Computer Science

UNDERGRADUATE PROGRAMS

The Department of Computer Science provides undergraduate instruction leading to the bachelor's degree in computer science. This program in computer science is accredited by the Computer Science Accreditation Board (CSAB), a specialized accrediting body recognized by the Council for Higher Education Accreditation (CHEA). The objective is to prepare professionals for meaningful careers in areas that require a thorough grounding in the underlying principles of computer systems, how they function and how they may be effectively applied to practical problems in a diversity of disciplines. Graduates will be prepared for the pursuit of graduate studies and for continued self-education. The department serves incoming freshmen, community college graduates, transfers from this and other institutions, nontraditional students continuing their education and others seeking instruction in computer science.

The department encourages students to earn an international studies certificate in parallel with the BS in computer science. Students interested in this program should seek advice from the Watson School advising office prior to initial registration.

Requirements for BS Degree in Computer Science

To receive the BS degree in computer science, the student must earn a minimum of 125 credit hours, including transfer credits, with an average of at least C (2.0 GPA), and a minimum of a C average in the major program.

A. *Credit Requirements*

A minimum of 125 semester credits of which:

1. a minimum of 60 credits must be in liberal arts and sciences courses.
2. a minimum of 30 credits must be earned in Watson School courses.

B. *Area Requirements* *credits*

1. Communications 8
   English writing and/or speech electives
2. Humanities/social science electives 16
3. Science 16
   PHYS 121 or 131. General Physics I
   PHYS 122 or 132. General Physics II

   Two science electives chosen from the following:
   BIOL 103. Unity of Life
   BIOL 104. Diversity of Life
   BIOL 113. Introductory Biology: Cell and Molecular
   BIOL 114. Introductory Biology: Organisms and Populations
   CHEM 107. Introductory Chemistry I
   CHEM 108. Introductory Chemistry II
   CHEM 111. Chemical Principles
   GEOL 111. Physical Geology
   GEOL 112. Oceanography
   GEOL 113. Geological Processes: An Introduction
   GEOL 114. The Earth's Dynamic Interior
   ASTR 114. The Sun, Stars and Galaxies
   PSYC 111. General Psychology and
   PSYC 112. Laboratory in General Psychology
4. Mathematics 19
   MATH 221. Calculus I
   MATH 222. Calculus II
   MATH 314. Discrete Mathematics
   MATH 341. Probability with Statistical Methods
   One elective chosen from:
   MATH 304. Linear Algebra
   MATH 358. Numerical Analysis I
   MATH 371. Mathematical Methods in Science I
   MATH 381. Graph Theory
5. Free electives 16
   Six credits must be in humanities, social sciences, arts and other disciplines (excluding computer science) that provide breadth of background.
6. Computer Science 50
   CS 140. Introduction to Computer Programming
   CS 210. Logic Design
   CS 220. Computer Organization and Assembly Language Programming
   CS 240. Data Structures
   CS 333. Algorithms
   CS 350. Operating Systems
   CS 373. Automata Theory and Formal Languages
   CS 471. Programming Languages
   CS 495. Senior Seminar in Computer Science

   Four electives chosen from at least two of the following four areas:
   Software Design—
   CS 340. Object-Oriented Programming
   CS 345. Software Engineering
   CS 348. The Human Computer Interface
   CS 460. Computer Graphics
   CS 472. Compiler Design
   Programming Languages—
   CS 340. Object-Oriented Programming

CS 342. Program Design Patterns
CS 360. GUI and Windows Programming
CS 465. Introduction to Artificial Intelligence
CS 472. Compiler Design

Computer Elements and Architecture—
CS 312. Intro Fault-Tolerant Computing
CS 323. Microcomputer Systems
CS 325. Advanced Computer Organization
CS 428. Computer Networks
CS 451. Operating Systems Implementation
CS 452. Systems Programming

Data Structures—
CS 432. Database Systems

One course from the following list may be used as a CS elective. This course will not count in any of the areas given above.
CS 395. Computer Science Internship
CS 396. Computer Science Co-op
CS 498. Undergraduate Project
CS 499. Undergraduate Research

TOTAL 125

C. *General Education Requirements*

The General Education requirements are described elsewhere in this *Bulletin*. Computer science majors can fulfill their General Education requirements within the 125-credit program described above. General Education courses should be taken during the freshman and sophomore years. For more information, see the "General Education and your Watson School Major" handout available in the Watson School advising office.

COMPUTER SCIENCE MINOR

The computer science minor consists of seven courses (CS 140, CS 210, CS 220, CS 240, MATH 314 and two CS courses at the 300 level or above). More information is available in the Watson School advising office.

