
  - CMSC 25400 Machine Learning
  - STAT 37601 Machine Learning and Large-Scale Data Analysis
  - CMSC 29900 Bachelor's Thesis as approved for this specialization
- **Human Computer Interaction**
  - CMSC 20300 Introduction to Human-Computer Interaction *(either CMSC 20300 or CMSC 20310 is required for this specialization)*
  - CMSC 20310 Introduction to Designing Interaction *(either CMSC 20300 or CMSC 20310 is required for this specialization)*

- CMSC 20370 Inclusive Technology: Designing for Underserved and Marginalized Populations
- CMSC 20380 Actuated User Interfaces and Technology
- CMSC 20630 Human-Robot Interaction: Research and Practice
- CMSC 23210 Usable Security and Privacy
-
- CMSC 23218 Surveillance Aesthetics: Provocations About Privacy and Security in the Digital Age
-
- CMSC 23240 Emergent Interface Technologies
- CMSC 23400 Mobile Computing
- CMSC 23900 Data Visualization
- CMSC 29900 Bachelor's Thesis as approved for this specialization

- **Machine Learning**
  - CMSC 25040 Introduction to Computer Vision
  - CMSC 25300 Mathematical Foundations of Machine Learning
  - CMSC 25400 Machine Learning
  - CMSC 25460 Introduction to Optimization
  - CMSC 25500 Introduction to Neural Networks
  - CMSC 25800 Adversarial Machine Learning
  - STAT 37601 Machine Learning and Large-Scale Data Analysis
  - Some TTIC courses are approved for this specialization. See course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu/).
  - CMSC 29900 Bachelor's Thesis as approved for this specialization
- **Programming Languages**
  - CMSC 22100 Programming Languages
  - CMSC 22300 Functional Programming
  - CMSC 22400 Programming Proofs
  - CMSC 22500 Type Theory
  - CMSC 22600 Compilers for Computer Languages
  - CMSC 29900 Bachelor's Thesis as approved for this specialization
- **Theory**
  - CMSC 27410 Honors Combinatorics
  - CMSC 27500 Graph Theory
  - CMSC 27502 Advanced Algorithms
  - CMSC 27530 Honors Graph Theory
  - CMSC 28000 Introduction to Formal Languages
  - CMSC 28100 Introduction to Complexity Theory
  - CMSC 28130 Honors Introduction to Complexity Theory
  - CMSC 28400 Introduction to Cryptography
  - MATH 27700 Mathematical Logic I
  - MATH 27800 Mathematical Logic II
  - CMSC 29900 Bachelor's Thesis as approved for this specialization

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 counted towards the major. Any CMSC 200-level course taken as an elective beyond requirements for the major may, with consent of the instructor, be taken for P/F grading. Courses fulfilling general education requirements must be taken for quality grades.

Non-majors may take courses either for quality grades or, subject to College regulations and with consent of the instructor, for P/F grading. A Pass grade is given only for work of C- quality or higher.

Honors

Students can 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 courses from an approved list of graduate and undergraduate courses. The list of approved courses, which draws from 300-level PhD Core courses and PhD Elective courses, and from 200-level "honors" courses, can be found at course-info.cs.uchicago.edu (https://course-info.cs.uchicago.edu/). The three courses counted towards the honors requirement must be courses counting towards the major requirements. At most one of these three courses may be taken during the graduation quarter, while the other two must be completed at least one quarter prior to graduation.

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 Bachelor's Thesis. This thesis must be based on an approved research project that is directed by a faculty member and approved by the Bx Thesis Program Director. See the Bx Thesis Program (https://cs.uchicago.edu/academics/undergraduate/bx-thesis-program/) page for more information.

Computer Science Minor

The Department of Computer Science offers a seven-course minor. Courses in the minor must be taken for quality grades, with a grade of C– or higher in each course.

Summary of Requirements

CMSC 14100	Introduction to Computer Science I	100
CMSC 14200	Introduction to Computer Science II	100
CMSC 14300	Systems Programming I *	100
CMSC 14400	Systems Programming II *	100
Three electives numbered CMSC 20000 or above *		300
Total Units		700

* Students may replace CMSC 14300 and/or CMSC 14400 with CMSC 200-level electives (for which the prerequisites have been met).

Upper-Level Courses

The computer science minor must include at least three courses chosen from CMSC 200-level courses and above. A 200-level course must replace each 100-level course in the list above that was used to meet general education requirements or the requirements of a major. A small number of CMSC 200-level courses may not be used for minor credit. Courses that fall into this category will be marked as such.

Additional Requirements

As per College policy (http://collegecatalog.uchicago.edu/thecollege/minors/), 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. Prospective minors should arrange to meet the Computer Science Minor Advisor no later than May 1 of their third year. The Minor Advisor must approve the student's Consent to Complete a Minor Program (https://humanities-web.s3.us-east-2.amazonaws.com/college-prod/s3fs-public/documents/Consent_Minor_Program.pdf) form, and the student must submit that form to the student's College adviser by the end of Spring Quarter of the student's third year.

Joint Bx/MS Program

Outstanding undergraduates may apply to complete an MS in computer science along with a BA or BS (generalized to "Bx") during their four years at the College. Students must be admitted to the joint MS program. There are three different paths to a Bx/MS (https://www.cs.uchicago.edu/undergraduate/ba-ms-or-bs-ms-program/): a research-oriented program for computer science majors, a professionally oriented program for computer science majors, and a professionally oriented program for non-majors.

Graduate Courses

Graduate courses offered by the Department of Computer Science are open to College students with consent of the instructor. Not all graduate courses, however, may count as major or minor electives; only those that are designated as PhD Core or PhD Elective courses (not Seminars) for the graduate program can be used to satisfy

undergraduate requirements. See course-info.cs.uchicago.edu (http://course-info.cs.uchicago.edu/) for more information on the designations for graduate courses.

Schedule Changes

Please be aware that course information is subject to change, and the catalog does not necessarily reflect the most recent information. Students should consult course-info.cs.uchicago.edu (http://course-info.cs.uchicago.edu) for up-to-date information.

Contact Us

The contacts listed below work together on all administrative and scheduling aspects of the academic programs described above. Each of us is happy to hear from and help you, but depending on the specific nature of your question you may choose to contact a particular person as follows.

The Computer Science Director of Undergraduate Studies is responsible for approval of specific courses and petitions, and responds as needed to changing course offerings in our program and other programs.

The Computer Science Major and Minor Advisors are available to consult with students regarding questions about specific courses they are considering taking to meet the major and minor requirements, respectively.

Computer Science Courses

CMSC 11111. Creative Coding. 100 Units.
This course is an introduction to programming, using exercises in graphic design and digital art to motivate and employ basic tools of computation (such as variables, conditional logic, and procedural abstraction). We will write code in JavaScript and related languages, and we will work with a variety of digital media, including vector graphics, raster images, animations, and web applications.
Note(s): Students who have taken CMSC 11800, STAT 11800, CMSC 12100, CMSC 15100, or CMSC 16100 are not allowed to register for CMSC 11111.
Equivalent Course(s): MADD 21111

CMSC 12100-12200-12300. Computer Science with Applications I-II-III.
This three-quarter sequence teaches computational thinking and skills to students who are majoring in the sciences, mathematics, and economics, etc. Lectures cover topics in (1) programming, such as recursion, abstract data types, and processing data; (2) computer science, such as clustering methods, event-driven simulation, and theory of computation; and to a lesser extent (3) numerical computation, such as approximating functions and their derivatives and integrals, solving systems of linear equations, and simple Monte Carlo techniques.

CMSC 12100. Computer Science with Applications I. 100 Units.
This course is the first in a three-quarter sequence that teaches computational thinking and skills to students in the sciences, mathematics, economics, etc. The course will cover abstraction and decomposition, simple modeling, basic algorithms, and programming in Python. Applications from a wide variety of fields serve both as examples in lectures and as the basis for programming assignments. In recent offerings, students have written programs to simulate a model of housing segregation, determine the number of machines needed at a polling place, and analyze tweets from presidential debates. Students can find more information about this course at http://bit.ly/cmsc12100-aut-20.
Prerequisite(s): First year students are not allowed to register for CMSC 12100. Placement into MATH 15100 or completion of MATH 13100.
Note(s): First year students are not allowed to register for CMSC 12100. This course meets the general education requirement in the mathematical sciences.