COURSE OFFERINGS/ UNDERGRADUATE

NOTE: Unless otherwise noted, all undergraduate courses carry 4 credits and are offered every year.

CS 100. FUNDAMENTALS OF PROGRAMMING USING BASIC *every semester*
Elementary computer programming concepts: variables, expressions, statements, sequential execution, branching, selection, iteration, subroutines, simple data structures. Problem solving techniques and applications using BASIC. To obtain a working knowledge of the language, students are required to develop programs of moderate difficulty. Not open to students who have taken any other CS course and not applicable toward degree in computer science.

CS 101. FRESHMAN SEMINAR *1 credit*
Introduction to and discussion of topics of interest to computer science majors: university, school, and department resources; current developments in computer science. Open to and recommended for computer science freshmen. Prerequisites: none.

CS 105. INTRODUCTION TO COMPUTING *every semester*
Computing and its place in our society, including ethics and privacy. Basic concepts of computer hardware and systems. Data flow in computer systems. Understanding and using common application programs: word processors, spreadsheets and databases. Computers in communications. Basic concepts of algorithms, programming and the programming process.

CS 120. PRINCIPLES OF COMPUTER SCIENCE
Introduction to the principles that underlie the design and use of computers. Concept of an algorithm. Designing machines that can execute algorithms. Making hardware useful by building virtual machines. Use of programming languages to express algorithms. Discussion of what can be computed. How machines communicate. Unifying theme is the need for formal specification in order to automate the solution to a problem.

CS 140. INTRODUCTION TO COMPUTER PROGRAMMING *every semester*
Algorithms and programs. Design, coding, debugging, documentation of programs in structured high-level language. Programming applications selected from variety of areas. Supervised laboratory involves use of computing facilities and software development tools. Prerequisite: CS 105 or some familiarity with computers and programming.

CS 205. ADVANCED COMPUTER APPLICATIONS
Advanced networking applications, HTML and Web page development, effective computer-based presentations, project planning and scheduling tools, using the Internet for research. Prerequisite: CS 105 or experience with personal computer applications.

CS 210. LOGIC DESIGN *every semester*
Basic concepts in the design and implementation of combinational and sequential circuits. Logic families and digital integrated circuits. Number representation and basic computer arithmetic. Supervised laboratory work involves digital system design and implementation using digital ICs. Prerequisite: CS 140.

CS 220. COMPUTER ORGANIZATION AND ASSEMBLY LANGUAGE PROGRAMMING *every semester*
The architecture and organization of digital computer systems: data representation, algorithms and circuits for computer arithmetic, processor, memory and I/O organization. Instruction encoding and addressing modes. I/O techniques. Interrupt logic and interrupt handling. Assemblers and macroprocessors. Assembly language programming. Supervised laboratory work involves assembly language programming. Prerequisite: CS 210.

CS 240. DATA STRUCTURES *every semester*
Introduction to modern imperative languages, development tools and methodologies for modular programming. Emphasis on software design using functional and data abstraction. Specification use and implementation of abstract data types such as stacks, queues, lists, tree and graphs. Programming language features such as recursion, dynamically allocated data structures and separate compi-

lation. Introduction to algorithm analysis, searching and sorting. Prerequisite: CS 140.

CS 244. INTRODUCTION TO C PROGRAMMING *2 credits*
C syntax. Programming techniques and applications appropriate for C language. Students write several programs. Prerequisite: CS 240 or equivalent.

CS 245. INTRODUCTION TO ADA PROGRAMMING *2 credits*
Ada syntax. Programming techniques and applications appropriate for Ada language. Students write several programs. Prerequisite: CS 240 or equivalent.

CS 248. INTRODUCTION TO JAVA *2 credits*
Introduction to the principles of Java programming: object-oriented style, coding style, Java API libraries, code reuse, writing stand-alone applications and applets for the Web. Prerequisite: CS 240 or equivalent.

CS 249. INTRODUCTION TO UNIX *2 credits*
Overview of the UNIX operating system and shell programming. History and design of UNIX commands (sed, grep, chmod, pattern matching, etc.). File and information management. Shell programming (scripts, flow control, programming environment, etc.). UNIX network programming (signals, pipes, sockets, etc.).

CS 312. FAULT-TOLERANT COMPUTING
Representation and classification of faults, techniques for fault-tolerant design of digital systems, fault detection and location, design of easily testable systems, error detecting and correcting codes, software fault-tolerance, time redundancy techniques for tolerating transient faults. Current and future applications of fault-tolerant design. Prerequisites: CS 220 and 240, or CS 341.

CS 323. MICROCOMPUTER SYSTEMS
Microprocessor architecture and microcomputer system hardware. Advanced assembly language programming and use of advanced assembler functions. Microprocessor support chips: memory, programmable ports, DMA controllers, USARTs, CRT controllers and disk controllers. Comparison of contemporary microprocessor systems. Supervised laboratory work involves microprocessor programming and interfacing experiments. Prerequisite: CS 220.

CS 325. ADVANCED COMPUTER ORGANIZATION
Processing and input/output overlapping techniques: interrupts, DMA, and channels. Memory organization: cache memory, interleaving, secondary storage devices, paging and segmentation. Instruction set design. High-speed arithmetic circuits. Control design: hard-wired and microprogrammed control. Pipelined, array, and multiprocessor systems. Fault-tolerant architectures. Case studies of contemporary microprocessors, medium/large-scale mainframes, and multiprocessors. Prerequisite: CS 220.

CS 328. INTERNET PROGRAMMING
Definition of Internet protocol and how to write programs that use the Internet. Starting with sockets and the TCP and UDP protocols, covers basic Internet protocols and applications, including ping, telnet, ftp, mail and http. More advanced protocols include RMI and Cobra. Java-based clients (applets and applications) and server-side Java servlets also examined. Projects include practical use of these protocols. Prerequisites: CS 240 and a working knowledge of Java.

CS 333. ALGORITHMS *every semester*
Analysis of common algorithms for processing strings, trees, graphs and networks. Comparison of sorting and searching algorithms. Algorithm design strategies: divide and conquer, dynamic, greedy, back tracking, branch and bound. Introduction to NP-completeness and parallel algorithms. Prerequisites: CS 240 and MATH 314.

CS 340. OBJECT-ORIENTED PROGRAMMING
Object-oriented analysis (OOA) and object-oriented design (OOD) concepts applied to object-oriented programming (OOP) using selected language. Method-driven and model-driven (OOA) approaches. Methodologies and tools. Objects, messages, classes, encapsulation, inheritance, polymorphism. Prototyping, code reuse and message connection simplicity. Students learn to formulate object solutions to practical problems through the use of projects. Prerequisites: CS 240 and MATH 304.

CS 341. DATA STRUCTURES AND ALGORITHMS FOR ELECTRICAL ENGINEERING *fall, 3 credits*
Program specification using functional and data abstraction. Implementation of stacks, queues, lists, trees and graphs. Recursion, dynamic allocation, compilation. Introduction to algorithm analysis and synthesis. Sorting and searching algorithms. Design projects. Prerequisite: EE 351 or equivalent and prior programming experience in a high-level language. For EE students only. Computer science students may not take this course to satisfy program requirements.

CS 342. PROGRAM DESIGN PATTERNS
Introduction to Smalltalk language and programming environment. Patterns for program design; examples of patterns used in existing software libraries, such as a windowing tool kit. Exercises in programming with design patterns and communicating designs to other programmers using the language of patterns. Programming graphical-interface and network applications in an object-oriented language other than Smalltalk. Prerequisite: CS 333, MATH 314 or 330 and a knowledge of Java.

CS 345. SOFTWARE ENGINEERING
Theory and practice of software engineering, especially as applied to life cycle of large software and computer systems. Project management, system requirements and specifications. Design representation and documentation. Implementation, installation and maintenance. Application of automation to development effort. Semester-long, term-oriented project allows students to execute theory. Prerequisite: CS 240.

CS 348. THE HUMAN-COMPUTER INTERFACE
Broad overview of issues in human-computer interaction, including methodologies for design and evaluation, user friendliness, use of input devices, dialogue design, voice input/output, training and cognitive models and theories. Prerequisite: CS 240.

CS 350. OPERATING SYSTEMS *every semester*
Introduction to fundamental concepts underlying the design and implementation of operating systems. Process concept and process management; processor and memory management; file systems; input/output subsystems; protection; security issues. Introduction to distributed systems. Prerequisite: CS 220.

CS 360. GUI AND WINDOWS PROGRAMMING
An overview of the issues involved in the design and implementation of graphical user interfaces (GUI) and windows applications. A practical, hands-on course that teaches many of the interactive, pointer-based, graphical techniques that comprise the modern desktop interaction metaphor. Microsoft Windows; the X Window System; event-driven programming; client/server model; Microsoft's API; Xlib; interface tools; window managers; widgets; resources; graphics and text in windows; future directions of GUIs; multimedia; 3D interaction. This is a project-oriented course that emphasizes the programming of windows applications rather than the aesthetical and psychological issues involved in user-interface design. Prerequisites: CS 220 and CS 240.