CMSC 12200. Computer Science with Applications II. 100 Units.
This course is the second in a three-quarter sequence that teaches computational thinking and skills to students in the sciences, mathematics, economics, etc. Lectures cover topics in (1) data representation, (2) basics of relational databases, (3) shell scripting, (4) data analysis algorithms, such as clustering and decision trees, and (5) data structures, such as hash tables and heaps. Applications and datasets from a wide variety of fields serve both as examples in lectures and as the basis for programming assignments. In recent offerings, students have written a course search engine and a system to do speaker identification. Students will program in Python and do a quarter-long programming project.
Prerequisite(s): CMSC 12100
Note(s): This course meets the general education requirement in the mathematical sciences.

CMSC 12300. Computer Science with Applications III. 100 Units.
The course revolves around core ideas behind the management and computation of large volumes of data ("Big Data"). Topics include (1) Statistical methods for large data analysis, (2) Parallelism and concurrency, including models of parallelism and synchronization primitives, and (3) Distributed computing, including distributed architectures and the algorithms and techniques that enable these architectures to be fault-tolerant, reliable, and scalable. Students will continue to use Python, and will also learn C and distributed computing tools and platforms, including Amazon AWS and Hadoop. This course includes a project where students will have to formulate hypotheses about a large dataset, develop statistical models to test those hypotheses, implement a prototype that performs an initial exploration of the data, and a final system to process the entire dataset.

Prerequisite(s): CMSC 12200

CMSC 14100. Introduction to Computer Science I. 100 Units.
This course is the first of a pair of courses that are designed to introduce students to computer science and will help them build computational skills, such as abstraction and decomposition, and will cover basic algorithms and data structures. Students will also be introduced to the basics of programming in Python including designing and calling functions, designing and using classes and objects, writing recursive functions, and building and traversing recursive data structures. Students will also gain basic facility with the Linux command-line and version control.
Prerequisite(s): Placement into MATH 15100 or completion of MATH 13100, or instructor's consent, is a prerequisite for taking this course.

CMSC 14200. Introduction to Computer Science II. 100 Units.
This course is a direct continuation of CMSC 14100. Students will explore more advanced concepts in computer science and Python programming, with an emphasis on skills required to build complex software, such as object-oriented programming, advanced data structures, functions as first-class objects, testing, and debugging. The class will also introduce students to basic aspects of the software development lifecycle, with an emphasis on software design. Students will also gain further fluency in working with the Linux command-line, including some basic operating system concepts, as well as the use of version control systems for collaborative software development.
Prerequisite(s): CMSC 14100, or placement into CMSC 14200, is a prerequisite for taking this course.

CMSC 14300. Systems Programming I. 100 Units.
This course is the first in a pair of courses designed to teach students about systems programming. In this course, students will develop a deeper understanding of what a computer does when executing a program. In order to make the operations of the computer more transparent, students will study the C programming language, with special attention devoted to bit-level programming, pointers, allocation, file input and output, and memory layout. In the context of the C language, the course will revisit fundamental data structures by way of programming exercises, including strings, arrays, lists, trees, and dictionaries. Furthermore, the course will examine how memory is organized and structured in a modern machine. Students will gain basic fluency with debugging tools such as gdb and valgrind and build systems such as make.
Prerequisite(s): CMSC 14200, or placement into CMSC 14300, is a prerequisite for taking this course.

CMSC 14400. Systems Programming II. 100 Units.
This course is a direct continuation of CMSC 14300. This course covers the basics of computer systems from a programmer's perspective. Topics include machine language programming, exceptions, code optimization, performance measurement, system-level I/O, and concurrency. Students will gain further fluency with debugging tools and build systems.
Prerequisite(s): CMSC 14300, or placement into CMSC 14400, is a prerequisite for taking this course.

CMSC 15100-15200. Introduction to Computer Science I-II.
This sequence, which is recommended for all students planning to take more advanced courses in computer science, introduces computer science mostly through the study of programming in functional (Scheme) and imperative (C) programming languages. Topics include program design, control and data abstraction, recursion and induction, higher-order programming, types and polymorphism, time and space analysis, memory management, and data structures including lists, trees, and graphs. NOTE: Non-majors 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.

CMSC 15100. Introduction to Computer Science I. 100 Units.
This sequence, which is recommended for all students planning to take more advanced courses in computer science, introduces computer science mostly through the study of programming in functional (Scheme) and imperative (C) programming languages. Topics include program design, control and data abstraction, recursion and induction, higher-order programming, types and polymorphism, time and space analysis, memory management, and data structures including lists, trees, and graphs. NOTE: Non-majors 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.
Prerequisite(s): Placement into MATH 15100 or completion of MATH 13100.
Note(s): This course meets the general education requirement in the mathematical sciences. Non-majors 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.

CMSC 15200. Introduction to Computer Science II. 100 Units.
This sequence, which is recommended for all students planning to take more advanced courses in computer science, introduces computer science mostly through the study of programming in functional (Scheme) and imperative (C) programming languages. Topics include program design, control and data abstraction, recursion and induction, higher-order programming, types and polymorphism, time and space analysis, memory management, and data structures including lists, trees, and graphs. NOTE: Non-majors 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.
Prerequisite(s): CMSC 15100, CMSC 16100, CMSC 12100, or CMSC 10500.
Note(s): This course meets the general education requirement in the mathematical sciences. Non-majors 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.

CMSC 15400. Introduction to Computer Systems. 100 Units.
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.
Prerequisite(s): CMSC 12100, 15100, or 16100, and CMSC 15200, 16200, or 12300.

CMSC 16100-16200. Honors Introduction to Computer Science I-II.
Both courses in this sequence meet the general education requirement in the mathematical sciences; students who are majoring in Computer Science must use either CMSC 15200 or 16200 to meet requirements for the major.

CMSC 16100. Honors Introduction to Computer Science I. 100 Units.
Programming in a functional language (currently Haskell), including higher-order functions, type definition, algebraic data types, modules, parsing, I/O, and monads. Basic data structures, including lists, binary search trees, and tree balancing. Basic mathematics for reasoning about programs, including induction, inductive definition, propositional logic, and proofs.
Prerequisite(s): Placement into MATH 16100 or equivalent and programming experience, or by consent.
Note(s): This course meets the general education requirement in the mathematical sciences.

CMSC 16200. Honors Introduction to Computer Science II. 100 Units.
This course emphasizes the C Programming Language, but not in isolation. Instead, C is developed as a part of a larger programming toolkit that includes the shell (specifically ksh), shell programming, and standard Unix utilities (including awk). Nonshell scripting languages, in particular perl and python, are introduced, as well as interpreter (#!) files that use the command-line version of DrScheme. We cover various standard data structures, both abstractly, and in terms of concrete implementations-primarily in C, but also from time to time in other contexts like scheme and ksh. The course uses a team programming approach. There is a mixture of individual programming assignments that focus on current lecture material, together with team programming assignments that can be tackled using any Unix technology. Team projects are assessed based on correctness, elegance, and quality of documentation. We teach the "Unix way" of breaking a complex computational problem into smaller pieces, most or all of which can be solved using pre-existing, well-debugged, and documented components, and then composed in a variety of ways.
Prerequisite(s): CMSC 16100, or CMSC 15100 and by consent.
Note(s): Students who have taken CMSC 15100 may take 16200 with consent of instructor. This course meets the general education requirement in the mathematical sciences.

CMSC 19911. Introduction to Creative Coding. 100 Units.
This course is an introduction to programming, using exercises in graphic design and digital art to motivate and employ basic tools of computation (such as variables, conditional logic, and procedural abstraction). We will write code in JavaScript and related languages, and we will work with a variety of digital media, including vector graphics, raster images, animations, and web applications. This course is offered in the Pre-College Summer Immersion program.