CS 373. AUTOMATA THEORY AND FORMAL LANGUAGES *every semester*
Theory and application of automata and the languages they recognize. Regular languages, finite-state automata, regular expressions, context-free languages, normal forms, pushdown automata, context-sensitive languages, linear bounded automata, Turing machines, computability, transducers. Application of concepts. Prerequisites: CS 240 and MATH 314.

CS 380. TOPICS IN COMPUTER SCIENCE
Topic varies, depending on interests of instructor.

CS 395. COMPUTER SCIENCE INTERNSHIP
every semester, 2-4 credits
On-the-job experience in computer science. Student interns have opportunities to work in local industrial, commercial or educational institutions and to apply their knowledge to practical professional problems. Formal classroom meetings in which interns share their experiences. Open only to juniors or seniors in computer science major. Registration competitive and by permission of instructor.

CS 396. COMPUTER SCIENCE CO-OP
every semester
On-the-job experience in computer science. Co-op students work 20 hours/week, September-May, in local industrial, commercial or educational organization and apply their knowledge to practical, professional problems. Students share experiences and discuss job search techniques in formal class meetings. Compensation provided by sponsor organization. Prerequisites: four courses in computer science; open only to matriculated juniors and seniors in computer science. Registration, by permission of instructor, is competitive and requires sponsor interview.

CS 397. INDEPENDENT STUDY *variable credit*
Individual study under direct supervision of faculty member investigating topic of interest to student. Special registration form required with signature of supervising faculty member.

CS 428. COMPUTER NETWORKS
Survey of data communications and computer networking history, technology, and systems. Fundamentals of data communications (data transmission and encoding, error detection techniques, flow control, etc.). Data communication networking (circuit-switched networks, packet switched networks, local area networks, etc.). Computer communications architecture, algorithms and protocols (X.25, TCP/IP, etc.). Internetworking. Contemporary features and issues (ISDN, ATM, FDDI, etc.). Prerequisite: one of CS 350, MATH 147 or MATH 341.

CS 432. DATABASE SYSTEMS
Associations between data elements and data models: entity-relationship, relational and object-oriented. Relational database design techniques. Various query languages. Introduction to query processing, transaction management and concurrency control. Prerequisite: CS 333.

CS 451. OPERATING SYSTEMS IMPLEMENTATION
Practical aspects of the implementation of operating systems. Issues and trade-offs involved in design of operating systems and their components. Assignments and project work involving design and implementation of key areas of multiprogrammed operating systems. Prerequisite: CS 350.

CS 452. SYSTEMS PROGRAMMING
Fundamental concepts in systems programming. Input/output programming: design and implementation of assemblers, loaders, linkage editors and macroprocessors; secondary storage organization and file processing; introduction to data communications. Prerequisites: CS 220 and 333.

CS 460. COMPUTER GRAPHICS
Concepts, structure, techniques and algorithms for use of modern interactive computer graphics systems. Graphics hardware, software system structure. Techniques and algorithms for basic graphics input/output functions. Matrix techniques for transformations and projections. Techniques for two- and three- dimensional modeling, rendering and visualization. Prerequisite: CS 240. Corequisite: MATH 304.

CS 462. NEURAL NETWORKS AND COMPUTATIONAL INTELLIGENCE
Introductory biological- or statistical-based neural network paradigms. Introduction to genetic algorithms, differential evolution and evolutionary programming. Discussion of both local and global training methods. Introduction to hybrid training methods that include numerical optimization and simple differential evolution. Group projects (which involve the design, implementation, training and testing of neural networks, in addition to a group project based either on evolutionary programming technology or genetic algorithm technology). Prerequisites: CS 333, MATH 304 and MATH 341.

CS 465. INTRODUCTION TO ARTIFICIAL INTELLIGENCE
Introduction to programming languages used in artificial intelligence and coverage of one particular language in depth. Assorted topics in artificial intelligence: search techniques for artificial intelligence applications, knowledge representation and expert systems. Prerequisite: CS 333.

CS 471. PROGRAMMING LANGUAGES
every semester
Characteristics of several types of programming languages: for example, procedural, functional, declarative and object-oriented languages. Formal syntax specification, Backus-Naur Form, introduction to language semantics. Language facilities for data types, control structures, subprograms. Run-time environments. Introduction to language processing. Prerequisite: CS 333.

CS 472. COMPILER DESIGN
Fundamentals of programming language translation. Compiler design concepts. General aspects of lexical analysis and parsing of context-free languages. Grammars and parsing techniques. Syntax-directed translation. Declarations and symbol management. Semantic processing and code generation. Principles, methods and examples of code optimization. Prerequisite: CS 471.