CMSC 19925. Understanding AI: Challenges, Changes for How We Communicate. 100 Units.
Generative AI, large language models (LLMs) -- these buzzwords have been popping up in newsrooms, classrooms, and dinner tables. Questions about safety, environmental impacts, economic impacts and educational effects make people wonder how AI works, how it might change and augment the way we communicate and write, and what we should do about it. This class will give students the opportunity to: understand how AI powered applications for writing such as Gmail's Smart Compose feature and Grammarly's personalized revision suggestions work, get hands-on-experiences working with various AI-powered writing tools, speak with researchers and industry professionals to understand the design, impact, and motivation of these AI-based tools, consider historical events such as the development of the printing press and the internet to contextualize the effects of technology on human communication, and create a final project that analyzes and reflects on how technologies change the way that we communicate and write. Through these experiences, students will better understand the present AI landscape, with a focus on LLMs and their impacts on communication and writing, and form their own perspectives on the opportunities and risks of AI.
Terms Offered: Summer

CMSC 20300. Introduction to Human-Computer Interaction. 100 Units.
An introduction to the field of Human-Computer Interaction (HCI), with an emphasis in understanding, designing and programming user-facing software and hardware systems. This class covers the core concepts of HCI: affordances, mental models, selection techniques (pointing, touch, menus, text entry, widgets, etc), conducting user studies (psychophysics, basic statistics, etc), rapid prototyping (3D printing, etc), and the

fundamentals of 3D interfaces (optics for VR, AR, etc). We compliment the lectures with weekly programming assignments and two larger projects, in which we build/program/test user-facing interactive systems.
Prerequisite(s): CMSC 14200 or CMSC 15400 or CMSC 22000
Equivalent Course(s): CMSC 30300, MADD 25300

CMSC 20310. Introduction to Designing Interaction. 100 Units.
We are surrounded by interactive systems and designs, from the user interfaces for our computers and smartphones to home appliances, doors, and furniture. This class will introduce basic techniques, theories, and practices in designing interactive objects and systems. Specifically, it will introduce students to developing user experience-centric perspectives on critically reflecting on existing (and future) interactive designs. It also provides practices for students to ideate new design solutions and prototype them using design and engineering methods. Centered in interaction design, the class covers broad topics touching human-computer interaction, industrial design, user experience design, and communication design.
Prerequisite(s): CMSC 15200, CMSC 15400, CMSC 14200, CMSC 14300, or CMSC 14400.
Equivalent Course(s): MADD 20310

CMSC 20360. Creating Interactive Systems With User-Centered Design. 100 Units.
Creating technologies that fit into people's lives involves more than having technically sophisticated algorithms, systems, and infrastructure. It involves deeply understanding various community needs and using this understanding coupled with our knowledge of how people think and behave to design user-facing interfaces that can enhance and augment human capabilities. When dealing with different kinds of users, achieving these goals requires us to think through how different constraints such as costs, access to resources, and various cognitive and physical capabilities shape what socio-technical systems can best address a particular issue. This course leverages human-computer interaction and the tools, techniques, and principles that guide research on people to introduce you to the concepts of user centered design. You will learn about different user groups such as children, older adults, users facing financial constraints as well as those needing assistive technology and their particular needs. In addition, you will learn how to be mindful of working with different types of user populations and how to think creatively of technology solutions that enhance human capabilities while minimizing harm. You will also put your skills into practice in a quarter long group project involving the creation of an interactive system for one of the user populations we study.
Prerequisite(s): CMSC 14400 or CMSC 15400 or CMSC 12300 or CMSC 22000 or CMSC 20300 or CMSC 20310.
Note(s): Students who have taken 20370 may not enroll in 20360.
Equivalent Course(s): CMSC 30360

CMSC 20370. Inclusive Technology: Designing for Underserved and Marginalized Populations. 100 Units.
Creating technologies that are inclusive of people in marginalized communities involves more than having technically sophisticated algorithms, systems, and infrastructure. It involves deeply understanding various community needs and using this understanding coupled with our knowledge of how people think and behave to design user-facing interfaces that can enhance and augment human capabilities. When dealing with under-served and marginalized communities, achieving these goals requires us to think through how different constraints such as costs, access to resources, and various cognitive and physical capabilities shape what socio-technical systems can best address a particular issue. This course leverages human-computer interaction and the tools, techniques, and principles that guide research on people to introduce you to the concepts of inclusive technology design. You will learn about different underserved and marginalized communities such as children, the elderly, those needing assistive technology, and users in developing countries, and their particular needs. In addition, you will learn how to be mindful of working with populations that can easily be exploited and how to think creatively of inclusive technology solutions. You will also put your skills into practice in a semester long group project involving the creation of an interactive system for one of the user populations we study.
Prerequisite(s): CMSC 14400 or CMSC 15400 or CMSC 12300 or CMSC 22000 or CMSC 20300 or CMSC 20310
Equivalent Course(s): CMSC 30370, MADD 20370

CMSC 20380. Actuated User Interfaces and Technology. 100 Units.
The recent advancement in interactive technologies allows computer scientists, designers, and researchers to prototype and experiment with future user interfaces that can dynamically move and shape-change. This class offers hands-on experience in learning and employing actuated and shape-changing user interface technologies to build interactive user experiences. The class provides a range of basic engineering techniques to allow students to develop their own actuated user interface systems, including 3D mechanical design, digital fabrication (e.g. 3D Printing), electronics (Arduino microcontroller), and actuator control (utilizing different kinds of motors). Through multiple project-based assignments, students practice the acquired techniques to build interactive tangible experiences of their own.
Prerequisite(s): CMSC 20300 or CMSC 20310
Equivalent Course(s): MADD 20380, CMSC 30380

CMSC 20600. Introduction to Robotics. 100 Units.
The University of Chicago's CMSC 20600 Introduction to Robotics course gives students a hands-on introduction to robot programming covering topics including sensing in real-world environments, sensory-motor control, state estimation, localization, forward/inverse kinematics, vision, and reinforcement learning. This course is centered around 3 mini projects exploring central concepts to robot programming and 1 final project whose topic is chosen by the students. Each of these mini projects will involve students programming real, physical robots

interacting with the real world. The use of physical robots and real-world environments is essential in order for students to 1) see the result of their programs 'come to life' in a physical environment and 2) gain experience facing and overcoming the challenges of programming robots (e.g., sensor noise, edge cases due to environment variability, physical constraints of the robot and environment).
Prerequisite(s): CMSC 14200 or CMSC 15400
Equivalent Course(s): CMSC 30600

CMSC 20630. Human-Robot Interaction: Research and Practice. 100 Units.
Robots are increasingly common in our everyday spaces: tutoring elementary students, assisting human workers in manufacturing contexts, providing museum tours, interacting with families within their homes, and helping to care for the elderly. One critical factor to the success of these robots is their ability to effectively interact with people: human-robot interactions. This course focuses on the core concepts and cutting-edge research in the field of human-robot interaction (HRI), covering topics that include nonverbal robot behavior, verbal robot behavior, social dynamics, norms & ethics, collaboration & learning, group interactions, applications, and future challenges of HRI. In class meetings, students lead discussions about cutting-edge peer-reviewed research HRI publications. In weekly labs, students engage in hands-on activities to learn the essential skills of human-robot interaction research, including experiment design, robot programming, and data analysis techniques. Students also participate in a quarter-long collaborative research project, where they pursue an HRI research question that often involves conducting their own human-subjects research study where they recruit human subjects to interact with a robot.
Prerequisite(s): CMSC 14200 or CMSC 15400
Equivalent Course(s): CMSC 30630

CMSC 20900. Computers for Learning. 100 Units.
Over time, technology has occupied an increasing role in education, with mixed results. Massive Open Online Courses (MOOCs) were created to bring education to those without access to universities, yet most of the students who succeed in them are those who are already successful in the current educational model. This course focuses on one intersection of technology and learning: computer games. This course covers education theory, psychology (e.g., motivation, engagement), and game design so that students can design and build an educational learning application. Labs focus on developing expertise in technology, and readings supplement lecture discussions on the human components of education.
Prerequisite(s): CMSC 14300 or CMSC 15400 or CMSC 22000
Equivalent Course(s): MADD 20900, CMSC 30900

CMSC 21010. Mathematical Foundations. 100 Units.
This course is an introduction to formal tools and techniques which can be used to better understand linguistic phenomena. A major goal of this course is to enable students to formalize and evaluate theoretical claims.
Equivalent Course(s): LING 21010, LING 31010, CMSC 31010

CMSC 21400. Creative Machines and Innovative Instrumentation. 100 Units.
An understanding of the techniques, tricks, and traps of building creative machines and innovative instrumentation is essential for a range of fields from the physical sciences to the arts. In this hands-on, practical course, you will design and build functional devices as a means to learn the systematic processes of engineering and fundamentals of design and construction. The kinds of things you will learn may include mechanical design and machining, computer-aided design, rapid prototyping, circuitry, electrical measurement methods, and other techniques for resolving real-world design problems. In collaboration with others, you will complete a mini-project and a final project, which will involve the design and fabrication of a functional scientific instrument. The course will be taught at an introductory level; no previous experience is expected. The iterative nature of the design process will require an appreciable amount of time outside of class for completing projects. The course is open to undergraduates in all majors (subject to the pre-requisites), as well as Master's and Ph.D. students.
Instructor(s): TBD (Autumn Quarter); John Carlstrom (Winter Quarter); Derek Buzasi (Spring Quarter) Terms Offered: Autumn Spring Winter
Prerequisite(s): PHYS 12200 or PHYS 13200 or PHYS 14200; or CMSC 12100 or CMSC 12200 or CMSC 12300; or consent of instructor.
Equivalent Course(s): PHYS 21400, ASTR 31400, ASTR 21400, PSMS 31400

CMSC 21800. Data Science for Computer Scientists. 100 Units.
Data-driven models are revolutionizing science and industry. This course covers computational methods for structuring and analyzing data to facilitate decision-making. We will cover algorithms for transforming and matching data; hypothesis testing and statistical validation; and bias and error in real-world datasets. A core theme of the course is "generalization"; ensuring that the insights gleaned from data are predictive of future phenomena. The course will include bi-weekly programming assignments, a midterm examination, and a final.
Prerequisite(s): CMSC 14300 or CMSC 15400 or CMSC 22000

CMSC 22000. Introduction to Software Development. 100 Units.
Besides covering a number of topics in software engineering, with an emphasis on software design, this course focuses on imparting a number of skills and industry best practices that are valuable in the development of large software projects, such as source control techniques and workflows, issue tracking, code reviews, testing, continuous integration, working with existing codebases, integrating APIs and frameworks, generating documentation, deployment, and logging and monitoring. The course also emphasizes the importance of

collaboration in real-world software development, including interpersonal collaboration and team management. The course will be organized primarily around the development of a class-wide software project, with students organized into teams. Collaboration both within and across teams will be essential to the success of the project.
Prerequisite(s): CMSC 14200

CMSC 22001. Software Construction. 100 Units.
Large software systems are difficult to build. The course discusses both the empirical aspects of software engineering and the underlying theory. Topics will include, among others, software specifications, software design, software architecture, software testing, software reliability, and software maintenance. Students will be expected to actively participate in team projects in this course.
Prerequisite(s): CMSC 14400 or CMSC 15400.

CMSC 22010. Digital Fabrication. 100 Units.
Digital fabrication involves translation of a digital design into a physical object. While digital fabrication has been around for decades, only now has it become possible for individuals to take advantage of this technology through low cost 3D printers and open source tools for 3D design and modeling. In this course we will cover the foundations of 3D object design including computational geometry, the type of models that can and can't be fabricated, the uses and applications of digital fabrication, the algorithms, methods and tools for conversion of 3D models to representations that can be directly manufactured using computer controlled machines, the concepts and technology used in additive manufacturing (aka 3D printing) and the research and practical challenges of developing self-replicating machines. We will have several 3D printers available for use during the class and students will design and fabricate several parts during the course.
Prerequisite(s): CMSC 15400 and some experience with 3D modeling concepts.

CMSC 22100. Programming Languages. 100 Units.
This course is an introduction to scientific programming language design, whereby design choices are made according to rigorous and well-founded lines of reasoning. The curriculum includes the lambda calculus, type systems, formal semantics, logic and proof, and, time permitting, a light introduction to machine assisted formal reasoning. Practical exercises in writing language transformers reinforce the the theory. While this course is not a survey of different programming languages, we do examine the design decisions embodied by various popular languages in light of their underlying formal systems.
Prerequisite(s): CMSC 14300 or CMSC 14400 or CMSC 15400

CMSC 22200. Computer Architecture. 100 Units.
Computing systems have advanced rapidly and transformed every aspect of our lives for the last few decades, and innovations in computer architecture is a key enabler. Residing in the middle of the system design layers, computer architecture interacts with both the software stack (e.g., operating systems and applications) and hardware technologies (e.g., logic gates, interconnects, and memories) to enable efficient computing with unprecedented capabilities. In this course, students will learn the fundamental principles, techniques, and tradeoffs in designing the hardware/software interface and hardware components to create a computing system that meets functional, performance, energy, cost, and other specific goals. Example topics include instruction set architecture (ISA), pipelining, memory hierarchies, input/output, and multi-core designs. In addition, we will discuss advanced topics regarding recent research and trends. This course also includes hands-on labs, where students will enhance their learning by implementing a modern microprocessor in a C simulator.
Prerequisite(s): CMSC 14400 or CMSC 15400

CMSC 22240. Computer Architecture for Scientists. 100 Units.
Designed to provide an understanding of the key scientific ideas that underpin the extraordinary capabilities of today's computers, including speed (gigahertz), illusion of sequential order (relativity), dynamic locality (warping space), parallelism, keeping it cheap - and low-energy (e-field scaling), and of course their ability as universal information processing engines. These scientific "miracles" are robust, and provide a valuable longer-term understanding of computer capabilities, performance, and limits to the wealth of computer scientists practicing data science, software development, or machine learning. This course can be used towards fulfilling the Programming Languages and Systems requirement for the CS major.
Prerequisite(s): CMSC 14300 or CMSC 15400

CMSC 22300. Functional Programming. 100 Units.
Programming languages often conflate the definition of mathematical functions, which deterministically map inputs to outputs, and computations that effect changes, such as interacting with users and their machines. In this course, students will develop an enriched perspective about these two related but distinct mechanisms, by studying the statically-typed pure functional programming language Haskell. Topics include: algebraic datatypes, an elegant language for describing and manipulating domain-specific data; higher-order functions and type polymorphism, expressive mechanisms for abstracting programs; and a core set of type classes, with strong connections to category theory, that serve as a foundational and practical basis for mixing pure functions with stateful and interactive computations. In addition to small and medium sized programming assignments, the course includes a larger open-ended final project.
Prerequisite(s): CMSC 14300 or CMSC 14400 or CMSC 15200

CMSC 22400. Programming Proofs. 100 Units.
In this course, we will explore the use of proof assistants, computer programs that allow us to write, automate, and mechanically check proofs. These tools have two main uses. They allow us to prove properties of our programs, thereby guaranteeing that our code is free of software errors. They also allow us to formalize mathematics, stating and proving mathematical theorems in a manner that leaves no doubt as to their meaning or veracity. At the intersection of these two uses lies mechanized computer science, involving proofs about data structures, algorithms, programming languages and verification itself.
Prerequisite(s): (CMSC 27100 or CMSC 27130 or CMSC 37000 or CMSC 37110), and (CMSC 14100 or CMSC 15100 or CMSC 16100 or CMSC 22100 or CMSC 22300)
Note(s): This course can count toward the Programming Languages & Systems requirement for the CS Major.
Equivalent Course(s): CMSC 32400

CMSC 22500. Type Theory. 100 Units.
Church's λ-calculus, β-reduction, the Church-Rosser theorem. Simple type theory, strong normalization. The Barendregt cube of type theories. Dependent types. The Curry-Howard Isomorphism. Formal constructive mathematics.
Prerequisite(s): (CMSC 15100 or CMSC 16100 or CMSC 22100 or CMSC 22300) and (CMSC 27100 or CMSC 27130 or MATH 28130 or CMSC 37110 or CMSC 27700 or MATH 27700)

CMSC 22600. Compilers for Computer Languages. 100 Units.
This course covers principles of modern compiler design and implementation. Topics include lexical analysis, parsing, type checking, optimization, and code generation. This is a project oriented course in which students will construct a fully working compiler, using Standard ML as the implementation language.
Prerequisite(s): CMSC 22100 or CMSC 22300

CMSC 22880. Introduction to Quantum Computing. 100 Units.
This introduction to quantum computing will cover the key principles of quantum information science and how they relate to quantum computing as well as the notation and operations used in QIS. We will then take these building blocks and linear algebra principles to build up to several quantum algorithms and complete several quantum programs using a mainstream quantum programming language.
Prerequisite(s): CMSC 14400 or CMSC 15400
Note(s): Students who have completed MENG 26400 may not enroll in CMSC 22880.