CS 495. SENIOR SEMINAR IN COMPUTER SCIENCE
every semester
Computer science as a profession. Ethical and social implications of computing. Development and application of written and oral communication skills. Teamwork and programming as a group activity. Prerequisite: senior standing.

CS 498. UNDERGRADUATE PROJECT
Participation in a project under supervision of a faculty member. Written report and oral presentation required.

CS. 499. UNDERGRADUATE RESEARCH
Participation in a project under supervision of a faculty member. Written report and oral presentation required.

GRADUATE PROGRAMS
Master of Science in Computer Science

REQUIREMENTS
Holders of the baccalaureate degree in computer science or a related field are invited to apply for admission to the MSCS program. Students whose undergraduate degree is not in computer science may be required to complete some preparatory work in addition to fulfilling the requirements listed below.

1. Complete at least one course in each of the following core areas:
    a. Architecture and Operating Systems
        CS 522. Computer Organization and Architecture
        CS 552. Operating Systems
    b. Programming Languages and Software Design
        CS 571. Programming Languages
        CS 572. Compiler Construction
    c. Theoretical Computer Science
        CS 573. Automata Theory and Formal Languages
        CS 575. Design and Analysis of Computer Algorithms
2. Complete one of the following options:
    a. Complete seven courses approved by the student's faculty adviser (making a total of 10 courses) and pass a comprehensive examination.
    b. Complete six courses approved by the student's faculty adviser (making a total of nine courses) and develop and present a project.
    c. Complete five courses approved by the student's faculty adviser (making a total of eight courses) and write and defend a thesis.
3. Maintain a B average in all coursework.

With faculty adviser approval, courses may be taken from other departments in the Watson School or from other schools within the University.

Doctoral Program in Computer Science
For more information about the PhD sequence, see "Graduate Information" above.

The doctoral program leads to a PhD degree in computer science. Students admitted into the program typically have a master's degree in computer science or a closely related discipline. Students with a BS degree and a strong academic record may also be directly admitted.

PhD students are required to have a minimum of 24 credit hours in residence. Students have to pass two qualifying examinations: a general comprehensive exam and a specialization exam covering the intended area of research. The general comprehensive exam covers the following five areas: (a) algorithms, (b) architecture, (c) operating systems, (d) programming languages and (e) any one of the following: artificial intelligence, compilers, database, automata theory and computability, or networks. The PhD student is also required to present and defend a prospectus that describes the intended research topic. Finally, the PhD dissertation has to be successfully defended.

Students in the PhD program must, at an early stage, identify a dissertation adviser from one of the full-time computer science faculty who shares their research interests.

COURSE OFFERINGS/ GRADUATE

Unless otherwise noted, all graduate courses carry 3 credits.

**Pending Graduate Council approval.*

CS 511. DESIGN AUTOMATION IN DIGITAL SYSTEMS
Principles for efficient computer-aided design; computer hardware description languages; hardware compiler (translator); system and logic level simulation; test generation, design verification, computer-aided logic design; physical construction. Prerequisite: CS 210 or knowledge of logic design.

CS 512. DIAGNOSIS AND RELIABLE DESIGN OF

DIGITAL SYSTEMS I
Digital system reliability and maintainability. Design for testability and built-in self-test. Fault modeling, test generation, functional testing. Fault-tolerant design techniques, architectures and software. Error detecting and correcting codes, self-checking and fail-safe logic. Prerequisite: CS 210 or knowledge of logic design.

CS 514. INTRODUCTION TO VLSI DESIGN
CMOS layout design rules, CMOS logic families, basic cell designs (gates, latches, memory cells, etc.), floor planning. Project involves use of VLSI design tools to design a small chip (such as small CPU, associative memory, array multiplier) that will eventually be fabricated using the MOSIS facilities. Prerequisite: CS 210.

CS 515. VLSI PROCESSOR DESIGN
Advanced issues in VLSI microprocessor design: datapath and control design techniques and trade-offs, using cell libraries of datapath components. Team project involves the specification, design, and implementation of a (pipeline) RISC CPU that will eventually be fabricated using the MOSIS facilities. Prerequisite: CS 514 (alternatives not acceptable).

CS 522. COMPUTER ARCHITECTURE AND ORGANIZATION *normally offered fall*
Pipelined processors: basic theory, instruction pipelines, multifunction units, instruction scheduling, precise interrupts. Pipelined vector machines. Superscalar and VLIW architectures. High-speed memory system design. Overview of parallel architectures: SIMD/MIMD systems, interconnection networks, synchronization and cache coherence. Prerequisite: CS 325.