CMSC 22900. Quantum Computer Systems. 100 Units.
This course will explore the design, optimization, and verification of the software and hardware involved in practical quantum computer systems. The course will provide an introduction to quantum computation and quantum technologies, as well as classical and quantum compiler techniques to optimize computations for technologies. Verification techniques to evaluate the correctness of quantum software and hardware will also be explored.
Prerequisite(s): CMSC 14400 or CMSC 15400, and CMSC 22880 or MENG 26400 or CMSC 38410 or MATH 38410 or CMSC 38420 or MATH 38420
Equivalent Course(s): CMSC 32900

CMSC 23000. Operating Systems. 100 Units.
This course provides an introduction to basic Operating System principles and concepts that form as fundamental building blocks for many modern systems from personal devices to Internet-scale services. Basic topics include processes, threads, concurrency, synchronization, memory management, virtual memory, segmentation, paging, caching, process and I/O scheduling, file systems, storage devices. The course will also cover special topics such as journaling/transactions, SSD, RAID, virtual machines, and data-center operating systems. The course project will revolve around the implementation of a mini x86 operating system kernel.
Prerequisite(s): CMSC 14400 or CMSC 15400

CMSC 23010. Parallel Computing. 100 Units.
This course provides an introduction to the concepts of parallel programming, with an emphasis on programming multicore processors. Topics include: Processes and threads, shared memory, message passing, direct-memory access (DMA), hardware mechanisms for parallel computing, synchronization and communication, patterns of parallel programming. The course will involve a substantial programming project implementing a parallel computations.
Prerequisite(s): CMSC 14400 or CMSC 15400 and (CMSC 22200, CMSC 22240, CMSC 23000, CMSC 23300, OR CMSC 23320)

CMSC 23200. Introduction to Computer Security. 100 Units.
This graduate course covers foundational research in computer and network security. The goal of this course is to provide students with a broad understanding of classical and modern security problems, the mental models and techniques commonly used in the field, and practical experience conducting security research. This class will consist of reading and discussing academic papers, as well as conducting and presenting an original research project. The course topics will span a wide cross-section of security research, ranging from hardware and systems security, to applied cryptography and machine learning security, to the economics and human aspects of cybercrime.
Prerequisite(s): CMSC 14400 or CMSC 15400

CMSC 23206. Security, Privacy, and Consumer Protection. 100 Units.
This course will cover the principles and practice of security, privacy, and consumer protection. Topics include: basic cryptography; physical, network, endpoint, and data security; privacy (including user surveillance and tracking); attacks and defenses; and relevant concepts in usable security. The course will place fundamental security and privacy concepts in the context of past and ongoing legal, regulatory, and policy developments, including: consumer privacy, censorship, platform content moderation, data breaches, net neutrality, government surveillance, election security, vulnerability discovery and disclosure, and the fairness and accountability of automated decision making, including machine learning systems. Students will learn both technical fundamentals and how to apply these concepts to public policy outputs and recommendations.
Instructor(s): Feamster, Nicholas
Prerequisite(s): CMSC 14300 or CMSC 15400 or equivalent, and instructor consent.
Note(s): Prerequisites: CMSC 14300 or CMSC 15400 or equivalent, or graduate student. Instructor consent required.
Equivalent Course(s): CMSC 30350, CAPP 30350

CMSC 23210. Usable Security and Privacy. 100 Units.
Regardless of how secure a system is in theory, failing to consider how humans actually use the system leads to disaster in practice. This course will examine how to design for security and privacy from a user-centered perspective by combining insights from computer systems, human-computer interaction (HCI), and public policy. We will introduce core security and privacy technologies, as well as HCI techniques for conducting robust user studies. Topics will include usable authentication, user-centered web security, anonymity software, privacy notices, security warnings, and data-driven privacy tools in domains ranging from social media to the Internet of Things. Students will complete weekly problem sets, as well as conduct novel research in a group capstone project. No prior experience in security, privacy, or HCI is required.
Prerequisite(s): CMSC 12300 or CMSC 14200 or CMSC 15400
Equivalent Course(s): CMSC 33210

CMSC 23218. Surveillance Aesthetics: Provocations About Privacy and Security in the Digital Age. 100 Units.
In the modern world, individuals' activities are tracked, surveilled, and computationally modeled to both beneficial and problematic ends. Jointly with the School of the Art Institute of Chicago (SAIC), this course will examine privacy and security issues at the intersection of the physical and digital worlds. Through both computer science and studio art, students will design algorithms, implement systems, and create interactive artworks that communicate, provoke, and reframe pervasive issues in modern privacy and security. The course will unpack and re-entangle computational connections and data-driven interactions between people, built space, sensors, structures, devices, and data. Synthesizing technology and aesthetics, we will communicate our findings to the broader public not only through academic avenues, but also via public art and media. The first phase of the course will involve prompts in which students design and program small-scale artworks in various contexts, including (1) data collected from web browsing; (2) mobility data; (3) data collected about consumers by major companies; and (4) raw sensor data. Students will receive detailed feedback on their work from computer scientists, artists, and curators at the Museum of Science & Industry (MSI). The course culminates in the production and presentation of a capstone interactive artwork by teams of computer scientists and artists; successful products may be considered for prototyping at the MSI.
Prerequisite(s): One of CMSC 23200, CMSC 23210, CMSC 25900, CMSC 28400, CMSC 33210, CMSC 33250, or CMSC 33251 recommended, but not required.
Note(s): Students interested in this class should complete this form to request permission to enroll: https://uchicago.co1.qualtrics.com/jfe/form/SV_5jPT8gRDXDKQ26a
Equivalent Course(s): CMSC 33218, MADD 23218

CMSC 23230. Engineering Interactive Electronics onto Printed Circuit Boards. 100 Units.
In this class we will engineer electronics onto Printed Circuit Boards (PCBs). We will focus on designing and laying out the circuit and PCB for our own custom-made I/O devices, such as wearable or haptic devices. In order for you to be successful in engineering a functional PCB, we will (1) review digital circuits and three microcontrollers (ATMEGA, NRF, SAMD); (2) use KICAD to build circuit schematics; (3) learn how to wire analog/digital sensors or actuators to our microcontroller, including SPI and I2C protocols; (4) use KICAD to build PCB schematics; (5) actually manufacture our designs; (6) receive in our hands our PCBs from factory; (7) finally, learn how to debug our custom-made PCBs.
Prerequisite(s): CMSC 14400 or CMSC 15400, and CMSC 23220 or CMSC 21400 or CMSC 20380 or PHYS 13300 or PHYS 14300 or PHYS 22700.
Equivalent Course(s): CMSC 33230

CMSC 23240. Emergent Interface Technologies. 100 Units.
In this class, we critically examine emergent technologies that might impact the future generations of computing interfaces, these include: physiological I/O (e.g., brain and muscle computer interfaces), tangible computing (giving shape and form to interfaces), wearable computing (I/O devices closer to the user's body), rendering new realities (e.g., virtual and augmented reality), haptics (giving computers the ability to generate touch and forces) and unusual auditory interfaces (e.g., silent speech and microphones as sensors). In this class you will: (1) learn about these new developments during the lectures, (2) read HCI papers and summarize these in short weekly assignments, and lastly, (3) start inventing the future of computing interfaces by proposing a new idea in the

form of a paper abstract, which you will present at the end of the semester and have it peer-reviewed in class by your classmates.
Prerequisite(s): CMSC 15400

CMSC 23260. Internet Censorship and Online Speech. 100 Units.
Information dissemination and online discourse on the Internet are subject to the algorithms and filters that operate on Internet infrastructure, from network firewalls to search engines. This course will explore the technologies that are used to control access to online speech and information, and cutting-edge technologies that can empower citizens in the face of these information controls. Students will learn about and experiment with technologies to control online discourse, ranging from firewalls that perform network traffic filtering to algorithms for content personalization and content moderation. We will also explore underlying technical trends, such as the increasing consolidation of Internet infrastructure and protocols, and the implications of consolidation for control over online discourse. Each course meeting will include a technical overview, reading discussion, and a hands-on laboratory activity.
Prerequisite(s): None
Equivalent Course(s): PARR 33260, CMSC 33260, MADD 23620

CMSC 23300. Networks and Distributed Systems. 100 Units.
This course focuses on the principles and techniques used in the development of networked and distributed software. Topics include programming with sockets; concurrent programming; data link layer (Ethernet, packet switching, etc.); internet and routing protocols (IP, IPv6, ARP, etc.); end-to-end protocols (UDP, TCP); and other commonly used network protocols and techniques. This is a project-oriented course in which students are required to develop software in C on a UNIX environment.
Prerequisite(s): CMSC 15400.

CMSC 23310. Advanced Distributed Systems. 100 Units.
In recent years, large distributed systems have taken a prominent role not just in scientific inquiry, but also in our daily lives. When we perform a search on Google, stream content from Netflix, place an order on Amazon, or catch up on the latest comings-and-goings on Facebook, our seemingly minute requests are processed by complex systems that sometimes include hundreds of thousands of computers, connected by both local and wide area networks. Recent papers in the field of Distributed Systems have described several solutions (such as MapReduce, BigTable, Dynamo, Cassandra, etc.) for managing large-scale data and computation. However, building and using these systems pose a number of more fundamental challenges: How do we keep the system operating correctly even when individual machines fail? How do we ensure that all the machines have a consistent view of the system's state? (And how do we ensure this in the presence of failures?) How can we determine the order of events in a system where we can't assume a single global clock? Many of these fundamental problems were identified and solved over the course of several decades, starting in the 1970s. To better appreciate the challenges of recent developments in the field of Distributed Systems, this course will guide students through seminal work in Distributed Systems from the 1970s, '80s, and '90s, leading up to a discussion of recent work in the field.
Prerequisite(s): CMSC 23300 with at least a B+, or by consent.

CMSC 23320. Foundations of Computer Networks. 100 Units.
This course focuses on the principles and techniques used in the development of networked and distributed software. Topics include programming with sockets; concurrent programming; data link layer (Ethernet, packet switching, etc.); internet and routing protocols (IP, IPv6, ARP, etc.); end-to-end protocols (UDP, TCP); and other commonly used network protocols and techniques. This is a project-oriented course in which students are required to develop software in C on a UNIX environment. This course can be used towards fulfilling the Programming Languages and Systems requirement for the CS major.
Prerequisite(s): CMSC 14400 or CMSC 15400
Note(s): This course can be used towards fulfilling the Programming Languages and Systems requirement for the CS major. Students who have taken CMSC 23300 may not take CMSC 23320.

CMSC 23360. Advanced Networks. 100 Units.
Advanced networks
Prerequisite(s): CMSC 23300 or CMSC 23320
Note(s): A more detailed course description should be available later.

CMSC 23400. Mobile Computing. 100 Units.
Mobile computing is pervasive and changing nearly every aspect of society. Sensing, actuation, and mediation capabilities of mobile devices are transforming all aspects of computing: uses, networking, interface, form, etc. This course explores new technologies driving mobile computing and their implications for systems and society. Current focus areas include new techniques to capture 3d models (depth sensors, stereo vision), drones that enable targeted, adaptive, focused sensing, and new 3d interactive applications (augmented reality, cyberphysical, and virtual reality). Labs expose students to software and hardware capabilities of mobile computing systems, and develop the capability to envision radical new applications for a large-scale course project.
Prerequisite(s): CMSC 14400 or CMSC 15400

CMSC 23500. Introduction to Database Systems. 100 Units.
This course is an introduction to database design and implementation. Topics include DBMS architecture, entity-relationship and relational models, relational algebra, concurrency control, recovery, indexing, physical data organization, and modern database systems. The lab section guides students through the implementation of a relational database management system, allowing students to see topics such as physical data organization and DBMS architecture in practice, and exercise general skills such as software systems development.
Prerequisite(s): CMSC 14400 or CMSC 15400
Equivalent Course(s): CMSC 33550

CMSC 23530. Advanced Database Systems. 100 Units.
This course focuses on advanced concepts of database systems topics and assumes foundational knowledge outlined in CMSC 23500. Topics will include distribute databases, materialized views, multi-dimensional indexes, cloud-native architectures, data versioning, and concurrency-control protocols.
Prerequisite(s): CMSC 23500

CMSC 23700. Introduction to Computer Graphics. 100 Units.
The course provides an introduction to computer graphics, covering fundamental concepts and techniques including rasterization, sampling, image/signal processing basics, convolutions, coordinate spaces, transformations, camera viewing, 3D transformations, ray tracing, 3D processing, parameterization, animation/deformations, and more. Assignments will be in Python.
Prerequisite(s): CMSC 14400 or CMSC 15400
Note(s): Prior experience with basic linear algebra (matrix algebra) is recommended.
Equivalent Course(s): CMSC 33700

CMSC 23710. Scientific Visualization. 100 Units.
Scientific visualization combines computer graphics, numerical methods, and mathematical models of the physical world to create a visual framework for understanding and solving scientific problems. The mathematical and algorithmic foundations of scientific visualization (for example, scalar, vector, and tensor fields) will be explained in the context of real-world data from scientific and biomedical domains. The course is also intended for students outside computer science who are experienced with programming and computing with scientific data. Programming projects will be in C and C++.
Prerequisite(s): CMSC 14300 or CMSC 15400
Equivalent Course(s): CMSC 33710

CMSC 23740. Introduction to Real-time Graphics. 100 Units.
This course provides an introduction to Computer Graphics with a focus on real-time rendering techniques, such as those found in computer-game engines. Topics include coordinate systems and transformations; geometric modeling; the programmable graphics pipeline; shadows; level-of detail optimizations; and other rendering techniques. The course covers both the theory and practice of computer graphics. The lectures, homework assignments and exams focus on algorithms, data structures, and mathematical foundations, while the lab sessions and programming projects deal with translating theory into practice.
Prerequisite(s): CMSC 14400 or CMSC 15400. Prior experience with linear algebra is recommended.

CMSC 23900. Data Visualization. 100 Units.
Data visualizations provide a visual setting in which to explore, understand, and explain datasets. This class describes mathematical and perceptual principles, methods, and applications of "data visualization" (as it is popularly understood to refer primarily to tabulated data). A range of data types and visual encodings will be presented and evaluated. Visualizations will be primarily web-based, using D3.js, and possibly other higher-level languages and libraries.
Prerequisite(s): CMSC 12200, CMSC 14200, CMSC 15200, OR CMSC 16200

CMSC 25025. Machine Learning and Large-Scale Data Analysis. 100 Units.
This course is an introduction to machine learning and the analysis of large data sets using distributed computation and storage infrastructure. Basic machine learning methodology and relevant statistical theory will be presented in lectures. Homework exercises will give students hands-on experience with the methods on different types of data. Methods include algorithms for clustering, binary classification, and hierarchical Bayesian modeling. Data types include images, archives of scientific articles, online ad clickthrough logs, and public records of the City of Chicago. Programming will be based on Python and R, but previous exposure to these languages is not assumed.
Terms Offered: Spring
Prerequisite(s): [CMSC 25300 or CMSC 35300 or STAT 27700 or TTIC 31020] and [STAT 24400 or STAT 24410 or STAT 24500 or STAT 24510]
Note(s): The prerequisites are under review and may change.
Equivalent Course(s): STAT 37601

CMSC 25040. Introduction to Computer Vision. 100 Units.
This course covers the fundamentals of digital image formation; image processing, detection and analysis of visual features; representation shape and recovery of 3D information from images and video; analysis of motion. We also study some prominent applications of modern computer vision such as face recognition and object

and scene classification. Our emphasis is on basic principles, mathematical models, and efficient algorithms established in modern computer vision.
Prerequisite(s): CMSC 25300, CMSC 25400, CMSC 25025, STAT 24300, STAT 24500, or TTIC 31020