CS 524. MICROCOMPUTER SYSTEMS
Advanced concepts in microprocessor systems such as interrupt handling, A-D and D-A conversion, programmable peripheral controllers, caches, multitasking, protection, memory management and virtual memory. Laboratory work will involve construction of a nontrivial microprocessor system. Prerequisite: CS 323.

CS 528. COMPUTER NETWORKS AND DATA COMMUNICATIONS
Survey of computer communication networks. Fundamental concepts of circuit and packet switching, local and remote networks, OSI reference model, protocols and network control algorithms. Prerequisites: CS 350 and some probability theory.

CS 529. HIGH PERFORMANCE COMPUTER NETWORKS
New technologies for high speed networking (such as ATM, Fibre channel, Myrient). Interaction among operating system, protocol stack and the network interface hardware. OS kernel and mapping requirements for low latency networking. Lightweight protocols; multicasting protocols. Synchronizing high speed streams. Innovative applications of low latency LANs (such as clustered supercomputing, distributed single-image servers, multimedia servers). Hands-on laboratory exercises on several high-speed LANs. Prerequisites: CS 350 and CS 428 or equivalent. Undergraduate students with the appropriate background can take this course as an elective.

CS 532. DATABASE SYSTEMS
Associations between data elements and data models: entity-relationship, relational and object-oriented. Relational database design techniques. Formal and commercial query languages. Introduction to query processing, transaction management and concurrency control. Prerequisite: CS 333.

CS 533. INFORMATION RETRIEVAL
Indexing and data structures for storing and searching the index. Boolean, statistical, inference nets and knowledge-based models. Thesaurus construction. Query expansion. Natural language and linguistic techniques. Evaluation. Distributed information retrieval. Information integration and fusion. Dissemination of information. Summaries, themes and reading tours. Hypertext. Internet tools. Intelligent agents. Digital libraries. Prerequisite: CS 333.

CS 541. CONCEPTS IN COMPUTER PROGRAMMING
All phases of problem solving by computer: definition of problems, design, implementation, verification. Hierarchical design tools, correctness of programs (structured programming, program reading), elementary data structures. Prerequisite: Programming at level of CS 140. Cannot be used for MSCS credit.

CS 544. OBJECT-ORIENTED SYSTEMS DEVELOPMENT *4 credits*
Object-oriented analysis, design and programming (OOA/OOD/OOP) methodology. Develop process that begins with system concept and finishes with operational OOP code. Formulate encapsulated base classes using OOA concepts and derive classes using inheritance and polymorphism. Translate OO system design. Translate OOD into OOP code making use of HIPOS or other representation. Reinforce methodology using "case studies" of interesting complexity. Prerequisites: algorithms, linear algebra.

CS 545. SOFTWARE ENGINEERING
Techniques for software development. Software life cycles. Software cost factors, estimation techniques. Software design concepts; design methodologies, notations. Language support for life-cycle; software verification, testing. Individual, team software design projects. Prerequisite: CS 333.

CS 546. SOFTWARE ENGINEERING ANALYSIS
Analytic methodologies associated with software engineering and its application to large projects. Software economics, verification and testing, software metrics, performance, design of experiments. Prerequisite: CS 333.

CS 548. INTRODUCTION TO MULTIMEDIA SYSTEMS
Multimedia's opportunities, problems, and solutions. Creating and interacting with video and audio, as well as with text, data and graphics. Prerequisite: CS 333 or equivalent.

CS 552. OPERATING SYSTEMS *fall*
Advanced topics in operating systems. Process synchronization, linguistic support for concurrency, virtual memory, deadlock theory, robustness, security, mathematical models and correctness of concurrent programs. Treatment of selected topics in distributed and multiprocessor operating systems. Prerequisite: CS 350 or equivalent.

CS 560. COMPUTER GRAPHICS
Concepts, structure, techniques, algorithms for use of modern interactive computer graphics systems. Graphics hardware, software system structure. Techniques and algorithms for basic graphics input-output functions. Matrix techniques for transformations and projections. Techniques

for three-dimensional modeling and visualization. Prerequisites: CS 333 and linear algebra.

CS 562. NEURAL NETWORKS/GENETIC OPTIMIZATION APPLICATIONS
Emphasis on tool building and applications. Neural networks: multilayer propagation, multi-temporal paradigms, pre- and post-processing, training. Real domain neural networks; network sizing. Evolutionary computing. Genetic optimization: coding, fitness functions, reproduction and convergence. Comparison with gradient methods, iterated search and simulated annealing. Implementation in an object-oriented language using libraries of object-oriented reusable components. Prerequisites: CS 333 and MATH 304.

CS 565. INTRODUCTION TO ARTIFICIAL INTELLIGENCE
An introduction to programming languages used in artificial intelligence and coverage of one particular language in depth. Assorted topics in artificial intelligence, including search techniques for artificial intelligence applications, knowledge representation and expert systems. Prerequisite: CS 333.

CS 566. TOPICS IN ARTIFICIAL INTELLIGENCE
Topics in artificial intelligence selected from natural language processing, learning, automated theorem proving, logics for artificial intelligence, planning, robotics and vision. Prerequisite: CS 565.

CS 571. PROGRAMMING LANGUAGES *normally offered fall*
Selected topics in programming languages and alternative programming paradigms. Functional and imperative languages. Logic programming and object-oriented programming paradigms. Languages for concurrent computation. Semantics of programming languages. Prerequisite: CS 471.

CS 572. COMPILER CONSTRUCTION *normally offered spring*
Fundamentals of programming language translation. Compiler design concepts. General aspects of lexical analysis and parsing of context-free languages. Grammars and parsing techniques. Syntax-directed translation. Declarations and symbol management. Semantic processing and code generation. Principles, methods and examples of code optimization. Prerequisite: CS 471.

CS 573. AUTOMATA THEORY AND FORMAL LANGUAGES *normally offered fall*
Regular languages, finite automata and regular expressions. Context-free languages and grammars, normal forms, pushdown automata. Recursive and recursively enumerable languages. Turing machines. Introduction to undecidability. Prerequisite: MATH 314.

CS 575. DESIGN AND ANALYSIS OF COMPUTER ALGORITHMS *normally offered spring*
Analysis of programs and review of design techniques. Lower bound theory and NP-completeness. Heuristic, approximation, probabilistic and parallel algorithms. Prerequisites: CS 373 and 333.

CS 576. COMPUTER SYSTEM PERFORMANCE
Concepts, methods involved in computer system performance modelling, measurement, evaluation. Workload characterization, problems involved with improvement of existing systems, design of future systems. Laboratory experience involving computer system performance monitoring under actual workload conditions. Prerequisites: CS 350 and probability and statistics.

CS 577. QUEUING THEORY AND NETWORKS
Summary of queuing theory concepts, bounds on waiting times, priority queue models. Modeling time-sharing and multiprocessor systems. Analysis and design of computer communication networks. Recent topics in networks performance analysis. Prerequisite: CS 350.

CS 578. FORMAL SPECIFICATION AND VALIDATION
Introduction to formal techniques for specification and validation of communication systems, computer systems and software. Topics include finite state machine methods, communicating sequential processes, calculus of communicating systems, axiomatic program semantics, Petri nets, temporal logic. Prerequisite: CS 373.

CS 580. SPECIAL TOPICS *1-4 credits*
Topics in specialized areas varying from semester to semester.

CS 594. INDUSTRIAL INTERNSHIP
Computer science, engineering and other professional experience. Daily log book memo progress reports and a formal report required. The internship may replace no more than one lecture course for the MSCS or MEng degree. Prerequisite: permission of department chair.

CS 595. TERMINATION PROJECT IN COMPUTER SCIENCE
A theoretical or practical project carried out under the supervision of a member of the Computer Science Department. Project documentation must be submitted to the department library and a public presentation is required. Further information is available in the department office. Prerequisite: consent of instructor and committee members.

CS 597. INDEPENDENT STUDY *1-12 credits*
Independent study supervised by a computer science faculty member. Student must obtain consent of instructor, who then determines description of study program, number of credits, frequency of meetings and location.

CS 599. MASTER'S THESIS *1-6 credits*
Research for and preparation of thesis. Must be approved by department chair.

CS 611. DESIGN AUTOMATION SEMINAR
Presentations by experts in industry and instruction on recent developments and current trends in various areas of design automation, such as design languages, efficient translation, hierarchical simulation, design verification, test generation, silicon compilation, physical design. Each student works on a project and gives a presentation. Prerequisite: CS 511.

CS 612. DIAGNOSIS AND RELIABLE DESIGN OF DIGITAL SYSTEMS II
Design to simplify testing of digital systems. Fault simulation. Advanced techniques in modeling, testing, error detection and fault isolation; system diagnosis; architecture and software fault-tolerance. Future trends in fault-tolerant computing. Prerequisite: CS 512.

CS 622. SEMINAR IN ALTERNATIVE COMPUTING CONCEPTS
Architecture/compiler synergism and design issues in the implementation of alternative paradigms such as object-oriented functional-based compiling and logic programming. Topics may vary from semester to semester to reflect current trends. Prerequisites: CS 522 and 571.