CMSC 25300. Mathematical Foundations of Machine Learning. 100 Units.
This course is an introduction to the mathematical foundations of machine learning that focuses on matrix methods and features real-world applications ranging from classification and clustering to denoising and data analysis. Mathematical topics covered include linear equations, regression, regularization, the singular value decomposition, and iterative algorithms. Machine learning topics include classification and regression, support vector machines, kernel methods, clustering, matrix completion, neural networks, and deep learning. Students are expected to have taken calculus and have exposure to numerical computing (e.g. Matlab, Python, Julia, R).
Prerequisite(s): CMSC 11900 or CMSC 12200 or CMSC 14100 or CMSC 15200 or CMSC 16200
Note(s): Undergraduate students are not allowed to enroll in CMSC 35300.
Equivalent Course(s): CMSC 35300, STAT 27700

CMSC 25400. Machine Learning. 100 Units.
This course introduces the foundations of machine learning and provides a systematic view of a range of machine learning algorithms. Topics covered include two parts: (1) a gentle introduction of machine learning: generalization and model selection, regression and classification, kernels, neural networks, clustering and dimensionality reduction; (2) a statistical perspective of machine learning, where we will dive into several probabilistic supervised and unsupervised models, including logistic regression, Gaussian mixture models, and generative adversarial networks.
Prerequisite(s): CMSC 25300 or CMSC 35300 or STAT 24300 or STAT 24500 or ((MATH 18600 or MATH 20250) and (CMSC 12100 or CMSC 14100 or CMSC 15100 or CMSC 16100) and (STAT 25100 or STAT 25150))
Equivalent Course(s): STAT 27725

CMSC 25422. Machine Learning for Computer Systems. 100 Units.
This course will cover topics at the intersection of machine learning and systems, with a focus on applications of machine learning to computer systems. Topics covered will include applications of machine learning models to security, performance analysis, and prediction problems in systems; data preparation, feature selection, and feature extraction; design, development, and evaluation of machine learning models and pipelines; fairness, interpretability, and explainability of machine learning models; and testing and debugging of machine learning models. The topic of machine learning for computer systems is broad. Given the expertise of the instructor, many of the examples this term will focus on applications to computer networking. Yet, many of these principles apply broadly, across computer systems. You can and should think of this course as a practical hands-on introduction to machine learning models and concepts that will allow you to apply these models in practice. We'll focus on examples from networking, but you will walk away from the course with a good understanding of how to apply machine learning models to real-world datasets, how to use machine learning to help computer systems operate better, and the practical challenges with deploying machine learning models in practice."
Instructor(s): Nick Feamster
Prerequisite(s): CMSC 14300 or CMSC 15400
Equivalent Course(s): DATA 35422, DATA 25422, CMSC 35422

CMSC 25440. Machine Learning in Medicine. 100 Units.
In this course we will study the how machine learning is used in biomedical research and in healthcare delivery. We will build and explore a range of models in areas such as infectious disease and drug resistance, cancer diagnosis and treatment, drug design, genomics analysis, patient outcome prediction, medical records interpretation and medical imaging. Students will become familiar with the types and scale of data used to train and validate models and with the approaches to build, tune and deploy machine learned models. We will use traditional machine learning methods as well as deep learning depending on the problem. The course will be fast moving and will involve weekly program assignments. We will introduce the machine learning methods as we go, but previous familiarity with machine learning will be helpful. Programming assignments will be in python and we will use Google Collaboratory and Amazon AWS for compute intensive training.
Prerequisite(s): CMSC 12200 or CMSC 15200 or CMSC 16200. Proficiency in Python is expected.

CMSC 25460. Introduction to Optimization. 100 Units.
Introduction to Optimization
Prerequisite(s): (CMSC 27200 or CMSC 27230 or CMSC 37000) and CMSC 25300

CMSC 25500. Introduction to Neural Networks. 100 Units.
This course will provide an introduction to neural networks and fundamental concepts in deep learning. It will cover the basics of training neural networks, including backpropagation, stochastic gradient descent, regularization, and data augmentation. It will explore network design principles, spanning multilayer perceptrons, convolutional and recurrent architectures, attention, memory, and generative adversarial networks. Students will gain experience applying neural networks to modern problems in computer vision, natural language processing, and reinforcement learning. Note: students can use at most one of CMSC 25500 and TTIC 31230 towards the computer science major.
Prerequisite(s): CMSC 25300 or CMSC 25400 or TTIC 31020
Note(s): Students can use at most one of CMSC 25500 and TTIC 31230 towards a CS major or CS minor.

CMSC 25610. Undergraduate Computational Linguistics. 100 Units.
This course is an introduction to topics at the intersection of computation and language. We will study computational linguistics from both scientific and engineering angles: the use of computational modeling to address scientific questions in linguistics and cognitive science, as well as the design of computational systems to solve engineering problems in natural language processing (NLP). The course will combine analysis and discussion of these approaches with training in the programming and mathematical foundations necessary to put these methods into practice. The course is designed to accommodate students both with and without prior programming experience. Our goal is for all students to leave the course able to engage with and evaluate research in cognitive/linguistic modeling and NLP, and to be able to implement intermediate-level computational models.
Equivalent Course(s): LING 28610

CMSC 25800. Adversarial Machine Learning. 100 Units.
Machine learning tools are now integrated into all facets of our lives, ranging from facial recognition running on deep neural network classifiers on our phones, to large language models as productivity and educational tools. Yet they are dynamic and ever evolving tools with significant limitations and weaknesses, some of which can be used to manipulate their results, while others can be used to limit the harms produced by their misuse and weaponization. This course takes a pragmatic perspective to the intersection between machine learning and security and privacy. Students will learn about specific topics related to both security (robustness, consistency, attacks and defenses) and privacy of machine learning models, ranging from classifiers on deep neural network architectures to generative tools for text (large language models) and images (diffusion models). We will study both concepts and implementations of a range of attacks and defenses, from using optimization methods (SGD, PGD, CW) that generate model evasion attacks (e.g. adversarial examples) to broader classes of poisoning attacks (dirty-label, clean-label), privacy attacks (model-inversion/extraction, membership-inference), and evolving attacks against large language models. We will also cover techniques to improve model robustness and to defend against a variety of attacks.
Prerequisite(s): CMSC 25300 or CMSC 25400 or TTIC 31020.

CMSC 25900. Ethics, Fairness, Responsibility, and Privacy in Data Science. 100 Units.
This course takes a technical approach to understanding ethical issues in the design and implementation of computer systems. Tensions often arise between a computer system's utility and its privacy-invasiveness, between its robustness and its flexibility, and between its ability to leverage existing data and existing data's tendency to encode biases. The course will demonstrate how computer systems can violate individuals' privacy and agency, impact sub-populations in disparate ways, and harm both society and the environment. It will also introduce algorithmic approaches to fairness, privacy, transparency, and explainability in machine learning systems. Through hands-on programming assignments and projects, students will design and implement computer systems that reflect both ethics and privacy by design. They will also wrestle with fundamental questions about who bears responsibility for a system's shortcomings, how to balance different stakeholders' goals, and what societal values computer systems should embed.
Prerequisite(s): CMSC 20300 or CMSC 20600 or CMSC 21800 or CMSC 22000 or CMSC 22001 or CMSC 23000 or CMSC 23200 or CMSC 23300 or CMSC 23320 or CMSC 23400 or CMSC 23500 or CMSC 23900 or CMSC 25025

CMSC 25910. Engineering for Ethics, Privacy, and Fairness in Computer Systems. 100 Units.
This course takes a technical approach to understanding ethical issues in the design and implementation of computer systems. Tensions often arise between a computer system's utility and its privacy-invasiveness, between its robustness and its flexibility, and between its ability to leverage existing data and existing data's tendency to encode biases. The course will demonstrate how computer systems can violate individuals' privacy and agency, impact sub-populations in disparate ways, and harm both society and the environment. It will also introduce algorithmic approaches to fairness, privacy, transparency, and explainability in machine learning systems. Through hands-on programming assignments and projects, students will design and implement computer systems that reflect both ethics and privacy by design. They will also wrestle with fundamental questions about who bears responsibility for a system's shortcomings, how to balance different stakeholders' goals, and what societal values computer systems should embed. Students may not take CMSC 25910 if they have taken CMSC 25900 or DATA 25900.
Prerequisite(s): CMSC 14400 or CMSC 15400.
Note(s): anti-requisites: CMSC 25900, DATA 25900

CMSC 27100. Discrete Mathematics. 100 Units.
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.
Prerequisite(s): CMSC 14200, CMSC 15200 or CMSC 16200 or CMSC 12200), or (MATH 15910 or MATH 16300 or MATH 16310 or MATH 19900 or MATH 20300 or MATH 20310 or MATH 20400 or MATH 20410 or MATH 20700 or MATH 25400 or MATH 25500 or MATH 25700)

Note(s): This is a directed course in mathematical topics and techniques that is a prerequisite for courses such as CMSC 27200 and 27400.