CS 624. PARALLEL PROCESSING ARCHITECTURES
SIMD and MIMD systems, programming issues and case studies. Advanced topics in interconnection network design, synchronization and cache coherence. Data and demand-driven architectures, systolic and wavefront arrays and other innovative approaches to parallel processing. Prerequisite: CS 522.

CS 625. PARALLEL PROCESSING SOFTWARE
Overview of parallel architectures. Parallel algorithms. Parallel programming languages and environments. Parallelizing and vectorizing compilers, optimization techniques, elimination of globals, hot-spots. Loop synchronization. Compiling for VLIW architectures. Schedulers for parallel machines, operating systems issues. Prerequisites: CS 522 and 575.

CS 628. COMPUTER AND COMMUNICATION NETWORKS SEMINAR
Current and advanced issues in the design, specification, analysis and verification of computer communication networks. Prerequisite: CS 528 or 577.

CS 632. ADVANCED DATABASE SYSTEMS
Coverage of selected advanced areas in database systems and applications. Specific topics vary from year to year to reflect current research interests and may include but are not limited to topics in distributed database, heterogeneous database, Internet-based text database, object-oriented database, data mining. Prerequisite: CS 432 or CS 532.

CS 652. OPERATING SYSTEMS SEMINAR
Issues in operating systems design, analysis and implementation. Specific topics vary from year to year and are chosen from current literature in distributed multiprocessing and real-time systems. Students present reports based on analysis of reading from the current literature. Prerequisite: CS 552.

CS 654. DISTRIBUTED SYSTEMS
Fundamental issues in distributed systems. Distributed synchronization and concurrency control. Distributed process management (scheduling, remote invocation, task forces, load balancing). Protection and security. Robust distributed systems. Case studies. Prerequisite: CS 552.

CS 660. ADVANCED COMPUTER GRAPHICS
A comprehensive review of the techniques needed to produce computer-generated shaded images of three-dimensional scenes. Recent research results are presented. Students design and implement portions of a three-dimensional graphics package. Topics selected from: modern graphics standards (PHIGS, X-Windows), user interface issues, 3-D viewing, geometric modeling, image synthesis, image manipulation, animation, scientific visualization. Prerequisite: CS 560.

CS 667. TOPICS IN LOGIC PROGRAMMING
Coverage of some advanced areas in logic programming that should prepare students to do research in the field. Selected topics may include the theory of logic programming, implementation, languages for parallel logic programming, analysis of logic programs. Prerequisite: knowledge of PROLOG programming, as may be acquired in CS 565.

CS 673. COMPUTABILITY AND COMPLEXITY THEORY
Coverage of important areas of computability and complexity theory. Topics may include primitive recursive functions, general recursive functions and their enumeration via Turing machines, Kleene's theorem, Blum's theory, Chaitin's theory, program schemata, incomputable functions, the structure of NP, time and space complexity, serial, parallel, deterministic, probabilistic and non-deterministic computation. Prerequisite: CS 575.

CS 681. TOPICS IN COMPUTER SCIENCE *2 credits*
Seminar course, primarily for students active or interested in advanced graduate work in computer science. Seminars based on recent research given by faculty and students. Prerequisite: completion of at least three courses at 500 level in computer science.

CS 688. COMPUTER SCIENCE GRADUATE SEMINAR *every semester, 1 credit*
Weekly seminar presentation by invited speakers, department faculty, and graduate students on contemporary topics in computer science and related fields. Cannot be used toward the MSCS. Prerequisite: graduate standing in computer science.

CS 697. ADVANCED INDEPENDENT STUDY *1-12 credits*
Reading and research on special advanced topics under direction of computer science adviser. Student must obtain consent of professor who then determines description of study program, number of credits, frequency of meetings, location.

CS 698. PREDISSERTATION RESEARCH *1-9 credits*
Reserved for exploratory research oriented toward dissertation.

CS 699. DISSERTATION *1-12 credits*
Research for and preparation of dissertation. Registration restricted to those admitted to candidacy.

CS 700. CONTINUOUS REGISTRATION *1 credit/semester*
Required for maintenance of matriculated status in graduate program when no other course taken. No credit toward graduate degree requirements.

CS 701. PRACTICUM FOR RESEARCH AND TEACHING ASSISTANTS *every semester*
Required for all funded graduate assistants. Research or teaching supervised by faculty adviser.

CS 707. RESEARCH SKILLS *1-4 credits*
Development of research skills required within graduate programs. May not be applied toward course credits for any graduate degree. Prerequisite: approval of relevant graduate program directors or department chairs.