CMSC 27130. Honors Discrete Mathematics. 100 Units.
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. Further topics include proof by induction; number theory, congruences, and Fermat's little theorem; relations; factorials, binomial coefficients and advanced counting; combinatorial probability; random variables, expected value, and variance; graph theory and trees. Time permitting, material on recurrences, asymptotic equality, rates of growth and Markov chains may be included as well. The honors version of Discrete Mathematics covers topics at a deeper level.
Prerequisite(s): (CMSC 14200 or CMSC 15200 or CMSC 16200 or CMSC 12200) or (MATH 15910 or MATH 16300 or MATH 16310 or MATH 19900 or MATH 20300 or MATH 20310 or MATH 20400 or MATH 20410 or MATH 20700 or MATH 25400 or MATH 25500 or MATH 25700)
Equivalent Course(s): MATH 28130

CMSC 27200. Theory of Algorithms. 100 Units.
This course covers design and analysis of efficient algorithms, with emphasis on ideas rather than on implementation. Algorithmic questions include sorting and searching, graph algorithms, elementary algorithmic number theory, combinatorial optimization, randomized algorithms, as well as techniques to deal with intractability, like approximation algorithms. 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, amortized analysis, analysis of probabilistic algorithms, the concepts of polynomial-time algorithms, and of NP-completeness.
Prerequisite(s): CMSC 27100 or CMSC 27130 or CMSC 37110

CMSC 27230. Honors Theory of Algorithms. 100 Units.
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. The honors version of Theory of Algorithms covers topics at a deeper level.
Prerequisite(s): CMSC 27100 or CMSC 27130 or CMSC 37110 or consent of the instructor.

CMSC 27410. Honors Combinatorics. 100 Units.
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.
Prerequisite(s): MATH 19900 or MATH 15900 or MATH 25400 or MATH 25700, or CMSC 27100 or CMSC 27130 or CMSC 37110. Experience with mathematical proofs.
Note(s): This course is offered in alternate years.
Equivalent Course(s): MATH 28410

CMSC 27500. Graph Theory. 100 Units.
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.
Prerequisite(s): CMSC 27100, or MATH 20400 or higher.

CMSC 27502. Advanced Algorithms. 100 Units.
TBD

CMSC 27530. Honors Graph Theory. 100 Units.
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, directly acyclic graphs, and tournaments. Techniques studied include the probabilistic method.
Prerequisite(s): CMSC 27100, CMSC 27130, or CMSC 37110, or MATH 20400 or MATH 20800.
Equivalent Course(s): MATH 28530

CMSC 27700-27800. Mathematical Logic I-II.
Mathematical Logic I-II

> **CMSC 27700. Mathematical Logic I. 100 Units.**
> 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.
Terms Offered: Autumn
Prerequisite(s): MATH 25400 or 25700
Equivalent Course(s): MATH 27700

CMSC 27800. Mathematical Logic II. 100 Units.
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).
Terms Offered: Winter
Prerequisite(s): MATH 27700 or equivalent
Equivalent Course(s): MATH 27800

CMSC 28000. Introduction to Formal Languages. 100 Units.
This course is a basic introduction to computability theory and formal languages. Topics include automata theory, regular languages, context-free languages, and Turing machines.
Prerequisite(s): CMSC 27100 or CMSC 27130 or CMSC 37110 or MATH 15900 or MATH 15910 or MATH 16300 or MATH 16310 or MATH 19900 or MATH 25500 or LING 21010
Equivalent Course(s): MATH 28000

CMSC 28100. Introduction to Complexity Theory. 100 Units.
Computability: Turing machines, Universal Turing machines and the Church-Turing thesis. Undecidability. Reducibilities. Complexity--the study of the amount of resources -- time, space, communication, randomness, etc -- needed in computations: Time and space complexity classes, nondeterministic and probabilistic computations. Complete problems. Lower bounds, and the big open problems: P vs NP, space vs. time, etc. Communication Complexity.
Prerequisite(s): CMSC 27200 or CMSC 27230 or CMSC 37000, or MATH 15900 or MATH 15910 or MATH 16300 or MATH 16310 or MATH 19900 or MATH 25500; experience with mathematical proofs.
Equivalent Course(s): MATH 28100

CMSC 28130. Honors Introduction to Complexity Theory. 100 Units.
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.
Prerequisite(s): CMSC 27100 or CMSC 27130, or MATH 15900 or MATH 19900 or MATH 25500; experience with mathematical proofs.

CMSC 28400. Introduction to Cryptography. 100 Units.
Cryptography is the use of algorithms to protect information from adversaries. Though its origins are ancient, cryptography now underlies everyday technologies including the Internet, wifi, cell phones, payment systems, and more. This course is an introduction to the design and analysis of cryptography, primarily from a theoretical perspective. It will cover how cryptography works, how security goals are formalized and analyzed theoretically, and the relationships between various cryptographic tasks. Topics will include symmetric encryption (aka: secret-key encryption), pseudorandomness, message authentication codes, collision-resistant hashing, asymmetric encryption (aka: public-key encryption), and digital signatures. We assume some background in discrete mathematics (e.g., the basics of discrete probability, counting, and modular arithmetic) and algorithms (e.g., big-O notation). We do not assume any prior knowledge of computer security, cryptography, or advanced mathematics. Problem sets will include writing proofs and will require applying the (many) new definitions and techniques that will be introduced over the course of the quarter. There may be a small number of assignments with a programming component.
Prerequisite(s): (CMSC 12200 or CMSC 14100 or CMSC 15200 or CMSC 16200) and ((CMSC 27200 or CMSC 27230 or CMSC 37000) or ((MATH 20300 or MATH 20310 or MATH 20320 or MATH 20700) and (MATH 25400 or MATH 25700)))

CMSC 28510. Introduction to Scientific Computing. 100 Units.
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.
Prerequisite(s): 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

CMSC 28515. Introduction to Numerical Partial Differential Equations. 100 Units.
This course deals with finite element and finite difference methods for second-order elliptic equations (diffusion) and the associated parabolic and hyperbolic equations. Some methods for solving linear algebraic systems will be used. Scalar first-order hyperbolic equations will be considered.
Prerequisite(s): A year of calculus (MATH 15300 or higher), a quarter of linear algebra (MATH 19620 or higher), CMSC 12200 or higher, and MATH 21100 or 21200 or CMSC 28510; or by consent.

CMSC 28540. Numerical Methods. 100 Units.
This is a practical programming course focused on the basic theory and efficient implementation of a broad sampling of common numerical methods. Each topic will be introduced conceptually followed by detailed exercises focused on both prototyping (using matlab) and programming the key foundational algorithms efficiently on modern (serial and multicore) architectures. The ideal student in this course would have a strong interest in the use of computer modeling as predictive tool in a range of discplines -- for example risk management, optimized engineering design, safety analysis, etc. The numerical methods studied in this course underlie the modeling and simulation of a huge range of physical and social phenomena, and are being put to increasing use to an increasing extent in industrial applications. After successfully completing this course, a student should have the necessary foundation to quickly gain expertise in any application-specific area of computer modeling.
Prerequisite(s): CMSC 15200 or CMSC 16200. Basic apprehension of calculus and linear algebra is essential

CMSC 29700. Reading and Research in Computer Science. 100 Units.
Students do reading and research in an area of computer science under the guidance of a faculty member. A written report is typically required.
Note(s): 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.

CMSC 29900. Bachelor's Thesis. 100 Units.
Open to fourth-year CS majors.
Note(s): Students should visit the Bx Thesis Program page for more details. https://cs.uchicago.edu/academics/undergraduate/bx-thesis-program/